A P4 Based Approach on Real-Time DDos Attack Detection in FABRIC

Madhav Rajesh
School of Computing and Augmented Intelligence
Arizona State University
Tempe, USA
mrajesh3@asu.edu

***Abstract*—Distributed Denial of Service (DDoS) attacks represent a major threat to network resource availability and can lead to significant disruptions in essential infrastructure systems. Due to the growing sophistication of DDoS attacks, there is a need for effective defense mechanisms that can identify and counter these attacks in real-time. Programmable data planes have emerged as a promising approach for developing DDoS defense strategies. This project focuses on creating a prototype system that combines P4-based packet processing with classification techniques for DDoS attack detection, as proposed by Lapolli et al. The implementation takes place in the FABRIC testbed environment. Using a realistic DDoS attack dataset, we evaluate the performance of the classifier in terms of accuracy, precision, recall, F1-score, and AUC-ROC. Our results demonstrate that the proposed system effectively detects DDoS attacks with high sensitivity (recall) and relatively low false positives. Furthermore, we discuss potential areas for improvement and future research directions to further enhance the performance of our DDoS attack detection and mitigation system. Overall, this study provides valuable insights into the development and evaluation of robust DDoS defense mechanisms.**

I. Introduction and Motivation

Distributed Denial of Service (DDoS) attacks pose a significant threat to the availability of network resources and can cause major disruptions in critical infrastructure systems. The increasing sophistication of DDoS attacks demands the deployment of effective defense mechanisms that can detect and mitigate these attacks in real time. In this context, programmable data planes have emerged as a promising technology for developing DDoS defense mechanisms that can operate at line-rate speeds and scale to handle large traffic volumes.

We used a P4-based real-time DDoS attack detection approach in the FABRIC testbed [1]. P4 [2] is a domain-specific language for programming the data plane of networking devices and allows for flexible and efficient implementation of packet processing functions. We leverage P4 to enable fine-grained network traffic monitoring and efficient filtering of attack traffic.

We implemented a prototype system that combines P4-based packet processing with classification techniques for DDoS attack detection by Lapolli et al (2019) [3]. The system will be deployed in the FABRIC testbed and evaluated using a range of realistic DDoS attack scenarios. The results demonstrated the effectiveness of the proposed approach in detecting and mitigating DDoS attacks in real-time while maintaining high throughput and low latency.

This project hopes to contribute to the ongoing efforts to develop effective and efficient DDoS defense mechanisms using programmable data planes. The proposed approach should be deployed in real-world networks and improve the resilience of critical infrastructure systems against DDoS attacks..

II. Related Work

The literature proposes several approaches for detecting and mitigating DDoS attacks. In this section, the paper discusses some of the related works in this area.

Traditional approaches for DDoS attack detection and mitigation typically rely on network-based [4]–[6] or host-based intrusion detection systems (IDSs) [7]–[9]. Network-based IDSs monitor network traffic for signs of attack, while host-based IDSs monitor system logs and events for signs of compromise. However, these approaches suffer from high false positive rates, limited scalability, and low real-time speed to detect and mitigate attacks.

More recent approaches have focused on using machine learning techniques for DDoS attack detection [10]–[12]. These techniques leverage statistical and behavioural analysis of network traffic to identify anomalous patterns that may indicate an attack. Some studies have used supervised learning algorithms [12], such as neural networks, to classify network traffic as normal or malicious. Other studies have used unsupervised learning algorithms [10], such as clustering, to identify anomalies in network traffic. However, these approaches may also suffer from high false positive rates and limited accuracy, particularly in the case of zero-day attacks.

In the context of programmable data planes, several approaches have been proposed for DDoS attack detection and mitigation using P4 [13] [14] [15]. For example, a recent study proposed a P4-based system that uses machine learning algorithms to classify network traffic as normal or malicious [14]. Another study proposed a P4-based system that uses bloom filters to efficiently filter out attack traffic [15]. These approaches demonstrate the potential of programmable data planes for developing efficient and effective DDoS defense mechanisms.

DDoS attack have been recently increasing in sophistication which can be attributed to several factors such as amplification

attacks [16], adaptive attacks [17], IoT-bbased botnets [18], DDoS-for-hire services [19], and encryted traffic [20], which are designed to bypass traditional security measures and cause more significant disruption to target systems.

In summary, we must develop more advanced and efficient defense mechanisms to meet the increasing sophistication of DDoS attacks, despite the various approaches proposed for their detection and mitigation. The proposed P4-based approach presented in this paper is implemented based on the existing literature and offers a promising solution for real-time DDoS attack detection.

III. Experimental Set-up

A. Dataset

We performed a packet trace-driven evaluation using representative datasets of legitimate and malicious traffic. As legitimate traffic, the CAIDA Anonymized Internet Traces 2016 [21] dataset is used, recorded from high-speed Internet backbone links. The CAIDA DDoS Attack 2007 [22] dataset is also used, which contains attack traffic aimed at depleting a target server's computing resources and saturating its Internet connection. Despite not being recent, the DDoS dataset is renowned for its thoroughness and applicability to assess system performance under attack [3], [13], [23].

