



CSE3502: Information Security Management

**TOPIC: NETWORK INTRUSION DETECTION
AND PREVENTION USING SNORT TOOL**

TEAM MEMBERS:

SHIVANGI CHAURASIA : 19BCE0607

UTKARSH SHARMA : 19BCE2506

SOHAN KANTIMAHANTHI : 19BCT0207

Submitted to:

Prof. Ruby D

Scope:

Every action in today's information era results in the creation of data in some or the other form. According to estimates, over 300,000 tweets are sent every minute, and over 4 million Facebook postings are made every minute. Knowing that additional users and data requires increased security. Security and dependability have become key concerns for individuals and organizations.

So observing the current scenario we have decided to address various terminologies, strategies, and practical procedures associated with the SNORT Intrusion Detection and Prevention System (IDPS) in our project. Therefore, through our project we present an alternative approach to the implementation of IDPS which focuses on the concept of an Intrusion Detection System (IDS) utilizing Snort, a popular tool for this purpose.

Abstract:

Security and reliability are the most important considerations in our day-to-day network usage. However, as network technology advances, assaults are becoming more sophisticated than defenses. As a result, the idea of SNORT Intrusion Detection System (IDS) and SNORT Intrusion Prevention System (IPS) is attracting security specialists to protect the network from illegal access.

We propose a level based architecture to tackle numerous difficulties such as low detection rate, inability to handle large amounts of traffic, unsupported automatic tweaking, and so on. All stages in the project will be incremental, meaning that they will be capable of providing the needed functionality as well as its lower levels.

Introduction:

Living withinside the age of information, every and each movement bring about a few shapes of statistics creation. According to statistics, over three hundred thousand tweets and over four million Facebook posts are being generated in minutes. Knowing the truth that extra customers and extra statistics require extra safety. In the present day era, safety and reliability have turn out to be the most important issues for an man or woman or an organization. In this venture we additionally mentioned approximately the diverse terminologies, strategies and realistic methodologies associated with SNORT Intrusion Detection and Prevention System (IDPS). This venture affords distinctive methods on implementation of IDPS this is primarily based totally on in-depth take a look at of diverse studies endeavors. It majorly offers with the idea of Intrusion Detection System the use of Snort that's a famous device for community safety. It is broadly customary with the aid of using company sectors on the way to steady their organization's community. The paper offers a truthful information of Snort, approximately its purpose, the modes it associated with, its implementations and the applications. Review has been made on the premise of the research and studies carried out withinside the literature section.

Technical analysis

In this project we have decided to design multiple attacks on our own to test it against our snort tool rules set configuration.

1) Tool Based attacks -

- DDOS attack using hping3 using kali Linux
- MAC spoofing using WI-FI LAN adapter hardware device
- Hacking windows PC using metasploit
- UDP attack using fping command on kali Linux
- Change of MAC address using NETCUT

2) Code based attacks -

- Environment variable manipulation - Using this attack we can modify system files and system environment variable
- SYN Flood attack (Type of DOS): This will be flooding the server with unacknowledged Sync requests over TCP.

Literature Survey:

S.No	Title of the Research papers	Name of the author with year	Major technologies used	Results/Outcome of their research	Drawbacks if any
1	Intrusion Detection Prevention System using SNORT	A Tasneem, A Kumar, S Sharma - 2018	IDPS (Intrusion Detection & Prevention System)	The main points of this study are the IDPS technologies and approaches, as well as how SNORT may help with the complete process. The success of IDPs is determined by the rate of detection and the number of false positives.	The success of IDPs is determined by the rate of detection and the number of false positives.
2	Cloud based Security Framework for Anomaly Based Intrusion Detection using Machine Learning Techniques	N Thakkar, M Karamta, S Joshi - 2019	IDS, Cloud security	In this paper, SNORT IDS method is used in cloud environments to detect intrusions.	Different cloud architectures require different SNORT systems.
3	Early Intrusion Detection System (IDS) using Snort and Telegram approach	A Erlansari, FF Coastera, A Husamuddin - 2020	IDS, SNORT, Real time monitoring system	Snort supported IDS to detect early intrusion and notify the server administrator through the Telegram	More rules scenarios on SNORT are needed to get better security in network systems.

				application.	
4	Performance Evaluation of <i>Snort</i> and <i>Suricata</i> Intrusion Detection Systems on Ubuntu Server	A Gupta, LS Sharma - 2020	SNORT, parallel architecture, signature based pattern matching	<p>The goalPreprocessor takes the packets and check them against set plug-ins</p> <p>like RPC plug-in, HTTP plug-in, port scanner plug-in. These plug-ins check for a certain type of behavior from the packet.</p> <p>Keyr</p> <p>distri</p> <p>Riya C</p> <p>Intrus</p> <p>On that particular behavior plug-in send that packet to Detection engine.</p> <p>Devil's</p> <p>Plug-ins can be enabled and disabled on need basis.</p> <p>Hel</p> <p>Riya CI</p> <p>Snort support</p>	More surveys on a larger scale are required to check the feasibility of these techniques highlighted.

				<p>many kind of preprocessors and their attendant plug ins, covering many commonly used protocols.</p> <p>of this work is to provide a categorical survey of the many research strategies used to improve the performance of open-source intrusion detection systems.</p>	
5	Intrusion Detection System Techniques and Tools	<p>Resmi, A. M., & Manicka, R. (2017)., A. M., & Manicka, R. (2017). : A Survey.</p> <p><i>Scholars Journal of Engineering and Technology (SJET)</i>.</p>	SNORT, Software Defined Networking(SD N)	<p>The author of this work did a survey on the entire study of IDS, its nature and methodology, as well as technologies utilised in the intrusion detection field.. The survey provides an overview of the tools used to detect and prevent intrusions, as well as their benefits and drawbacks.</p>	<p>Out of the various different intrusion system tools few tools only support a limited range of security threats and issues. Since rules were not configured properly the detection rate became low. Whereas with the very minimum of hardware and sensor support, several tools were still having problems detecting accurate invaders.</p>

6) MF Kabir, S Hartmann - 2018. Cyber security challenges: An efficient intrusion detection system design. 2018 International Young Engineers Forum (YEF-ECE)

Summary-

This paper highlights how combining the concept IDS and a data mining technique indexing will improve the accuracy of the intrusion detection in real time. This will solve the detection efficiency problem such as real time detection rate, false positives etc in distributed environments.

Drawbacks-

Although combining the use of SNORT and data mining is an effective tool to reduce the problems with NIDs based systems, there isn't enough data collected to make this feasible.

7) Gaddam, R., & Nandhini, M. (2017, March). An analysis of various snort based techniques to detect and prevent intrusions in networks proposal with code refactoring snort tool in Kali Linux environment. In 2017 International Conference on Inventive Communication and Computational Technologies (ICICCT) (pp. 10-15). IEEE.

Summary-

After becoming familiar with IDS and its classification, the author explores numerous snort-based intrusion detection strategies for maintaining an organization's security against various assaults. In high-speed and cloud environments, Snort-based IDPS using efficient rules such as Bayesian Network and Honeypot-like approaches can protect from simple intrusions to dangerous DoS and DDoS type attacks. This paper proposes an architecture to enhance the efficiency of Snort IDS.

Drawbacks-

Although snort based IDS can protect the system from simple to complex attacks in high speed and cloud environments still it has few drawbacks such as improper detection rate with high number of false alarms. So there could be other ways to enhance the security of snort based IDS.

8) Bul'ajoul, W., James, A., & Shaikh, S. (2019). A new architecture for network intrusion detection and prevention. *IEEE access*, 7, 18558-18573.

Summary-

In this paper the author came up with a novel approach of deployment of a NIDPS architecture which was designed, evaluated as well as implemented. In the context of high-speed and volume attacks, the author also examined the performance of an open source NIDPS. The test's objective was to see how well the NIDPS performed in high-speed traffic when limited by off-the-shelf hardware, and then to figure out how to enhance it.

Drawbacks-

Signature-based NIDPS can identify known attacks, but the main drawback is that each signature must be stored in a database and compared to incoming packets. New signatures appear on a regular basis, and keeping track of them is a challenge. Another issue is the amount of time it takes to check all signatures. Knowledge sharing could be the answer.

9) F Erlacher, F Dressler - 2018 FIXIDS: A high-speed signature-based flow intrusion detection system. NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium

Summary-

FIXIDS makes use of HTTP intrusion signatures from the widely used Snort NIDS and applies them to incoming IPFIX Flows. In the experimental evaluation, the authors try to show a performance gain of a factor of three compared to Snort while maintaining the same detection ratio.

Drawbacks- No drawbacks observed

10) de la Cruz, J. E. C., Goyzueta, C. A. R., & Cahuana, C. D. (2020, September). Intrusion Detection and Prevention System for Production Supervision in Small Businesses Based on Raspberry Pi and Snort. In 2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON) (pp. 1-4). IEEE.

Summary-

The author has explored various topics in this work of the IDS and IPS system such as how the average use of outgoing traffic from LAN network to WAN network is approximately 29Mbps in tests with five simultaneous clients in a teleworking collaborative environment. As well as function behavior of IDS and IPS as the IDS will detect and log malicious behavior, but the IPS will detect, log, and prevent it, with the option of blocking the source IP address, destination IP address, or both, depending on the administrator's setup.

Drawbacks-

Business snort refers to the use of the snort ruleset for businesses, non-profit organizations, colleges and universities, government agencies, consultancies, etc. where snort sensors are in use in a production or lab environment. This is the case for big business so is considered in case Personal snort ruleset is not enough.

11) Hussein M. Elshafie; Tarek M. Mahmoud; Abdelmgeid A. Ali (2019) Improving the Performance of the Snort Intrusion Detection Using Clonal Selection

Summary-

The author has explored various topics in this work of Snort NIDS using clonal selection algorithm (CSA). The proposed approach is evaluated using the 1999 DARPA Intrusion Detection Evaluation Data Sets of MIT (Massachusetts Institute of Technology) as a testbed. The conducted experiments compare the recall, precision, and F-score of Snort NIDS on its own, Snort NIDS improved by negative selection algorithm (NSA), and the proposed approach. The obtained results show that the proposed approach is more powerful than the others.

12) Jain, G. (2021, March). Application of SNORT and Wireshark in network traffic analysis. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1119, No. 1, p. 012007). IOP Publishing.

Summary-

The attributes of Snort, which is used for intrusion detection by using predefined rules and alerting the user by directing alert messages, are discussed in this paper. The SNORT programme generates a log file including data packets and alert messages. The captured data packets are then examined using Wireshark, which is exported from the recorded log file.

