DDOS DETECTION AND MITIGATION USING MACHINE LEARNING

By

ARPIT RAMESH GAWANDE

A thesis submitted to the

Graduate School - Camden

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Scientific Computing

Written under the guidance of

Dr. Jean-Camille Birget

and approved by

Dr. Jean-Camille Birget

Dr. Sunil Shende

Dr. Suneeta Ramaswami

Camden, New Jersey

May 2018

THESIS ABSTRACT

DDOS DETECTION AND MITIGATION USING MACHINE LEARNING

By

ARPIT RAMESH GAWANDE

Thesis Director:

Dr. Jean-Camille Birget

Distributed Denial of Service (DDoS) attacks are very common now days. It is evident that the current industry solutions, such as completely relying on the Internet Service Provider (ISP) or setting up a DDoS defense infrastructure, are not sufficient in detecting and mitigating DDoS attacks, hence consistent research is needed. In this thesis we first tried to understand how DDoS attacks happens then we discussed a way to detect DDoS attacks using machine learning tools at the routers, instead of setting a centralized analysis system. We have proposed a standard communication architecture which can be used across all the networking devices for mitigating DDoS attacks. We have also created a simulation program to demonstrate our detection technique.

Contents

1 Understanding a DDoS attack

1.1 What is a DDoS attack?

A Distributed Denial of Service (DDoS) attack is a way to jam a host network or its resources with a large number of data packets[1] [1] —or connections, so that the host becomes disabled. There are different types of DDoS attacks such as : 1. Volume-based, e.g., SYN Flood Attacks, in which the victim is flooded with a high volume of Transmission Control Protocol (TCP)/ User Datagram Protocol (UDP) packets or connections. 2. Application-based, in which an application such as Domain Name Service (DNS), Voice over Internet Protocol (VoIP), or Hypertext Transfer Protocol (HTTP) are attacked. 3. Low-rate, in which the attacker exploits a vulnerability in the application design, e.g., Slowloris. [2]

1.2 Challenges in dealing with a DDoS attack

Many DDoS attacks happen every day [3]. Some are reported on the news, while many remain unnoticed. The real challenge in detecting and defending the DDoS attack is its dynamic nature. The source[2] is not just a single node or a system on the Internet. There can be many systems participating in a DDoS attack, and often these systems are distributed over different regions of the Internet. Also, the source of the packet is often spoofed[3] [4]. The original attack source is changed in a spoofed data packet, which makes it harder to know the actual IP address of the system from where the attack has originated. In addition oftentimes the source system itself is not aware that it is compromised, and it is being used as a bot [5] by an attacker to launch a DDoS attack.

As the source address can't be a reliable way of knowing the attack source

[1] A messages sent on the Internet is broken into shorter messages for transmission. These short messages are called packets. Packet term was coined by Donald Watts Davies.

[2] The source is a system/device on the Internet that has an IP address and that is under a DDoS attack

[3] Spoofing is the way to change the source IP address of the message. This is a known issue in the protocol itself, not in the implementation

(because of spoofing), detecting and mitigating the attack at the destination[4] is not very useful. The destination may know that the attack is happening but to stop it happening it will have to block all the incoming traffic including the legitimate traffic. To avoid blocking legitimate traffic, companies, such as Cisco and Netgear have come up with some solutions. Many of the solutions provided by these giants—like most of the research that is done in this field—is focused on collecting network traffic flow information [6] at routers(gateways). A flow consists of several Internet packets captured during a fixed time interval. The router sends that captured flow information to the central system for analysis. Central system is a hardware and software infrastructure which is capable of processing and analyzing large flow information.

Internet Protocol Flow Information Export (IPFIX) protocol created by the Internet Engineering Task Force (IETF), NetFlow protocol created by Ciscos [7], and Sflow (Sampled flow) [8] are some of the major protocols which are widely used for flow collection and analysis. These protocols have defined a standard way to export the flow information from the router and similar devices. All these flow monitoring protocols gather information and send the consolidated flow information to the centralized server where the user can login and perform functions; such as Security Monitoring, Bandwidth monitoring, Resource Management, Traffic Analysis, Performance Management etc. On such systems, there are some modules which are specifically used for anomaly/DDoS attack detection.

For example, Cisco netflow has a flow exporter, flow collector, and analysis modules. Flow exporter modules are installed on the routers. The routers, which have flow exporter module, send flow information to the collector module installed on the server. Along with the collector module, the server also has an analysis module, which can be used to detect different patterns in the flow.

These technologies scale well and can be sufficient to indicate trends in network traffic; however, they have limitations. 1) They are not cross platform, e.g., the

[4]System under a DDoS attack

router with Sflow protocol won't work with Cisco routers. 2) They involve setting up expensive hardware, which acts as a collector server. 3) The source address is used for flow analysis, which is not reliable due to IP spoofing in the case of a DDoS attack.

1.3 A DDoS attack Detection and Mitigation

A DDoS attacks can be detected by checking if there is any anomalous behavior in the network traffic, such as a sudden increase in the number of packets going to a destination. Detection can occur at the server by observing all of the incoming traffic or it can be done by observing all of the outgoing/incoming traffic at the ISP's or at every router. The attack can be mitigated if the anomalous packets are blocked from reaching their destination.

From the previous section we know that the router-based flow analysis can be useful for anomaly detection, but it has limitations. We don't want to set up expensive hardware, we want to have a protocol or a system which is compatible with other routers. Also, we don't want a source IP address for detection analysis. So, if we can come up with a way by which we can detect anomaly in the traffic at the intermediate devices on the Internet such as gateway devices (routers) and create a communication protocol between such gateway devices and the destination server or network, then better decisions on regulating the packet flow can be taken.

2 Network Functioning

A switch creates a network and router connects those networks. A router links a computer to the Internet through other routers. Routers are the backbone of the network who helps to forward packet from one point to other point on the Internet. Every packet traveling on the Internet goes through a router [9]. Router knows where the packet is destined, hence it could serve as first point of knowledge about the change in the packet flow information for a destine network. Each router has interfaces to which hosts, or other networks are connected. So, router is aware to whom it is connected. Router uses protocols to communicate between other networking devices and by that it gathers knowledge about other networks or routers on the Internet. Internet Control Message Protocol (ICMP) [10] is one of the most frequently use protocol by routers for sharing operational information.

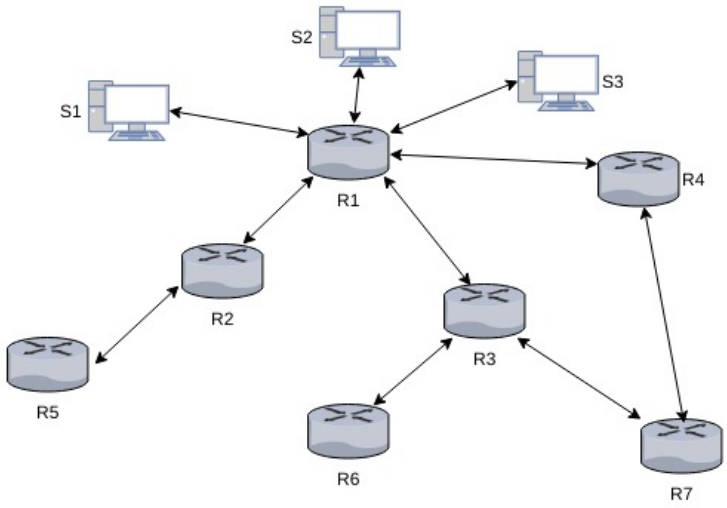

Figure 1: Network Example

Let's illustrate this using an example. In the above figure we can see that hosts S1, S2, S3 are connected to a router R1. Router R1 is connected to the Internet through router R2, R3 and R4, thus every packet reaching to the system S2 is coming from either of these three routers. All three routers are located in different geographical region. Most of the websites are regional, either county, state or national (if we leave out few global websites) and hence they are mostly

accessed from those regions they are meant for, e.g., the Rutgers University website is accessed mostly from the eastern region of United States and that too mostly from the New Jersey State or the Philadelphia region.