B. Testbed

We conducted experiments to evaluate the effectiveness of the Lapolli et al's P4-based approach for real-time DDoS attack detection and mitigation by implementing the prototype system in the FABRIC testbed environment. For the purpose of this experiment, FABRIC testbed provides us with state-of-the-art servers connected by a high-speed network and switch hardware with programmable data planes.

In the process of implementing the model in FABRIC testbed, a straightforward topology (Figure 1) was chosen to facilitate the evaluation of the DDoS attack detection algorithm. This topology features four hosts ($h1$, $h2$, $h3$, and $h4$) and a single switch (s1). The host $h1$ acts as the traffic input to the switch, $h2$ as output for regular packets, $h3$ as output for suspect packets, and $h4$ as statistics output.

The swtich device $s1$ is running a Ubuntu 18.04 image and runs on 8 cores with 16GB of RAM, the P4 program for the classifer is deployed in a dockererized enviorment. The host devices $h1$, $h2$, $h3$ and $h4$ is running a Ubuntu 18.04 image and runs on 8 cores with 8GB of RAM.

C. Attack Scenario and Threat Model

We implemented and trained a statistical model based on IP address Shannon entropy to characterize legitimate traffic and calculate anomaly detection thresholds proposed by Lapolli, Marques, and Gaspary (A. C. Lapolli, J. A. Marques, and L. P. Gaspary, "Offloading real-time DDoS attack detection to programmable data planes")." [3] in the FABRIC testbed. The source code for the classifier is available in the Lapolli's

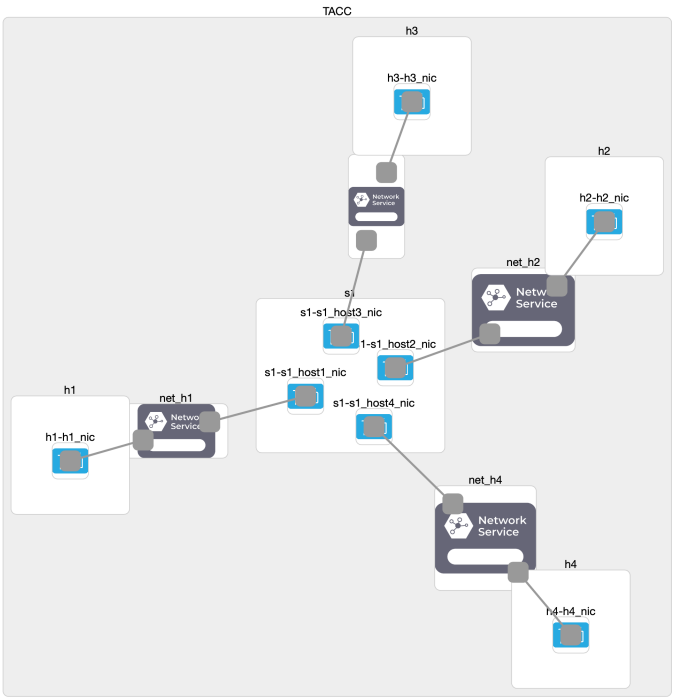

Fig. 1. Network Topology in FABRIC testbed

Github repository [1] and has been used as a starting point for this project.

Distributed denial-of-service attacks encompass a variety of methods aimed at compromising or disrupting externally accessible services. This projects considers a scenario where an attacker can orchestrate a group of hosts distributed across the globe to send illegitimate service requests to a single target. These requests can either overwhelm the target's network with high traffic volume or exploit specific protocol vulnerabilities to consume the target server's computing resources. The attacker may employ spoofed IP addresses to make detecting and characterizing attack packets difficult.

D. Traffic Characterization and Anomaly Detection

High-rate in-network packet processing imposes strict time and memory constraints, necessitating a straightforward yet effective detection strategy. We assume DDoS attacks involve many hosts directing traffic towards one or a few victims, causing source and destination IP address distributions to deviate from legitimate patterns during malicious activities. The idea of calculating Shannon entropy is a proven method for accurately identifying such deviations.

Let X be a set of IP addresses within a total of m packets, and $f_0, f_1, \ldots, f_N$ the frequencies of each unique address. The entropy $H(X)$ of X is given by:

$$H(X) = \log_2(m) - \frac{1}{m}\sum_{x=0}^{N} f_x \log_2(f_x), \qquad (1)$$

[1] Available at https://github.com/aclapolli/ddosd-p4

where the summation $S(X) = \sum_{x=0}^{N} f_x \log_2(f_x)$, represents the entropy norm. The entropy has a negative relationship with the entropy norm. The minimum entropy ($H(X) = 0$) occurs when all IP addresses are the same, meaning there is no diversity or randomness in the IP addresses. In this situation, the traffic is highly predictable, and the entropy is at its lowest value, $zero$. The maximum entropy norm ($S(X) = m \log_2(m)$) is associated with this situation because there is no variation in the frequencies of IP addresses. The IP address with the maximum frequency is the only one in the observation window.

