International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Classifying IP Spoofing Requests Using Heuristic Fuzzy Computer Network Approach

Ravula Kartheek[1], Battula Siva Kanaka Raju[2], D Jaya Sonia[3], Pinjala Gopinadh[4]

[1,3,4]Assistant Professor, St.Ann's Engineering College: Chirala
[2]Assistant Professor, Vignan Institute of Technology and Science: Telangana

Abstract:
Distributed Denial of Service (DoS) Attack This represents a serious problem for Internet communications. This problem is exacerbated if an attacker tampers with the data. Source IP address. Attackers can easily change the source IP address leads to unauthorized access to network resources. Many solutions are available. Please identify this issue in your research community. In this paper, we proposed a detection method that takes data flow into account. Metric. An attacker can change any field in the IP, so expect a packet hop count field. In this article, he will introduce one. A Heuristic Fuzzy Logic Approach to Spoofing Detection package. Effective implementation of fuzzy rules the two member functions μHC and μPTT are the Receive IP header. The classification of forged packets is as follows: Identify changes up close Hop count and packet transmission time due to fuzzy Membership function using fuzzy triangles Membership features.

Keywords: spoofed IP address, DDOS attacks, hip count, data packet transfer

I. INTRODUCTION

Distributed Denial of Service (DDoS) attacks often exploit vulnerabilities in IP networks. Spoofing techniques are employed to obscure the origins of flooding, resulting in the watering of traffic and the obfuscation of attack locations [1]. This manipulation aims to deceive legitimate hosts, enticing them to unwittingly serve as reflectors and redirecting traffic towards unintended destinations. Consequently, the amplification of traffic occurs as a consequence of the pervasive flooding.

Many DDoS attack tools adopt a strategy of concealing attacking sources and diffusing localities in attacking traffic by randomizing the 32-bit source address field in the IP header [2]. This technique serves to obfuscate the origins of the attack, contributing to the challenge of attribution and complicating efforts to trace back to the actual perpetrators. A recent study on "backscatter" [3] corroborates the prevalent adoption of randomness in spoofing IP addresses within the context of DDoS attacks.

Furthermore, certain well-documented DDoS attacks, including the Smurf attack [4] and more contemporary instances of Distributed Reflection Denial of Service (DRDoS) attacks [5], hinge on the utilization of IP spoofing. These attacks involve the manipulation of source IP addresses in each spoofed packet to replicate the victim's IP address. Generally, DDoS attacks employing IP spoofing pose significantly greater challenges in terms of defense.

In the quest to identify and mitigate DDoS attacks, two distinct methodologies emerge: router-based and host-based approaches. The router-based strategy involves embedding detection mechanisms within IP routers to trace the source(s) of the attack [6] or identify and block malicious traffic [7]. However, the efficacy of these router-based solutions hinges not only on router support but also necessitates coordination across diverse routers and networks, requiring widespread deployment to realize their full potential.

In contrast, the host-based approach offers a more immediate deployment option. Furthermore, the host-based approach has a notable advantage, as end systems possess a heightened incentive to implement defense mechanisms compared to network service providers.

Contemporary host-based approaches safeguard an Internet server through the implementation of sophisticated resource-management schemes [9]. Alternatively, they achieve protection by significantly curbing each request's resource consumption to withstand the traffic deluge, employing techniques such as SYN cookies [10] and Client Puzzle [11].

Existing host-based solutions operate at the transport layer and above, yet they fall short in preventing the victim server from expending CPU resources in responding to interrupts generated by spoofed IP traffic. Particularly at high speeds, the influx of incoming IP packets triggers numerous interruptions, posing a substantial risk of slowing down the victim server [12].

Consequently, the imperative to detect and filter spoofed packets at the IP layer, independent of router support, becomes indispensable for effective defense against DDoS attacks. This paper introduces a straightforward scheme designed to authenticate incoming IP packets at the victim's internet server, abstaining from the use of cryptographic methods or reliance on router support. The primary objective of this proposal is not to attain flawless authentication but rather to effectively filter out a majority of spurious traffic with minimal collateral damage. The underlying concept revolves around leveraging intrinsic network information, specifically considering factors such as the number of hops and the time it takes for a received packet to traverse from its source to its destination.

