Networked Systems
ETH Zürich — seit 2015

Detecting anomalies with traffic distributions

Bachelor Thesis

Author: Nick Tuninga

Tutor: Alexander Dietmüller, Georgia Fragkouli

Supervisor: Prof. Dr. Laurent Vanbever

March 2023 to June 2023

Abstract

This report proposes a distribution-based method for anomaly detection in network traffic. Its performance is compared to other detection methods on a variety of DDoS attacks. Experimental evaluations conducted on different attack scenarios showcase the effectiveness and potential of the distribution-based approach for detecting network anomalies.

Contents

Chapter 1

Introduction

Anomaly detection in network traffic refers to the process of finding suspicious activity or any other out of the ordinary traffic patterns. It is critical for applications like cybersecurity or intrusion detection. These anomalies can have many origins, including but not limited to network intrusions or attacks, device malfunctions or human error related causes.

The initial step in combating these attacks involves collecting data, which can be obtained from different points within the network infrastructure, such as routers or firewalls. The collected data has to be preprocessed by extracting important information like packet headers and then be formatted into usable input for the detection system.

There are a wide variety of detection methods of which I will be looking at three of them. One class are referred to as rule-based methods. They utilize thresholds or employ other rule-based techniques to flag anomalies. Gaining momentum in recent times is the employment of machine learning and neural network approaches for anomaly detection. These methods have gained significant popularity due to their ability to automatically learn patterns and detect anomalies based on the learned models. By combining the strengths of different detection methods, organizations can improve their ability to detect and respond to network anomalies, mitigating potential threats and enhancing the security of their network infrastructure. Alternitavely, statistical approaches, for instance, operate on the assumption that data points conform to a distribution curve. These methods heavily rely on probability theory to identify deviations.

This report specifically addresses the issue of distributed denial of service (DDoS) attacks. These types of attacks typically involve high volumes of traffic. Furthermore, the detection methods under investigation are specifically designed to gather data at the level of routers.

This report compares three distinct detection methods, each differing at their fundamental core. I evaluated and tested these against a range of different DDoS attacks. By conducting this comparison, the aim is to gain insights into the strengths and weaknesses of each detection method and determine their efficacy in detecting and mitigating DDoS attacks.

Chapter 2

Background

In this section, I provide a brief overview of important networking and anomaly detection concepts, starting with the data and control plane of a router.

The data plane and control plane are two separate components within a router that serve distinct purposes. The data plane is responsible for the actual forwarding of data packets from one network to another. Its primary function is to make decisions on how to direct incoming packets based on their destination addresses. The data plane operates at wire speed, efficiently forwarding packets without requiring much processing power.

The control plane is responsible for managing the overall operation and configuration of the router. It handles tasks such as building and updating routing tables and maintaining network reachability information. The control plane ensures that the data plane has the necessary information to make forwarding decisions accurately. The control plane also handles administrative functions like router management, security configurations, and software updates.

Another important term is the so called flow. It refers to a tuple of a network packets features, which could include IP addresses or ports. The most common flow identifier is the five-tuple consisting of source IP address, source port, destination IP address, destination port and the transport protocol.

When analyzing the performance of the detection methods, it is important to know what true/-false positives/negatives are. When a detection method makes a prediction about an incoming packet and whether it believes it to be an attack packet or not, then that prediction refers to the first term (true or false). In this report a positive will always refer to an attack. The second term (positive or negative) is the true label or groundtruth of that packet. In other words if that packet is actually an attack packet or not. So a correct classification of an attack packet would be a true positive and a misclassification of an attack packet would be a false positive.

A commonly used metric is the true positive rate (TPR), which is a measure of the proportion of actual positive instances that are correctly identified as positive. The false positive rate (FPR) is a measure of the proportion of actual negative instances that are incorrectly identified as positive by the model.