Let us consider an example with an observation window (OW) containing four packets with IP addresses: $\{192.168.1.1, 192.168.1.1, 192.168.1.1, 192.168.1.1\}$. In this case, all IP addresses are the same $(192.168.1.1)$, and the entropy $H(X)$ is at its minimum value, $zero$. The set X contains only one unique source IP address $X = \{192.168.1.1\}$. The frequency of the IP address in X is f_0 (192.168.1.1) = 4 (occurs four times). Now calculating the entropy norm $S(X)$, $S(X) = f_0 \times \log_2(f_0) = 4 \times \log_2(4) = 4 \times 2 = 8$,which is the maximum entropy norm. The entropy value and entropy norm provide information about the distribution of IP addresses within an observation window, which can be used to identify unusual patterns and potential DDoS attacks.

During a DDoS attack, the entropy of source IP addresses is expected to increase as malicious packets introduce new values to the distribution. Conversely, the entropy of destination IP addresses should decrease as the victim becomes a more frequent target. This effect is only noticeable when the number of packets, m, provides a robust representation of the current distributions. However, increasing m results in a higher attack detection latency, as more packets are needed for each observation window measurement. When calculating entropy over a small number of packets, changes in distributions due to malicious traffic may be indistinguishable from short-term fluctuations of legitimate traffic.

Lapolli et al suggests setting dynamic thresholds for source and destination IP address entropies to tackle this trade-off. However for this experiment we uses a preset value of m.

E. Collecting Traffic Statistics

The detection mechanism has been built using the P4 behavioral model reference implementation (BMv2) [24], which has a limited set of processing primitives that mirror the current programmable hardware device's constraints. This subsection discusses overcoming these limitations to achieve real-time DDoS attack detection.

The proposed mechanism's top-level scheme is shown in Figure 2. The P4 program estimates IP address entropies for consecutive segments of the incoming packet stream, referred to as the observation window (OW). At the end of each observation window, traffic characterization units read entropy values to generate a legitimate traffic model. The anomaly detection unit then calculates detection thresholds based on this model and triggers an attack alarm when the latest entropy estimates exceed the thresholds.

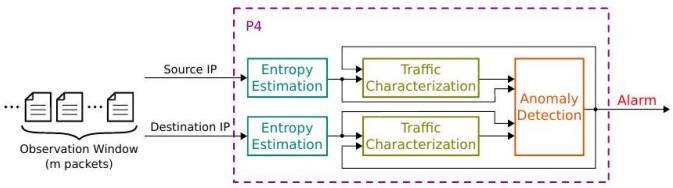

Fig. 2. DDoS Attack Detection Top-Level Scheme [25]

1) Entropy Estimation: Due to the P4 behavioral model's lack of support for the binary logarithm function, the program assumes a fixed (but adjustable) value for the observation window size m, which makes the first term in Equation 1 a constant. Consequently, real-time entropy estimation is reduced to calculating the second term, which depends on the frequencies of each distinct observed address. The model then describe approximating these frequencies from the packet stream while adhering to processing constraints.
2) Traffic characterization: The classifier builds a model of legitimate traffic by processing successive entropy estimates. The entropy time series of source and destination IP addresses are independently summarized in terms of central tendency and dispersion indices. The proposed mechanism updates this model in real-time, with entropy measurements flagged as malicious by the anomaly detection unit being discarded from characterization.
3) Anomaly Detection: In identifying potential DDoS attacks, the anomaly detection process focuses on discovering unusual patterns or behaviors in the frequency of source and destination IP addresses. The detection relies on two conditions:
   a) For source IP addresses: $\hat{H}_n > M_{n-1} + kD_{n-1}$
      If the estimated count of an IP address in the current observation window is greater than the sum of the mean and k times the dispersion index from the previous observation window.
   b) For destination IP addresses: $\hat{H}_n < M_{n-1} - kD_{n-1}$
      If the estimated count of an IP address in the current observation window is less than the difference between the mean and k times the dispersion index from the previous observation window.

$\hat{H}_n$ is the estimated entropy of source (or destination) IP addresses at time window n, M_{n-1} is the mean of the entropy values up to the previous time window $(n-1)$, k is a scaling factor (a constant multiplier), and D_{n-1}: standard deviation of the entropy values up to the previous time window $(n-1)$

A DDoS attack alarm is set off when either of these conditions is met. The k parameter is an adjustable sensitivity coefficient influencing detection thresholds. The effect of k is proportional to the traffic characteristics since it is multiplied by the dispersion index.

Increasing k leads to more stringent detection conditions, which results in higher accuracy when identifying DDoS attacks. However, it also increases the chances of more subtle attacks going undetected. Decreasing k can broaden anomaly detection coverage but may lead to more false alarms.