Drawbacks-

No such drawbacks found as the papers discusses about the use of two different technology snort and wireshark

13) Hassan, Z., Odarchenko, R., Gnatyuk, S., Zaman, A., & Shah, M. (2018, October). Detection of distributed denial of service attacks using snort rules in cloud computing & remote control systems. In *2018 IEEE 5th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC)* (pp. 283-288). IEEE.

Summary-

The proposed model by the author describes cloud computing and its various service models as well as security issues in cloud computing. Cloud computing users trust third parties, whose other major cloud security issue is computing. A DDoS attack is an attack that floods the server with a large number of unnecessary packets, making them unavailable to legitimate users. Thesis launched DDoS attacks and discussed tools to launch DDoS attacks.

Drawbacks-

There are certain shortcomings of the snort tool. So in order to avoid any issues in further development a newer and updated version of snort which is USNORT can be used which detects intrusions according to the rules identified by SNORT.

14) Guezzaz, A., Asimi, A., Asimi, Y., Tbatou, Z., & Sadqi, Y. (2019). A Global Intrusion Detection System using PcapSockS Sniffer and Multilayer Perceptron Classifier. *Int. J. Netw. Secur.*, 21(3), 438-450.

Summary-

In the paper the authors demonstrate that they will be carrying out the evaluation of the actions of a list of hue and intrusion detection tools, and at the end of these tools will be susceptible to vulnerabilities. They also propose an optimal approach of intrusion detection based on multilayer perceptron technique aiming to improve the accuracy of detection

Drawbacks-

In the future work the authors explain that they would be working in detail implementation and validation of various steps of this global system describing the proposed solutions.

15) Resmi, A. M., & Manicka, R. (2017). Intrusion Detection System Techniques and Tools: A Survey. *Scholars Journal of Engineering and Technology (SJET)*.

Summary-

The author of this work did a survey on the entire study of IDS, its nature and methodology, as well as technologies utilised in the intrusion detection field. The survey provides an overview of the tools used to detect and prevent intrusions, as well as their benefits and drawbacks. The IDS types are network based, host based intrusion detection system, hybrid IDS, net-host based, anomaly based IDS, and misuse intrusion detection systems.

Drawbacks-

Out of the various different intrusion system tools few tools only support a limited range of security threats and issues. Since rules were not configured properly the detection rate became low. Whereas with the very minimum of hardware and sensor support, several tools were still having problems detecting accurate invaders.

16) Visoottiviseth, V., Chutaporn, G., Kungvanruttana, S., & Paisarnduangjan, J. (2020, October). PITI: Protecting Internet of Things via Intrusion Detection System on Raspberry Pi. In *2020 International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 75-80). IEEE.

Summary-

The paper suggests performing hybrid detection by combining Snort IDS and behavior-based profile analysis together in order to efficiently detect malicious and suspicious traffic.

Drawbacks-

This paper highlights a very basic architecture and it should be developed into an Intrusion Prevention System (IPS) which can drop the suspicious packets and detect more types of attacks.

17) HM Elshafie, TM Mahmoud - 2019. Improving the Performance of the Snort Intrusion Detection Using Clonal Selection. 2019 International Conference on Innovative Trends in Computer Engineering (ITCE)

Summary-

The paper suggests performing hybrid detection by combining Snort IDS and behavior-based profile analysis together in order to efficiently detect malicious and suspicious traffic.

Drawbacks- No drawbacks observed

18) SM Kattamuri, V Kakulapati - 2018. Performance Analysis of Mail Clients on Low Cost Computer With ELGamal and RSA Using SNORT

Summary-

The authors applied encryption for text files by using cryptographic algorithms like Elgamal and RSA.

Drawbacks-

Within a low cost, low power computer, they observed that as the size of the file increases, the run time is constant for compressed data; whereas in plain text, it changes significantly.

19) Baykara, M., & Das, R. (2018). A novel honeypot based security approach for real-time intrusion detection and prevention systems. *Journal of Information Security and Applications*, 41, 103-116.

Summary-

A honeypot primarily based totally method is proposed, which may be used at the community protection for the real-time intrusion detection and prevention systems. For this proposed novel method, a powerful software program device became advanced. The advanced machine is a hybrid honeypot that mixes the advanced residences of low and excessive interplay honeypots in an unmarried structure.

Drawbacks-

Certain limitations while working on the multi thread design with animations were also discussed in context of response time of the application that becomes short in performance while loading tests. Developed honeypot-based IDPS can receive, drop, analyze and filter the network traffic using SNORT. Hence, the limitation of the developed system depends on SNORT limitation

20) Intrusion Detection and Prevention Systems: An Updated Review. Nureni Ayofe Azeez Taiwo Mayowa Bada, Sanjay Misra, Adewole Adewumi, Charles Van der Vyver, Ravin Ahuja

Summary-

Given that the most current review on the issue was published in 2016, this paper provides an updated review of IDPSs. It will also cover the usage of IDPSs to discover

vulnerabilities in various channels via which data on a network or system is accessed, as well as the preventative techniques used to prevent infiltration.

Drawbacks-

No such drawbacks observed

21) Liang, W., Li, K. C., Long, J., Kui, X., & Zomaya, A. Y. (2019). An industrial network intrusion detection algorithm based on multifeature data clustering optimization model. *IEEE Transactions on Industrial Informatics*, 16(3), 2063-2071.

Summary-

The proposed algorithm can effectively improve the detection rate and real-time performance of detecting abnormal behavior for the multifeature data in industrial networks. The novel features are twofold, to rapidly select a node with high-security coefficient as the cluster center, and match the multifeature data around the center into a cluster. The results shows that the proposed algorithm has good superiority in terms of detection rate and time compared to other algorithms.

Drawbacks-

In the industrial network, the detection accuracy of abnormal data reaches 97.8%, and the FP of detection is decreased by 8.8%.

22) RT Gaddam, M Nandhini - 2017 An analysis of various snort based techniques to detect and prevent intrusions in networks proposal with code refactoring snort tool in Kali Linux environment

Summary-

This paper briefs various trends in Intrusion Detection & Prevention. To understand various techniques in IDS, this paper analyses various approaches proposed by security researchers specifically using popular open source software Snort as their IDS tool. Being an open source IDS, Snort can be easily configured and deployed in any environment. To assess the efficiency, these research papers are analyzed in various performance aspects like Detection Accuracy, Scalability and Capability of detecting unknown attacks.

Drawbacks-

To overcome various challenges like low detection rate, incapable of handling huge traffic, unsupported automated tuning, etc. that are identified during literature review, this paper proposes a level based architecture. All the levels are designed as incremental i.e. capable of providing the desired functionality and also its lower levels.

23)A categorical survey of state-of-the-art intrusion detection system-*Snort*

Summary-

This paper provides a categorical assessment of the many research strategies used to improve the performance of Snort, the open-source intrusion detection system that has become the de facto standard in the field of intrusion detection and prevention. All of the techniques' advantages and disadvantages have been discussed. In order to improve the performance of Snort-IDS in a high-speed network, a novel parallel design has been developed. The architecture is based on the network traffic's constituent protocol proportions.

24) Karim, QT Vien, TA Le, G Mapp - Computers, 2017 A Comparative Experimental Design and Performance Analysis of Snort-Based Intrusion Detection System in Practical Computer Networks

Summary-

This article examines the performance of Snort-based NIDS (S-NIDS) on a real-world network using cutting-edge technologies in a variety of network circumstances, including high data rates, heavy traffic, and huge packet sizes. An effective testbed is built using Snort and several multi-core CPUs, such as the i5 and i7, as well as various operating systems, such as Windows 7, Windows Server, and Linux.

25)A Gupta, LS Sharma - Proceedings of ICRIC 2019, 2020 - Springer Performance Evaluation of *Snort* and *Suricata* Intrusion Detection Systems on Ubuntu Server

Summary-

In this paper, *Snort* and *Suricata* are compared experimentally through a series of tests to identify more scalable and reliable *IDS* by putting the systems under high traffic. Results indicated that *Snort* had a lower system overhead than *Suricata* and utilized only one processor on a multi-core environment. However, *Suricata* evenly utilized all the processing elements of the multi-core environment and provided a higher packet analysis rate.

26) M. Firoz Kabir; Sven Hartmann 2018 Cyber security challenges: An

efficient intrusion detection system design

Summary-

This paper proposed a concept IDS to investigate the experimental performance of Snort based NIDS. We have used an open source network intrusion detection and prevention system Snort to implement our two different indexing methods. We used Snort version 2.9.7.5 which has almost 26k Snort rules and is very efficient for online network auditing. We implemented prefix and random indexing methods to all Snort rules to create primary patterns that reduce packet inspection time.

Drawbacks-

Since all highly sensitive positive alerts need instant action from network administrators, our concept IDS also reduces the false positive (wrong alert) rate even at high network traffic.

27) Indian Journal of Science and Technology, 2018 Snort-Based Smart and Swi Intrusion Detection System

Summary-

This study proposes a smart Intrusion Detection System (IDS) that identifies network threats in less time after monitoring incoming data, resulting in improved performance. Statistical Analysis/Methods: The back-propagation algorithm is used to extract the features. Then, using a multi-layer perceptron supervised neural network, only these important features are trained. MATLAB is used to run the simulation. Findings: The proposed system was found to have a high rate of accuracy, sensitivity, and a lower percentage of false positives.

Drawbacks-

Denial-of-Service (DoS), User-to-Root (U2R), Remote-to-Local (R2L), and Probe attacks are the four types of invasions that have been identified.

28) Singh, V. K., Ebrahim, H., & Govindarasu, M. (2018, September). Security evaluation of two intrusion detection systems in smart grid scada environment. In 2018 North American Power Symposium (NAPS) (pp. 1-6). IEEE.

Summary-

The authors have shown ways on how to use IDS tools like Snort and Bro to build an anomaly-based intrusion detection system (AbIDS) to identify generation-altering attacks on SCADA-based protection schemes, also known as corrective action schemes. Listening and filtering network traffic between the controller and the power system simulator is part of the proposed multistage IDS strategy.

Drawbacks-

For future studies, the authors intend to construct several rule sets based on system behavior in multi-dimensions to detect different types of attacks in future investigations.