The subsequent sections of this paper are structured as follows. In Section II, we elucidate the TTL-based hop-count computation. Section III provides a demonstration of the computation of packet transfer time. Following this, Section IV delves into the construction of mapping tables. Section V elucidates the correlation between hop count and packet transfer time. Section VI discusses the fuzzy membership function-based search strategy for identifying spoofed packets. The ensuing section, Section VII, presents the results and discussions. The paper culminates by outlining future extensions in Section VIII.

II. HOP COUNT COMPUTATION

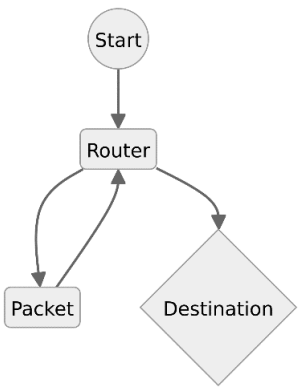

Fig1: Hop Count Computation

The determination of the number of hops a packet undergoes en route to its destination is encoded in the Time-to-Live (TTL) field of the IP header. Additionally, each intermediate router diminishes the TTL value by one before forwarding a packet to the subsequent hop. Since hop count information is not explicitly stored in the IP header, it is computed from the final TTL value.

A. Time to Live - TTL

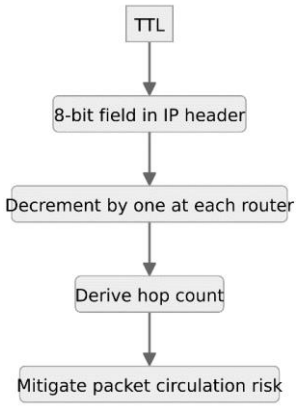

Fig2: Time to Live - TTL

The Time to Live (TTL) parameter serves as a crucial indicator of the permissible duration for a packet's presence within a network [19]. Its primary function is to mitigate the risk of a packet indefinitely circulating within the network. This is achieved by decrementing the TTL value by one each time the packet traverses a router. Consequently, the TTL value can be utilized to derive the hop count, representing the number of intermediary devices a packet has passed through. Encoded as an 8-bit field in the IP header, TTL was initially introduced to stipulate the maximum lifespan of each packet on the Internet.

B. Determining Hop Count: Methods and Approaches

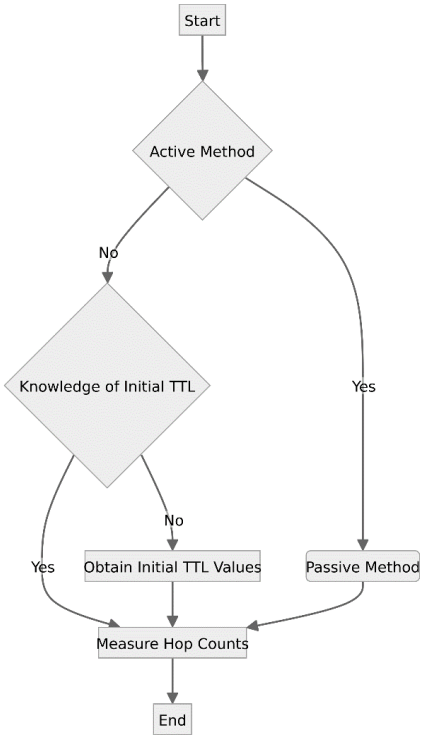

Fig3: Determining Hop Count: Methods and Approaches

Hop count computation involves two distinct approaches: active measurement and passive measurement. The active method employs ICMP ECHO packets, typically yielding an accurate hop count. However, applying this technique to a large number of hosts becomes impractical due to the recommendation against sending numerous ICMP packets for measurement purposes.

On the other hand, the passive method involves subtracting the Time to Live (TTL) of a received IP packet from its original value. This technique doesn't require sending sample packets, making it an ideal choice for measuring the hop counts of numerous hosts. Mathematically, the hop count can be expressed as the difference between the initial TTL and the TTL of the received packet:

(Hop count)} = (Initial TTL)

It is important to note that for this method to be effective, knowledge of the initial TTL values is essential beforehand.