Lapolli et al [3] provide additional background on Entropy Estimation, Traffic characterization, and Anomaly Detection.

F. P4 Implementation

In this section, we breakdown the key parts of the code

1) Header definitions: Headers are the data structures that define the layout of protocol fields in a packet. This code defines headers for Ethernet, IPv4, and TCP/UDP protocols.

```
header ethernet_t {...}
header ipv4_t {...}
header tcp_t {...}
header udp_t {...}
```

2) Metadata: The code uses metadata to carry information within the switch but not as part of the packet header. In this case, it defines custom metadata containing information about source and destination IP addresses and their frequencies.

```
struct metadata_t {
    bit<32> srcAddr;
    bit<32> dstAddr;
    bit<16> srcPort;
    bit<16> dstPort;
    bit<16> srcFreq;
    bit<16> dstFreq;
}
```

3) Parser: The parser processes incoming packets and extracts the fields from the packet headers. It identifies Ethernet, IPv4, and TCP/UDP headers and pulls their fields into the corresponding header structures.

```
parser MyParser(...) {
    ...
}
```

4) Control block: This block implements the packet processing logic. In this implementation, the MyIngress control block processes the packet headers and updates the metadata fields accordingly.

```
control MyIngress(...) {
    ...
}
```

5) Tables and actions: The code defines tables and their associated actions to perform lookups and modify packets based on specified rules. It defines two tables: srcIpTable and dstIpTable. These tables count the frequencies of source and destination IP addresses. The actions srcIpHit and srcIpMiss increment the source IP address frequency count, while dstIpHit and dstIpMiss do the same for destination IP addresses.

```
table srcIpTable {...}
action srcIpHit(bit<16> cnt) {...}
action srcIpMiss(bit<32> addr) {...}

table dstIpTable {...}
action dstIpHit(bit<16> cnt) {...}
action dstIpMiss(bit<32> addr) {...}
```

6) Deparser: The deparser reassembles the packet after processing. It takes the modified headers and metadata and constructs a new packet for forwarding.

```
control MyDeparser(...) {
    ...
}
```

7) Final composition: The main P4 program comprises the parser, ingress control block, and deparser.

```
V1Switch(...) main;
```

In summary, this BMv2 target switch processes incoming packets, extracts source and destination IP addresses, and updates the frequency count for each address to aid in detecting DDoS attacks.

G. Traffic Generation

In the study, a traffic generation strategy was employed using the TRAFG tool [25] to simulate network traffic for evaluating the performance of the DDoS attack detection algorithm. The study combines the CAIDA Anonymized Internet Traces 2016 [21] and CAIDA DDoS Attack 2007 [22] datasets to generate synthetic workloads. This approach allowed for the creation of realistic traffic patterns and the assessment of the effectiveness of the detection mechanisms in a controlled environment. The key elements of the Traffic Generation strategy using the TRAFG tool are highlighted below.

1) Tool Parameters: The TRAFG tool requires five parameters as input:
   - packet count: the number of packets composing the detection phase
   - attack proportion: the proportion of malicious packets within the overall traffic during the attack
   - legitimate pcap filename: a pcap file containing packet traces of legitimate traffic
   - malicious pcap filename: a pcap file containing packet traces of a DDoS attack
   - output pcap filename: the output filename for the workload pcap
2) Configuration: Various configurations were set up, such as packet rates, packet sizes, and transmission durations, to tailor the traffic characteristics to the testing requirements. Additionally, a choice was made between different traffic distributions, including uniform and exponential, to reflect real-world network conditions better.