Chapter 3

Dataset and Simulator

The experiments conducted in this report utilize the DDoS CIC-DDoS2019 dataset [3], which is a comprehensive dataset comprising multiple different types of attacks. The captured traffic in the dataset was simulated to represent real-world scenarios. The normal background traffic in the dataset was generated by simulating the behavior of 25 interacting users. The aim was to mimic the characteristics of real-world traffic as closely as possible. This normal traffic served as the baseline for comparison and provided a reference for detecting anomalies. Throughout the duration of the traffic simulation, various attacks were conducted on the network from external sources outside of the 25-user network. These attacks were designed to simulate different types of modern DDoS attacks and serve to test the performance of the detection methods under different attack scenarios.

The CIC-DDoS2019 website [1] provides a comprehensive list of timestamps which specify the start and duration of each attack. Inbetween each attack only benign background traffic should be observed. It's important to note that the timestamps are offset by five hours, necessitating proper adjustment. The simulation and capturing of traffic took place on two seperate days. I only use data of the second day. That dataset consists of 819 equal-sized packet capture files. These files are arranged in chronological order, providing a sequential representation of the captured traffic over time. The capture period on the second day began at 10:35 and concluded at 17:15, encompassing a total duration of six hours and forty minutes of traffic data.

To isolate a specific attack, I extracted and combined the relevant packets based on their timestamps, which constitutes the initial step. The objective was to create a simulator input that includes a ten-minute pre-attack phase consisting solely of benign traffic, followed by the attack itself, and concluding with another phase of approximately ten minutes of benign traffic.

To prepare the input for the detection algorithms, the next step involved extracting the necessary features from the packet headers. I achieved this by passing the packets through a single buffer simulator. This simulator takes in the data as a packet capture file and simulates the traffic as it were to move through a router with a single buffer. As each packet passes thorugh the router, the relevant header information gets extracted and written to a csv. The simulator may drop the occasional packet but this doesn't impact the traffic that I analyse with the detection methods. This step is important as it significantly decreases size of the input file for the detection methods.

However, at this stage the detection algorithm input file was not guaranteed to have only benign traffic at the beginning and end as expected. The implementation of the detection methods had to account for this uncertainty by disregarding attack packets outside the desired time range.

[1] https://www.unb.ca/cic/datasets/ddos-2019.html

Limitations The network traffic was simulated in a way that all malicious packets had the same source IP and the destination IP was part of the same /24 network. When using the dataset one has to be aware of the fact that even during times of no logged attack, that is between attacks when only background benign traffic should be encountered, there might still be one ongoing.

Chapter 4

Detection methods

4.1 Jaqen

One of the simplest and straightforward detection methods is the rule-based approach that utilizes counters. Jaqen [2] combines threshold based detection in the data plane with additional processing and attack mitigiation in the control plane.

When estimating the attack, it considers factors such as attack volume and type based on data collected from the data plane. This information is then transmitted to the control plane, where the appropriate countermeasures are calculated and implemented. The control plane can only be as good as the information it receives from the data plane. This is important, since the control plane defines a signature which should match only packets of the ongoing attack and has to be accurate in order not to count unnecessary packets.

The detection involves assigning a counter to each flow, which keeps track of the number of packets encountered within that flow, given the signature assigned by the control plane matches. It then checks whether the counter surpasses a predefined threshold, and if it does, the flow is dropped. These counters are reset after a certain period of time.

This rule-based method can be highly effective in thwarting volume-based attacks, particularly those with minimal flow variation. By setting appropriate thresholds, it becomes possible to detect and mitigate such attacks efficiently.

However, this report implements Jaqen without the additional attack estimation, focusing solely on the use of flow counters. Although this implementation may lack the advanced attack estimation capabilities, evaluating its effectiveness will provide valuable insights into its overall detection performance.