Using traceroute we can see how many hops[5] away the destination is. Following is one of the captured traceroute for Rutgers University website.

```
arpit@omega:~$ traceroute camden.rutgers.edu
traceroute to camden.rutgers.edu (128.6.34.90), 30 hops max, 60 byte packets
 1  192.168.0.1 (192.168.0.1)  1.067 ms  1.697 ms  1.684 ms
 2  10.240.177.197 (10.240.177.197)  7.617 ms  9.975 ms  10.302 ms
 3  67.59.225.66 (67.59.225.66)  10.803 ms  12.759 ms  13.074 ms
 4  dstswr1-ge1-2.rh.mhwhnj.cv.net (67.83.247.130)  18.962 ms  18.952 ms  18.902 ms
 5  67.59.239.121 (67.59.239.121)  18.844 ms 451be043.cst.lightpath.net (65.19.114.67)  18.314 ms
 6  451be031.cst.lightpath.net (65.19.98.49)  19.762 ms 64.15.3.138 (64.15.3.138)  10.763 ms  17.2
 7  * * *
 8  * * *
 9  RUTGERS-THE.ear3.Newark1.Level3.net (4.14.216.6)  33.338 ms  32.792 ms  33.274 ms
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  web-www.camden.rutgers.edu (128.6.34.90)  22.632 ms  23.859 ms  23.866 ms
```

Figure 2: Trace Route: All the routers in the path to destination

We can see that the packet traverse through the 15 routers to reach to camden.rutgers.edu server. This trace route is taken from a location in the New Jersey State.

[5]hops are intermediate routers in the communication channel

3 Our Approach

3.1 The router as a point of analysis

From Figure 1 and 2 we know that routers are located at different geographical locations and also there are specific regions from which a given website or web server is accessed (except few). To know the locations from where the web server/site has been accessed we can use services, called as GeoIP services. These services can detect the geographical location of the system from which the IP packet has originated, but that is just an approximate, based on the source IP and not always correct. In case of a DDoS attack, this information is unreliable because the packet source address is often spoofed, hence it is difficult to know the actual geographical location from which the packet has come, but if the router through which that packet has traveled can provide its own geographical information, then such information can be useful to understand the path through which a packet has traveled and thus we can know the region from which the packet was originated.

Also, in the normal scenario there is a consistency in which a website is accessed from different geographical regions, and this consistency can be found by measuring the number of requests or packets traveled from a router to a destination server. This behavior of accessing different websites from a router can be learned over the time. Thus, finding this behavior in a flow[6] at the router can form the basis of analysis in this paper. If there is a deviation from the learned behavior of accessing a particular destination, then that change in behavior as well as router geographical region information can be communicated to the destination network. The destination network on receiving that information, can decide, whether it wants the reporting router to discard or forward the traffic for it. This is a selective process in which traffic from only specific router is blocked while traffic from other routers remain unaffected.

[6]In this paper we will consider flow only in the sense of destination addresses

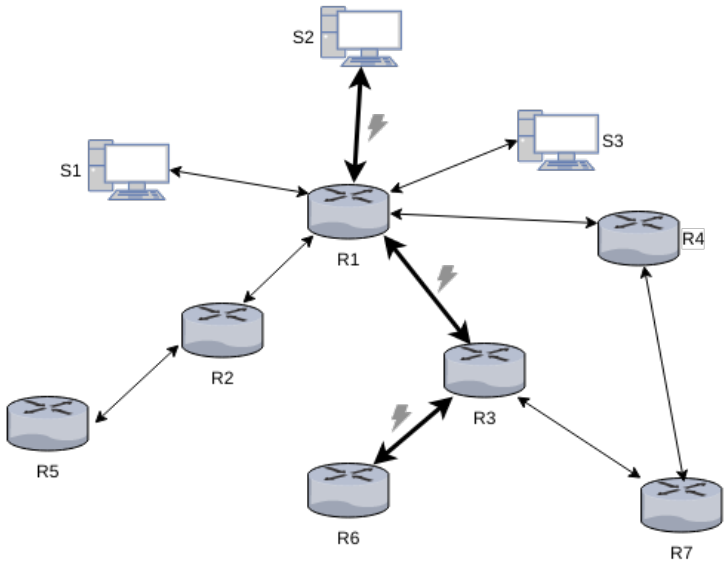

Figure 3: DDoS Attack path

In the above figure an attack is initiated from the region where router R6 is located and from router R6 data packets travel to the victim[7] system S2. Attack packets traveled through router R3 and R1 to reach system S2. If we can detect an attack at router R6, then router R6 can discard all the packets heading towards system S2. In this process, only traffic from router R6 is affected but traffic from all other routers remain unaffected.

With the advance of electronics and the Internet of Things, processing and storage capacity of the electronic devices has increased. Routers are also not left behind, but storage capacity of routers is always very less compared to server which collect flow data for network traffic analysis. If we use learning techniques that don't need much storage, then we don't have to store large chunk of packets on the router. Instead of storing data packets for longer time for analysis, we can learn from a small number of packets and then discard packets once learning is done, leaving behind only the learned information on the router. This is necessary because the number of entries on the Internet routing table has steadily grown. Now that the table has passed 500,000 routes [11], so storing each and every flow

[7]Victim is a computer system which is under a DDoS attack

information for these routes could be difficult.

3.2 Internet packet flow capturing at the router

To do the analysis, we have first gather the flow information during a time window (e.g., 300 sec) whose size will be fixed at the beginning. Thus, a flow will contain all the packets that traveled from a router to different destinations during a time span of 300 seconds. We can also combine such flows to form a new flow with bigger time windows, e.g., if we combine two flows of 300 seconds then we will have flow of 600 seconds. This flow information can be captured during a particular period or throughout the day. Once we have flow information we can apply learning techniques on each flow iteratively to gain deeper knowledge about normal behavior of the flow, e.g., on average, how many packets of different protocols are destined for a given destination from a given router/region during a particular time of the day.

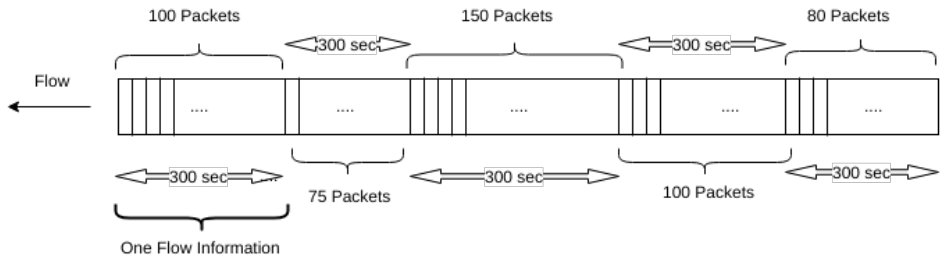

Figure 4: A 300 second time segment corresponds to a flow of information. The number of packets can vary in a flow.