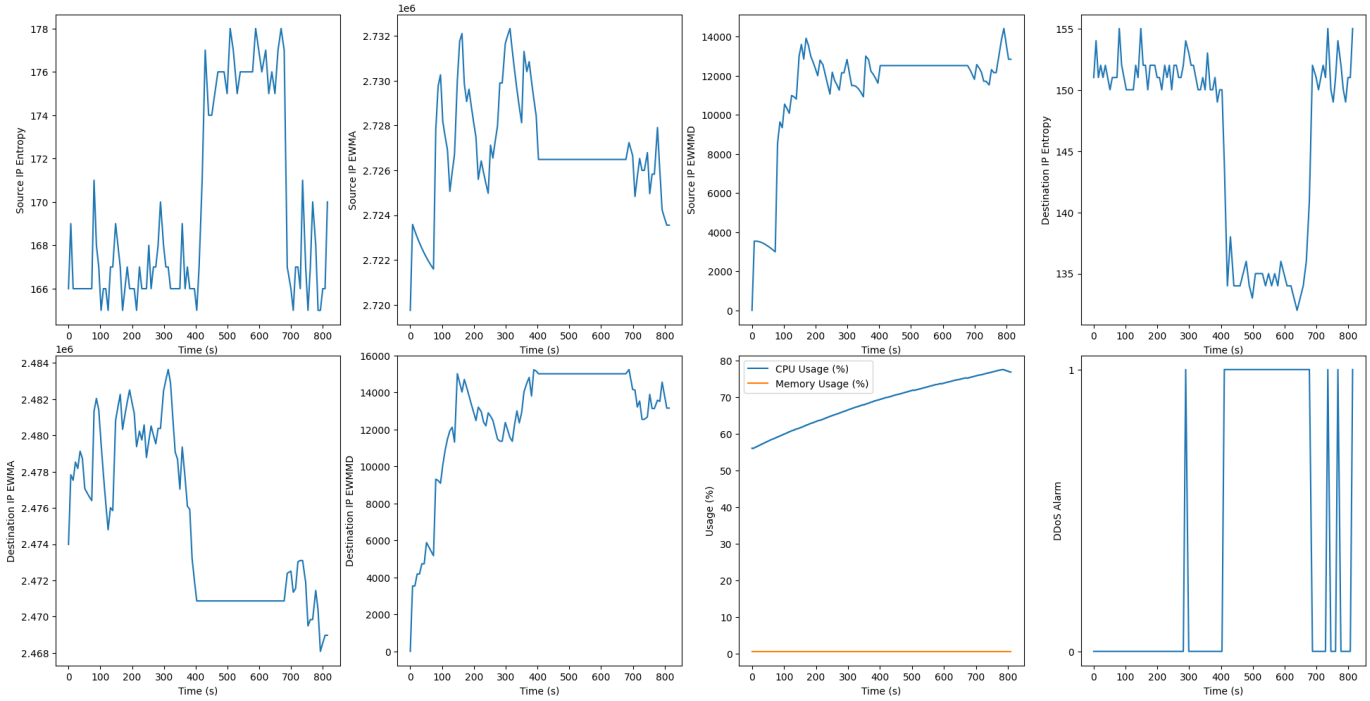

Fig. 3. Switch Statistics

3) Random Number Generation: The code initializes a random number generator to create packet inter-arrival times and sizes based on the chosen distribution. This approach guarantees consistency in the traffic pattern and maintains the desired packet rate throughout the simulation. By utilizing the TRAFG tool and incorporating the above components, the Traffic Generation strategy effectively simulated network traffic to assess the performance of the DDoS attack detection algorithm in the study.

For the purpose of this experiment, a value of 1,900,544 was used. This consist of 116 OWs with each window consisting of 2^{14} packets, 16 OWs were used as training data for the training phase and 100 OWs were used as testing data for validating the results of the classifier. An attack proportion of 0.2 was selected, which signifies that 20% of the traffic consists of malicious packets.

The pcap file generated by the TRAFG tool is used as the workload inputted to the switch by $h1$ (Host 1). The pcap file size was 13 GB and a data size of 10 GB. The capture duration spans 348,322.107 seconds, with the first packet time recorded at 2016-04-06 10:03:00,000000, and the last packet time noted at 2016-04-06 10:08:48,322107. The data byte rate is 29 MBps, while the data bit rate stands at 234 Mbps. The average packet size is 52 bytes, and the average packet rate is 564 kpackets/s.

The pcap file maintains a strict time order and comprises one interface. The information for Interface 0 is as follows:

- Encapsulation = Ethernet (1 - ether)
- Capture length = 1500
- Time precision = microseconds (6)
- Time ticks per second = 1,000,000
- Number of stat entries = 0
- Number of packets = 1,900,544

In order to simulate the DDoS attack using the generated pcap file, the Tcpreplay [26] tool is employed. This tool is designed to replay previously captured network traffic stored in pcap files, enabling the evaluation of various network devices and configurations under realistic traffic conditions.

These statistics and information provide valuable insights into the generated DDoS attack data, which is subsequently used to assess the performance of the implemented classifier in detecting and mitigating DDoS attacks in real-time.

H. Evaluation Metrics

In order to assess the performance of the implemented classifier for real-time DDoS attack detection and mitigation, various evaluation metrics are employed. This section discusses the chosen evaluation metrics and their corresponding mathematical equations.

1) Confusion Matrix: A confusion matrix is a tabular representation of the classification results produced by the model, allowing for an assessment of its performance. It consists of four elements: true positive (TP), false positive (FP), true negative (TN), and false negative (FN). These elements are defined as follows:

- True Positive (TP): The number of instances correctly classified as positive by the model.

- False Positive (FP): The number of instances incorrectly classified as positive by the model.
- True Negative (TN): The number of instances correctly classified as negative by the model.
- False Negative (FN): The number of instances incorrectly classified as negative by the model.