C. Challenges Associated with Initial TTL Values

Under RFC 1700, the suggested initial Time to Live (TTL) value is 64. However, real-world internet practices frequently deviate from this recommendation. Extensive research conducted by the Swiss Academic & Research Network (SWITCH) on various Operating Systems (OS) reveals a range of six distinct initial TTL values: 30, 32, 60, 64, 128, and 255.

Distinguishing packets based on their initial Time to Live (TTL) values becomes relatively straightforward when dealing with extreme values such as 255 or 128. However, the assessment becomes more intricate for packets with TTL values below 60 or 64, and even more challenging for values below 30.

Notably, widely used operating systems such as Microsoft Windows, Linux, and Free BSD employ 32 and 64 as their respective initial TTL values. Consequently, the following formula is proposed for converting TTL to hop count.

Hop Count =
32-TTL TTL<=32
64-TTL TTL<=62
128-TTL TTL<=128
255-TTL TTL<=255

III. PACKET TRANSFER TIME

In the assessment of packet transfer time between the measurement point and the target host, it becomes imperative to implement a reliable measurement approach. The absence of a timestamp field in the IP header complicates the direct calculation of packet transfer time through packet header analysis. While employing measurement software capable of sending and receiving sample packets provides the most accurate results, this method is restricted to specific hosts due to the necessity of executing a measurement program. Consequently, it is unsuitable for collecting packet transfer times across numerous hosts.

An alternative approach involves sending an ICMP ECHO packet and measuring the time until its reply is received from the host. Although effective, this method proves cumbersome when dealing with a large volume of ICMP packets, and the round-trip time of ICMP packets may vary from that of IP packets.

To address these challenges, a passive method is proposed, involving the capture and analysis of TCP handshaking packets. This method facilitates the collection of roundtrip times for a substantial number of hosts without the need to dispatch unnecessary packets. During the establishment of a TCP connection,

the initiating host transmits a TCP packet with a SYN flag bit, signaling a request for connection establishment. In response, the destination host promptly sends back a TCP packet with ACK and SYN flag bits to accept the request and initiate the connection in the reverse direction. The negotiation concludes when the initiating host acknowledges the process with an ACK packet.

Notably, TCP inherently endeavors to establish separate upstream and downstream connections to achieve full-duplex transmission. This procedural sequence is commonly known as the 3-way handshake [20]. This passive method proves instrumental in capturing and analyzing TCP handshaking packets, offering an efficient means to collect roundtrip times across a diverse array of hosts without the need for extraneous packet transmission.

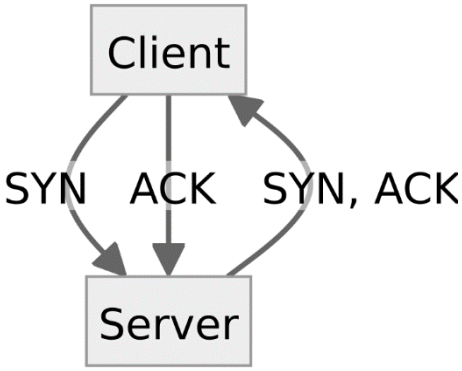

Figure 4. 3-Way handshake

The transmission of the ACK packet promptly follows the reception of the SYN packet, effectively making the ACK packet analogous to an ECHO packet corresponding to the SYN packet. Consequently, the time disparity between sending the SYN packet and receiving the subsequent ACK packet can be utilized as an approximate measure of the packet transfer time between the two hosts. This method offers a practical means to gauge the latency in transferring packets between host entities.

