International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11s
DOI: https://doi.org/10.17762/ijritcc.v11i11s.8172
***Article Received:** 27 June 2023 **Revised:** 20 August 2023 **Accepted:** 08 September 2023*

A Method for Malicious Network Packet Detection based on Anomalous TTL Values

Geetika Sharma[1], Rajni Ranjan Singh Makwana[2]
[1]Computer Science and Engineering
Madhav Institute of Technology & Science
Gwalior, INDIA
geetika.sharma03@gmail.com
[2]Computer Science and Engineering
Madhav Institute of Technology & Science
Gwalior, INDIA
ranjansingh06@gmail.com

Abstract—In the current digital age, a pervasive shift towards digitalization is evident in all aspects of life, encompassing entertainment, education, business, and more. Consequently, the demand for internet access has surged, paralleled therefore unfortunate escalation in cybercrimes. This study undertakes an exploration into the intrinsic nature of network packets, aiming to discern their potential for malice or legitimacy. In the internet, 32 intermediate nodes are encountered by a Network packet before it reaches its final host. Our findings suggest that the time-to-live (TTL) parameter in certain IP packets diverges from the initial TTL by more than 32 intermediary hops. It's likely that these packets are generated by specialized software. We anticipate that malicious IP packets exhibit unconventional TTL values, influenced by factors such as the source machine's operating system and protocols like TCP/ICMP/UDP, etc. To gauge the effectiveness and value of the proposed method, an experiment was conducted utilizing the SNORT NIDS system. Filtering rules based on signatures were formulated to thoroughly analyze the traffic. Real network data, along with DARPA and MACCDC 2012 datasets, were employed as inputs for the SNORT NIDS, and it has been observed that the suggested approach successfully detects the anomalous network packets.

Keywords-NIDS, SNORT, Network Security.

I. INTRODUCTION

We live in a digital universe where data is transmitted via the internet. Cyberattacks on the core network infrastructure or Internet services are always a possibility. The volume of malicious traffic is continuously increasing. We must safeguard our data and also our system against malicious actions that could harm our device or data. To verify all data packets traveling throughout the network, we employ an intrusion detection system (IDS). [3]SNORT is an intrusion detection system that analyses network data and generates alerts.

A. Network Intrusion Detection system (IDS)

Network traffic is monitored by an NIDS, which warns users when it detects any unusual activity. It is a piece of software that checks networks for illegal or dangerous activity. Any harmful activity is usually reported to an administrator or centralised data is gathered by a SIEM system (security information and event management). A SIEM system combines the outputs from several sources and uses alarm filtering techniques to separate valid alarms from false alarms [6][23]. A SNORT NIDS performs the following functions:

1) Packet Capture- The above graphic depicts the SNORT working model, where we can observe networks in the form of skies, datasets, or any other type of data originating from clients that are saved in packet capture recorded mode. Based on the IP addresses they have, these networks are organized into hierarchical groups. [25]

2) Packet Analysis- Protocol analysis is performed here with the network sniffing utility SNORT, which also captures packets within protocol levels for a deeper examination. Now, the network supervisor may devote greater attention to potentially harmful data packets. [26]

3) Rule set- SNORT enables users to quickly develop new rules following their software's filter rules. This enables network administrators to modify SNORT conversation's functionality for them and the operations it should perform.[28]

4) Detection System- According to the rule set, SNORT detection depends on packet headers. It arranges rules into categories based on ports, protocols like IP and TCP, and rules with and without content. Employing multi-pattern matches in rules which do involve content improves performance. Especially when we are using protocols like HTTP. The creation of rules without a purpose gives no useful results.[27]

5) Alert Generation- SNORT produces notifications for users based on the rule steps outlined within its configuration

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11s
DOI: https://doi.org/10.17762/ijritcc.v11i11s.8172
***Article Received:** 27 June 2023 **Revised:** 20 August 2023 **Accepted:** 08 September 2023*

file. These rules must precisely define the conditions under which a packet should be considered suspicious or malicious. This includes identifying the risks associated with exploiting vulnerabilities and the potential for such actions to violate the organization's security policies or pose a threat to the network, thus prompting the issuance of warnings.[21]

6) Log File- SNORT will record each IP message that enters the network once it's set to record packet traffic. The administrator of the network will then be able to see who has connected to it, what operating system as well as protocols have been employed, & identify those individuals.[29]