3.3 Analysis techniques to be implemented

In the proposed system, each router will itself act as an analyzer. Each packet will be analyzed, and a flow statistic will be created based on the destination IP address. Flow statistic will be used to create feature vectors and those feature vectors will be used for clustering based on destination IP address. Clustering is a process of examining the collection of points, and grouping the points into clusters

according to some distance measure. The goal is to minimize the distance of every point in the cluster to other points in the same clusters. [12].

Once the clustering is learned (i.e. we know which destination IP address is tending to be in which cluster), the learned information will be used as a benchmark for all future flows. The routers will constantly keep clustering destination IP addresses and if there is deviation for the normal traffic at a router for any destination then that will affect the clustering and it will cause the destination IP address to be placed in different cluster. This change in the cluster for a given destination IP address can be marked as a change in the behavior of the traffic for that destination. Along with the cluster we will also use the Novelty Detection algorithm to achieve more accurate result. This change in traffic behavior will be reported to destination network, which then decides, buy doing its own analysis, on regulating the traffic coming to itself from the router which has sent the information.

4 Implementation

As discussed previously, there are different types of DDoS attacks, such as volume-based, application-based and also low-rate DDoS attacks. Among these different types of DDoS attacks, the volume-based attacks are most common. In the volume-based attack, a victim is flooded with a high volume of Internet packets (TCP, UDP, HTTP or ICMP), which make it unable to serve the requests.

For the demonstration of the suggested approach we are simulating a volume-based bot attack. In the real world, the volume-based attacks are orchestrated using bots. Bots use compromised computer systems, controlled by an attacker for launching an attack. They are not bounded by geographical boundaries, so they can be anywhere in the Internet. Botnet (i.e. network of bots) are employed by an attacker to launch a DDoS attack. As we know that the Internet is connection of different computer system that communicate with each other, through different channels such as cables, satellite or radio device and these communication channels run throughout the globe; connecting different computer systems at different locations. For our simulation, instead of any bot program, we are using Low Orbit Ion Canon(LOIC) tool which is a free DDoS attack launching tool. This tool is even used by attackers in the real world to launch a DDoS attacks.

We used Wireshark, an open source tool, for capturing Internet packets. Wireshark can capture all digital information received or sent through different devices such as Ethernet or Wi-Fi devices, which connect computers to the Internet. It also helps identify different protocol packets (e.g., TCP, UDP) within the wrapper packet created at Data Link Layer packet.

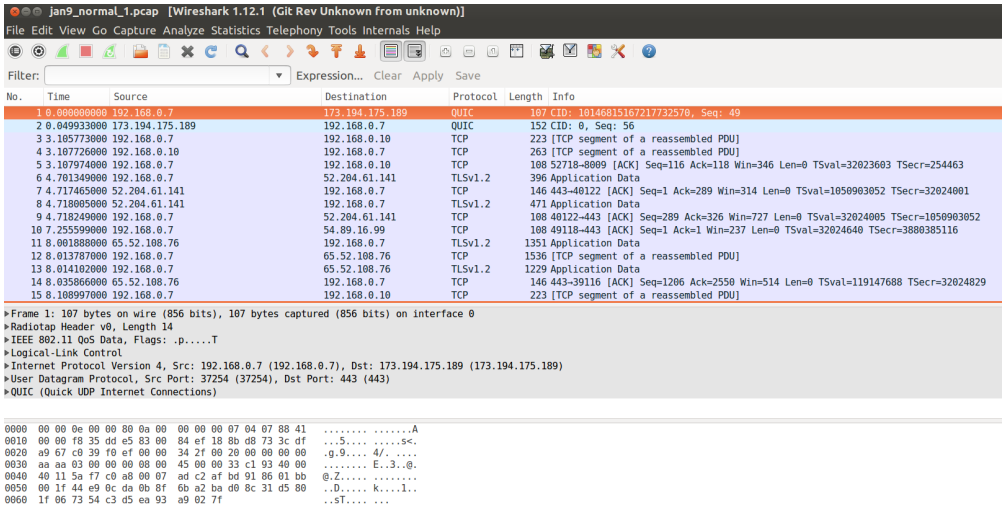

Figure 5: Wireshark Tool: snippet of captured packets

To gather data for the demonstration, we have created a small network which has one router and couple of host machine. Each machine can be a victim of a DDoS attack. We have installed Wireshark tool on one of the machine in this network. For capturing traffic in the network, we are using the Promiscuous mode of Wireshark. In Promiscuous mode, a network interface can record not only the traffic that is intended to itself but all the traffic on the network, so we can see all in/out traffic in our setup network. This setup is similar to any router on the Internet which is connected with different routers and hosts.

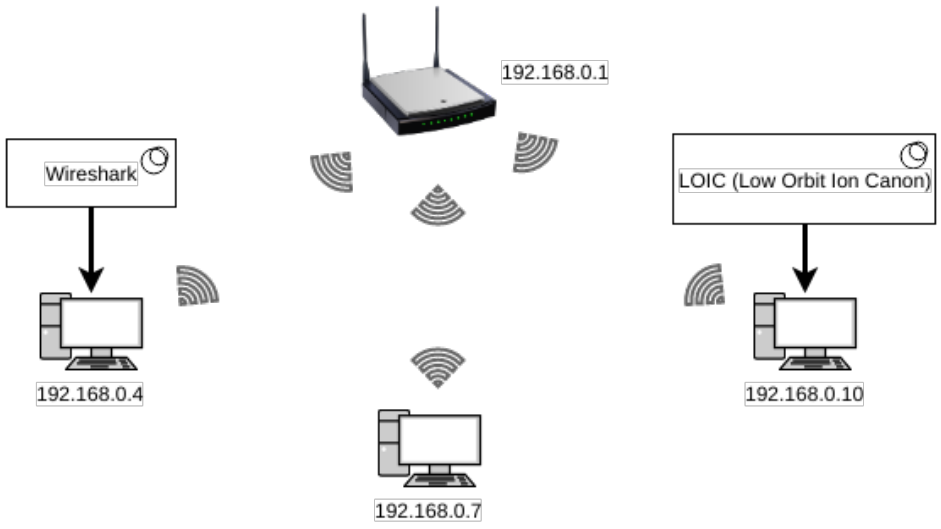

Figure 6: Network for simulating a DDoS attack detection

To collect packets traveling in our network, we start the Wireshark tool. We let it run for a while and then we orchestrate a DDoS attack on one of the host (e.g., 192.168.0.7 in Figure 6) in the network. This attack is engineered using the Low Orbit Ion Canon(LOIC) tool that is installed on one of the host (e.g., 192.168.0.10 in Figure 6) in our network. This tool allows us to launch TCP and UDP flood attack on any destination. In this paper we are analyzing the flow-based TCP and UDP attack, which is one of the most common type of DDoS attacks.

In the LOIC tool we need to give the IP address and the port number of the destination where we want to orchestrate attack while rest of the work is done by it. This tool floods destination with packets and if we choose TCP then it will try to create multiple connections and send packets over them. We start flood attack on one of the system (e.g., 192.168.0.7) in demo network and let it continue for few minutes. We launched such attacks few times in between the packet capturing session.