29) PRajesh KannaM.E.(Assistant Professor) PSanthiM.E., Ph.D.(Professor) Unified Deep Learning approach for Efficient Intrusion Detection System using Integrated Spatial–Temporal Features

Summary-

Pre-processing, feature extraction via network training and network testing, and final classification are all performed by the proposed IDS model. The Lion Swarm Optimization (LSO) is utilised in the OCNN–HMLSTM model to optimise the hyper-parameters of CNN for the best configuration of learning spatial features. The HMLSTM extracts the time features after learning the hierarchical relationships between the various features. Finally, the unified IDS technique categorises network data using the extracted spatial–temporal properties. The tests are run on the NSL-KDD, ISCX-IDS, and UNSWNB15 public IDS datasets.

Drawbacks-

When comparing the performance of OCNN–HMLSTM to that of current IDS approaches, the proposed model outperforms them all with a high accuracy of over 90%, fewer false values, and superior classification coefficients.

30) Z.K. Baker; V.K. Prasanna A methodology for synthesis of efficient intrusion detection systems on FPGAs

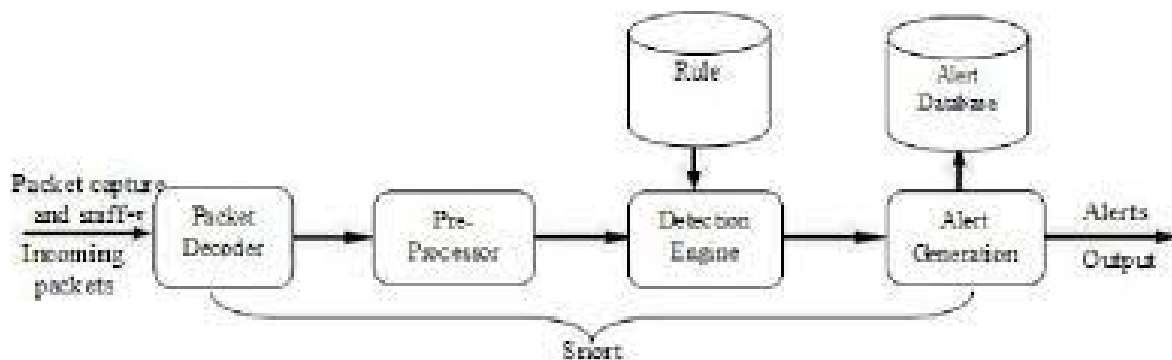
Summary-

This paper presents a methodology for system-wide integration of graph-based partitioning of large intrusion detection pattern databases. Integrating ruleset-based graph creation and min-cut partitioning, our methodology allows efficient multi-byte comparisons and partial matches for high performance FPGA-based network security.

Drawbacks-

Through pre-processing, this methodology yields designs with competitive clock frequencies that are a minimum of 8x more area efficient than previous non-pre decoded shift-and-compare architectures.

High level Architecture Diagram:



Packet Decoder: Decodes the specific protocol elements for each packet.

Pre-Processor: Examines packets for suspicious activity and modifies them so that the detection engine can properly interpret them. This checks the packet against set plug-ins eg: http plug-in, plug-ins can be enabled or disabled on need of basis.

Detection Engine: This builds attack signatures by parsing Snort rules.

Alert Generation: Generates a series of alerts that are sent if the packets do not adhere to any set of active rules.

Rule Database: Stores the set of SNORT rules that are used by the detection engine.

Alert Database: Creates a log of all alerts generated while running.

Goals:

1. Define the various terminologies, techniques and practical methodologies related to SNORT Intrusion Detection and Prevention System (IDPS).
2. How to set up and install SNORT.

Specifications:

Martin Roesch, the creator and former CTO of Sourcefire, designed Snort in 1998 as a network intrusion detection and prevention solution that is free and open source. Cisco, which bought Sourcefire in 2013, now develops Snort.

IDS/IPS:

An intrusion is defined as illegal entry to someone's property or area in general, but in computer science, it is defined as an act that compromises the essential computer network security goals of confidentiality, integrity, and privacy. Intrusion detection is the process of continuously monitoring and analysing events in a computer system or network for signals of potential threats and violations of computer security practises, acceptable usage regulations, or standard security policies. The existence of infiltration in the network is detected by the Intrusion Detection System (IDS). It's made to keep track of what's going on in a computer system or network, and to respond to events that show symptoms of prospective security policy violations. Intrusion Prevention System (IPS), on the other hand, is a network security system or technology that can identify intrusion activities as well as take the necessary countermeasures to prevent them.

Technologies Types :

IDPS technologies come in a variety of shapes and sizes. They are classified into the four groups below for the purposes of this text, based on the kind of events they monitor and how they are deployed.

Network based IDPS

An intrusion detection system that monitors and evaluates network traffic for any unusual behaviour or threats in the network is known as a network-based intrusion detection system. It scans the packets passing over the network and looks for strange patterns in the packets to look for malicious activities. If a threat is found, the system will take action based on the threat, such as contacting administrators.

Wireless based IDPS

Wireless Intrusion Detection and Prevention System analyses wireless network traffic by examining wireless protocol behaviours and takes relevant measures. It identifies the presence of unauthorised wireless local area networks. It is unable to detect suspicious activity at the application, transport, and protocol layers. It's set up in a specific range so that the company can keep an eye on the wireless network.

Network Behavior Analysis

NBA analyses network traffic to identify threats like as DDoS (Distributed Denial of Service) attacks, malware (e.g. worms, backdoors), and policy violations that cause irregular traffic patterns. These systems are used to monitor the flow of data on an organization's internal network, but they can also be used to monitor internal and external networks.

Host based IDPS

HIDPS analyses a single host's characteristics and detects intrusions by looking at the file system, file access, system calls, and network events. It is capable of preventing system-level attacks and detecting attacks that NIDPS is unable to detect. The burden on the hosts can be dispersed across the network. It can even examine actions transmitted in encrypted end-to-end connections.

Methodologies of IDPS:

IDPS employs a variety of approaches to identify and prevent any infiltration. These approaches are utilised in accordance with the system's requirements.

Anomaly based Methodology

These forms of attacks are used to detect unknown attacks, or activity that has never been seen before. This methodology does not necessitate the creation of any rules. Based on regular network traffic patterns, it detects malicious activity. The disadvantage of this technology is that it has a high rate of false alarms.

Signature based Methodology

This methodology is used to detect unknown threats that have already been pre-defined as signatures and preserved. When data is delivered over the network, it is first forwarded to a server, which scans it for harmful material. It compares the network packet to the signature database that is already stored in the network, discarding any packets that match, and sending any packets that do not match to the network. It is beneficial to organisations that are concerned about known threats.

Stateful Protocol Analysis

It compares observed behaviour to established profiles of how protocols should act. It checks the information about the connections between the hosts to the entries in the state table. The state table keeps track of computer connections, including the port and source IP address, port and the destination IP address, and the protocol used. The system's key benefit is that it can identify attacks from within a network.

Hybrid based

It's the combination of two intrusion detection systems — anomaly and signature-based detectors – or a mixture of any two approaches. It compiles the results of anomaly and signature-based detectors before calculating the attack likelihood. This method changes the normal network model of the anomaly detector as well as the rule set of signature-based detectors. It uses the anomaly and signature-based detectors' collected outputs to make a final conclusion on the possibility of an attack.

Attack on IDS

There are many different types of attacks which can corrupt a system. Those attacks can be generally grouped into the following categories:

Confidentiality: allows the attacker to gain access to information without authorization.

Integrity: allows the unauthorized attacker to affect the state of the system. This could mean either affecting the system state or any data residing on or passing through the system.

Availability: principle of availability is violated if the attacker can prevent an authorized user from accessing a system resource.

Control: attack grants an unauthorized attacker a privilege in violation of the access control policy of the system, the attack is on the control principle. This attack can give means to further attacks on confidentiality, integrity, and/or availability.

Scanning Attack

Scanning attacks can be used to gather data about the system under attack. The attacker can obtain topological information, types of network traffic allowed through a firewall, active hosts on a network, OS and kernel of hosts on a network, server software operating, version numbers of software, and so on, using scanning techniques. The attacker could use this knowledge to conduct assaults directed at more specific exploits. The information above was collected via a covert SYN scan. Because it never completes TCP connections, this scan is dubbed stealth. Because the attacker does not initiate a full TCP connection, this approach is known as half open scanning. As if you were establishing a legitimate TCP connection, the attacker sends a SYN message. If the attacker receives a SYN/ACK, the port is open for assault. If there is no response, the attacker may conclude that the port is closed. This is just one type of scan technique; there are a plethora of others.

Denial of Service Attack

Flooding and vulnerability exploitation are the two most common types of denial of service (DoS) attacks. Flooding assaults are frequently fairly simple to execute. For example, a DoS attack can be launched with the ping command: ping -f victim. As a result, the victim will receive an enormous quantity of ping packets. If the attacker has more bandwidth than the victim, he or she will be readily and rapidly overwhelmed. A SYN flood attack, for example, sends a flood of TCP/SYN packets to a victim with a faked source address. The victim will open half-open TCP connections as a result of this - the victim will

send a TCP SYN/ACK packet and wait for an ACK response. The victim will gradually exhaust available resources waiting for ACKs from a non-existent host because the ACK never arrives.

Penetration Attack

All assaults that allow an unauthorised attacker access to system resources, privileges, or data are classified as penetration attacks. Exploiting a software defect is a frequent way for this to happen. In July of 2002, for example, an exploit was discovered in the ssh challenge answer handling code that let an attacker to run arbitrary code as the person running sshd (often root). This would be classified as a penetration attack. The ability to run code as root provides an attacker access to any system resource imaginable. Furthermore, the user may be able to conduct additional forms of attacks on this system, as well as attack other systems from the compromised system.

Computer Vulnerabilities

There are different types of errors that arise in a system by which an attacker can gain access to the system.

Input Validation Error

The two most common forms are buffer overflows and boundary condition errors. These both occur when a program does not properly check the input it receives from the system. A buffer overflow attack exploits the boundaries of some buffer, resulting in some data overwriting adjacent memory locations. A boundary condition error occurs when input to a program causes the program to exceed some boundary. For instance, the input may cause the system to run out of memory.

Access Validation Error

This occurs when the access control policy of the system is flawed. As a result, the attacker can utilize this to gain control of the system. Exceptional Condition Handling Error: system becomes vulnerable because some type of exception has arisen. This exception could either not be caught, or handled incorrectly, thus allowing the attacker to exploit the system.

Configuration Error

This error results as a fault of the end administrator who is responsible for configuring the system. This is not a fault of how the system was designed, but a fault of the user who incorrectly configured the system.