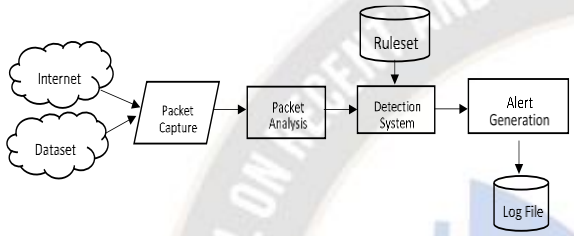

Figure 1: Working Model of Snort

B. *TIME-TO-LIVE (TTL)*

In the realm of computer networking, TTL (Time-to-Live) refers to a field situated in the header of an IP (Internet Protocol) datagram. This field governs the maximum number of hops, or intermediate devices, that the datagram is permitted to traverse before its disposal becomes necessary. The sender of the datagram establishes the initial TTL value, which then undergoes decrementing with each subsequent device forwarding the datagram.[24] In the event that the TTL value dwindles to zero, the datagram is promptly discarded, prompting the dispatch of an ICMP (Internet Control Message Protocol) message back to the sender. This message serves to apprise the sender of the occurrence of the failure[1] [2].

In the following figure, we can see Host A is sending network packets with an initial TTL value of 128 and when it passes through a router 1, it decreases its TTL value by 1. Similarly, when it reaches its destination host B, it shows its final TTL value is 124. So we can calculate the hop_count value by subtracting the TTL initial value (ti) from the TTL final value (tf), which equals 128-124 = 4ms.

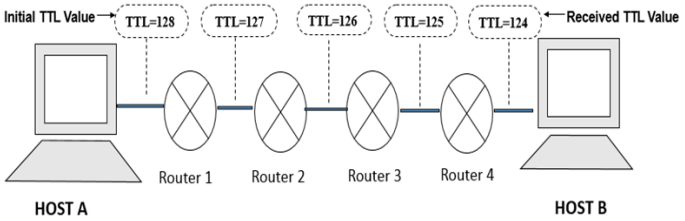

Figure 2: TTL decrement by Routers

As we know, the TTL value is used in data packets by the operating system. Every OS has its own reserved TTL values for TCP/UDP and ICMP. Every protocol has a fixed TTL value.

$$hop\_count(t) = ti - tf \quad \text{- equation(1)}$$

In equation 1, ti is the initial or origin TTL value and tf is the final or received TTL value. Ordinarily, the hop count value for most network packets doesn't exceed 32. However, a minuscule portion of TTL values surpass this threshold. Research indicates that merely 1.28% of IPv4 addresses and a mere 0.01% of IPv6 addresses showcase TTL values that fall beyond the scope of fewer than 32 hops [11]. Table 1 contains the initial TTL values of all versions of operating systems [4][5]. In this table we observe that older versions carried 30 or 32 minimum shows the TTL values and newer versions carried at least 64 TTL values.

Table 1: List of Operating System [4][5]

Device / OS	Version	Protocol	TTL
AIX	-	TCP	60
AIX	-	UDP	30
Android	3.2.1, 5.1.1	TCP and ICMP	64
AIX	3.2, 4.1	ICMP	255
BSDI	BDS/ OS 3.1/4.0	ICMP	255
Compa	Tru64 v5.0	ICMP	64
Cisco	-	ICMP	254
DEC Pathworks	V5	TCP and UDP	30
Foundry	-	ICMP	64
FreeBSD	2.1R	TCP and UDP	64
FreeBSD	3.4, 4.0	ICMP	255
FreeBSD	5	ICMP	64
HP-UX	9.0x	TCP and UDP	30
HP-UX	10.01	TCP and UDP	64
HP-UX	10.02	ICMP	255
HP-UX	11	ICMP	255
HP-UX	11	TCP	64
Irix	5.3, 6.x	TCP and UDP	60
Irix	6.5.3, 6.5.8	ICMP	255
juniper	-	ICMP	64
MPE/IX(HP)	-	ICMP	200
Linux	2.0.x kernel	ICMP	64
Linux	2.0.14 kernel	ICMP	255
Linux	2.4 kernel	ICMP	255
Linux	Red Hat 9	ICMP and UDP	64
MacOS/MacTCP	2.0.x	TCP and UDP	60
MacOS/MacTCP	X(10.5.6)	ICMP/UDP/TCP	64
NetBSD	-	ICMP	255
Netgear FVG318	-	ICMP and UDP	64
OpenBSD	2.6 & 2.7	ICMP	255
OpenVMS	07.01.2002	ICMP	255
OS/2	TCP/IP 3.0	-	64
OSF/1	V3.2A	TCP	60
OSF/1	V3.2A	UDP	30
Solaris	2.5.1, 2.6, 2.7, 2.8	ICMP	255
Solaris	2.8	TCP	64
Stratus	TCP_OS	ICMP	255
Stratus	TCP_OS(14.2-)	TCP and UDP	30
Stratus	TCP_OS(14.3+)	TCP and UDP	64
Stratus	STCP	ICMP/TCP/UDP	60
SunOS	4.1.3/4.1.4	TCP and UDP	60
SunOS	5.7	ICMP and TCP	255
Ultrix	V4.4/V4.2A	TCP	60
Ultrix	V4.4/V4.2A	UDP	30
Ultrix	V4.2/V4.2A	ICMP	255
VMS/Multinet	-	TCP and UDP	64
VMS/TCPware	-	TCP	60
VMS/TCPware	-	UDP	64
VMS/Wollongong	1.1.1.1	TCP	128
VMS/Wollongong	1.1.1.1	UDP	30
VMS/UCX	-	TCP and UDP	128
Windows	For Workgroups	TCP and UDP	32
Windows	95	TCP and UDP	32
Windows	98	ICMP	32
Windows	98, 98 SE	ICMP	128
Windows	98	TCP	128
Windows	NT 3.51	TCP and UDP	32
Windows	NT 4.0	TCP and UDP	128
Windows	NT 4.0 SP5-	-	32
Windows	NT 4.0 SP6+	-	128
Windows	NT 4 WRKS SP 3, SP 6a	ICMP	128
Windows	NT 4 Server SPA	ICMP	128
Windows	ME	ICMP	128
Windows	2000 pro	ICMP/TCP/UDP	128
Windows	2000 family	ICMP	128
Windows	XP/Vista/7	ICMP/TCP/UDP	128
Windows	Server 2008/10	ICMP/TCP/UDP	128