2) Accuracy: Accuracy is the fraction of correctly classified instances out of the total instances. It is calculated as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{2}$$

3) Precision: Precision, also referred to as positive predictive value, measures the proportion of true positive instances out of the total predicted positive instances. It is calculated as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3}$$

4) Recall: Recall, which is synonymous with the true positive rate (TPR), quantifies the proportion of actual positive instances that are correctly identified by the model. It is calculated as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

5) F1-score: The F1-score is the harmonic mean of precision and recall, providing a balance between these two metrics. It is calculated as:

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

6) AUC-ROC: The area under the receiver operating characteristic curve (AUC-ROC) is a performance measurement that evaluates the trade-off between the true positive rate (TPR) and the false positive rate (FPR) across various decision thresholds. A higher AUC-ROC value indicates a better-performing model. The ROC curve is generated by plotting the TPR against the FPR for each threshold, providing a visual assessment of the model's performance. The closer the curve is to the top-left corner of the plot, the better the model's performance.

In conclusion, these evaluation metrics provide a comprehensive assessment of the implemented classifier for real-time DDoS attack detection and mitigation. By analyzing the model's performance using the confusion matrix, true positive rate, false positive rate, accuracy, precision, recall, F1-score, AUC-ROC, and ROC curve, a thorough understanding of its effectiveness under various conditions can be obtained. This, in turn, enables the identification of potential areas for improvement and facilitates the development of more robust DDoS detection and mitigation solutions.

IV. Results

A. Switch Metrics

In this section, we present (Figure 3) various statistics collected by the switch that are instrumental in identifying and understanding the patterns associated with distributed denial-of-service (DDoS) attacks. For the purpose of the experiment a sensitivity coefficient k value of 0.33 was used. These statistics are collected and send to $h4$ (Host 4). The metrics collected are as follows:

1) Source IP Entropy: A metric that quantifies the randomness of source IP addresses in the network traffic. A higher entropy value implies a greater diversity of source IP addresses, which can be an indication of a DDoS attack.
2) Source IP Exponentially Weighted Moving Average (EWMA): A metric that smooths out fluctuations in the source IP addresses by giving more weight to recent observations. This helps in identifying trends and patterns in the network traffic.
3) Source IP Exponentially Weighted Moving Mean Deviation (EWMMD)**:** A metric that measures the variability in the source IP addresses over time, which can indicate the presence of unusual traffic patterns, possibly due to a DDoS attack.
4) Destination IP Entropy: A metric that quantifies the randomness of destination IP addresses in the network traffic. A higher entropy value implies a greater diversity of destination IP addresses.
5) Destination IP Exponentially Weighted Moving Average (EWMA): A metric that smooths out fluctuations in the destination IP addresses by giving more weight to recent observations, aiding in the identification of trends and patterns in the network traffic.
6) Destination IP Exponentially Weighted Moving Mean Deviation (EWMMD): A metric that measures the variability in the destination IP addresses over time, which can reveal the presence of unusual traffic patterns, potentially caused by a DDoS attack.
7) CPU Usage and Memory Usage: A metric that measures the percentage of system CPU resource and memory resource used by the switch while running. The CPU usage increased steadily from 56.1% to 77.9% during the process, however the memory usage remained constant at 0.6%.
8) DDoS Alarm: A binary value indicating whether a DDoS attack has been detected (1) or not (0). By analyzing patterns and correlations between the other metrics and the DDoS Alarm values, we can improve the detection of DDoS attacks and better protect our network infrastructure.

These metrics, when plotted and analyzed in conjunction, provide valuable insights into the network traffic patterns and help in the early detection and prevention of DDoS attacks.

B. Performance Metrics

The given confusion matrix (Figure 4) provides insight into the performance of the classifier. From the matrix, we can observe the following:

1) True Negatives (TN): 605,112 instances were correctly classified as negative.

2) False Positives (FP): 42,084 instances were incorrectly classified as positive.
3) False Negatives (FN): 0 instances were incorrectly classified as negative.
4) True Positives (TP): 80,004 instances were correctly classified as positive.

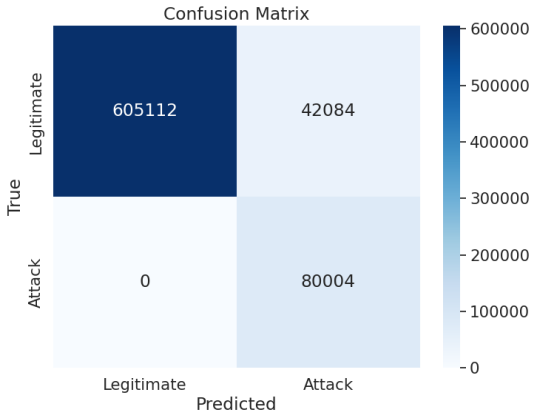

Fig. 4. Confusion Matrix of the classifier

Based on this confusion matrix, the classifier demonstrates high sensitivity (recall) since there are no false negatives. It means that the classifier correctly identifies all positive instances. However, there are some false positives, which might affect the precision of the classifier. Overall, the classifier performs well in detecting positive instances (high recall) but may misclassify some negative instances as positive (lower precision). The low false positive rate enables us to discard suspect traffic with little to no interference on legitimate requests. However, further mitigation strategy like Deep Packet Inspection [27] would be required to further classify True Positive and False Positives.

