**DISTRIBUTED DENIAL OF SERVICE (DDOS) DETECTION & MITIGATION BY PREVENTING SYNCHRONIZE (SYN) FLOODING ATTACKS**

1Dr. A. Kannaki @ VasanthaAzhagu, 2Tamalika Singhadar, 3Neha Biswas, 4Lavanya, 5Samapeeka Bepari

1B.E., M.E., B.Ed., Ph.D., 2,3,4,5 B. Tech,

1 Associate Professor and Head

Department of Computer Science and Engineering,

Achariya college of Engineering Technology, Puducherry, India

2,3,4,5 Department of Computer Science and Engineering,

Achariya college of Engineering Technology, Puducherry, India

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Abstract:* This paper introduces a robust Distributed Denial of Service (DDoS) detection and mitigation system employing cuckoo hashing and TCP reset techniques to counter TCP SYN flood attacks. Cuckoo hashing facilitates efficient storage and retrieval of IP addresses, enabling rapid identification of potential threats. The system monitors incoming SYN packets, leveraging cuckoo hashing for quick look-up and identification of suspicious patterns indicative of DDoS activity. Upon detection, the system employs TCP reset packets strategically to mitigate the attack, thwarting malicious connections and preserving network integrity. Experimental results demonstrate the effectiveness of the proposed approach in accurately detecting and efficiently mitigating SYN flood attacks, ensuring uninterrupted service availability and bolstering network security against evolving threats.

*Index Terms* - Distributed Denial of Service (DDoS), TCP SYN flood attacks, Cuckoo Hashing, TCP reset.

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

# **Introduction**

In the landscape of evolving networking technologies, cybersecurity threats pose significant challenges to the integrity and performance of online systems. Among these threats, denial-of-service (DoS) and distributed DoS (DDoS) attacks, such as TCP SYN flooding, UDP flood, Smurf, and ICMPv6-based flooding attacks, have emerged as particularly disruptive. The TCP SYN flooding attack, in particular, leverages the TCP handshake process to overwhelm server memory rapidly. Traditional mitigation strategies often rely on expensive firewalls, but the advent of software-defined networking (SDN) offers novel approaches for combating such threats. SDN, facilitated by protocols like OpenFlow and programmable data planes like P4, provides a standardized framework for separating control and data planes. This separation enables more efficient network management and introduces the possibility of distributing control plane tasks to switches, thereby reducing reliance on central controllers. Leveraging this paradigm shift, P4 emerges as a key tool for implementing innovative mitigation strategies. In this paper, a novel approach combining cuckoo hashing and TCP reset methods is proposed for SYN flooding mitigation. Unlike traditional methods that primarily focus on IP addresses, the proposed scheme records source MAC addresses in both legitimate and illegitimate tables to thwart attackers who frequently change IP addresses. By integrating a detection engine and illegitimate table, attackers can be promptly identified and blocked, rather than merely returning SYN/ACK packets as in conventional TCP reset methods. The contributions of this research are manifold. Firstly, the proposed approach demonstrates a significant reduction in malicious traffic while enhancing detection accuracy and optimizing register usage. Secondly, by focusing on MAC addresses, the method effectively counters attackers attempting to evade detection by IP address manipulation. Finally, experimental validation on the bmv2 P4 software switch showcases the practical viability and superiority of the proposed approach over existing methods, promising enhanced network security in SDN environments.

The remaining part of the paper is organized as follows: Section II describes about the related works for the project. Section III explains the system architecture about how the packet checking process is implemented within the SDN system. Section IV provides comparative analysis with the existing TCP Reset method and the SMDMTS model. We present our conclusion and future work in Section V accompanied with the reference description in Section VI.