All the traffic, including the normal and the attack traffic, will get captured using Wireshark. Packets are captured for about five hours in the given network. Once the packets are captured they are saved as pcapng file, which is a Wireshark file format for captured packets. Captured packets during the normal operation and during the attacks are saved separately. The normal packet flow information is used for training and testing the learning algorithms (we will explain it in later sections) while attack packets flow information is used for detecting the attack. Wireshark captures every detail of the packets, but we don't need all of the information, we are interested only in the IP layer information of the packet. Most of the routers analyze IP layer of the packet for routing, having said that, there is no reason that other layers of the packet cannot be analyzed, but for our demonstration purpose we are analyzing only IP packets.

A data extraction program is written in Python language to extract IP layer information from the captured packets. This program extracts address, port and

time information from each of the captured IP packets[8].

Destination IP Address	Protocol	Time stamp(Sec.)	Sample Number
52.6.129.72	6	1512094785.928596000	1
192.168.0.4	6	1512094785.946987000	1
192.168.0.4	17	1512094786.148488000	1

Table 1: Sample file snippet (row represents an IP packet)

Our data extraction program also divides the captured data into 300 seconds capture windows, thus creating sample data which are a collection of IP packets captured over the time of 300 seconds. It then writes each sample in the separate file for further processing. This sampling of the packets is the continuous process. We then run another program which extracts the flow information from those sample files. One 'flow' contains the number of different packets capture for a given destination. We will store this flow information as sample flow. Then we will train learning algorithms using those sample flows. Once learning is done, those sample files will be discarded, and new samples will be created for further training. This training process has to be continuous process in order to correctly reflect the current status of the flow at given router. Whatever the new information learned, is augmented with the previous learning to have the correct understanding of the flow. This learning can be done for the time during a day or during a week of a year, e.g., We can have separate learning information for flow from morning 9 am to 12 pm and also can have information for evening 6 pm to 12 pm.

Flow based model is built, as it is more reliable and fast. Packet analyzing is often difficult due to the size and encryption. Also, destination port number is not a reliable information in detecting attacks because of the fact that attacker uses different ports during an attack.

Creating a training set for the learning algorithms is an intermediate step in which IP packet count for each destination for a given protocol (e.g., TCP, UDP)

[8]packets containing IP information

is calculated. The training set gives us the flow information for each destination (i.e. how many packets of a particular protocol are recorded for a given destination IP address during a time window, e.g., 300 seconds).

	IP Packet count		
Destination IP Address	ICMP	TCP	UDP
172.217.10.134	0	8	12
65.19.96.252	5	0	192
68.67.178.134	0	78	0

Table 2: Training Set with three training examples

We are using a Python program to create training sets, as given in above table 2, from the sample files. Each row in a training set is one training example with destination IP address as label. A training example is an $\mathbb{R}^3$ vector whose elements are the number of packets observed for a particular protocol for a destination during a fixed time (e.g., 300 second). There are around 150 protocols managed and assigned by the Internet Assigned Numbers Authority (IANA) but most commonly used protocols in a DDoS attack are ICMP, TCP and UDP protocols. For the training and analysis purpose we are using only these three protocols as the desired feature. In the larger system such as routers managed by ISP, other protocols can also be used as features if required.

Each sample file is processed, and corresponding train/test set file is created. Train and test sets files are stored on the file systems, so that those can be used by other programs for training and testing the learning algorithms.

4.1 Machine Learning

According to Tom Mitchell [13], a computer program is said to learn from an experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience.

A learning algorithm builds a hypothesis using a training set as input. Then

that hypothesis is used to perform predictions. The most common categorization of machine learning algorithms is Supervised and Unsupervised.

Let $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$ be the function that we need to guess from input vectors $x^1, x^2, ..., x^n$ also called as 'input variables' or 'feature vectors'. Let Ξ be the set of such input vectors. Let n be the number of input vectors in a training set Ξ. Let H be the set of some functions from $\mathbb{R}^d$ to $\mathbb{R}^d$. Let $h \in H$ be the hypothesis about function $f \in H$. We select h based on a training set $X \subseteq \Xi$, of m input vectors. In Supervised learning we know the values of f for m samples in the training set X. We assume that if we can find a hypothesis h that closely agrees with f for the members of X, then this hypothesis will be a good guess for f when X is large. In Unsupervised learning, we simply have a training set of vectors without function values for them. The problem in this case, typically, is to partition the training set into subsets, $X_1, .., X_N$, in some appropriate way. [12]

A supervised algorithm such as One Class Support Vector Machine (SVM) [14] is efficient at identifying the anomalies in the data but this algorithm is process and memory intensive, so training the algorithm for each and every IP address is very costly in terms of resources. Because of the resource constraints of the router, our approach is to first cluster the IP addresses based on the features using Unsupervised learning algorithms such as k-means and then apply One Class SVM on the clusters to decide on the boundaries of those clusters. The k-means algorithm is fast and consumes less resource compared to One Class SVM, that makes it good choice to be used on the devices such as routers which have less processing power and less memory.

4.1.1 Feature Scaling

Before feeding training data, that were acquired in an earlier stage, to a learning algorithm, we have to do feature scaling, also called Standardizing. Feature scaling is done by subtracting the mean and scaling the feature to a unit variance value. It is necessary because different features which are at different scales could cause one

feature dominating the others in the algorithm output result, e.g., consider two vectors (1, 2, 3000), (1, 3, 2000). If we calculate the Euclidean distance between these two vectors using the formula $\sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$, then the distance will be $\sqrt{(1-1)^2 + (2-3)^2 + (3000-2000)^2}$. Form this, it is evident that the larger term is dominating the result.

First, we will convert training examples into vectors. Each vector is one training example and each coordinate in the vector is a feature. To standardize the input vector, the mean and the standard deviation are calculated for the set of input vectors. Then a new vector is created by subtracting the mean from each feature vector and dividing that feature vector by its standard deviation. The new vector created after this step is standardized vector. A set of such standardized vectors is used as input to the learning algorithms.

$$\text{Standardization formula: } x' = \frac{x - \bar{x}}{\sigma}$$

where x is a feature vector, $\bar{x}$ is the mean and σ is its standard deviation.

4.1.2 Clustering

k-means [15] clustering is one of the most efficient algorithms for creating clusters. This algorithm takes any k randomly chosen points as centroids (also called, cluster centers) $\mu_1, \mu_2, ..., \mu_k$ as input from the training set $X = \{x^1, x^2, ..., x^m\}$, $x^i \in \mathbb{R}^d$, $i = 1, 2, ..m$.

The following is Lloyd's algorithm which is the most popular heuristic algorithm for k-means clustering. The clustering that we will do is of destination IP addresses, such that each cluster will have some number of destination IP addresses.

Algorithm 1 k-means

1: Random choose any k points $\in X$ as centroids.

2: **repeat**

3: Calculate the distance for each element (or data point) of the training set to all the centroids.

4: For each element, assign the element to the centroid which is nearest.

5: Recalculate the new centroid for all the element in one cluster by taking the mean. If $x_1^j, x^{j_2}, ..., x^{j_n}$ are the elements of the cluster j, then for cluster j, the new centroid will be $\mu_j = \frac{1}{n}[x^{j_1} + x^{j_2} + ... + x^{j_n}]$, where n is number of points assigned to cluster j. Do this for all the clusters.