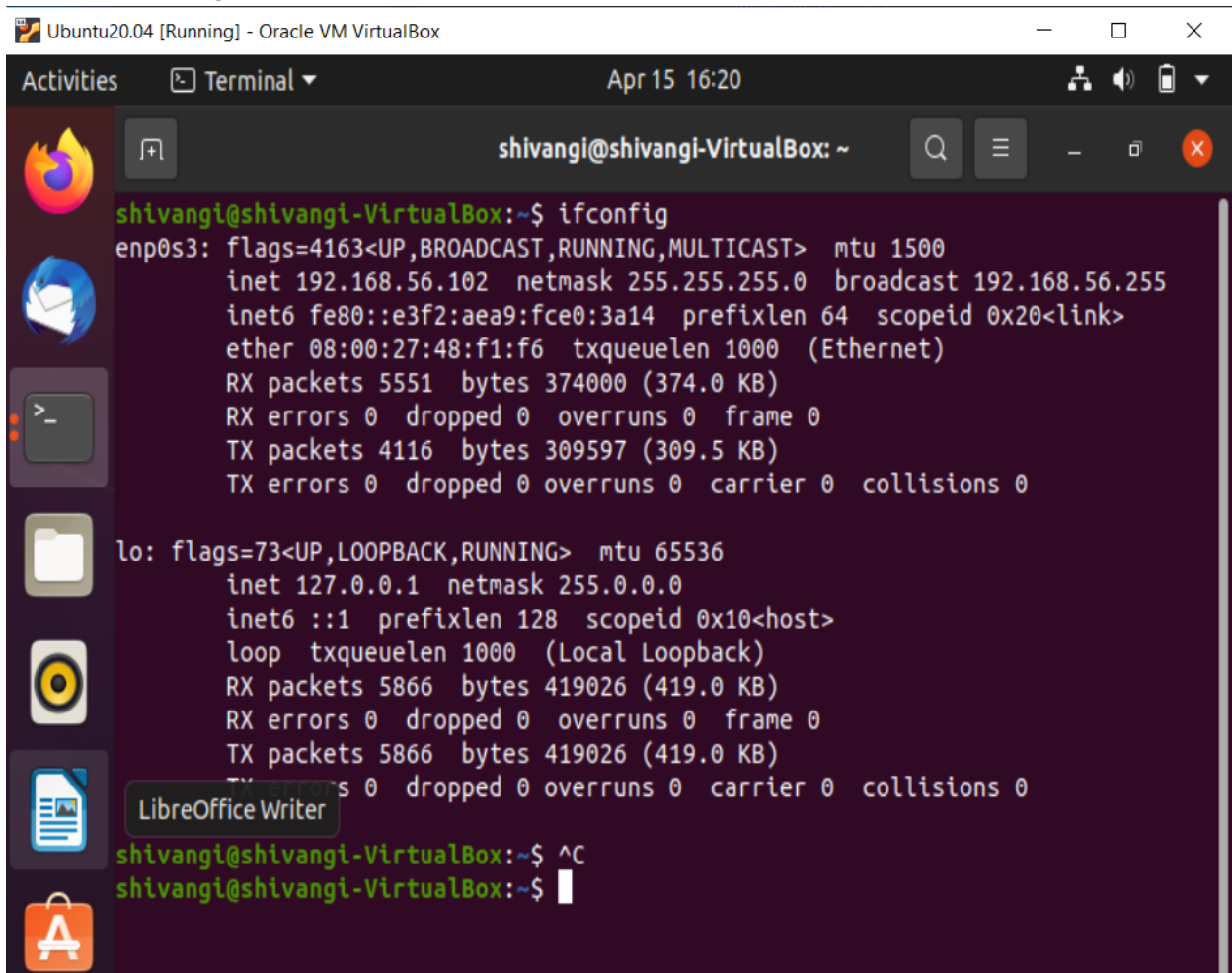
Race Condition

A flaw in a system where the output exhibits unexpected dependence on the timing of events. Attackers can exploit race conditions with respect to denial of service attacks. Also, attackers can take advantage of

programs which reach race conditions while stuck in a privileged state. While in this privileged state, the attacker could convince systems to perform illegal operations.

How to Install SNORT:

I. ifconfig command on Ubuntu:



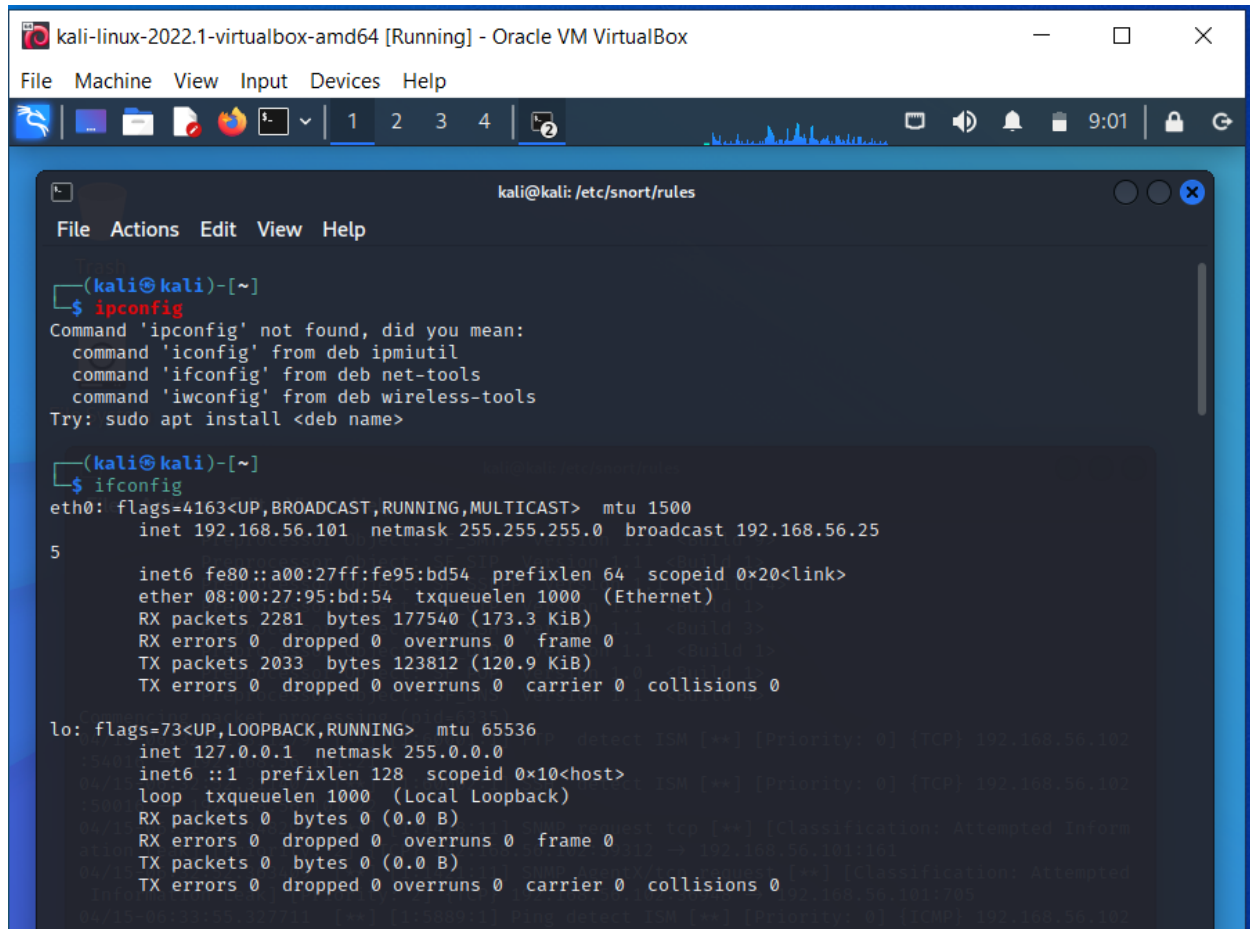
The screenshot shows a terminal window titled "Ubuntu20.04 [Running] - Oracle VM VirtualBox". The terminal prompt is "shivangi@shivangi-VirtualBox: ~". The user has entered the command "ifconfig". The output shows details for the "enp0s3" network interface and the "lo" loopback interface. The "enp0s3" interface has an IP address of 192.168.56.102 and a netmask of 255.255.255.0. The "lo" interface has an IP address of 127.0.0.1 and a netmask of 255.0.0.0. The terminal also shows the status of the interfaces, including flags, mtu, and statistics.

```
shivangi@shivangi-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.102 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::e3f2:aea9:fce0:3a14 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:48:f1:f6 txqueuelen 1000 (Ethernet)
    RX packets 5551 bytes 374000 (374.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4116 bytes 309597 (309.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 5866 bytes 419026 (419.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5866 bytes 419026 (419.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

shivangi@shivangi-VirtualBox:~$ ^C
shivangi@shivangi-VirtualBox:~$
```

II. ifconfig command on kali linux:



The screenshot shows a Kali Linux terminal window titled "kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox". The terminal displays the output of the `ifconfig` command. It shows details for the `eth0` interface, including its flags, IP address (192.168.56.101), netmask (255.255.255.0), and broadcast address (192.168.56.255). It also shows statistics for RX and TX packets, bytes, errors, and collisions. The `lo` (loopback) interface is also shown with its IP address (127.0.0.1) and netmask (255.0.0.0).