# **Related Works**

In today’s digital landscape, cybersecurity threats have escalated in tandem with the widespread adoption of mobile devices and applications. As highlighted by Almaiah et al. [[10]](#_REFERNCES), effectively addressing these threats begins with classification, a crucial step preceding problem-solving. This section delves into various methodologies aimed at combating emerging cybersecurity challenges, particularly focusing on detection and mitigation strategies underpinned by OpenFlow and P4 technologies. One prevalent approach, TCP Reset [[8]](#_REFERNCES), safeguards legitimate packets and facilitates benign client connections through an authentication mechanism independent of controller intervention. However, this method, while effective against SYN flood attacks, inadvertently amplifies network traffic by responding to SYN/ACK packets from attackers, thus necessitating a more refined solution. SAFETY [[11]](#_REFERNCES) addresses this need by dynamically setting thresholds using Shannon’s entropy calculation. Nonetheless, its reliance on controller-driven packet collection inflates the load between controller and switch, undermining scalability. Similarly, SLICOTS [[13]](#_REFERNCES) attempts to alleviate packet ingress frequency by implementing temporary forwarding rules, yet still grapples with increased controller-switch load as traffic surges. To tackle flooding, Malik et al. [[14]](#_REFERNCES) propose the Flooding Factor based Framework for Trust Management (F3TM), leveraging trust values for identifying malicious nodes. Conversely, Sunil et al. [[15]](#_REFERNCES) employ a delimited anti-jammer scheme to pinpoint vehicle locations via anomaly detection in vehicle-to-vehicle communication data, bolstered by the foster rationalizer and morsel supple filter functions. Paolucci et al. [[16]](#_REFERNCES) introduce a P4-based method assigning two registers per IP match table to manage data. However, scalability concerns arise as register demands surge with escalating connection counts, necessitating packet discards beyond set thresholds. Alternatively, the counting Bloom filter (CBF) [[18]](#_REFERNCES) , utilized in some studies [[4,17]](#_REFERNCES), accrues traffic insights at the data plane. Yet, its efficacy diminishes when similar packet rates from attackers and normal users result in indistinguishable counters, leading to confusion between the two. As data exchange burgeons, encryption and blockchain security assume paramount importance. An innovative hybrid encryption approach, ECCHC [[19]](#_REFERNCES), seamlessly integrates elliptic curve cryptosystems and hill ciphers, fortifying data security without key sharing over the Internet. Similarly, Aitizaz et al. [[20]](#_REFERNCES) propose a blockchain mechanism allowing users to encrypt data locally before uploading it to the distributed ledger for heightened security and record-keeping.

# In essence, the evolving landscape of security and trust in data exchange underscores the imperative for robust encryption and blockchain integration. These advancements not only safeguard against attacks but also foster a climate of enhanced security, reliability, and confidentiality in digital transactions.

# **System Architecture**

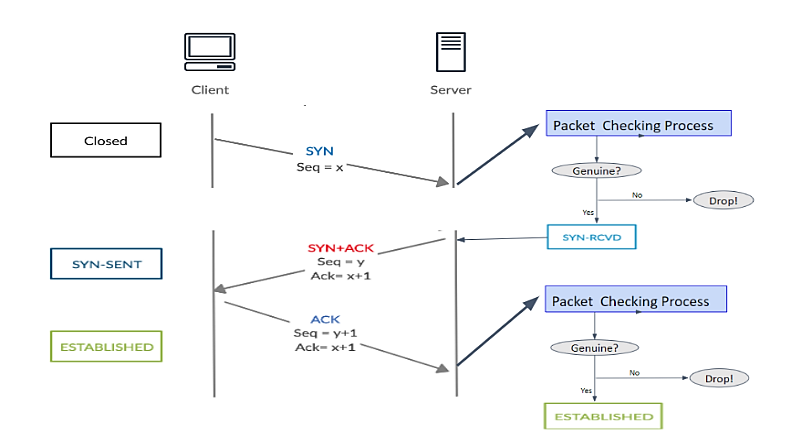
Our proposed system architecture consists of two modules: The Detection Module and The Mitigation Module. As soon as the packet enters the system, it first encounters the mitigation module. The packet's MAC address entry is checked in the illegitimate table, if the entry exists, it is detected as an attacker and the malicious packet is dropped. If the entry doesn't exist, its entry is checked in the legitimate table. If entry exists, the packet is forwarded to the forwarding table else its corresponding TCP flag is checked. If the TCP flag is SYN, add the SYN packet to the check\_syn table. Next, using Cuckoo Hashing technique, find the hash value (key) by taking the source IP address (server IP address) and destination MAC address (client MAC address) of the respective packet as input to the hashing function and producing a hash value (key) as the output. There exists 4 hash tables and 4 hashing functions that has slots where each slot contains a key and a counter. The key comprises of source IP address (server IP address) and destination MAC address (client MAC address). The counter value is initially set to zero. Now the system moves on to the detection module wherein the obtained hash value (key) entry is checked in the hash tables. If the key exists in the hash tables, increment the counter value by one and compare it against the threshold (the maximum number of times connection failures are acceptable, here 5). If the counter value is greater than the threshold specification, alert the SDN controller to add the client MAC address to the illegitimate table and block it henceforth. If the counter value is smaller than the threshold value, the packet is send to the in\_port (client port). If the key does not exist in the hash tables, check if there is any empty slot available. If empty slot is present, add the key to the table and increment counter value by one. If empty slot is not available, send the packet to the in\_port. If the flag is ACK, check for ACK packet entry (corresponding SYN and acknowledge number of the TCP ACK packet) in the check\_ack table. If the entry exists, modify ACK to RST (where RST is the TCP flag to specify a TCP RESET for the connection) through modify\_ack\_to\_rst action. If the entry does not exist, attacker is detected as it is impossible to receive an ACK packet without a corresponding SYN packet. Next, use cuckoo hashing as described above to find the hash value (key). If the key exists in the hash table, clear the slot it is present in using clear\_the\_slot action and add the client MAC address in the legitimate table, then provide the service requested. If the key does not exist in the hash table, send the packet to in\_port. Both check\_syn and Check\_ack table contain a maximum of 256 entries. Provide the legitimate users (clients) with their requested service and add their MAC address to the forwarding table after service has been provided. The hash values are stored in registers.