The performance of counter-based methods depends on several parameters which include the selection of features that constitute the flow, the threshold value that determines when a flow should be dropped, and the reset period of the counters. The choice of features incorporated in the flow directly impacts the number of counters employed. Reducing the number of features decreases the number of counters required. However, this may also lead to an increased benign drop rate, since the algorithm has less information to tell malicious packets from benign packets apart. But adding useless features will most likely spread the attack traffic across too many counters, reducing the permormance. Nevertheless, if the right features are excluded, this adjustment can effectively capture more malicious traffic than without the exclusion.

The threshold value plays a crucial role in determining the trade-off between false positives and false negatives in the detection process, which I investigate in chapter 5. Adjusting the reset period of the counters is particularly effective in detecting attacks characterized by short inter-arrival times between packets. By resetting the counters more frequently, the method becomes more effective at identifying such attacks. However, it is important to note that this may also result in slower attacks potentially slipping through undetected. These parameters should be adjusted accordingly to accurately mitigate attacks and align with specific traffic preferences.

4.2 ACC-Turbo

ACC-Turbo (Aggregate-Based Congestion Control) [1] utilizes a more advanced approach on the data plane, going beyond simple counters. Instead, it employs clusters to analyze incoming packets. During the setup phase, a predetermined number of clusters are selected by the user. During runtime each cluster is assigned a unique priority based on its size. The priority is inversely proportional to the throughput per cluster. In other words, clusters with higher throughput receive lower priority. Each cluster establishes a range for each feature. Let's consider an example with the "time to live" feature. A cluster may define its range for the "time to live" feature to be between 60 and 64 seconds. When a packet enters the switch, its feature values are compared to the feature ranges of each cluster. If the packet's "time to live" value falls within the range of that cluster, the packet is assigned to it. If more features are chosen to be part of the clustering process, the above proccedure is repeated for ech feature. However, if the packet's feature values do not fall within the range of any existing cluster, the distances between the packet and all clusters are calculated. The cluster that is closest to the packet, in terms of feature values, is identified. The feature ranges of this closest cluster are then updated to accomodate the new packet leading to the expansion of the cluster.

Unlike Jaqen, which drops attack packets, ACC-Turbo follows a congestion control algorithm where the primary objective is to maximise the flow of benign traffic while simultaneously deprioritizing the flow of malicious traffic without dropping it, as long as it doesn't interfere with the benign traffic. By deprioritizing attack packets instead of dropping them, ACC-Turbo aims to strike a balance between effectively managing congestion caused by malicious traffic and allowing the smooth flow of legitimate traffic. This approach ensures that the detection method focuses on maximising the overall throughput of benign traffic while minimising the impact of potential attacks on legitimate traffic.

In the ACC-Turbo detection method, there are several adjustable features that can impact its performance. The most important include the number of clusters, the reset period of the clusters, and the selected features. Unlike Jaqen, reducing the number of features in ACC-Turbo does not directly reduce the number of clusters. Instead, it reduces the dimension of the feature space. It is important to note that removing features can potentially lead to inaccurate clustering, as a packet that was initially distant from a cluster could end up at the center of that same cluster when a feature is excluded. However, there may be cases where prior knowledge suggests the importance of specific features. In such scenarios, excluding less relevant features could result in improved clustering accuracy.

Regularly resetting the clusters is crucial to prevent the formation of one oversized cluster that could compromise the effectiveness of the method. This is because having one big cluster will take away the space of the other clusters making it more likely that any packet, malicious or not, will be assigned to the big cluster. By resetting the clusters, the system maintains a balanced distribution of packets among the clusters. Varying the number of clusters has a similar effect to varying the number of counters when using Jaqen.

In comparison to Jaqen, one notable distinction of ACC-Turbo is that it doesn't require the installation of specific countermeasures to mitigate detected attacks. This characteristic potentially enables ACC-Turbo to react more swiftly in response to detected anomalies. However, it is important to emphasize that the focus of this report is primarily on the detection performance of the methods, rather than their reaction speed. By prioritizing the detection performance, the report aims to evaluate and compare the effectiveness of Jaqen and ACC-Turbo in identifying and distinguishing between benign and malicious network traffic.