6: **until** No data point is reassigned to a different centroid.

For this paper we will be using the k-means++ algorithm, which is an improvement of k-means, where an arbitrarily initialization step is replaced by simple, randomized seeding technique. K-means++ tries to find the centroids as far as away from each other. If $D(x)$ is the shortest distance from a data point $x \in X$ to the closest centroid we have already chosen, then the following is the k-means++ algorithm [16].

Algorithm 2 k-means++

1: Select a centroid μ_1, chosen uniformly at random from X, as first centroid.

2: Choose a new centroid $\mu_i \in X$, such that the probability $\frac{D(\mu_i)^2}{\sum_{x \in X} D(x)^2}$ is highest.

3: Repeat Step 2. until all k centroids are taken.

4: Proceed with the Lloyd's k-means algorithm skipping random initialization stage.

For our DDoS attack detection program, we will be using the Scikit-learn libraries. Scikit-learn is the most popular and rich open-source machine learning software library for the Python programming language and it has implementation for both k-means++ and One Class SVM machine learning algorithms, also it has data preprocessing programs such as feature scaling.

We will take a training set (in the format given in figure 2) and we will cluster training examples from that training set to using the k-means++ clustering algorithm from Scikit-learn library.

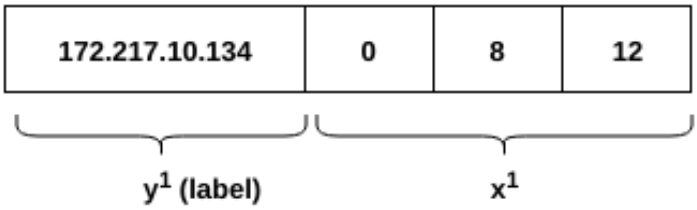

Figure 7: One training example

Before doing the clustering, we have to first determine the number of clusters and centroids.[9] Deciding on the number of clusters is important, because randomly choosing the number of clusters will not have correct clustering. So, we will use the Elbow method to find the optimal number of clusters in our training set. The Elbow method checks the portion of the variance explained by function of the number of clusters. Following is the example of a Elbow diagram.

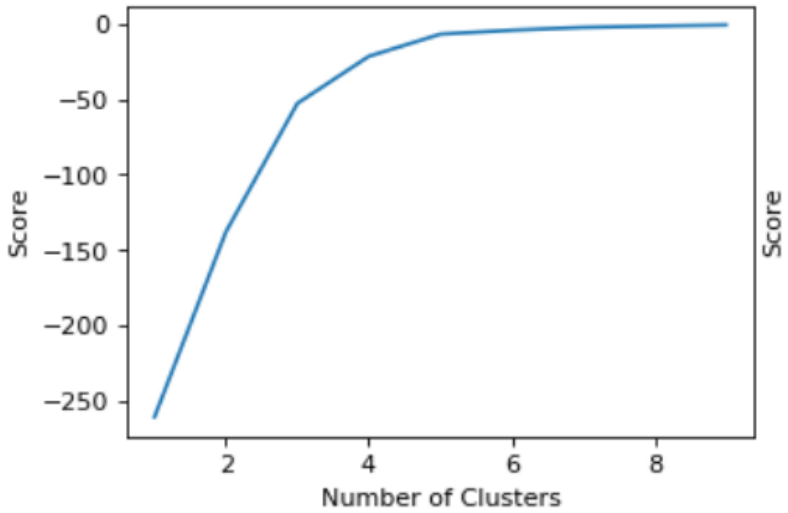

Figure 8: Elbow Method for cluster count detection

[9]The centroid is a vectors which is the arithmetic mean of all the points in the cluster.

Using the elbow method, the variance for each cluster number is calculated and the cluster number which produces less variance for the next cluster number is selected as best choice for the give training set.

To have correct clustering we first run the k-means++ algorithm to determine the central vectors called centroids. Clusters are represented by these centroids. As we have multiple training sets (also called 'samples'), we have cluster centroids for each sample. But in the future, we will require only one set of centroids which would be the more accurate clustering for all our training sets. To achieve this, we will save all the centroids, and then find the median of those centroids after removing the outliers, and that will give us a good estimate of centroids. These estimated centroids are used for clustering the samples in the future. Following is the example of centroid vector, where a row represents a cluster and a column represents a feature.

Cluster	ICMP	TCP	UDP
0	-0.16815612	-0.14928111	-0.16948046
1	-0.18181818	5.13527652	5.68956244

Table 3: Example of cluster centroids: A row is a cluster and a column is a feature

Once we have centroids we can use them to cluster training set again and get the correct clustering based on the centroid.

After clustering is done, it needs to be tested for the accuracy. I.e. how similar the test set clustering is with training set clustering. This can be done by checking, whether an IP address has the same cluster number assigned in both the training and the test sets or it differs. For measuring this similarity, Rand Index(RI) [17] is used. RI is a measure of how well does the test clustering matches with the trained model:

$$RI = \frac{TP+TN}{TP+FP+FN+TN},$$

where, $TP = true\ positives$, $TN = true\ negatives$, $FP = false\ positives$, and $FN = false\ negatives$

Because the training sets contain the flow information for different IP addresses on a router during the window of 300 seconds, there is a very high possibility that the same destination IP address is captured in multiple training set. Our goal is to find the correct cluster for the destination IP address, and to achieve this goal, we have labeled IP address with cluster in which it is found most of the time in all our training sets, e.g., If an IP address is assigned cluster '0' in five training sets and cluster '1' in two training sets, then that IP address would be assigned cluster '0' in the result, that we store for future referencing.

Destination IP Address	Cluster	Packet Count
74.125.141.106	1	113
72.30.2.182	0	16
64.94.191.14	0	22

Table 4: Learned information after clustering

The clustering information tells us the normal behavior of the packets traveling from the router to a given destination. As we have fixed the centroids for the clustering, every time in the future we should expect an IP address to be found in the same cluster mentioned in the clustering information table. But If there is a flooding DDoS attack on a destination, then we can expect to see that destination IP address assigned to a different cluster.

4.1.3 Anomaly Detection using One Class Support Vector Machine

From the experiments on different data sets, it is found that the destination under attack is labeled with the same cluster number. This happened because there is no other cluster it can be assigned to, so it gets assigned to the cluster whose centroid is nearest.

To avoid such a situation, we will have to create boundaries for the clusters, and if any IP address is outside of the cluster boundaries then that can be considered suspicious. This provision will make sure that the attack on the destination IP

address will be detected even if the destination is assigned to the same cluster it was assigned in the past.

To create cluster boundaries, we have used One Class SVM classifier. One Class SVM is a type of Support Vector Machine (SVM). SVM is a supervised machine learning algorithm, which means, the training example will have label. I our case, a cluster number will be treated as label for a training example (see table:labeled-set).

Support Vector Machine (SVM): SVM is a supervised learning algorithm which tries to classify training examples into two distinct classes. Classification is based on the labels of the training set. Consider the training set $\{(x^1, y^1), .., (x^m, y^m)\}$, where $x^i \in \mathbb{R}^d$ is the training example and $y^i \in \{-1, +1\}$ is a label (or a class) for x^i. SVM will separates those classes using a line or a curve.