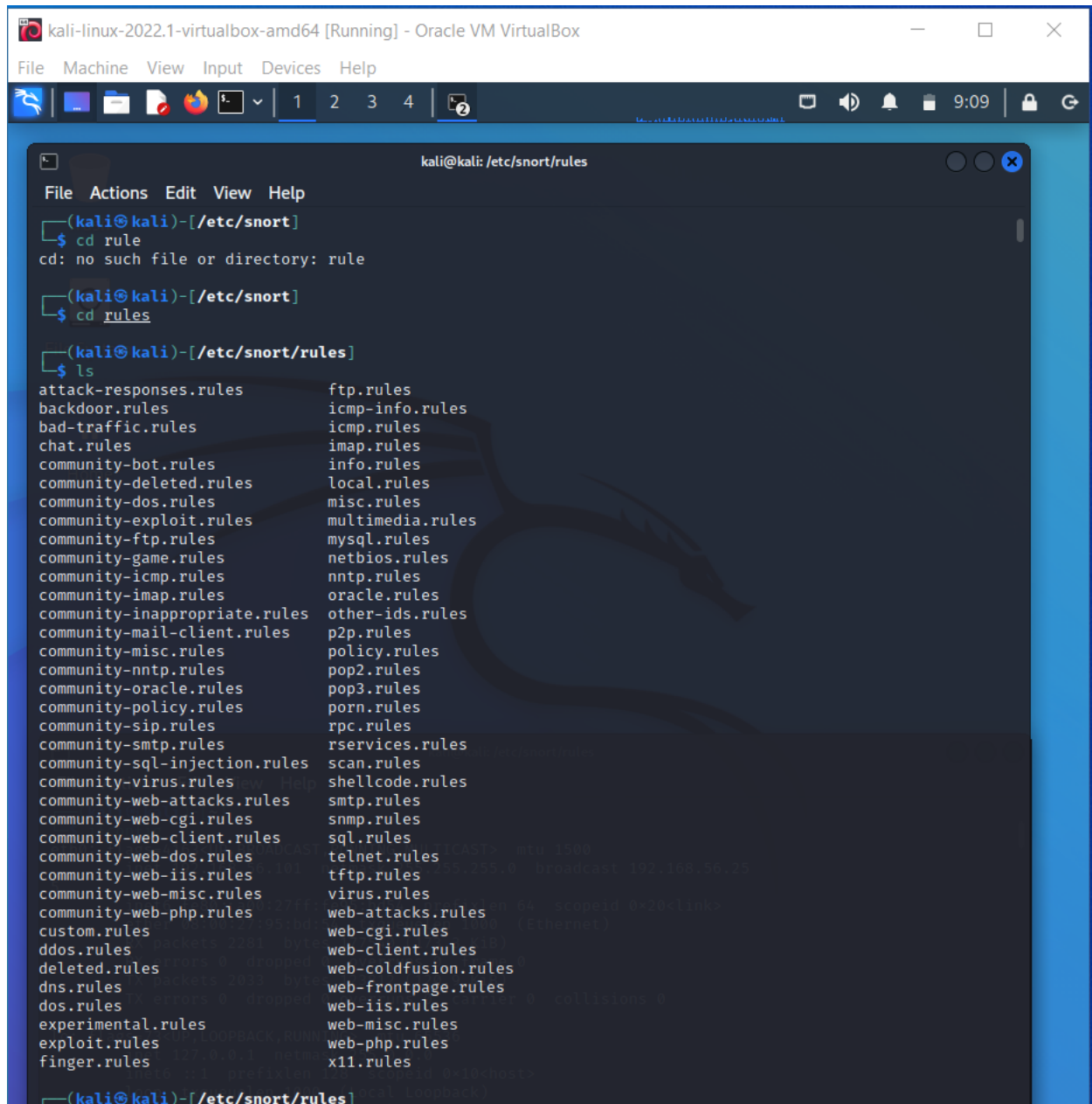
```
kali@kali: /etc/snort/rules
File Actions Edit View Help

(kali@kali)-[~]
$ ipconfig
Command 'ipconfig' not found, did you mean:
  command 'iconfig' from deb ipmiutil
  command 'ifconfig' from deb net-tools
  command 'iwconfig' from deb wireless-tools
Try: sudo apt install <deb name>

(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.56.101  netmask 255.255.255.0  broadcast 192.168.56.255
    ether 08:00:27:95:bd:54  txqueuelen 1000  (Ethernet)
    RX packets 2281  bytes 177540 (173.3 KiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 2033  bytes 123812 (120.9 KiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 0  bytes 0 (0.0 B)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 0  bytes 0 (0.0 B)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```


III. Create some required rules



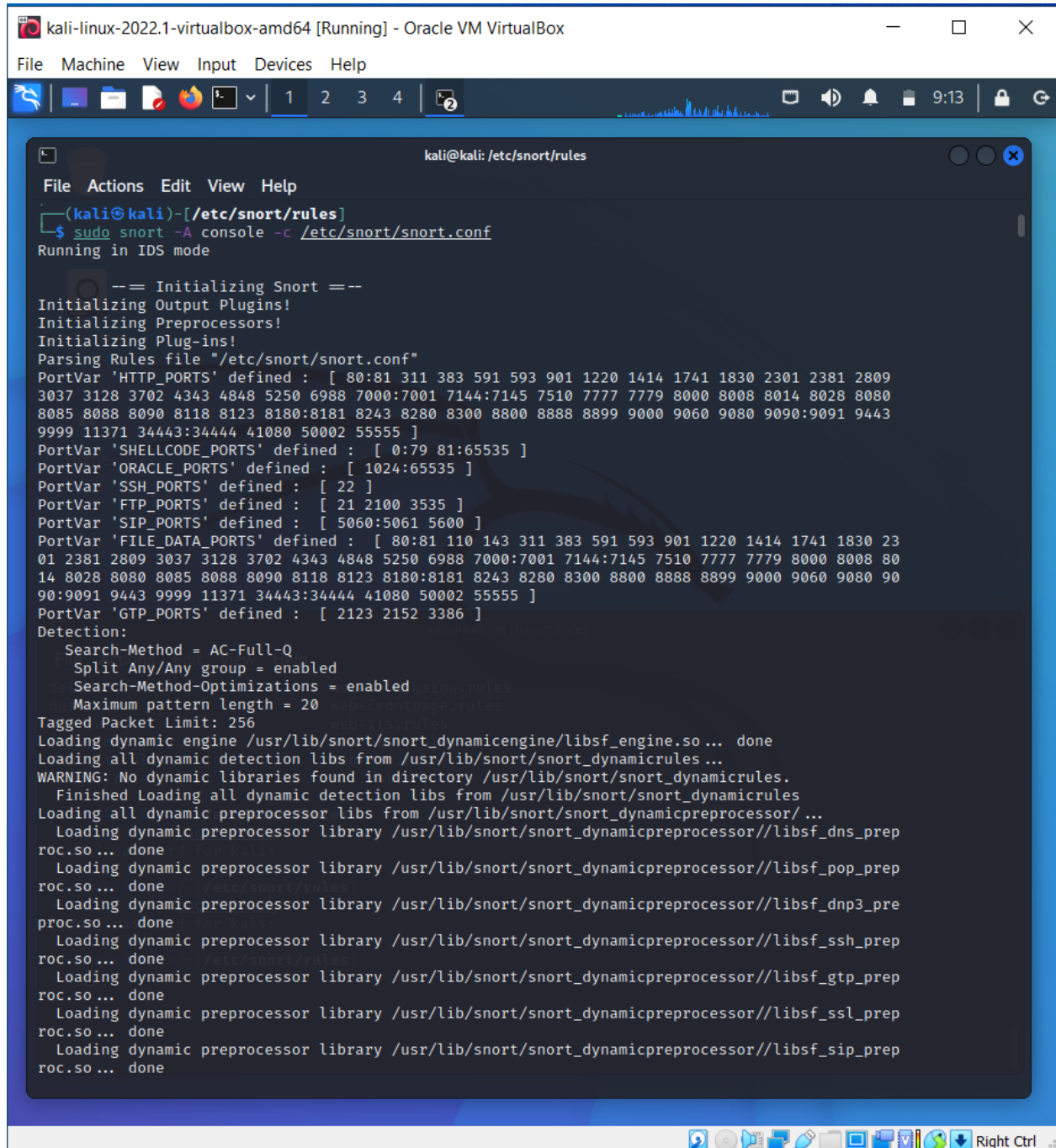
The screenshot shows a Kali Linux virtual machine window titled "kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox". The terminal window is open at the prompt "(kali@kali)-[/etc/snort]". The user enters the command "cd rule", which results in an error: "cd: no such file or directory: rule". The user then enters "cd rules", and the prompt changes to "(kali@kali)-[/etc/snort/rules]". Finally, the user enters "ls", which lists the contents of the directory. The list is as follows:

```
(kali@kali)-[/etc/snort/rules]
$ ls
attack-responses.rules      ftp.rules
backdoor.rules             icmp-info.rules
bad-traffic.rules          icmp.rules
chat.rules                 imap.rules
community-bot.rules        info.rules
community-deleted.rules    local.rules
community-dos.rules        misc.rules
community-exploit.rules    multimedia.rules
community-ftp.rules        mysql.rules
community-game.rules       netbios.rules
community-icmp.rules       nntp.rules
community-imap.rules       oracle.rules
community-inappropriate.rules other-ids.rules
community-mail-client.rules p2p.rules
community-misc.rules       policy.rules
community-nntp.rules       pop2.rules
community-oracle.rules     pop3.rules
community-policy.rules     porn.rules
community-sip.rules        rpc.rules
community-smtp.rules       rservices.rules
community-sql-injection.rules scan.rules
community-virus.rules      shellcode.rules
community-web-attacks.rules smtp.rules
community-web-cgi.rules    snmp.rules
community-web-client.rules sql.rules
community-web-dos.rules    telnet.rules
community-web-iis.rules    tftp.rules
community-web-misc.rules   virus.rules
community-web-php.rules    web-attacks.rules
custom.rules              web-cgi.rules
ddos.rules                web-client.rules
deleted.rules              web-coldfusion.rules
dns.rules                  web-frontpage.rules
dos.rules                  web-iis.rules
experimental.rules        web-misc.rules
exploit.rules              web-php.rules
finger.rules               x11.rules
```

Exploring the local.rules file and adding our custom rules to detect attack on kali:

[illegible]

Running the snort in IDS mode:

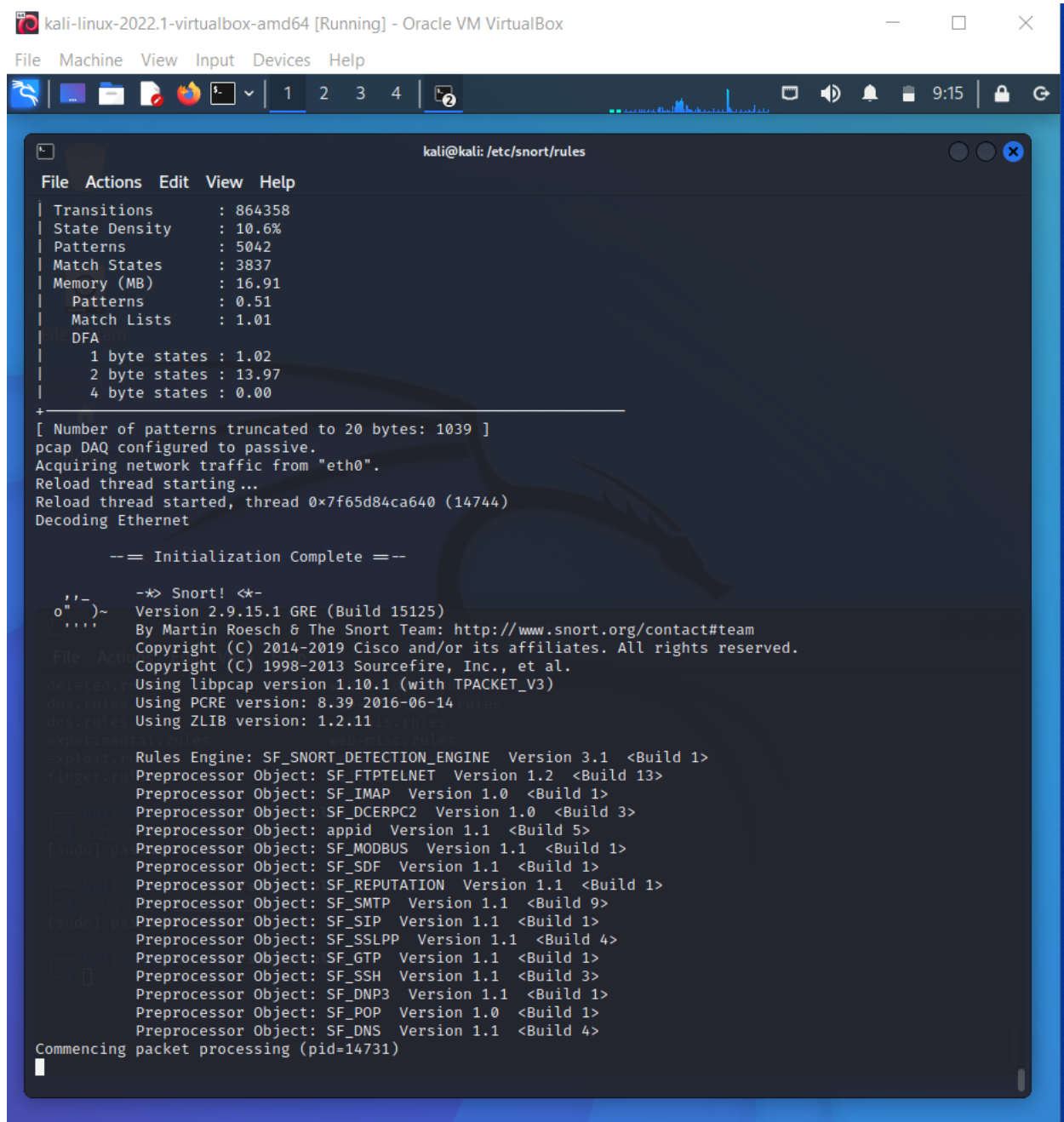


The screenshot shows a Kali Linux virtual machine window titled "kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox". The terminal window is titled "kali@kali: /etc/snort/rules" and shows the command `sudo snort -A console -c /etc/snort/snort.conf` being executed. The output indicates that Snort is running in IDS mode and is initializing various components, including output plugins, preprocessors, and rule sets. The terminal output is as follows:

```
kali@kali: /etc/snort/rules
(kali@kali)-[/etc/snort/rules]
$ sudo snort -A console -c /etc/snort/snort.conf
Running in IDS mode

--= Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined : [ 80:81 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809
3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7779 8000 8008 8014 8028 8080
8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443
9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'SHELLCODE_PORTS' defined : [ 0:79 81:65535 ]
PortVar 'ORACLE_PORTS' defined : [ 1024:65535 ]
PortVar 'SSH_PORTS' defined : [ 22 ]
PortVar 'FTP_PORTS' defined : [ 21 2100 3535 ]
PortVar 'SIP_PORTS' defined : [ 5060:5061 5600 ]
PortVar 'FILE_DATA_PORTS' defined : [ 80:81 110 143 311 383 591 593 901 1220 1414 1741 1830 23
01 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7779 8000 8008 80
14 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 90
90:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'GTP_PORTS' defined : [ 2123 2152 3386 ]
Detection:
  Search-Method = AC-Full-Q
  Split Any/Any group = enabled
  Search-Method-Optimizations = enabled
  Maximum pattern length = 20
Tagged Packet Limit: 256
Loading dynamic engine /usr/lib/snort/snort_dynamicengine/libsengine.so... done
Loading all dynamic detection libs from /usr/lib/snort/snort_dynamicrules...
WARNING: No dynamic libraries found in directory /usr/lib/snort/snort_dynamicrules.
Finished Loading all dynamic detection libs from /usr/lib/snort/snort_dynamicrules
Loading all dynamic preprocessor libs from /usr/lib/snort/snort_dynamicpreprocessor/...
Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libs_dns_pre
roc.so... done
Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libs_pop_prep
roc.so... done
Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libs_dnp3_pre
proc.so... done
Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libs_ssh_prep
roc.so... done
Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libs_gtp_prep
roc.so... done
Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libs_ssl_prep
roc.so... done
Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libs_sip_prep
roc.so... done
```

Before running of nmap attack from ubuntu side:



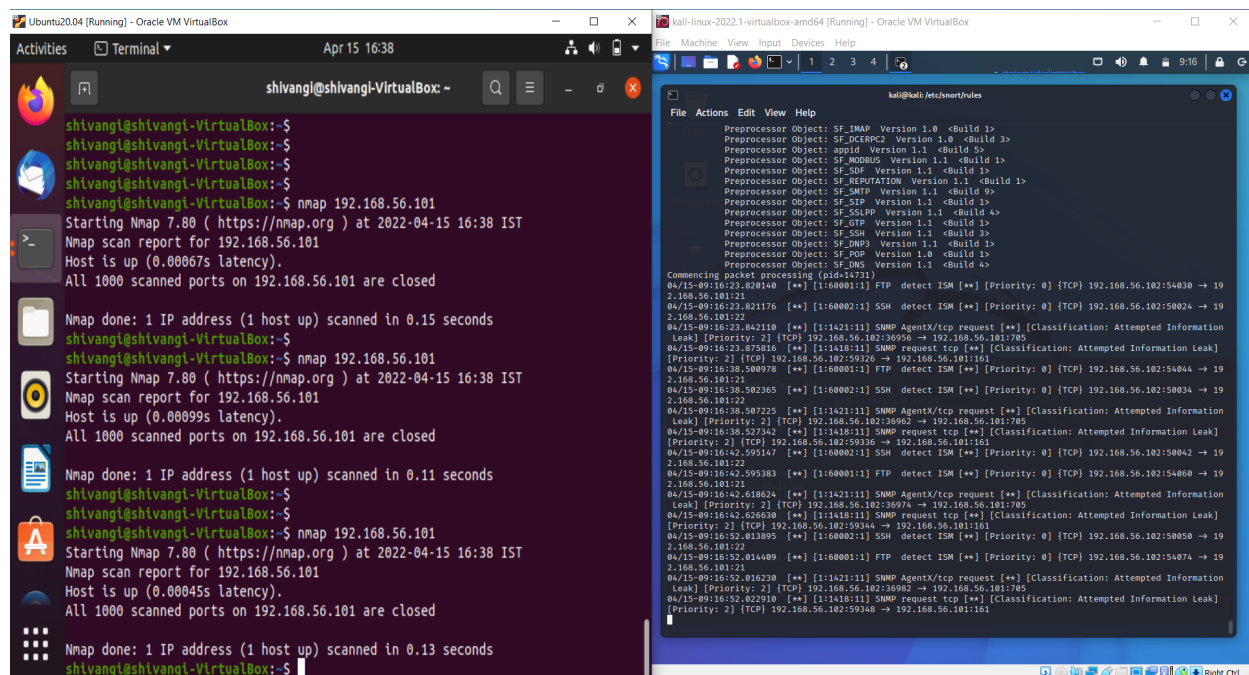
```
kali@kali: /etc/snort/rules

File Actions Edit View Help
| Transitions      : 864358
| State Density    : 10.6%
| Patterns         : 5042
| Match States     : 3837
| Memory (MB)      : 16.91
| Patterns         : 0.51
| Match Lists      : 1.01
| DFA
|   1 byte states  : 1.02
|   2 byte states  : 13.97
|   4 byte states  : 0.00
+-----+
[ Number of patterns truncated to 20 bytes: 1039 ]
pcap DAQ configured to passive.
Acquiring network traffic from "eth0".
Reload thread starting...
Reload thread started, thread 0x7f65d84ca640 (14744)
Decoding Ethernet

--= Initialization Complete ==--

--*- Snort! <*-
o" )~ Version 2.9.15.1 GRE (Build 15125)
' ' By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
File Act Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.
deleted Copyright (C) 1998-2013 Sourcefire, Inc., et al.
and rules Using libpcap version 1.10.1 (with TPACKET_V3)
des rules Using PCRE version: 8.39 2016-06-14
expirator Using ZLIB version: 1.2.11
exploit Rules Engine: SF_SNORT_DETECTION_ENGINE Version 3.1 <Build 1>
finger Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Preprocessor Object: appid Version 1.1 <Build 5>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Commencing packet processing (pid=14731)
```

After nmap command:

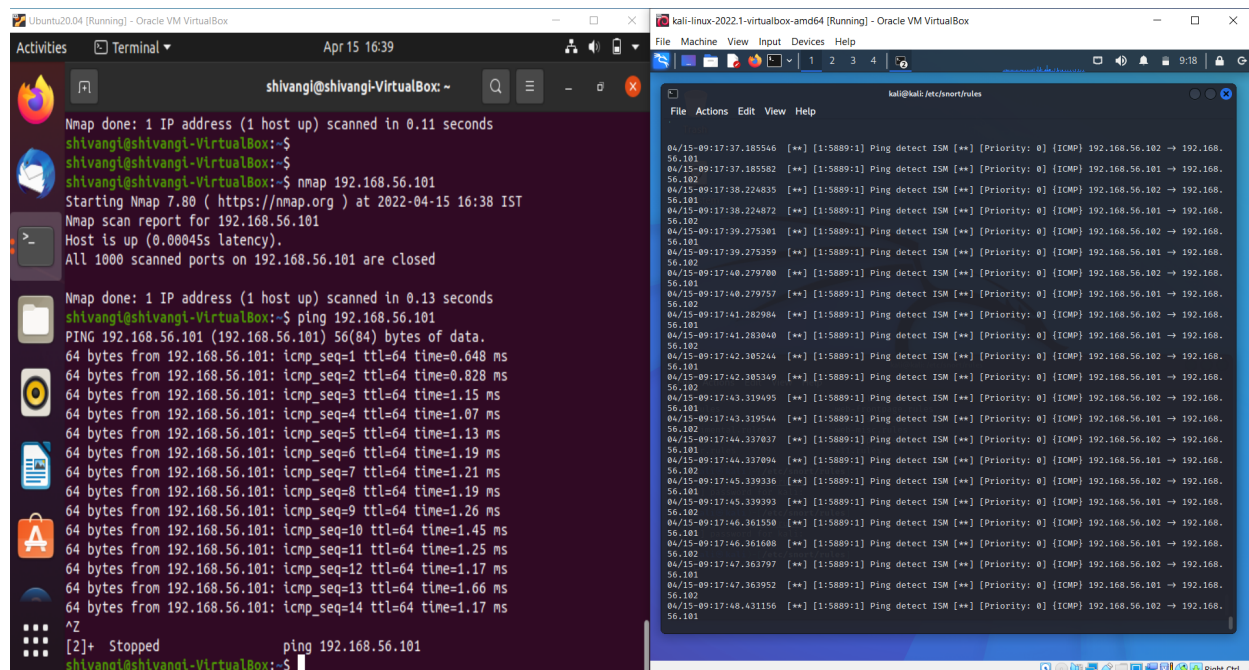


The image shows two terminal windows side-by-side. The left window is titled 'shivangl@shivangl-VirtualBox: ~' and shows the output of an nmap scan on 192.168.56.101. The right window is titled 'kali@kali-jetchnoorthules' and shows a list of installed packages.