4.3 Distribution

The distribution-based method offers another approach to detecting anomalies in network traffic. In this method, it is necessary to establish a baseline of what constitutes normal benign traffic whereas any significant deviation from this baseline could potentially indicate an attack.

I first lay out a simple method that utilizes the foundation of a distribution approach and discuss what challenges it may face. I will further propose another statistical method that hopes to solve some of these challenges.

Since this is a distribution-based method, the algorithm needs the mentioned baseline for normal traffic. This is obtained by undergoing a training or fitting phase. During this phase, certain packet features are extracted from collected data and fitted to a distribution. The resulting distribution represents the frequency of encountered flows, measured as encounters per total packets. This distribution represents the expected pattern of benign traffic that the router or detection system anticipates.

An initial approach could involve fitting the benign traffic data to a distribution and assigning an attack probability to each incoming packet based on its deviation from the learned distribution. However, a challenge with this approach is that it treats a single outlier with the same weight as a high volume of outliers. This is problematic because one of the main characteristics of a Distributed Denial-of-Service (DDoS) attack is its massive volume of traffic. Treating all outliers equally may lead to false positives or insufficient detection sensitivity.

To address this challenge, one potential solution is to use a window approach. This involves fitting a distribution within a specified window of time and calculating the deviation from the learned distribution within that window. By focusing on a specific timeframe, it becomes possible to detect not only the presence of an attack but also the intensity or volume of the attack. However, this approach primarily determines whether an attack is ongoing or not, rather than providing detailed information about the individual probability of attack for each packet.

The main challenge is that the method is not able to use traffic volume as information to detect anomalies. I propose here a method that hopes to solve this challenge. The implementation follows the following steps. Firstly, a fitting phase is conducted to establish the baseline distribution, which serves as the reference for normal benign traffic. This is done the same way as with the previous method. This fitting phase is crucial as it forms the foundation of the method.

As each packet enters the switch, it is added to the fitted distribution by adjusting the values of each flow, then the deviation from the fitted distribution is calculated. This deviation is then compared to the deviation that would occur if no packet from that flow had ever entered the switch after the training phase. By comparing these deviations, it becomes possible to assess whether the incoming packet exhibits anomalous behavior. If the deviation of the packet exceeds a certain predefined threshold, indicating a significant deviation from the expected distribution, the packet is dropped. This mechanism allows for the detection of anomalies based on both the volume of the packets and the probability of encountering such packets within the fitted distribution.

To maintain a balance between detection sensitivity and false positives, the accumulated distribution is periodically reset to the initial fitted distribution. This reset ensures that the algorithm does not overly penalize benign traffic and avoids excessive dropping of legitimate packets. However, one has to keep in mind that reducing the reset time too much can result in decreased sensitivity to volume-based attacks, as the algorithm may become less responsive to sudden increases in traffic volume.

The deviation measure used in this method is the Jensen-Shannon divergence. This divergence is a symmetrized version of the Kullback-Leibler divergence, which is a measure of the difference between two probability distributions. The Jensen-Shannon divergence quantifies the similarity or dissimilarity between two distributions, taking into account both their commonalities and differences. The Jensen-Shannon divergence (JSD) between two probability distributions P and Q is defined as:

$$JSD(P\|Q) = \frac{1}{2}KL(P\|M) + \frac{1}{2}KL(Q\|M)$$

where $KL(P\|Q)$ represents the Kullback-Leibler divergence between distributions P and Q, and M is the average distribution:

$$M = \frac{1}{2}(P + Q)$$

The JSD ranges between 0 and 1, where 0 indicates that the distributions are identical, and 1 indicates that the distributions are completely dissimilar.