SVM tries to create a non-linear separation boundary by projecting data points using a non-linear function $\Phi : x \to \phi(x)$ (called "kernel" trick) to higher dimension space. The data points in space I which can't be separated by a line are projected to the feature space $F \subseteq \mathbb{R}^d$ where there can be a hyperplane that separates data points of one class form another. If that separating hyperplane is projected back on the original space I then we get a non-linear curve. [14]

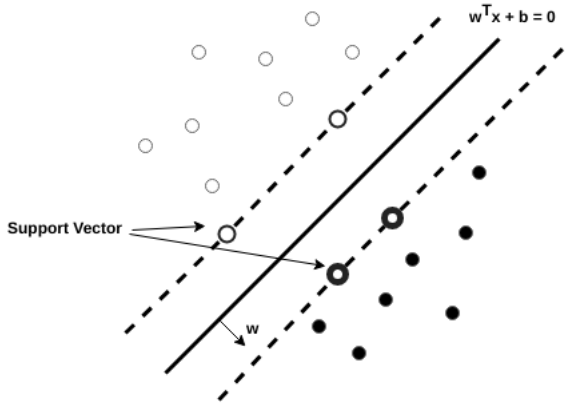

Figure 9: Linear Separation using Support Vector

The hyperplane is represented with the equation $w^T\phi(x)+b = 0$, where $x \in \mathbb{R}^d$,

$w \in F$ and $b \in \mathbb{R}$. This hyperplane separates the training example labeled with -1 and 1. The position of the hyperplane is such that the distance from the closest point from each class to the hyperplane is same. To avoid the over-fitting, slack variables ξ^i are introduced. Over-fitting happens because the learned hypothesis fits training examples so well that it fails to generalize the new examples. The SVM classification is an optimization problem which is stated as follows. [14] [18]

$$min_{w,b,\xi^i} \ \frac{||w||^2}{2} + C \sum_{i=1}^{m} \xi^i$$

subject to:

$$y^i(w^T \phi(x^i) + b) \geq 1 - \xi^i \text{ for all } i = 1, \ldots, m$$

where $y^i \in \{-1, +1\}$, $x \in \mathbb{R}^d$, $b \in \mathbb{R}$ and $\xi^i \geq 0$ for all $i = 1, \ldots, m$

The constant $C > 0$ is the regularization parameter. If C is chosen large, misclassification of training examples can be avoided. If chosen small, then we may misclassify few examples, but the margin will be large, so most of the points will be far away from the decision boundary. If this minimization problem is solved using Lagrange multipliers, then the classification function is:

$$sign(\sum_{i=1}^{m} \alpha^i y^i K(x, x^i) \ + b)$$

The α^i are the Lagrange multipliers, the $\alpha^i y^i$ are called the support vectors and the function $K(x, x^i) \Rightarrow \phi(x)^T \phi(x^i)$ is the kernel function. Kernel function is a similarity function that is a dot product of data points which are mapped in the higher dimensional feature space by function $\Phi. The intercept term \in \mathbb{R}$.

One Class Support Vector Machine: One Class Support Vector Machine (SVM) is the extension of SVM which detects boundaries of the training set so that every new training example will be classifies as belong to training set or not. It separates all the training set data point from feature space F and maximizes the distance of hyperplane from F. This creates a binary function which returns $+1$ for the training example that fits in the trained set region, otherwise it will return -1.

The minimization function of One Class SVM is slightly different than the SVM. [14]

$$min_{w,\rho,\xi^i} \ \frac{||w||^2}{2} + \frac{1}{\nu m}\sum_{i=1}^{m}\xi^i - \rho$$

$$\text{subject to:}$$

$$(w.\phi(x^i)) \geq \rho - \xi^i \text{ for all } i = 1, \ldots, m$$

$$\xi^i \geq 0 \text{ for all } i = 1, \ldots, m$$

The new parameter $\nu \in (0, 1]$ introduced in place of C in previous SVM equation is used to set upper bound on outliers/anomalies and lower bound on the number of training examples.

Because SVM solves a quadratic programming problem (QP), computing and storage requirements of SVM increase rapidly with the number of training vectors. The approach presented in this paper is more efficient because the number of clusters is limited and always be far less than the number of destination IP addresses. Training one class SVM on clusters will be far less process and memory intensive than training on a massive number of IP addresses, e.g., in our simulation training sets we have 4 four clusters, while the unique number of destination IP addresses in all the training sets is 268, which is more than 60 times the count of clusters we have found.

To classify whether a test example can be a part of a cluster, we need to have classifier for each cluster. I.e. If we have four clusters, we need to train four One Class SVM classifiers, one for each cluster.

5 Detection

To detect an attack, we have to capture the packet flow at the router during a time interval with which we have created our trained set (e.g., 300 second window). We have to transform the captured flow into a test set using same programs we have used to create training sets from the sample files. The created test set is then clustered using the k-means++ algorithm, using the centroids, that were found after clustering the training sets. We then check the cluster assignment for each example in the test set, and if there is any destination IP address[10] for which the new cluster assignment does not match with the already known cluster for that destination IP address, then that destination IP address is suspected to be under the DDoS attack and so it is added to the suspect list. The suspect list will contain the IP addresses which are probably under attack because they are not in the cluster they are supposed to.

In another scenario, if the test cluster label for an IP address matches with the trained cluster label, then the feature vector for that destination IP address is passed to the cluster classifier to check if it is inside the boundary of that cluster. As explained in previous section, we will use One Class SVM classifier to decide the boundaries, e.g., if an IP address is labeled with cluster number 0, then we will use One Class SVM classifier that was trained on cluster 0. If the output of the classifier is -1 then that destination IP address is added to the suspect list. Failing to be in the same cluster boundary, that it was in the past, is a sign of significant change in the behavior of the traffic for a given destination IP address on the router where this analysis is done. For every destination from the compiled suspect list, the total number of packets observed in the test set are also recorded. The packet count will act as a filter, to eliminate any misclassification. If there is a significant difference between the number of captured packets in the training and in the test, then that destination IP address will be considered under a DDoS attack.

[10] a destination IP address is a label of a test example

6 Detection Simulation

We have created a simulation for detecting DDoS attacks using the approach proposed in this paper. As explained in section 4 on page 13, we have a small network of three computers and a router. We captured the normal as well as orchestrated DDoS attacks traffic in the network. We processed the capture data, converted it into samples and use those samples to create training sets. We are using training sets created from the normal traffic for labeling IP addresses with cluster numbers, and then training the classifiers.

We have first used the elbow method explained in section 4.1.2 on page 19 to detect the optimal cluster count. Using that method, we were able to detect 4 clusters in our training sets.