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11s
DOI: https://doi.org/10.17762/ijritcc.v11i11s.8172
***Article Received:** 27 June 2023 **Revised:** 20 August 2023 **Accepted:** 08 September 2023*

II. RELATED WORK-

Various works have been presented by the authors for the detection of the anomalous network traffic, some most relevant are as follows:-

A. TTL-Based Review Paper-

Yamada 2013 et al. [9] this paper presents a new approach for detecting malicious packets based on the time-to-live (TTL) values of IP packets. The proposed method involves classifying packets as either normal or abnormal based on their TTL values, with abnormal TTL packets being filtered as potentially malicious. Additionally, the method does not require updating databases for discriminating malicious packets. The authors suggest conducting additional comparative studies with existing methods to evaluate the effectiveness of the proposed method. Furthermore, the paper outlines ongoing research to estimate TTL values by sending ICMP echo request packets to the source IP address of an abnormal TTL packet and comparing the estimated TTL value with the actual hop count indicated by the ICMP reply packet.

Paxson et al. [10] asserts that the majority of Internet routes (but not all) had less than 30 hops in 1997, he also draws the conclusion that the Internet's core has expanded beyond 30 hops and thus starting TTL values greater than 30 should be utilized. They have observed this, which supports it. Assuming insignificant, 1.28 percent of IPV4 addresses and 0.01% of IPV6 addresses have TTL values that are more than 32 hops or less below typical start values .

Scheitle 2016 et al. [11] This work presents the use of TTL values for anomaly detection in carrier-grade networks. The authors capture and analyze a dataset of TTL values and find that a majority of IP addresses are TTL-stable, while developing methods to analyze and quantify subgroups of unstable IP addresses. They also investigate the correlation between TTL data and BGP data and compare the results with ping-back scans to determine anchor Hop Count values for multi-TTL IP addresses. The survey presents insights gained from the research, including the observation that passive data carries largely unbiased Hop Count data and that different subnet sizes show uniform Hop Count behavior.

Kushwah D. et al. [12] presented a method for the detection of anomalous traffic based the anomalous TCP flag values. All the abnormal flag combination have been identified and packets are scrutinized by the SNORT system based on abnormal flag combinations.

Jin et al. in 2003 [13], The IP header's Time-to-Live information has two intriguing features. First, different IP stacks use various start TTL values. Second, the TTL value needs to be decreased for each router that is traversed. Several organizations have explored the idea of using inbound TTL values to detect suspicious and perhaps faked packets. A method to use hop count filtering to protect against fake DDoS traffic was put forth .Its data collection and evaluation are based on trace route measurements to a small set of targets and fake traffic, which restricts their relevance to a broader audience.

Vanaubel et al. in 2013 [14], stated that router fingerprinting may be useful for a variety of purposes, including spotting weak routers or strange TTL activity. Lightweight router fingerprinting method based on the router signature, or the n-tuple of initial TTL values that a router uses to spoof ICMP reply packets.