Chapter 5

Evaluation

In this section I compare the detection performance of the three introduced methods. I ran two different attacks through each detection method. One is a high volume attack and one a low volume attack. The comparison consists of showing how well the approaches were able to set benign traffic from attack traffic apart by analysing how often they mistaken a benign packet for a malicious packet and vice versa.

The performance of a binary classifier is commonly assessed by plotting the false positive rate against the true positive rate, which is known as the Receiver Operating Characteristic (ROC) curve. In the case of Jaqen, a data point is obtained for each threshold value. When the threshold is set to zero, all malicious packets are correctly identified, but all benign packets are also classified as malicious, resulting in a false positive rate (FPR) and true positive rate (TPR) of one. Conversely, setting the threshold to infinity ensures that no packets, whether malicious or benign, are classified as malicious, leading to an FPR and TPR of zero. By definition any point on the ROC curve where the TPR is equal to the FPR is a result of random guessing, indicating poor classifier performance. The objective is to approach the upper left corner of the graph, which corresponds to the point (0,1), indicating a high true positive rate and a low false positive rate. By plotting a data point for each threshold value and connecting them, we can visualise the ROC curve.

5.1 Experiment setup

The simulator takes in the output of the pcap converter with all necessary packet information. The simulator then loops through each packet extracting the selected features. In this case they are source IP, destination IP and destination port. The true label for each packet was easily determined since all attack packets have the same unique source IP.

5.1.1 ACC-Turbo setup

Calculating the TPR and FPR works well for Jaqen and the distribution method, whose goal it is to detect and drop all malicious traffic. But since ACC-Turbo is used for congestion control, its primary goal is to maximise benign throughput regardless of how much malicious traffic gets through.

In order to put ACC-Turbo on the ROC plane we need a way to define the TPR and FPR that makes for a sensical comparison. To determine the true positive rate (TPR) and false positive rate (FPR), one approach is to run a simulation with a rate limit and measure the percentage of benign and malicious traffic that successfully passes through. This allows us to define the TPR as one minus the percentage of malicious traffic that makes it through the classifier, and the FPR as the percentage of benign traffic that is incorrectly classified as malicious. It's worth mentioning that in this context, the FPR may not hold significant importance. The reason for this is that the focus is primarily on accurately detecting malicious traffic (achieving a high TPR) while minimising false negatives.

A rate limit of 80 Mbps was set for the simulation. In comparison, peak attack traffic ran at 1 Gbps. This rate limit is far lower that the attack throughput and high enough for peak benign throughput.

As each packet enters the router and the rate limit is reached it gets assigned to a buffer depending on the packet priority. The buffer size is 10MB in total. If there is space again packets from buffers are taken first. In the following runs six clusters were used and were reset after one second. These values don't hold any significance but seemed to work well for the attacks.

5.1.2 Jaqen setup

The flow of each packet is the tuple of the selected features (source IP, destination IP, destiantion port) and the counters were reset after five seconds. The TPR and FPR were calculated per definition.

5.1.3 Distribution setup

The input was the same file as with the other two methods. The training phase was the pre attack stage, where only benign traffic is encountered. The experiments fit a distribution to the flow. The distribution was reset every half second. This reset time worked well for the attacks and was chosen through trial and error. The TPR and FPR were calculated per definition. A false positive would encompass a benign packet that exceeds the threshold and gets dropped.