Figure 1: Packet checking process in TCP 3-way handshake

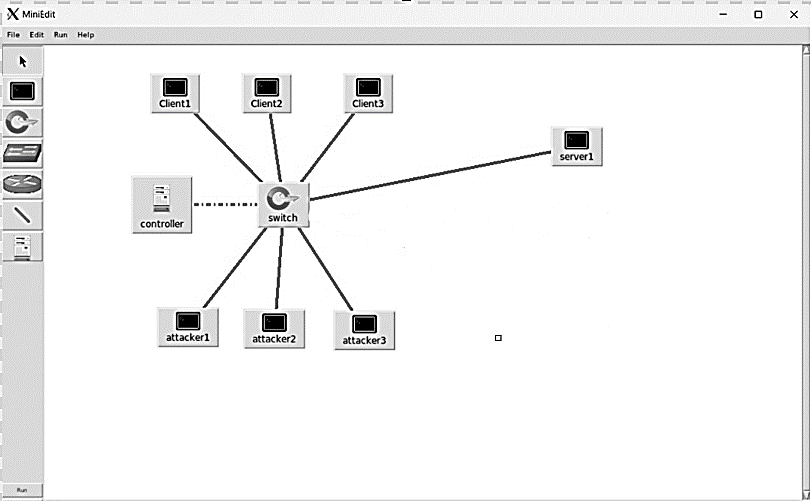
****

Figure 2: Designed network topology

# 

# **Comparative Analysis**

The existing system faces several setbacks, including a large number of modules, which can lead to complexity and potential performance issues. Moreover, the system struggles to effectively differentiate between distributed denial-of-service (DDoS) attacks and flash events using the information entropy method, resulting in inaccuracies in identification. Additionally, it lacks the ability to accurately discern DDoS attacks and flash events, which compromises its effectiveness in threat mitigation. Furthermore, the limited scale of network topology experiments restricts the validation of multiple intrusion detection system (IDS) collaborative parallel detection schemes, limiting the system's scalability and real-world applicability. In contrast, the proposed system boasts several strengths, such as a reduced number of modules, which simplifies implementation and management. Moreover, it demonstrates increased accuracy in detection, enabling more precise identification of DDoS attacks and flash events. Additionally, the proposed system requires minimal setup, making it accessible and practical for deployment across various network environments. Furthermore, its virtual implementability enhances its versatility and adaptability to different infrastructures and operational requirements.