TABLE I
SUMMARY OF PERFORMANCE METRICS

Metric	Value
Accuracy	0.9421
Precision	0.6553
Recall	1.0000
F1-score	0.7918
AUC-ROC	0.9675

The performance metrics presented in Table I provide insights into the performance of the classifier. Let's interpret each metric:

1) Accuracy (0.9421): The classifier correctly classifies 94.21% of the instances, indicating that its overall performance is relatively high.
2) Precision (0.6553): The classifier has a precision of 65.53%, which means that, out of all the instances predicted as positive, 65.53% are actually positive. The relatively lower precision suggests that the classifier may generate a substantial number of false positives.
3) Recall (1.0000): The classifier has a perfect recall of 100%, meaning that it is able to identify all positive instances without any false negatives. This indicates a high sensitivity in detecting positive instances.
4) F1-score (0.7918): The F1-score, which balances precision and recall, is 0.7918. Although this value is not perfect, it still signifies a reasonably good trade-off between precision and recall, considering the imbalance between these two metrics.
5) AUC-ROC (0.9675): The AUC-ROC value of 0.9675 (Figure 5) is close to 1, which suggests that the classifier performs well in distinguishing between positive and negative instances. A high AUC-ROC value indicates a robust model.

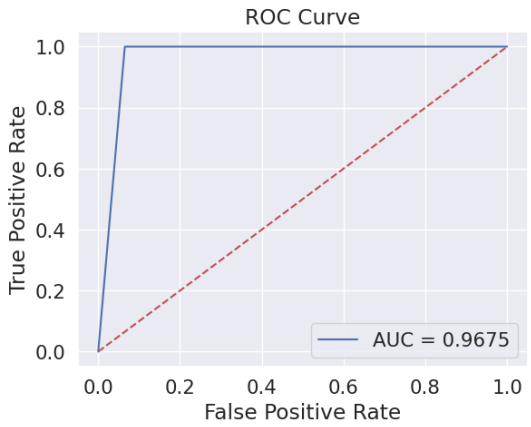

Fig. 5. ROC Curve

In summary, the classifier exhibits strong performance in terms of accuracy, recall, and AUC-ROC. However, it struggles somewhat with precision, which may result in a higher number of false positives. Overall, the classifier performs well, but there is room for improvement, particularly in reducing false positives to enhance precision.

V. SUMMARY, CONCLUSIONS, AND FUTURE WORK

In this study, we presented the development and evaluation of a real-time DDoS attack detection and mitigation system based on P4. The classifier demonstrated strong performance in terms of accuracy, recall, and AUC-ROC, while struggling with precision, resulting in a higher number of false positives.

The system relies on various metrics collected by the switch, including entropy, exponentially weighted moving averages, and mean deviations of source and destination IP addresses. These metrics, along with CPU and memory usage, provided valuable insights into network traffic patterns, enabling the early detection and prevention of DDoS attacks.

The classifier's performance was assessed using a range of evaluation metrics, including accuracy, precision, recall, F1-score, AUC-ROC, and ROC curve. The results indicated a high level of accuracy and recall but a relatively lower precision, suggesting that the classifier may generate a substantial number of false positives. Nevertheless, the overall

performance of the classifier is promising, and the low false positive rate allows for the effective discarding of suspect traffic with minimal interference to legitimate requests.

In conclusion, the proposed DDoS attack detection and mitigation system demonstrates the potential for effectively protecting network infrastructure against DDoS attacks. However, there is room for improvement, particularly in reducing false positives and enhancing precision.

Future work may involve the following aspects:

- Exploring the effect of various variables such as attack proportion, sensitivity coefficient, observation window size, length of training phase etc., on the performance metrics of the classifier.
- Refining the classifier by exploring alternative machine learning algorithms or ensemble methods to improve precision without sacrificing recall or overall accuracy.
- Investigating the use of additional network traffic features and more sophisticated feature selection techniques to enhance the system's ability to detect and mitigate DDoS attacks.
- Evaluating the system's performance under different network traffic conditions and attack scenarios to ensure robustness and adaptability.
- Integrating advanced mitigation strategies, such as deep packet inspection, to further distinguish between true positive and false positive instances.
- Developing a dynamic threshold mechanism for the classifier to adapt to changes in network conditions and maintain optimal performance.

By addressing these potential areas of improvement, we believe that future research can lead to more robust and efficient DDoS attack detection and mitigation systems, ultimately safeguarding network infrastructures and ensuring the smooth operation of critical online services.

REFERENCES

[1] I. Baldin, A. Nikolich, J. Griffioen, *et al.*, "Fabric: A national-scale programmable experimental network infrastructure," *IEEE Internet Computing*, vol. 23, no. 6, pp. 38–47, 2019.