III. PROPOSED METHODOLOGY

This paper proposed a methodology for detecting malicious packets using abnormal time-to-live TTL values in IP packet headers. We know that, generally, an IP packet takes less than or equal to 32 hops to travel between source and destination. If a packet has a higher TTL value, we assume it was generated by specialized software. We mark those packets that contain abnormal TTL values as malicious packets and generate alerts. SNORT NIDS is utilized for the implementation of the proposed work. To detect malicious network packets using TTL values, a network intrusion detection system (NIDS) can be used. The NIDS would monitor network traffic and examine the TTL values of incoming packets. If a packet is found to have a TTL value that deviates from the expected TTL value, it may be flagged as suspicious.

For example, if a packet is received with a TTL value of 1, it may indicate that the packet has been spoofed to make it appear as if it originated from within the network. Once a suspicious packet has been identified, further analysis can be carried out to determine whether it is indeed malicious or not. This may involve examining the packet's payload, source IP address, and other network attributes to identify any signs of malicious activity.

The TTL (time-to-live) value of an IP packet depends on the protocol used, such as TCP, UDP, or ICMP, as well as the operating system of the sending device. Therefore, when detecting malicious network packets using TTL values, it is important to take into account the protocol used, the operating system of the sending device, and the expected range of TTL values for the network. Generally an IP packet takes less than or equal to 32 hops to travel between source to destination [11]. If a packet takes extra ordinary TTL value than we assume that it is generated by some special software. We mark those packets who contains abnormal TTL values as a malicious packets and generating SNORT alerts.

As shown in figure 6, the proposed model captures data either through datasets or real incoming network traffic. The captured packets are analyzed by applying a signature based ruleset on it.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11s
DOI: https://doi.org/10.17762/ijritcc.v11i11s.8172
Article Received: 27 June 2023 Revised: 20 August 2023 Accepted: 08 September 2023

The first step is to check whether the packet is TCP or not. If it matches, the packet is again checked for invalid TTL values. If TTL range is invalid then it is marked as malicious otherwise legitimate.

Similar approach is used to identify malicious packets for UDP and ICMP network packets. This process ensures that all potentially malicious packets are identified and logged for review, while the legitimate packets are allowed to pass through the filtration process without interruption. By sending alerts to the administrator, any potential security threats can be addressed promptly and efficiently, reducing the risk of network downtime or compromise.