5.2 Results

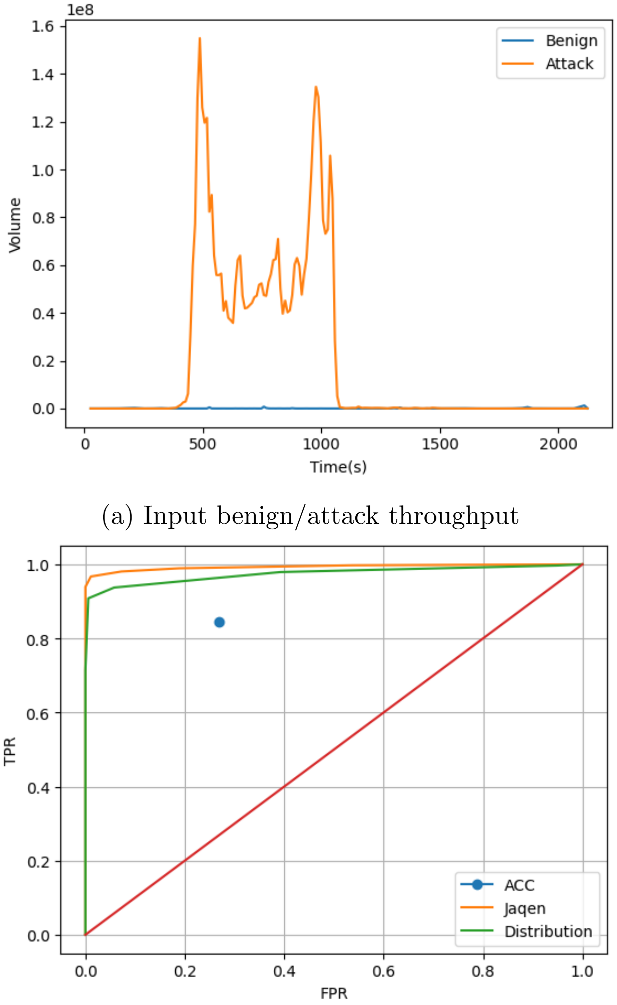
Throughout the experiments, all configurations remained consistent for the different attacks, except for the duration of the pre-attack benign phase. This means that some attacks have a longer or shorter period of benign only throughput before and after the attack. These phases are observable in figures 5.1a and 5.2a.

Although there is a slight variation in the duration of this phase among the attacks, it is not expected to have a significant impact on the overall results, since mitigating the attack phase is the hardest and most important part of the detection.

Figure 5.1a shows the input traffic over time. The traffic is seperated into benign (blue) and attack (orange) throughput. The volume is proportional to the length of the packet. Figure 5.1b shows the ROC plot of all detection methods and underlines their detection performance. The x-axis is the FPR, which can be interpreted as percentage of benign traffic dropped. Conversely, the y-axis represents the TPR or the percentage of attack traffic dropped.

Regarding the detection performance Jaqen takes the lead, with the two other methods lacking behind. This may be because there isn't a lot of flow distinction within the attack, since the dataset is limited that way.

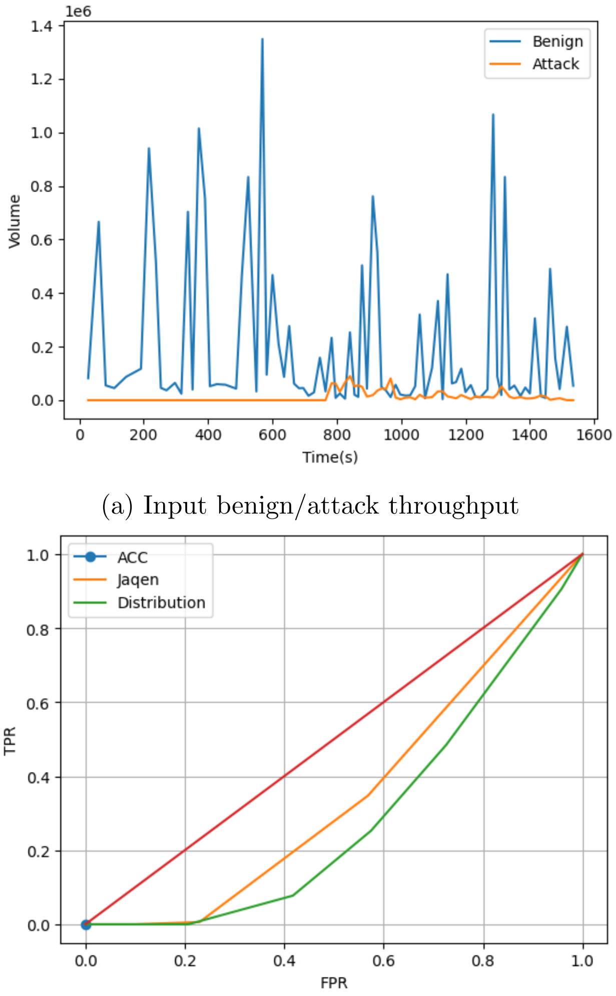
With this evaluation I only investigate the detection accuracy. However a real world switch with a limited buffer might show other results, since Jaqen would fill up the buffer with attack packets of a new flow, which will lead to the dropping of a benign packet, even if Jaqen is confident that packet is legitimate. ACC-Turbo wouldn't have this problem, because higher priority packets