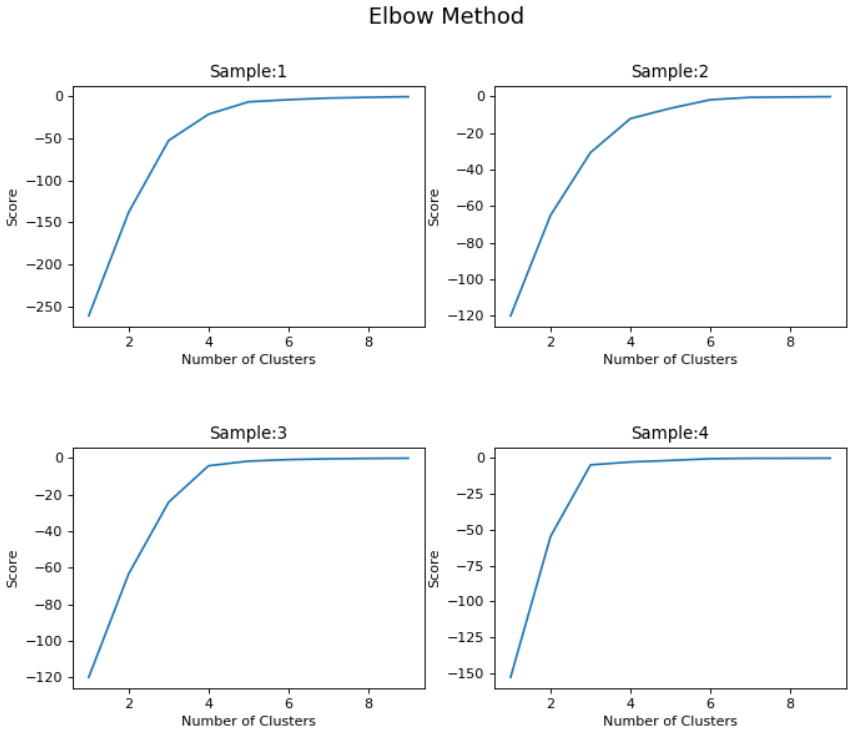

Figure 10: Elbow method to find number of clusters

We then applied k-means++ clustering, as explained in section 4.1.2 on page 19, to find the correct centroids for our training examples. We found following centroids for our training sets.

Cluster	ICMP	TCP	UDP
0	-0.16815612	-0.14928111	-0.16948046
1	-0.18181818	5.13527652	5.68956244
2	5.08663322	-0.27110845	-0.099885
3	-0.18670401	-0.18804342	-0.018538

Table 5: Cluster centroids for our training examples

We then used these centroids to cluster our training sets. To draw the clusters in two dimensional space we had to reduce the three- dimensional training example into two-dimensions without losing much information. We have achieved this by using Principal-Component Analysis (PCA). PCA is a technique for taking a dataset consisting of a set of tuples representing points in a high-dimensional space and finding the directions along which the tuples line up best. [19]. We have used the Scikit-learn PCA module for this purpose.

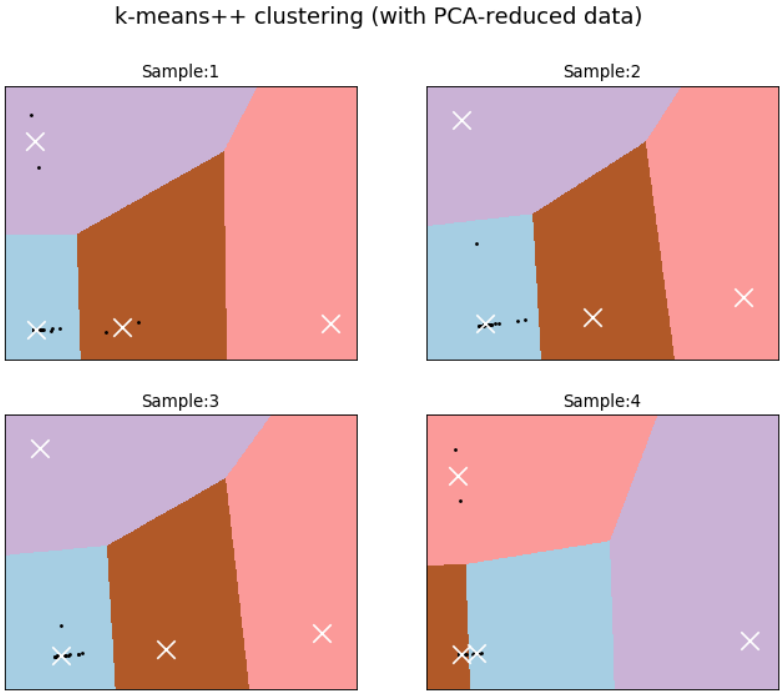

Figure 11: Clustering using k-means++ algorithm

k-means++ algorithm gives label to each training example. This label is the cluster number to which that training example has been assigned to. So, the new labeled data look like below (table 6).

Destination IP Address	ICMP	TCP	UDP	Cluster
172.217.10.134	0	8	12	1
65.19.96.252	5	0	192	0
68.67.178.134	0	78	0	2

Table 6: Labeled Training Set (with cluster number)

We also checked the clustering accuracy using the Rand Index explained in the section 4.1.2 on page 19. We observed that, with more training sets, the RI index improves. A tag file which contains IP addresses, their cluster labels and average packet count is created. This tag file will be used for detecting anomaly in the network traffic.

As explained in the section 4.1.3 on page 23, we used the training sets, that are labeled with cluster number, to get classifiers. We are using Scikit-learn's 'OneClassSVM' library to train the model and create classifier for each cluster. The input vector to the classifier is the set of all the training examples belonging to the same cluster. Thus, for our four clusters, we have four classifiers. The input vector to the classifier will be of the form shown in table 6 on page 30.

Following is the result of modeling on the training data sets. Each cluster has its own model. We used PCA component analysis to draw the model.

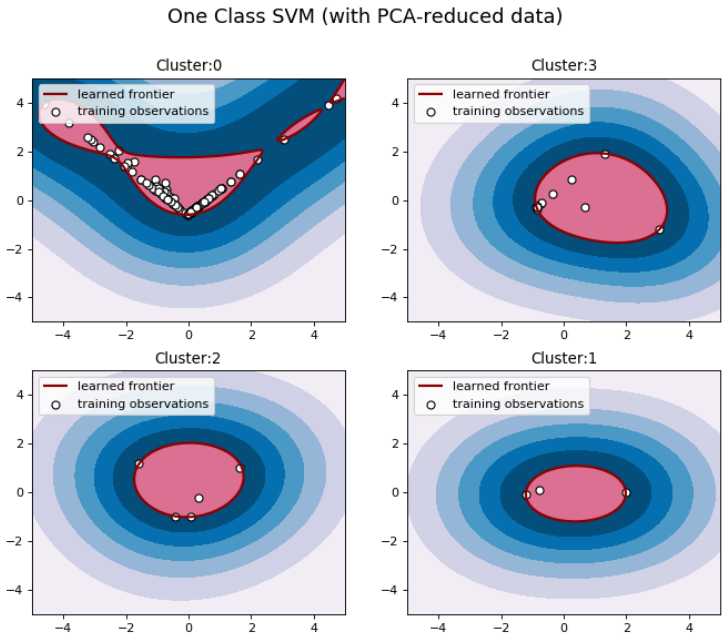

Figure 12: One Class SVM

At the end of the training, we have a file containing the IP addresses labeled with the cluster numbers and four classifiers for four clusters.