IV. CONSTRUCTION OF MAPPING TABLE

The meticulous creation of an accurate mapping table, encompassing hop count and packet transfer time (PTT) values for all IPs, is paramount for the effectiveness of our approach. To achieve this, the following methods were implemented for the construction and continuous updates of the mapping table:

1. Initialization of the Mapping Table:

To establish the mapping table initially, the Internet server administrator collects traces from clients, capturing both IP addresses and their corresponding hop-count values along with PTT. This initial data collection period is crucial for ensuring accuracy from the outset, with the duration tailored to the server's daily traffic volume. For highly frequented sites, a few days may suffice, while for less congested ones, a few weeks might be more suitable. After this initial population, the mapping table continues to grow by adding new entries whenever requests from previously unseen legitimate IP addresses are detected. Over time, the mapping table evolves to encompass accurate mappings between IP addresses and their associated hop counts, along with PTT, for all legitimate clients. This robust construction ensures the detection of spoofed IP traffic during a Distributed Denial of Service (DDoS) attack.

2. Updating the Mapping Table:

To maintain accuracy, the mapping table must be regularly updated as hop counts for existing IP addresses may change. Changes in hop counts could be triggered by network relocations, routing fluctuations, or

temporary network disruptions. While some of these events are transient and can be overlooked, longer-term changes in hop counts must be captured and reflected in the mapping table.

As per [22], the hop count starts at 4, with a maximum of 30 hops. Consequently, multiple IP addresses may share identical hop count values. This potential overlap necessitates scrutiny, particularly in detecting instances where attackers might exploit the same hop count values as those associated with spoofed IP addresses. Regularly examining hop count distributions across different internet locations is crucial to ensuring the effectiveness of our Hop Count Filtering (HCF) approach.

The critical aspect in both the initialization and updating processes lies in safeguarding the mapping table against invalid entries. During initialization, administrators must provision for new entries to accommodate emerging legitimate IP addresses. Given that hop counts can change due to router instability or network failures, the update function operates periodically, modifying entries only after the completion of the three-way handshake of TCP connections. This stringent approach guarantees the integrity of the mapping table, reinforcing its reliability in countering DDoS attacks.

V. CORRELATION BETWEEN HOP COUNT AND PACKET TRANSFER TIME

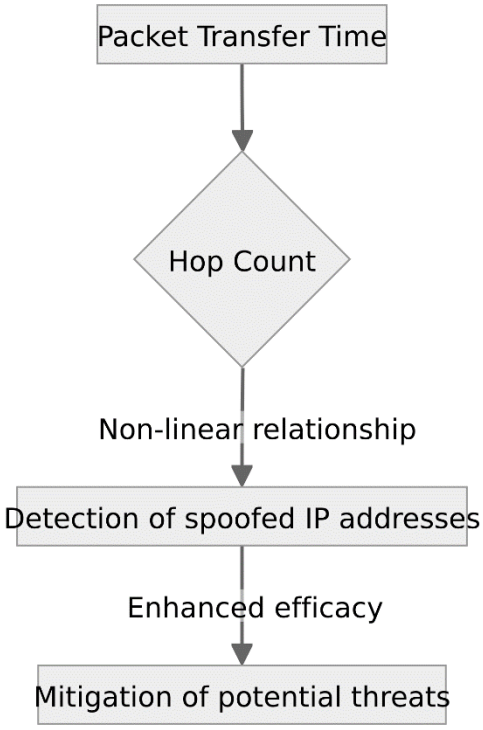

Fig5: Correlation Between Hop Count And Packet Transfer Time

As outlined in [21], a robust non-linear relationship has been identified between hop count and packet transfer time. This correlation has been reaffirmed through various graphical analyses, thereby substantiating the intrinsic association between these two metrics. Consequently, this research leverages the observed relationship between hop count and packet transfer time as integral components for the detection of spoofed IP addresses. The consideration of both metrics enhances the efficacy of our approach in identifying and mitigating potential threats associated with IP address spoofing.