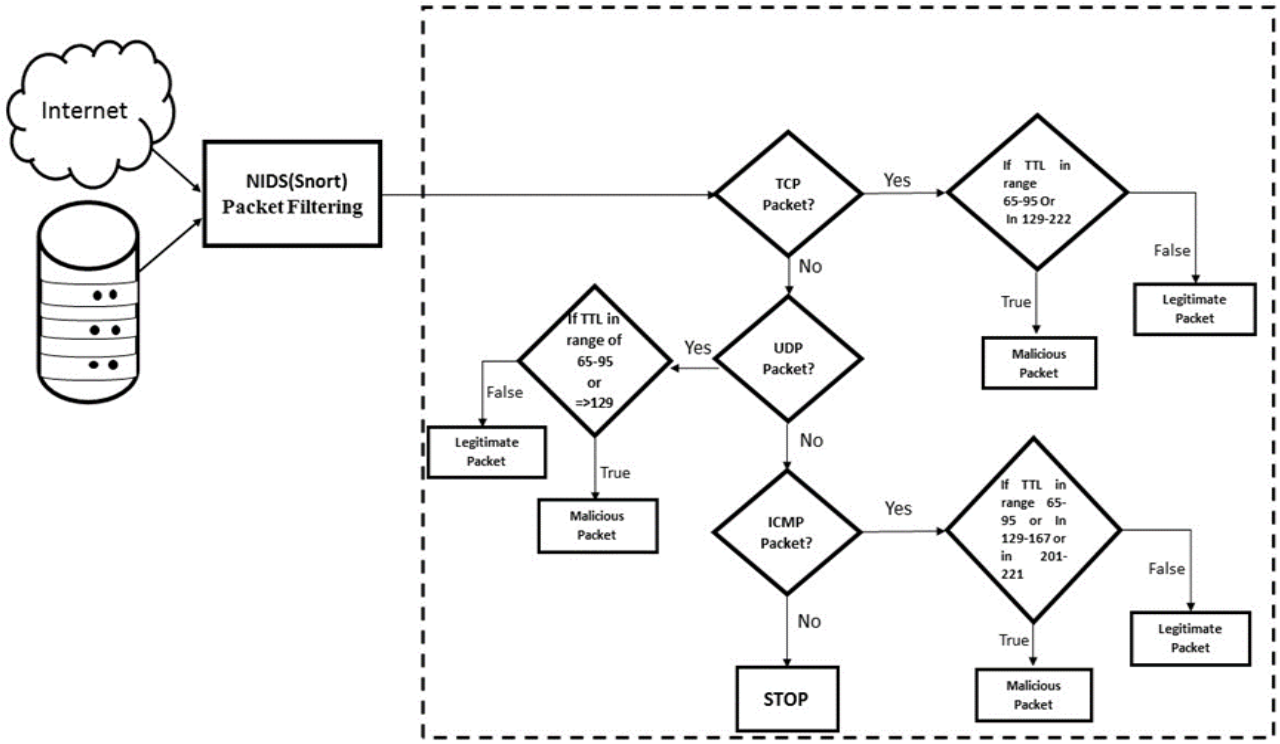

Figure 3: Proposed Methodology

In the above OS table 1 maximum all OS and their current and older versions included. Taking this OS data in our SNORT rules as TTL basis filtering of network packets.

From the above table we find some initial TTL value set for TCP packet:
{30, 32, 60, 64, 128, 255}
We got normal TTL range for TCP-
{1-64}, {96-128}, {223-255}
Now, from this normal TTL range we can easily get abnormal TTL range: {65-95}, {129-222}

Similarly, we can find normal or abnormal TTL value for UDP or ICMP too.
For UDP {30, 32, 60, 64, and 128} are the initial TTL value set
Normal TTL: {1-64}, {96-128}
Abnormal TTL: {65-95}, {129-255}

For ICMP {32, 60, 64, 128, 200, 254, and 255} are the initial TTL values
Normal TTL ranges: {1-64}, {96-128}, {168-200}, {222-255}
Abnormal TTL ranges: {65-95}, {129-167}, {201-221}

Any network packets with these abnormal TTL values are identified as suspicious and require further investigation.

IV. DATASETS

To analyse our ruleset, we need to apply it to a dataset. In this work, the following datasets are utilized: DARPA, MACCDC, and Real Network Data. Table 2 shows the dataset details.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11s
DOI: https://doi.org/10.17762/ijritcc.v11i11s.8172
***Article Received:** 27 June 2023 **Revised:** 20 August 2023 **Accepted:** 08 September 2023*

DARPA- To examine accurately an intrusion detection system, we need real-time capturing of network packets. DARPA contact off-line and on-line data were captured in 1998 and 1999 evaluations. DARPA Outside sniffing data (Tcpdump format) and inside sniffing data (Tcpdump format) are used [19].

MACCDC- The Mid-Atlantic Collegiate Cyber Defense Competition produced this dataset (MACCDC). The dataset that was used was taken in 2012. The dataset used in this study is around 1 GB in size and is in Tcpdump format. These attacks are carried out during a competition to instruct students by simulating real network circumstances [20].

Table 2. Datasets utilized

S.No.	DARPA MILL 98/99	MACCDC 2012	Real Network
1.	Outside Tcpdump	maccdc2012_00000	14 july 22 5pm to 6pm"1hr wireshark capture dataset"
2.	Inside Tcpdump	maccdc2012_00001	14 july 22 6pm to 7pm"1hr wireshark capture dataset"
3.	-	Maccdc2012_00008	14 july 22 8pm to 9pm"1hr wireshark capture dataset"

V. IMPLEMENTATIONS

SNORT filtering rules have been created as shown in Figure 4. SNORT is activated in NIDS mode. If any packet matches the rules, an alert will be generated and logs will be created as shown in figure 5.

```
alert tcp any any -> any any (msg: "Malicious Packet"; content: "ttl"; ttl:65-95;sid:1000005;)

alert tcp any any -> any any (msg: "Malicious Packet"; content: "ttl"; ttl:129-222; sid:1000006;)

alert udpany any -> any any (msg:"Malicious Packet"; content: "ttl"; ttl:65-95; sid:1000008;)

alert udp any any -> any any (msg:"Malicious Packet"; content: "ttl"; ttl:=>129; sid:1000009;)

alert icmpany any -> any any (msg:"Malicious Packet"; content: "ttl"; ttl:65-95; sid:1000010;)

alert icmp any any -> any any (msg:"Malicious Packet"; content: "ttl"; ttl:129-167; sid:1000011;)

alert icmp any any -> any any (msg:"Malicious Packet"; content: "ttl"; ttl:201-221; sid:1000012;)
```