```
shivangl@shivangl-VirtualBox: ~  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$ nmap 192.168.56.101  
Starting Nmap 7.80 ( https://nmap.org ) at 2022-04-15 16:38 IST  
Nmap scan report for 192.168.56.101  
Host is up (0.00067s latency).  
All 1000 scanned ports on 192.168.56.101 are closed  
  
Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$ nmap 192.168.56.101  
Starting Nmap 7.80 ( https://nmap.org ) at 2022-04-15 16:38 IST  
Nmap scan report for 192.168.56.101  
Host is up (0.00099s latency).  
All 1000 scanned ports on 192.168.56.101 are closed  
  
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$ nmap 192.168.56.101  
Starting Nmap 7.80 ( https://nmap.org ) at 2022-04-15 16:38 IST  
Nmap scan report for 192.168.56.101  
Host is up (0.00045s latency).  
All 1000 scanned ports on 192.168.56.101 are closed  
  
Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds  
shivangl@shivangl-VirtualBox:~$
```

```
kali@kali-jetchnoorthules  
File Actions Edit View Help  
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>  
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>  
Preprocessor Object: apid Version 1.1 <Build 5>  
Preprocessor Object: SF_MODULES Version 1.1 <Build 1>  
Preprocessor Object: SF_SDF Version 1.1 <Build 1>  
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>  
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>  
Preprocessor Object: SF_SIP Version 1.1 <Build 1>  
Preprocessor Object: SF_SSL/P Version 1.1 <Build 4>  
Preprocessor Object: SF_GTP Version 1.1 <Build 1>  
Preprocessor Object: SF_SSH Version 1.1 <Build 3>  
Preprocessor Object: SF_DNS Version 1.1 <Build 1>  
Preprocessor Object: SF_DNS Version 1.1 <Build 4>  
Commencing packet processing (pid=14731)  
04/15-09:16:23.820140 [**] [1:60001:1] FTP detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:54030 → 192.168.56.101:21  
04/15-09:16:23.821176 [**] [1:60002:1] SSH detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:50024 → 192.168.56.101:22  
04/15-09:16:23.821176 [**] [1:60002:1] SSH detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:50024 → 192.168.56.101:22  
04/15-09:16:23.842110 [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:36926 → 192.168.56.101:705  
04/15-09:16:23.875816 [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:59326 → 192.168.56.101:161  
04/15-09:16:38.508978 [**] [1:60001:1] FTP detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:54044 → 192.168.56.101:21  
04/15-09:16:38.508978 [**] [1:60002:1] SSH detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:50034 → 192.168.56.101:22  
04/15-09:16:38.508265 [**] [1:60002:1] SSH detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:50034 → 192.168.56.101:22  
04/15-09:16:38.507225 [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:36962 → 192.168.56.101:705  
04/15-09:16:38.523422 [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:59336 → 192.168.56.101:161  
04/15-09:16:42.595147 [**] [1:60002:1] SSH detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:50042 → 192.168.56.101:22  
04/15-09:16:42.595147 [**] [1:60001:1] FTP detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:54060 → 192.168.56.101:21  
04/15-09:16:42.595383 [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:36974 → 192.168.56.101:705  
04/15-09:16:42.620638 [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:59344 → 192.168.56.101:161  
04/15-09:16:52.018995 [**] [1:60002:1] SSH detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:50050 → 192.168.56.101:22  
04/15-09:16:52.018995 [**] [1:60001:1] FTP detect ISM [**] [Priority: 0] [TCP] 192.168.56.102:54074 → 192.168.56.101:21  
04/15-09:16:52.018995 [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:36982 → 192.168.56.101:705  
04/15-09:16:52.022910 [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.56.102:59346 → 192.168.56.101:161
```

Pinging on kali system from ubuntu system:

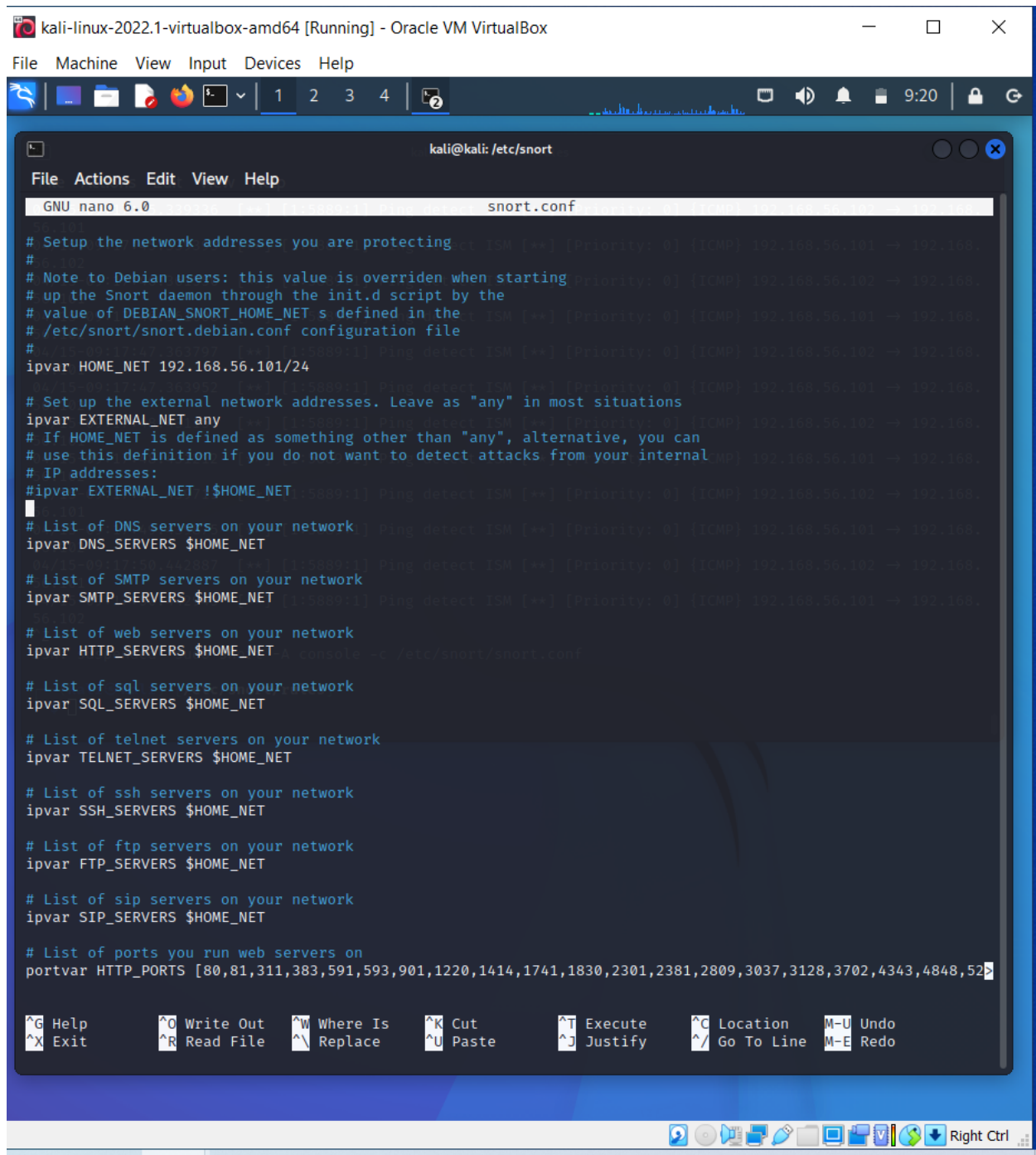


The image shows two terminal windows side-by-side. The left window is titled 'shivangl@shivangl-VirtualBox: ~' and shows the output of a ping command from 192.168.56.101 to 192.168.56.101. The right window is titled 'kali@kali-jetchnoorthules' and shows a list of installed packages.