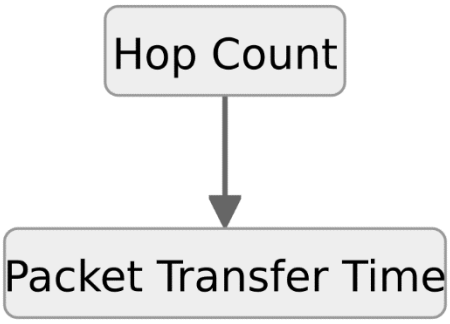

Figure 6. Correlation between Hop Count and Packet Transfer Time

VI. FUZZY MEMBERSHIP FUNCTION-BASED SEARCH STRATEGY FOR SPOOFED PACKETS

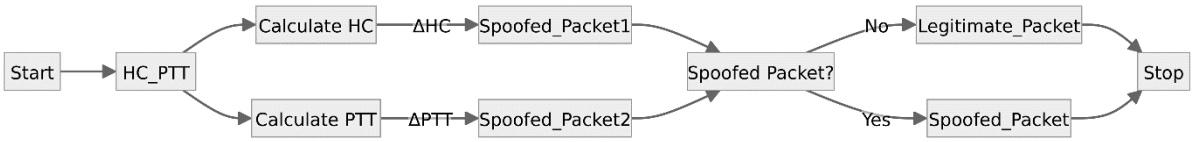

Fig7: Fuzzy Membership Function-Based Search Strategy For Spoofed Packets

This paper introduces an algorithm employing fuzzy operations to identify spoofed packets effectively. The algorithm leverages fuzzy rules, incorporating two distinct membership functions in the strategy for evaluating hop count (HC) and packet transfer time (PTT). Additionally, a heuristic-based fuzzy set approach is implemented to streamline computational complexity.

By the insights from Section V, predefined set values for HC and PTT, denoted as HCset and PTTset, are established. To implement a heuristic fuzzy triangular membership function, incoming packets are segregated based on their source IP addresses. Subsequently, within the sub-table, changes in HC and PTT are computed for every 100 milliseconds, represented as ΔHC and ΔPTT, respectively.

Two distinct fuzzy set models are constructed from the sub-table to identify potential spoofed packets.

A. Fuzzy Set Model for the Variation of Hop Count:

The received hop count value is assessed using a fuzzy triangular membership function, categorizing it as very close, close, or not close to the predefined HCset. Figure 3 illustrates the triangular membership function of HC, accompanied by μset. A minor difference between the response time of packets and HCset yields a higher membership value, and conversely, a substantial difference results in a lower membership value. This linguistic formulation is expressed through a membership function with defined conditions as follows:

IF ΔHC is very close TO HCset THEN μ(ΔHC) is high,

IF ΔHC is close TO HCset THEN μ(ΔHC) is medium,

IF ΔHC is not close TO HCset THEN μ(ΔHC) is low.

This fuzzy set model effectively captures variations in hop count, providing a nuanced assessment to identify potential instances of IP address spoofing.

$$\mu_{HC} = \begin{cases} 1 - \frac{\Delta HC}{HC_{set}} & \text{for } 0 < \Delta HC < HC_{set} \\ 1 + \frac{\Delta HC}{HC_{set}} & \text{for } -HC_{set} < \Delta HC < 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{--- (2)}$$

Where, $\Delta HC = HC - HC_{set}$

B FUZZY SET MODEL FOR THE VARIATION OF PPT

"The received PTT value is assessed using a fuzzy triangular membership function, categorizing it as 'very close,' 'close,' or 'not close' to the PTTset. Figure 4 illustrates the triangular membership function of PTT alongside µset. A minor disparity between packet response times and PTTset yields a high membership value, while a significant difference results in a lower membership value. The corresponding membership function, governed by specific conditions, is presented as follows:"

$$\mu_{PTT} = \begin{cases} 1 - \frac{\Delta PTT}{PTT_{set}} & \text{for } 0 < \Delta PTT < PTT_{set} \\ 1 + \frac{\Delta PTT}{PTT_{set}} & \text{for } -PTT_{set} < \Delta PTT < 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{--- (1)}$$

Where, $\Delta PTT = PTT - PTT_{set}$

"The convergence of µHC and µPTT ultimately classifies the spoofing characteristics of received packets. This classification is expressed as:"

$\mu Dn = \min(\mu HC, \mu PTT)$

Where \(\mu_{Dn} \) represents the fuzziness associated with the nth received packet from the host. Moreover, the union of individual packets' spoofing characteristics yields the overall spoofing characteristic of the host, expressed as:

$\mu D = \max(\mu D1, \mu D2, \ldots, \mu Dn)$

Where μDdenotes the fuzziness associated with the received packets from the host.