In a SYN flooding attack, the attacker inundates the target host with a high volume of fake SYN packets, overwhelming its capacity to handle legitimate connection requests. To assess the efficacy of our proposed system compared to the TCP reset approach, we conducted experiments simulating this attack scenario. In our experiment, the packet rate of 50 normal users was set at 0.25 (packets per second), while the attacker's packet rate was elevated to 5 (packets per second) over a duration of 30 seconds. The key distinction lies in the response mechanism: while TCP reset continues to send SYN/ACK packets even in the face of malicious SYN packets, our proposed system dynamically adapts by blocking the attacker's MAC address upon detection. Consequently, the proportion of malicious traffic in the network, as measured passing through the switch, significantly diminishes. Specifically, our system demonstrates a remarkable reduction, slashing the percentage of malicious traffic to some extent of that observed under the TCP reset method, highlighting its superior efficacy in mitigating SYN flooding attacks.

# **Conclusion And Future Works**

In conclusion, in the context of preventing TCP SYN flood attacks, this system typically operates in two phases: detection and mitigation. Detection involves monitoring incoming traffic patterns to identify abnormal behavior indicative of an attack. This may include a sudden surge in connection requests from multiple sources or an unusually high number of incomplete TCP handshake attempts. Various statistical and machine learning techniques can be employed for accurate detection. Once an attack is detected, the mitigation phase comes into play. One common strategy is to employ rate limiting or connection throttling mechanisms to filter out malicious traffic while allowing legitimate connections to pass through. This can be implemented at different levels of the network stack, such as at the firewall or load balancer. Another approach involves deploying specialized hardware or software solutions designed specifically for mitigating DDoS attacks. These solutions often utilize sophisticated algorithms and heuristics to identify and block malicious traffic in real-time, effectively mitigating the impact of the attack on the network infrastructure. Additionally, leveraging cloud-based DDoS protection services can provide scalable and resilient defense against large-scale attacks by offloading traffic filtering to specialized scrubbing centers. Overall, an effective DDOS detection and mitigation system combines robust detection mechanisms with adaptive mitigation strategies to safeguard networks against TCP SYN flood attacks and other types of DDoS threats, ensuring uninterrupted service availability for legitimate users. The proposed system ensures to improvise the system accuracy and efficiency with usage and maintains the security of the system.

Adding legitimate and illegitimate tables consume additional space and requires a free timeout for each item in the parameter. If an item in the table has not been accessed for a while, the switch will delete the item. However, if the idle time set is too short, it will cause the controller to update the content frequently. Therefore, an appropriate idle timeout parameter should be set in the future to save space in the table and to avoid frequent updates that can cause a large increase in load between the controller and the switch. In the future, it is expected that the method proposed in this paper can be implemented more effectively by setting the appropriate idle timeout parameter for each item with a real dataset of SYN flooding, or by setting the parameter automatically in a dynamic manner. It may also be possible to estimate the size of the registers required to stop all attackers within a given time by deriving an applicable equation. In the future, it is expected to be implemented on real P4 hardware (e.g., switch with Intel Tofino chip) to verify the method proposed in this paper.