[2] P. Bosshart, D. Daly, G. Gibb, *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[3] Â. C. Lapolli, J. A. Marques, and L. P. Gaspary, "Offloading real-time ddos attack detection to programmable data planes," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, IEEE, 2019, pp. 19–27.

[4] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[5] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *computers & security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[6] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks.," in *Lisa*, vol. 99, 1999, pp. 229–238.

[7] C. C. W. Ko, "Execution monitoring of security-critical programs in a distributed system: A specification-based approach," Ph.D. dissertation, Citeseer, 1996.

[8] D. Wagner and R. Dean, "Intrusion detection via static analysis," in *Proceedings 2001 IEEE Symposium on Security and Privacy. S&P 2001*, IEEE, 2000, pp. 156–168.

[9] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 darpa off-line intrusion detection evaluation," *Computer networks*, vol. 34, no. 4, pp. 579–595, 2000.

[10] Y. Xu and Y. Liu, "Ddos attack detection under sdn context," in *IEEE INFOCOM 2016-the 35th annual IEEE international conference on computer communications*, IEEE, 2016, pp. 1–9.

[11] S. Hosseini and M. Azizi, "The hybrid technique for ddos detection with supervised learning algorithms," *Computer Networks*, vol. 158, pp. 35–45, 2019.

[12] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and A. V. Vasilakos, "An effective network traffic classification method with unknown flow detection," *IEEE Transactions on Network and Service Management*, vol. 10, no. 2, pp. 133–147, 2013.

[13] A. da Silveira Ilha, Â. C. Lapolli, J. A. Marques, and L. P. Gaspary, "Euclid: A fully in-network, p4-based approach for real-time ddos attack detection and mitigation," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3121–3139, 2020.

[14] F. Musumeci, A. C. Fidanci, F. Paolucci, F. Cugini, and M. Tornatore, "Machine-learning-enabled ddos attacks detection in p4 programmable networks," *Journal of Network and Systems Management*, vol. 30, pp. 1–27, 2022.

[15] J. Hill, M. Aloserij, and P. Grosso, "Tracking network flows with p4," in *2018 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS)*, IEEE, 2018, pp. 23–32.

[16] C. Rossow, "Amplification hell: Revisiting network protocols for ddos abuse.," in *NDSS*, 2014, pp. 1–15.

[17] G. Somani, M. S. Gaur, D. Sanghi, M. Conti, and M. Rajarajan, "Ddos attacks in cloud computing: Collateral damage to non-targets," *Computer Communications*, vol. 107, pp. 170–181, 2017. DOI: 10.1016/j.comcom.2017.03.009.

[18] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "Ddos in the iot: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017. DOI: 10.1109/MC.2017.201.

[19] J. J. Santanna, R. van Rijswijk-Deij, A. Sperotto, M. Wierbosch, A. Pras, and C. Hesselman, "Booters: Can anything justify distributed denial-of-service (ddos) attacks for hire?" *Journal of Information, Communication and Ethics in Society*, vol. 13, no. 3/4, pp. 168–178, 2015. DOI: 10.1108/JICES-01-2015-0003.

[20] R. Dubin, G. Jethro, A. Herzberg, and H. Shulman, "Efficient ddos detection for encrypted traffic," in *2017 13th International Conference on Network and Service Management (CNSM)*, IEEE, 2017, pp. 1–9. DOI: 10.23919/CNSM.2017.8256020.

[21] *Anonymized Internet Traces 2016*, https://catalog.caida.org/dataset/passive_2016_pcap, Accessed: 2023-3-1.

[22] *DDoS 2007 attack*, https://catalog.caida.org/dataset/ddos_attack_2007, Accessed: 2023-3-1.

[23] P. Xiao, W. Qu, H. Qi, and Z. Li, "Detecting ddos attacks against data center with correlation analysis," *Computer Communications*, vol. 67, pp. 66–74, 2015.

[24] p4lang, *P4lang/behavioral-model: The reference p4 software switch*. [Online]. Available: https://github.com/p4lang/behavioral-model.

[25] Aclapolli, *Aclapolli/ddosd-cpp: "offloading real-time ddos attack detection to programmable data planes" c++ emulation tools*. [Online]. Available: https://www.github.com/aclapolli/ddosd-cpp.

[26] [Online]. Available: https://tcpreplay.appneta.com/.

[27] E. Özer and M. İskefiyeli, "Detection of ddos attack via deep packet analysis in real time systems," in *2017 International Conference on Computer Science and Engineering (UBMK)*, 2017, pp. 1137–1140. DOI: 10.1109/UBMK.2017.8093526.

APPENDIX

All the source code for P4 program, Python Jupyter notebooks for configuring the FABRIC enviorment, data preprocessing, reproducing the tables and figures in this project are available in the author's Github Repository [2]. The data used for this project can be found at CAIDA's website [21], [22]

[2] Available at https://github.com/maddyrajesh/CSE-534-DDOS