```
shivangl@shivangl-VirtualBox: ~  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$  
shivangl@shivangl-VirtualBox:~$ nmap 192.168.56.101  
Starting Nmap 7.80 ( https://nmap.org ) at 2022-04-15 16:38 IST  
Nmap scan report for 192.168.56.101  
Host is up (0.00045s latency).  
All 1000 scanned ports on 192.168.56.101 are closed  
  
Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds  
shivangl@shivangl-VirtualBox:~$ ping 192.168.56.101  
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.  
64 bytes from 192.168.56.101: icmp_seq=1 ttl=64 time=0.648 ms  
64 bytes from 192.168.56.101: icmp_seq=2 ttl=64 time=0.828 ms  
64 bytes from 192.168.56.101: icmp_seq=3 ttl=64 time=1.15 ms  
64 bytes from 192.168.56.101: icmp_seq=4 ttl=64 time=1.07 ms  
64 bytes from 192.168.56.101: icmp_seq=5 ttl=64 time=1.13 ms  
64 bytes from 192.168.56.101: icmp_seq=6 ttl=64 time=1.19 ms  
64 bytes from 192.168.56.101: icmp_seq=7 ttl=64 time=1.21 ms  
64 bytes from 192.168.56.101: icmp_seq=8 ttl=64 time=1.26 ms  
64 bytes from 192.168.56.101: icmp_seq=9 ttl=64 time=1.19 ms  
64 bytes from 192.168.56.101: icmp_seq=10 ttl=64 time=1.45 ms  
64 bytes from 192.168.56.101: icmp_seq=11 ttl=64 time=1.25 ms  
64 bytes from 192.168.56.101: icmp_seq=12 ttl=64 time=1.17 ms  
64 bytes from 192.168.56.101: icmp_seq=13 ttl=64 time=1.66 ms  
64 bytes from 192.168.56.101: icmp_seq=14 ttl=64 time=1.17 ms  
^Z  
[2]+  Stopped                  ping 192.168.56.101  
shivangl@shivangl-VirtualBox:~$
```

```
kali@kali-jetchnoorthules  
File Actions Edit View Help  
04/15-09:17:37.185546 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:37.185582 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:38.224835 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:38.224872 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:39.275301 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:39.275359 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:40.279700 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:40.279757 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:41.282984 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:41.283840 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:42.305244 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:42.305349 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:43.319495 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:43.319544 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:44.337037 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:44.337094 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:45.339336 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:45.339393 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:46.361550 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:46.361608 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:47.363797 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:47.363952 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101  
04/15-09:17:48.311356 [**] [1:5889:1] Ping detect ISM [**] [Priority: 0] [ICMP] 192.168.56.102 → 192.168.56.101
```


Making necessary HOME_NET changes in snort.conf file:



The screenshot shows a Kali Linux virtual machine running in Oracle VM VirtualBox. The main window displays the `snort.conf` file being edited with the GNU nano 6.0 text editor. The file contains configuration for Snort, including network addresses, external network settings, and various server lists. The `HOME_NET` variable is set to `192.168.56.101/24`. The `EXTERNAL_NET` variable is set to `!$HOME_NET`. The `DNS_SERVERS`, `SMTP_SERVERS`, `HTTP_SERVERS`, `SQL_SERVERS`, `TELNET_SERVERS`, `SSH_SERVERS`, `FTP_SERVERS`, `SIP_SERVERS`, and `HTTP_PORTS` are also configured. The bottom of the window shows a terminal window with the `snort` command being executed. The status bar at the bottom indicates the time is 9:20 and the system is running on a 64-bit architecture.

```
kali@kali: /etc/snort
File Actions Edit View Help
GNU nano 6.0 snort.conf
# Setup the network addresses you are protecting
#
# Note to Debian users: this value is overridden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 192.168.56.101/24

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET

# List of sip servers on your network
ipvar SIP_SERVERS $HOME_NET

# List of ports you run web servers on
portvar HTTP_PORTS [80,81,311,383,591,593,901,1220,1414,1741,1830,2301,2381,2809,3037,3128,3702,4343,4848,52]
```

Conclusion:

Following a review of IDS and its classifications, multiple Snort-based Intrusion Detection approaches are presented and shown in this project to maintain an organization's security against threats. Snort-based IDPS with efficient rules, Bayesian Networks, Honeypots, Hardware-assisted techniques, Neural Networks, and Multi-Sensors-like techniques can protect against simple intrusions to dangerous DoS and DDoS type attacks in high-speed and Cloud environments, but they have significant drawbacks. Several issues are recognised and explained that must be taken into account while creating effective IDS for any network. Many options to improve the efficiency of a Snort-based Intrusion Detection and Prevention System remain. In future we will integrate the proposed design into Snort tool and evaluate it to achieve better detection rate with less false alarms.

ACKNOWLEDGMENT

We would like to express our gratitude to our teacher (Ruby D) who provided us with the golden opportunity to undertake this wonderful project on the topic (Network Intrusion Detection and Prevention using SNORT tool), which also helped us in doing a lot of Research and we came to know about so many new things that we are really thankful for.

References:

1. Tasneem, A., Kumar, A., & Sharma, S. (2018). Intrusion Detection Prevention System using SNORT. *International Journal of Computer Applications*, 181(32), 21-24.
2. Thakkar, N., Karamta, M., Joshi, S., & Potdar, M. B. (2019). Anomaly Detection and Categorization in Cloud Environment using Deep Learning Techniques. *International Journal of Computer Sciences and Engineering (IJCSE)*, 7(5), 211-214.
3. Erlansari, A., Coastera, F. F., & Husamudin, A. (2020). Early Intrusion Detection System (IDS) using Snort and Telegram approach. *SISFORMA*, 7(1), 21-27.

4. Gupta, A., & Sharma, L. S. (2020). Performance evaluation of snort and suricata intrusion detection systems on ubuntu server. In *Proceedings of ICRIC 2019* (pp. 811-821). Springer, Cham.
5. Resmi, A. M., & Manicka, R. (2017). Intrusion Detection System Techniques and Tools: A Survey. *Scholars Journal of Engineering and Technology (SJET)*.
6. MF Kabir, S Hartmann - 2018. Cyber security challenges: An efficient intrusion detection system design. 2018 International Young Engineers Forum (YEF-ECE)
7. Gaddam, R., & Nandhini, M. (2017, March). An analysis of various snort based techniques to detect and prevent intrusions in networks proposal with code refactoring snort tool in Kali Linux environment. In *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)* (pp. 10-15). IEEE.
8. Bul'ajoul, W., James, A., & Shaikh, S. (2019). A new architecture for network intrusion detection and prevention. *IEEE access*, 7, 18558-18573.
9. Erlacher, F., & Dressler, F. (2018, April). FIXIDS: A high-speed signature-based flow intrusion detection system. In *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium* (pp. 1-8). IEEE.
10. de la Cruz, J. E. C., Goyzueta, C. A. R., & Cahuana, C. D. (2020, September). Intrusion Detection and Prevention System for Production Supervision in Small Businesses Based on Raspberry Pi and Snort. In *2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)* (pp. 1-4). IEEE.
11. Elshafie, H. M., Mahmoud, T. M., & Ali, A. A. (2019, February). Improving the performance of the snort intrusion detection using clonal selection. In *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)* (pp. 104-110). IEEE.
12. Jain, G. (2021, March). Application of SNORT and Wireshark in network traffic analysis. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1119, No. 1, p. 012007). IOP Publishing.
13. Hassan, Z., Odarchenko, R., Gnatyuk, S., Zaman, A., & Shah, M. (2018, October). Detection of distributed denial of service attacks using snort rules in cloud computing & remote control systems. In *2018 IEEE 5th*

International Conference on Methods and Systems of Navigation and Motion Control (MSNMC) (pp. 283-288). IEEE.

14. Guezzaz, A., Asimi, A., Asimi, Y., Tbatou, Z., & Sadqi, Y. (2019). A Global Intrusion Detection System using PcapSockS Sniffer and Multilayer Perceptron Classifier. *Int. J. Netw. Secur.*, 21(3), 438-450.
15. Resmi, A. M., & Manicka, R. (2017). Intrusion Detection System Techniques and Tools: A Survey. *Scholars Journal of Engineering and Technology (SJET)*.
16. Visoottiviseth, V., Chutaporn, G., Kungvanruttana, S., & Paisarnduangjan, J. (2020, October). PITI: Protecting Internet of Things via Intrusion Detection System on Raspberry Pi. In *2020 International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 75-80). IEEE.
17. Elshafie, H. M., Mahmoud, T. M., & Ali, A. A. (2019, February). Improving the performance of the snort intrusion detection using clonal selection. In *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)* (pp. 104-110). IEEE.
18. Kattamuri, S. M., & Kakulapati, V. (2018). Performance Analysis of Mail Clients on Low Cost Computer With ELGamal and RSA Using SNORT. In *Handbook of Research on Pattern Engineering System Development for Big Data Analytics* (pp. 332-353). IGI Global.
19. Baykara, M., & Das, R. (2018). A novel honeypot based security approach for real-time intrusion detection and prevention systems. *Journal of Information Security and Applications*, 41, 103-116.
20. Azeez, N. A., Bada, T. M., Misra, S., Adewumi, A., Vyver, C. V. D., & Ahuja, R. (2020). Intrusion detection and prevention systems: an updated review. *Data management, analytics and innovation*, 685-696.
21. Liang, W., Li, K. C., Long, J., Kui, X., & Zomaya, A. Y. (2019). An industrial network intrusion detection algorithm based on a multifeature data clustering optimization model. *IEEE Transactions on Industrial Informatics*, 16(3), 2063-2071.
22. Gaddam, R., & Nandhini, M. (2017, March). An analysis of various snort based techniques to detect and prevent intrusions in networks proposal with code refactoring snort tool in Kali Linux environment. In *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)* (pp. 10-15). IEEE.

23. Gupta, A., & Sharma, L. S. (2020). A categorical survey of state-of-the-art intrusion detection system-Snort. *International Journal of Information and Computer Security*, 13(3-4), 337-356.
24. Karim, I., Vien, Q. T., Le, T. A., & Mapp, G. (2017). A comparative experimental design and performance analysis of snort-based intrusion detection system in practical computer networks. *Computers*, 6(1), 6.
25. Gupta, A., & Sharma, L. S. (2020). Performance evaluation of snort and suricata intrusion detection systems on ubuntu server. In *Proceedings of ICRIC 2019* (pp. 811-821). Springer, Cham.
26. Kabir, M. F., & Hartmann, S. (2018, May). Cyber security challenges: An efficient intrusion detection system design. In *2018 International Young Engineers Forum (YEF-ECE)* (pp. 19-24). IEEE.
27. Olanrewaju, R. F., Khan, B. U. I., Najeeb, A. R., Zahir, K. N., & Hussain, S. (2018). Snort-based smart and swift intrusion detection system. *Indian Journal of Science and Technology*, 11(4), 1-9.
28. Singh, V. K., Ebrahim, H., & Govindarasu, M. (2018, September). Security evaluation of two intrusion detection systems in a smart grid scada environment. In *2018 North American Power Symposium (NAPS)* (pp. 1-6). IEEE.
29. Kanna, P. R., & Santhi, P. (2021). Unified deep learning approach for efficient intrusion detection system using integrated spatial-temporal features. *Knowledge-Based Systems*, 226, 107132.
30. Baker, Z. K., & Prasanna, V. K. (2004, April). A methodology for synthesis of efficient intrusion detection systems on FPGAs. In *12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines* (pp. 135-144). IEEE.