# **References**

1. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. Future Gener. Comput. Syst. 2013, 29, 1645–1660.
2. Kreutz, D.; Ramos, F.; Verı ́ssimo, P.; Rothenberg, C.; Azodolmolky, S.; Uhlig, S. Software-Defined Networking: A Comprehensive Survey. Proc. IEEE 2014, 103, 14–76.
3. Bouyeddou, B.; Kadri, B.; Harrou, F.; Sun, Y. DDOS-attacks detection using an efficient measurement-based statistical mechanism. Eng. Sci. Technol. Int. J. 2020, 23, 870–878.
4. Hill, J.; Aloserij, M.; Grosso, P. Tracking Network Flows with P4. In Proceedings of the 2018 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS), Dallas, TX, USA, 11 November 2018
5. Lin, T.Y.; Wu, J.P.; Hung, P.H.; Shao, C.H.; Wang, Y.T.; Cai, Y.Z.; Tsai, M.H. Mitigating SYN flooding attack and ARP spoofing in SDN data plane. In Proceedings of the 2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS), Daegu, Republic of Korea, 22–25 September 2020; pp. 114–119.
6. Bosshart, P.; Daly, D.; Gibb, G.; Izzard, M.; McKeown, N.; Rexford, J.; Schlesinger, C.; Talayco, D.; Vahdat, A.; Varghese, G.; et al. P4: programming protocol-independent packet processors. ACM SIGCOMM Comput. Commun. Rev. 2014, 44, 87–95.
7. Pagh, R.; Rodler, F.F. Cuckoo hashing. J. Algorithms 2004, 51, 122–144.
8. Luo, L.; Guo, D.; Ma, R.T.B.; Rottenstreich, O.; Luo, X. Network anti-spoofing with SDN data plane. In Proceedings of the IEEE INFOCOM 2017—IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017.
9. Lin, T.Y.; Wang, C.Y.; Tuan, Y.P.; Tsai, M.H.; Chenz, Y.R. A Study on Traffic Asymmetry for Detecting DDoS Attack in P4-based SDN. J. Inf. Sci. Eng. 2022, 38, 1265–1283. . .\_38(6).0009.
10. Almaiah, M.A.; Al-Zahrani, A.; Almomani, O.; Alhwaitat, A.K. Classification of cyber security threats on mobile devices and applications. In Artificial Intelligence and Blockchain for Future Cybersecurity Applications; Springer: Cham, Switzerland , 2021; pp. 107–123.
11. Kumar, P.; Tripathi, M.; Nehra, A.; Conti, M.; Lal, C. SAFETY: Early Detection and Mitigation of TCP SYN Flood Utilizing Entropy in SDN. IEEE Trans. Netw. Serv. Manag. 2018, 15, 1545 – 1559.
12. Shannon, C.E. Prediction and Entropy of Printed English. Bell Syst. Tech. J. 1951, 30, 50–64.
13. Mohammadi, R.; Javidan, R.; Conti, M. SLICOTS: An SDN-Based Lightweight Countermeasure for TCP SYN Flooding Attacks. IEEE Trans. Netw. Serv. Manag. 2017, 14, 487 – 497.
14. Ahmed, M.N.; Abdullah, A.H.; Chizari, H.; Kaiwartya, O. F3TM: Flooding Factor based Trust Management Framework for secure data transmission in MANETs. J. King Saud Univ.-Comput. Inf. Sci. 2017, 29, 269–280.
15. Kumar, S.; Singh, K.; Kumar, S.; Kaiwartya, O.; Cao, Y.; Zhou, H. Delimitated anti jammer scheme for Internet of vehicle: Machine learning based security approach. IEEE Access 2019, 7, 113311–113323.
16. Paolucci, F.; Civerchia, F.; Sgambelluri, A.; Giorgetti, A.; Cugini, F.; Castoldi, P. P4 Edge node enabling stateful traffic engineering and cyber security. J. Opt. Commun. Netw. 2019, 11, A84–A95.
17. Friday, K.; Kfoury, E.; Bou-Harb, E.; Crichigno, J. Towards a Unified In-Network DDoS Detection and Mitigation Strategy. In Proceedings of the IEEE Conference on Network Softwarization (NetSoft), Ghent, Belgium, 29 June–3 July 2020.
18. Luo, L.; Guo, D.; Ma, R.T.B.; Rottenstreich, O.; Luo, X. Optimizing Bloom Filter: Challenges, Solutions, and Comparisons. IEEE Commun. Surv. Tutor. 2018, 21, 1912 – 1949.
19. Almaiah, M.A.; Dawahdeh, Z.; Almomani, O.; Alsaaidah, A.; Al-Khasawneh, A.; Khawatreh, S. A new hybrid text encryption approach over mobile ad hoc network. Int. J. Electr. Comput. Eng. (IJECE) 2020, 10, 6461–6471.
20. Ali, A.; Almaiah, M.A.; Hajjej, F.; Pasha, M.F.; Fang, O.H.; Khan, R.; Teo, J.; Zakarya, M. An industrial IoT-based blockchain-enabled secure searchable encryption approach for healthcare systems using neural network. Sensors 2022, 22, 572.