(a) Input benign/attack throughput

(b) ROC of the methods

Figure 5.1: SSDP Attack



(a) Input benign/attack throughput

(b) ROC of the methods

Figure 5.2: WebDDoS Attack

can't be starved by lower priority packets in the buffer. The Distribution approach would probably run into the same issues as Jaqen, where the attack packets are able to starve benign traffic in the buffer.

Figure 5.2a shows the input traffic for the WebDDoS attack. This attack has a low volume of attack throughput. This is observable by looking at the orange curve. Figure 5.2b shows the ROC for the WebDDoS attack. Unsuprisingly, ACC-Turbo is in the bottom left, which means it didn't drop any packets. This makes sense, since the attack throughput doesn't interfere with the benign traffic. This however doesn't tell us much about its ability to tell attack form benign apart.

Both the Jaqen method and the distribution method exhibit a phenomenon where they drop more benign traffic than attack traffic. This is evident from the position of their curves below the diagonal line. Despite this observation, it provides valuable insights into the algorithms' ability to distinguish between the two types of traffic. The closer the plot hugs the bottom right corner, the better the method is at differentiating between benign and attack traffic.

It is expected that these methods drop more benign packets than attack packets since they are designed to react to the volume of traffic, and in this case, the benign traffic has a higher volume.

Challenges of the detection methods The methods obviously face some challenges that I will discuss here. In order to be performant, Jaqen has to set the right threshold beforehand. However, Jaqen's algorithm is not a one-size-fits-all approach. The optimal threshold for each specific attack may vary, requiring constant readjustment to ensure optimal performance. Similarly, the distribution-based approach encounters the same challenge, as the best threshold may differ for different attacks.

In contrast, ACC-Turbo's clustering method does not rely on fixed thresholds. This characteristic of ACC-Turbo's method can provide advantages in terms of adaptability and performance. If the attack however constists of a single or very few flows Jaqen will outperform the other methods.

If there is no ongoing attack and the link is not operating at its capacity, ACC-Turbo will not drop any packets, whereas Jaqen and the distribution method may still drop packets depending on the traffic conditions.

The requirement of a training phase to fit the data into a distribution can pose a vulnerability. In some cases, if attacks manage to infiltrate during the training phase, they may remain undetectable.

Chapter 6

Outlook

A distribution-based method offers a unique approach to detecting anomalies in network traffic. By establishing a baseline distribution of normal benign traffic, it provides a reference point for comparison. This method considers the deviation of incoming packets from the learned distribution to identify potential attacks. It takes into account both the volume of packets and the probability of encountering such packets within the fitted distribution.

While the distribution-based method offers valuable insights into network traffic analysis, it does face certain challenges that can be addressed through future optimizations.

One optimization is to extend the training phase and update it regularly. By collecting more extensive and up-to-date data during the training phase, a better baseline of legitimate network traffic can be established. This helps improve the accuracy of anomaly detection by ensuring that the learned distribution accurately reflects the current traffic patterns.