Figure 4: Developed SNORT Rules

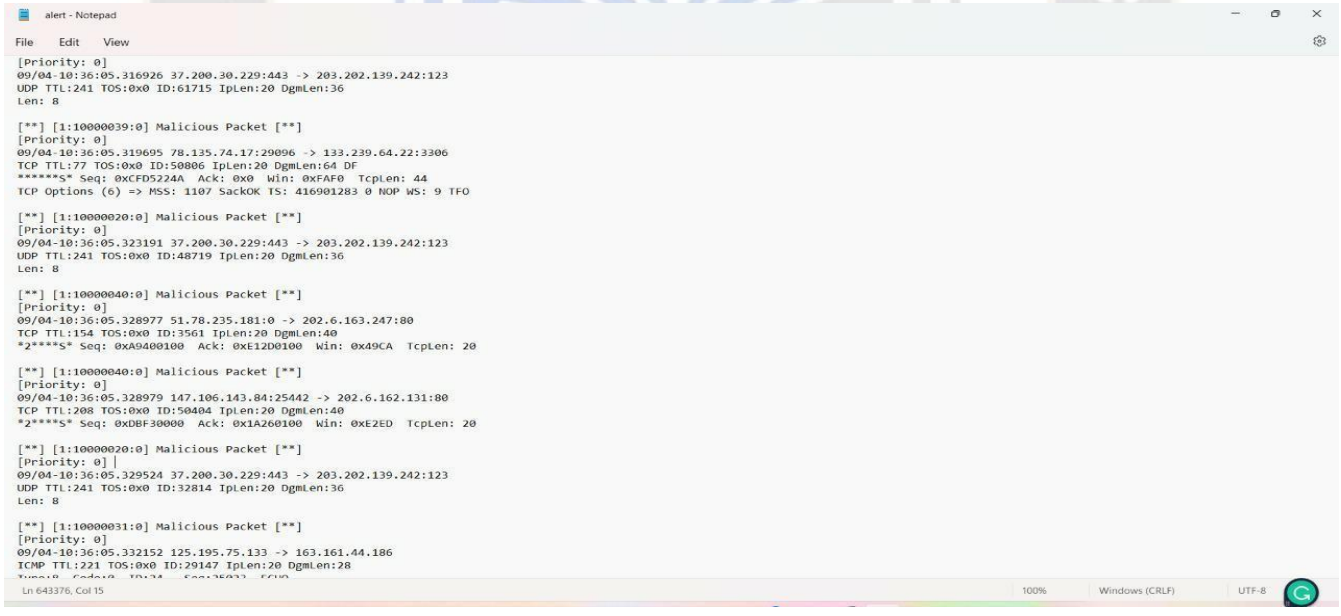

Figure 5. Snort alerts in log files

The set of SNORT rules as shown in Figure 4: are used to identify network traffic containing particular values in the Time to Live (TTL) field of the IP header. The TTL field is used to restrict the lifespan of a packet and prevent it from endlessly circulating in the network. Each time the packet is forwarded through a router, the value of the TTL field is reduced by one. If the TTL field reaches zero, the packet is discarded.

The seven rules shown in the Figure 4: are designed to detect packets with suspicious Time to Live (TTL) values in different protocols. The first two rules are designed to detect TCP packets, specifically between 65 and 95 or between 129

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11s
DOI: https://doi.org/10.17762/ijritcc.v11i11s.8172
***Article Received:** 27 June 2023 **Revised:** 20 August 2023 **Accepted:** 08 September 2023*

and 222. These values may indicate that the packet was generated by an attacker attempting to avoid detection by using an unusual TTL value.

Similarly, the next two rules are designed to detect UDP packets with TTL values outside the normal range, specifically between 65 and 95 or greater than 128.

The last two rules are designed to detect ICMP packets withTTL values between 65 and 95 or between 129 and 167 & 201 and 221. ICMP packets are typically used for diagnostic and error reporting purposes, and unusual TTL values in these packets may indicate malicious activity.

Each of these rules generates an alert with the message "Malicious Packet" if a packet matching the specified criteria is detected. The unique identification number assigned to each rule, known as the "sid" field, can be used to manage and track alerts generated by the rule. By using these rules, network administrators can detect and respond to potentially malicious network activity based on the TTL values in network packets.

After implementing above rules in snort.conf file. Following series of commands are executed:

C:\Snort\bin>snort.exe

C:\Snort\bin>snort –i 1 –c

C:\snort\etc\snort.conf –A full –l

C:\snort\log

In this snort console command i's value varies from 0 to 5 processor variation and –c is for consoling and –l is for creating log or "alert.ids" in logfolder.

After applying datasets and real network traffic, following results are obtained. It is observed that very few network packets having odd TTL values. It is possible that these packet are created by the special soft wares therefore treated as malicious packets.