VII. RESULTS AND DISCUSSIONS

The efficacy of the proposed scheme has been assessed through testing with neighbouring nodes proximate to the victim server. Following the deliberations in Section V, an initial mapping table has been meticulously constructed. Concurrently, at the victim server, the hop count and packet transfer time of received packets from Host 1 are systematically recorded in Table I at one-second intervals. Analogously, Table II presents analogous details from Host 2.

Given the potential reception of spoofed packets due to the victim server's connectivity with all neighbouring hosts, the tabulated information in these tables encompasses both legitimate and potentially spoofed packets. To discern the latter, a heuristic fuzzy logic approach has been employed. Membership function values of μ HC and μ PTT were calculated at 100-second intervals, with the results tabulated in Table III for Host 1 and Table IV for Host 2. This comprehensive analysis enables the identification of potential spoofed packets amid the received data, shedding light on the robustness of the proposed scheme in discerning malicious activities.

"Tables I and II present a comprehensive overview of received packet details during the initial 100 seconds at the victim server, originating from Hosts 1 and 2. In real-world scenarios, attackers may deliberately alter the source addresses of the packets dispatched from compromised hosts."

TABLE I. PACKETS DETAILS FROM HOST1:

S.No	HOP Count	PTT (ms)
1	13	305.658
2	11	372.308
3	16	318.133
4	15	322.143

TABLE II. PACKETS DETAILS FROM HOST2:

S.No	HOP Count	PTT (ms)
1	14	146.868
2	16	100.778
3	13	90.814
4	15	110.695

From Tables I and II, the changes in hop count (▲HC) and packet transfer time (▲PTT) were computed using the following equations:

$\blacktriangle HC = HC_{current} - HC_{previous}$

$\blacktriangle PTT = PTT_{current} - PTT_{previous}$

These calculations provide insights into the variations in hop count and packet transfer time between consecutive data points.

$$\Delta HC = HC - HC_{set} \quad -- (3)$$

$$\Delta PTT = PTT - PTT_{set} \quad -- (4)$$

Utilizing the values of ▲HC and ▲PTT, the respective membership function values μ_{HC} and μ_{PTT} were computed using the equations (1) and (2). The obtained results are systematically tabulated in Tables III

and IV, providing a comprehensive representation of the membership function values associated with the changes in hop count and packet transfer time.

TABLE III: DETAILS OF μHC and μPTT RECEIVED FROM HOST 1:

S.No	μHC	μPTT	μD
1	0.9285	0.9818	0.9285
2	0.7857	0.8039	0.7857
3	0.9780	0.8571	0.8571
4	0.9285	0.9651	0.9285

These values represent the membership function values μ_{HC}, μ_{PTT}, and μ_{D} associated with the changes in hop count and packet transfer time received from Host 1.

TABLE IV: DETAILS OF μHC and μPTT RECEIVED FROM HOST 2:

S.No	μHC	μPTT	Fuzziness
1	0.8235	0.7787	PS
2	0.9411	0.8380	PS
3	0.7647	0.7551	NS
4	0.8823	0.9204	PS

These values represent the membership function values μ_{HC} and μ_{PTT}, along with the corresponding fuzziness, associated with the changes in hop count and packet transfer time received from Host 2. The fuzziness is categorized as 'PS' (Potentially Spoofed) or 'NS' (Not Spoofed) based on the heuristic fuzzy logic applied.

From Tables III and IV, the proximity of μ_{HC} and μ_{PTT} was determined through the application of min-max computation. The host exhibiting the minimum mutual transfer function between μ_{HC} and μ_{PTT} has been identified as 'NS' (No Spoofing). The remaining packets have been categorized as 'PS' (Partially Spoofed), indicating a potential presence of spoofing characteristics.

VIII. CONCLUSION

The presented spoofing detection method classifies received source IP packets as NS (Not Spoofed), PS (Partially Spoofed), and FS (Fully Spoofed) using a heuristic fuzzy triangular membership approach. Legitimate requests are discerned based on the proximity of μ_{HC} and μ_{PTT} through the application of min-max computation. This comprehensive approach enhances the precision of spoofing identification, contributing to an effective and reliable method for securing network communication against potential malicious activities.