Another optimization involves considering alternative divergence measures for calculating the deviation. While the Jensen-Shannon divergence is commonly used due to its ability to capture both similarities and differences between distributions, alternative measures can be explored for faster computations or improved performance. Selecting a divergence measure that aligns well with the specific requirements and constraints of the detection system can enhance its efficiency and effectiveness.

These optimizations can help refine the distribution-based method and address some of its challenges. By improving the training phase and considering alternative divergence measures, the method can become more robust, accurate, and efficient in detecting anomalies in network traffic.

When used alongside other detection methods, the distribution-based approach can complement their strengths. It provides an additional layer of analysis by assessing the overall traffic pattern and by adjusting the way the deviation is calculated it could become more sensitive towards new IPs or devices and less so to traffic volume. This can help uncover anomalies that may not be easily captured by other methods, such as pattern-based or threshold-based approaches.

By integrating the distribution-based method with existing detection mechanisms, it may enhance the overall detection accuracy and robustness. The combination of different approaches allows for a more comprehensive understanding of network traffic behavior and improves the ability to detect various types of attacks.

Bibliography

[1] Gran Alcoz, A., Strohmeier, M., Lenders, V., and Vanbever, L. Aggregate-based congestion control for pulse-wave ddos defense. In *SIGCOMM '22: Proceedings of the ACM SIGCOMM 2022 Conference* (New York, NY, 2022-08), Association for Computing Machinery, pp. 693 – 706. 36th ACM SiGCOMM Conference (SIGCOMM 2022); Conference Location: Amsterdam, Netherlands; Conference Date: August 22-26, 2022; Conference lecture on August 26, 2022.

[2] Liu, Z., Namkung, H., Nikolaidis, G., Lee, J., Kim, C., Jin, X., Braverman, V., Yu, M., and Sekar, V. Jaqen: A High-Performance Switch-Native approach for detecting and mitigating volumetric DDoS attacks with programmable switches. In *30th USENIX Security Symposium (USENIX Security 21)* (Aug. 2021), USENIX Association, pp. 3829–3846.

[3] Sharafaldin, I., Lashkari, A. H., Hakak, S., and Ghorbani, A. A. Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In *2019 International Carnahan Conference on Security Technology (ICCST)* (2019), pp. 1–8.

Appendix A

My Appendix

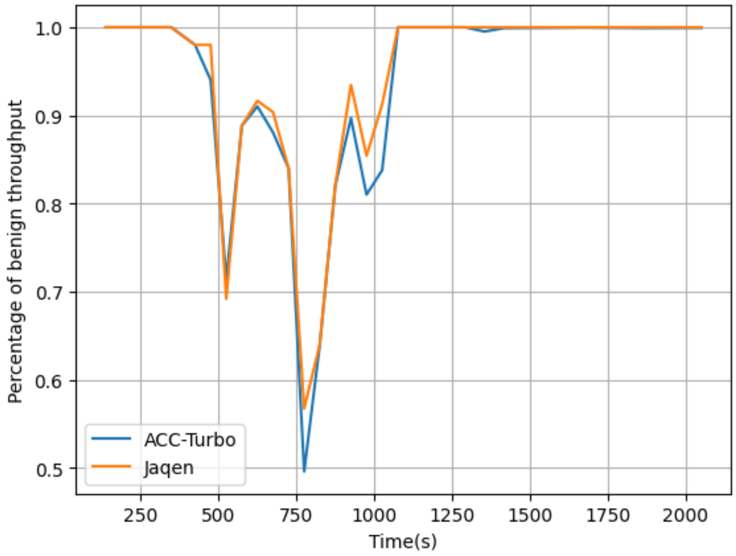


Figure A.1: Fraction of benign traffic for the SSDP attack

Both Jaqen and ACC-Turbo are implemented with the same rate limit and the benign traffic drop percentage is plotted over time. Figure A.1 shows the discussed starvation of benign traffic in the Jaqen algorithm. ACC-Turbo and Jaqen perform similarly when both have a rate limit implemented.