Table 3. No. of alerts captured by Snort

Dataset	Total Packet Captured	Total Alerts generated	TCP	UDP	ICMP
DARPA	958881	4256	0	4256	0
MACCDC	4198011	2123	3	1999	121
Real network Traffic	92885	17220	63	16845	312

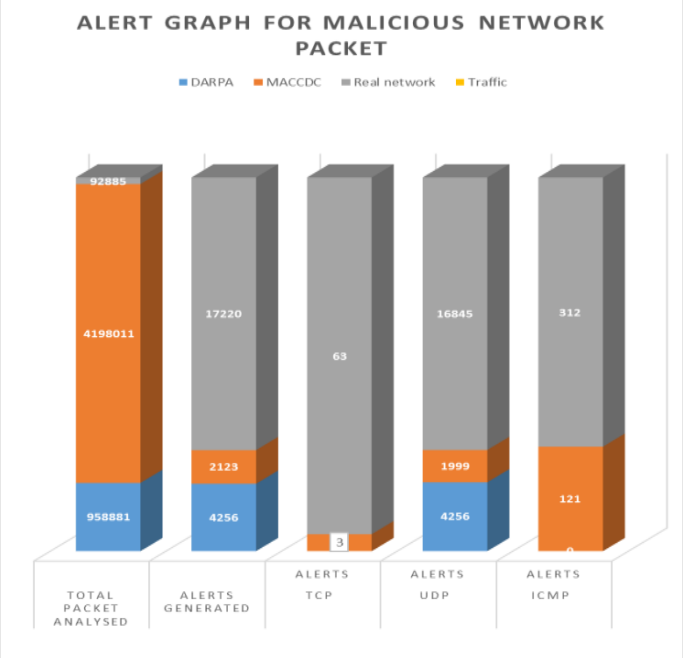

Figure 6: Result in graphical representation

VI. CONCLUSION AND FUTURE WORK

In the proposed work, we created snort rules to detect anomalous IP packets based on anomalous TTL values, and this work is beneficial to detecting malicious traffic. It is observed that very few packets have greater than 32 hop count values. They may be created by using specialised software. Further investigation of these packets is required. This work is totally focused on the anomalous TTL values. However, in the future, we may also check the anomalous flag values or window size to find out if the packet is legitimate or not.example, do not differentiate among departments of the same organization). This template was designed for two affiliations.

REFERENCES

[1] Andrew S. Tanenbaum, "Computer networks":4th Edition, 9788131701980,

[2] Forouzan, Behrouz A.. TCP/IP Protocol Suite. United Kingdom, McGraw-Hill, 2002.

[3] Roberto Di Pietro, Luigi Mancini, "Intrusion Detection Systems", ISBN: 9780387772660, Germany, Springer US, 2008.

[4] "Subin's blog" https://subinsb.com/default-device-ttl-values/

[5] "OSTECHNIX" https://ostechnix.com/identify-operating-system-ttl-ping/

[6] SNORT, "https://www.snort.org/", Network Intrusion Detection System.

[7] "Manual_Snort"http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node32.html

[8] "ResearchGate"https://www.researchgate.net/figure/Operating-Systems-TTL-Values_tbl1_260288614

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11s
DOI: https://doi.org/10.17762/ijritcc.v11i11s.8172
***Article Received:** 27 June 2023 **Revised:** 20 August 2023 **Accepted:** 08 September 2023*

[9] Yamada, R. and Goto, S., 2013. Using abnormal TTL values to detect malicious IP packets. *Proceedings of the Asia-Pacific Advanced Network, 34*, pp.27-34.

[10] Paxson, V., 1997. End-to-end routing behavior in the Internet. IEEE/ACM transactions on Networking, 5(5), pp.601-615.Q. Scheitle, O. Gasser, P. Emmerich, and G. Carle, "Carrier-Grade Anomaly Detection Using Time-to-Live Header Information," 2016, [Online]. Available: http://arxiv.org/abs/1606.07613

[11] Kushwah, D., Singh, R.R. and Tomar, D.S., 2019, January. An Approach to Meta-Alert Generation for Anomalous TCP Traffic. In International Conference on Security & Privacy (pp. 193-216). Springer, Singapore.

[12] Ahmed Al-Hunaiyyan, Asaad Alzayed, Rana Alhajri, Abdulwahed Khalfan. (2023). Using Social Networking Sites for Requirements Elicitation: Perspectives and Challenges. International Journal of Intelligent Systems and Applications in Engineering, 11(4s), 357–368. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2675.

[13] Jin, C., Wang, H. and Shin, K.G., 2003, October. Hop-count filtering: an effective defense against spoofed DDoS traffic. In Proceedings of the 10th ACM conference on Computer and communications security (pp. 30-41).

[14] Vanaubel, Y., Pansiot, J. J., Mérindol, P., & Donnet, B. (2013, October). Network fingerprinting: TTLbased router signatures. In Proceedings of the 2013 conference on Internet measurement conference (pp. 369-376).