REFERENCES

1. A. Chakrabarti and G. Manimaran, "Internet Infrastructure Security: A Taxonomy," in Proc. Conf. IEEE Network, vol. 16, no. 6, pp. 13-21, 2002.
2. CERT Coordinate Center, "Denial of Service Attacks," [Online]. Available: http://www.cert.org/tech tips/denial of service.html.

3. B. Al-Duwairi and G. Manimaran, "International dropping: A novel scheme for syn flooding Mitigation," in Proc. Conf. IEEE INFOCOM, April 2006.
4. Christos Douligeris and Aikaterini Mitrokotsa, "DDoS Attacks and Defense Mechanisms: A Classification."
5. R. R. Talpade, G. Kim, and S. Khurana, "WOW: Traffic-based Network Monitoring Framework for Anomaly Detection," Proc. 10th IEEE Symposium on Computers and Communications, Red Sea, Egypt, pp. 442-451, June 1999.
6. W. Lee and S. J. Stolfo, "Data mining approaches for intrusion detection," 7th USENIX Security Symposium, San Antonio, TX, pp. 79-93, January 1998.
7. I. B. D. Cabrera et al., "Proactive Detection of Distributed Denial of Service Attacks using MIB Traffic Variables - A Feasibility Study," IFIP/IEEE Int. Symp. On Integrated Network Management, Seattle, WA, May 2001.
8. Y. Huang, J. M Pullen, "Countering Denial-of-Service attacks Using Congestion Triggered Packet Sampling and Filtering," Proc. ICCCN, Arizona, USA, Oct. 2001.
9. T.M. Gil and M Poleto, "MULTOPS: a data-structure for bandwidth attack detection," Proc. 10th USENIX Security Symposium, Washington, DC, pp. 23-38, Aug. 2001.
10. Haining Wang, Cheng Jin, and Kang G. Shin, "Defense Against Spoofed IP Traffic Using Hop-Count Filtering," IEEE/ACM Transactions on Networking, Vol. 15, No. 1, February 2007.
11. S. M. Bellavin, "ICMP traceback messages," Internet Draft, 2001.
12. C. Banos, "A proposal for ICMP traceback messages," Internet Draft, Sept. 2000.
13. H. Burch and H. Cheswick, "Tracing anonymous packets to the approximate source," Proc. USENIX LISA, New Orleans, pp. 319-327, Dec. 2000.
14. R. Stone, "CenterTrack: An IP Overlay Network for Tracking DOS Roods," Proc. 9th USENIX Security Symposium, Denver, Colorado, pp. 199-212, Aug. 2000.
15. S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback," Proc. USENIX Transaction on Networking, vol. 9: (3), pp. 226-237, June 2001.
16. D. X. Sang and A. Penig, "Advanced and Authenticated Marking Schemes for IP Traceback," Proc. IEEE INFOCOM, Anchorage, AK.
17. A. C, Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, and T. Strayer, "Single-Packet IP Traceback," IEEE/ACM Transactions on Networking (TON), vol. 10: (4), pp. 721-734, Dec. 2002.
18. The Swiss Education and Research Network, "Default TTL Values in TCP/IP," 2002, [Online]. Available: http://secfr.nerim.net/docs/fingerprint/en/ttl_default.html.
19. Information Science Institute in University of Southern California, "Internet protocol," RFC 791, 1981.
20. W. Richard Stevens, "TCP/IP Illustrated, Volume 1," Addison Wesley Longman, Inc., 1994.
21. Keita Fujii and Shigeki Goto, "Correlation between Hop Count and Packet Transfer Time."
22. http://www.opus1.com on 12th October 2009.
23. Dr. N. Arumugam Lecturer (SG), International Research Journal of Engineering and Technology (IRJET), Volume: 05 Issue: 07 | July 2018, Classification of IP Spoofed Request using Heuristics Fuzzy Approach in Computer Network (pp:1118-1112)