[15] J. Ren, J. Guo, W. Qian, H. Yuan, X. Hao, and H. Jingjing, "Building an Effective Intrusion Detection System by Using Hybrid Data Optimization Based on Machine Learning Algorithms," Secur. Commun. Networks, vol. 2019, 2019, doi: 10.1155/2019/7130868

[16] R. T. Gaddam and M. Nandhini, "An analysis of various SNORT based techniques to detect and prevent intrusions in networks: Proposal with code refactoring SNORT tool in Kali Linux environment," Proc. Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2017, no. Icicct, pp. 10–15, 2017, doi: 10.1109/ICICCT.2017.7975177.

[17] M. Albanese, E. Battista, and S. Jajodia, "A deception based approach for defeating OS and service fingerprinting," 2015 IEEE Conf. Commun. NetworkSecurity, CNS 2015, pp. 317–325, 2015, doi: 10.1109/CNS.2015.7346842.

[18] R. Harang and P. Guarino, "Clustering of SNORT alerts to identify patterns and reduce analyst workload," Proc. - IEEE Mil. Commun. Conf. MILCOM, pp. 1–6, 2012, doi: 10.1109/MILCOM.2012.6415777.

[19] R. Harang and P. Guarino, "Clustering of SNORT alerts to identify patterns and reduce analyst workload," Proc. - IEEE Mil. Commun. Conf. MILCOM, pp. 1–6, 2012, doi: 10.1109/MILCOM.2012.6415777.

[20] 1999 DARPA Intrusion Detection Evaluation Dataset-https://www.ll.mit.edu/r-d/datasets/1999-darpaintrusion-detection-evaluation-dataset

[21] https://www.netresec.com/?page=MACCDC "National CyberWatch Mid-Atlantic Collegiate Cyber Defense Competition (MACCDC)"

[22] G. D. Kurundkar, N. A. Naik, and S. D. Khamitkar, "Network Intrusion Detection using SNORT," Int. J. Eng. Res. Appl., vol. 2, no. 2, pp. 1288–1296, 2012.

[23] R. R. Kompella, S. Singh, and G. Varghese, "On scalable attack detection in the network," IEEE/ACM Trans. Netw., vol. 15, no. 1, pp. 14–25, 2007, doi: 10.1109/TNET.2006.890115.

[24] V. A. Grønland, "Building IDS rules by means of a honeypot," Dep. Comput. Sci. Media Technol. Gjøvik Univ. Coll. 2006, vol. 77, 2006, [Online]. Available: http://brage.bibsys.no/hig/handle/URN:NBN:no-bibsys_brage_4248%5Cnhttp://brage.bibsys.no/hig/bitstream/URN:NBN:no-bibsys_brage_4248/1/Gronland - Building ids rules by means of a honeypot.pdf%5Cnhttp://brage.bibsys.no/hig/handle/URN:NBN:no-bibsys_brage_424

[25] R. Beverly, "A robust classifier for passive tcp/ip fingerprinting," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 3015, pp. 158–167, 2004, doi: 10.1007/978-3-540-24668-8_16.

[26] M. Roesch, "SNORT - Lightweight intrusion detection for networks," Proc. 13th Conf. Syst. Adm. LISA 1999, pp. 229–238, 1999.

[27] Kshirsagar, D. R. . (2021). Malicious Node Detection in Adhoc Wireless Sensor Networks Using Secure Trust Protocol. Research Journal of Computer Systems and Engineering, 2(2), 12:16. Retrieved from https://technicaljournals.org/RJCSE/index.php/journal/article/view/26.

[28] Kumar, Vinod, and Om Prakash Sangwan. "Signature based intrusion detection system using SNORT." International Journal of Computer Applications & Information Technology 1.3 (2012): 35-41.

[29] Z. Zhou, Z. Chen, T. Zhou, and X. Guan, "The study on network intrusion detection system of SNORT," 2010 Int. Conf. Netw. Digit. Soc. ICNDS 2010, vol. 2, pp. 194–196, 2010, doi: 10.1109/ICNDS.2010.5479341.

[30] Singh, R.R. and Tomar, D.S., 2015. Network forensics: detection and analysis of stealth port scanning attack. International Journal of Computer Networks and Communications Security, 3(2), pp.33-42.

[31] Kushwah D, Singh RR, Tomar DS. An approach to meta-alert generation for anomalous tcp traffic. InSecurity and Privacy: Second ISEA International Conference, ISEA-ISAP 2018, Jaipur, India, January, 9–11, 2019, Revised Selected Papers 2 2019 (pp. 193-216). Springer Singapore.