We took the test set which was created by capturing packets during the orchestrated DDoS attacks and used detection techniques explained in section 5 on page 27. Using the detection techniques on our data, we have successfully detected an attack on the destination IP address 92.168.0.7 in our modeled network.

ip	packet_count	percent_increase	cluster
173.194.68.189	2	-93.548387	0
192.168.0.10	14	-88.333333	3
192.168.0.4	31	-96.924603	2
192.168.0.7	29867	1684.169654	1
224.0.0.251	9	-95.945946	0
65.19.96.252	2	-91.304348	0
65.19.96.253	4	-77.777778	0

Figure 13: Program output

7 Mitigation

By now we have a computer program (consist of few modules, e.g., clustering, classification, detection) which can be run on a router and can be used to detect a DDoS attack. As there are many routers through which a destination can be reached, so to correctly detect a DDoS attack we should have this programs on every router in the Internet. One of the way to achieve this is by using Network Function Virtualization (NFV) [20]. NFV is an advanced technology which allows us to manage network devices remotely and also allows to add functionality to those networking devices. Because of this virtual technology, we don't have to manually install computer programs on the routers rather we can do it anytime from anywhere. Although this technology is new, its spreading very fast and soon will be available for every commercial router.

After detecting an attack, a router can choose to block all the packets going to the destination under attack, or it can communicate the attack information to the destination. A destination can be a server or a router. The attack information can contain region name and packet count. Region is the geographical location where the router is situated, and packet count is the average number of packets observed at the router for the destination during attack. Communicating the attack information to the destination can be more helpful because the destination will be aware of the nature of the attack, i.e. for the information provided by router it will know from where attack is originated and how intense the attack is. The destination can also query router for more information such as IP source of the attack packets.

The control of blocking the traffic at the router can also be given to the destination. In this case, destination can perform its own analysis, understand the nature of the attack and then decide whether the incoming packets from the reporting router should be block or not. There can be different parameter based on which the destination server can decide to block the flow and those can be provided by the router as additional information. Many questions can be asked before making

the decision at a destination, such as, How many routers have reported the change in traffic? Is the attack information coming from the region which never had traffic in the past?

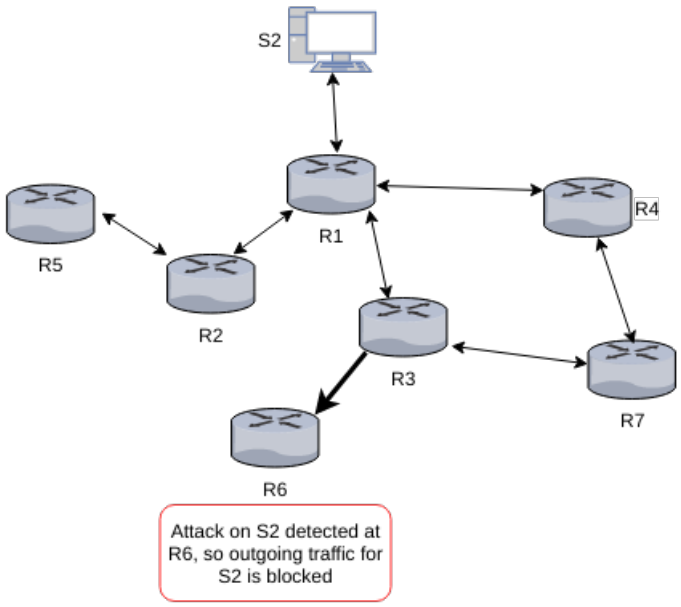

Figure 14: Blocking traffic at router: The out-bound traffic for S2 is blocked from router R6, because attack has been detected. While traffic from all other routers is unaffected

To Communicate with a destination network about the attack, router can use the existing ICMP protocol. ICMP protocol is used to provide feedback about the problems in the communication environment. ICMP messages are sent in several situations such as: 1) when a datagram cannot reach its destination. 2) when the gateway does not have the buffering capacity to forward a datagram. 3) when the gateway can direct the host to send traffic on a shorter route. [10] Similarly we can use ICMP protocol to inform destination about anomalies in the traffic. ICMP protocol has many unused type code (there can be 0-255 types but as of now only 0-41 are in use). We can create a new ICMP 'type' to send DDoS attack detection information from the router to the destination server and then destination server can send mitigation instructions to the routers.

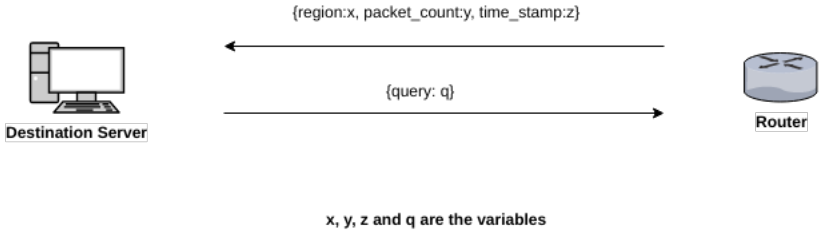

Figure 15: Communication between a router and a destination server

8 Conclusion

The DDoS attacks are real threat to everyone on the Internet, and its detection is difficult because of the spoofing techniques employed by the attackers. We discussed how we can combine two machine learning algorithms (clustering and classification) to efficiently detect DDoS attacks at the routers. We also discussed how to create programs that use the learning algorithms and then how to installed those programs on the routers, thus eliminate the need for extra infrastructure for detection of a DDoS attack. With the advancement of Network Function Virtualization (NFV), it is also easy to push the learning programs on the routers. We have also discussed how a router can stop an attack and also how it can communicate the attack information with the stake holders (e.g., server) for better decisions on blocking the attack traffic.

References

[1] D. W. Davies. A communication network for real-time computer systems. *Radio and Electronic Engineer*, 37(1):47–51, January 1969.

[2] Cisco Security portal. A cisco guide to defending against distributed denial of service attacks. `https://www.cisco.com/c/en/us/about/security-center/guide-ddos-defense.html`. Online; accessed 1-August-2017.

[3] Derek Kortepeter. Destructive ddos attacks increasing at a rapid rate. `http://techgenix.com/ddos-attacks-increasing/`, December 2017. Online; accessed 22-August-2017.

[4] S. M. Bellovin. A look back at "security problems in the tcp/ip protocol suite". In *20th Annual Computer Security Applications Conference*, pages 229–249, Dec 2004.

[5] Ken Dunham and Jim Melnick. *Malicious Bots: An Inside Look into the Cyber-Criminal Underground of the Internet*, chapter 1. Introduction to Bots. CRC Press, August 2008,.

[6] Network Working Group. Traffic flow measurement: Architecture. `https://www.ietf.org/rfc/rfc2722.txt`, October 1999.

[7] Cisco. Introduction to cisco ios netflow. `https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html`, May 2012. Online; accessed 22-August-2017.

[8] sFlow. Using sflow. `http://www.sflow.org/using_sflow/index.php`. Online; accessed 16-November-2017.

[9] Cisco. What is a network switch vs. a router? `https://www.cisco.com/c/en/us/solutions/small-business/resource-center/connect-employees-offices/network-switch-what.html`. Online; accessed 16-November-2017.

[10] Network Working Group. Icmp. `https://tools.ietf.org/html/rfc792`, Septmber 1981.

[11] Omar Santos. The size of the internet global routing table and its potential side effects. `https://supportforums.cisco.com/t5/network-infrastructure-documents/the-size-of-the-internet-global-routing-table-and-its-potential/ta-p/3136453`, May 2014. Online; accessed 22-August-2017.

[12] Jeffrey D. Ullman Jure Leskovec, Anand Rajaraman. *Mining of Massive Datasets*, chapter Clutering. Stanford University, 2014.

[13] T.M. Mitchell. *Machine Learning*. McGraw-Hill international editions - computer science series. McGraw-Hill Education, 1997.

[14] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, pages 582–588, Cambridge, MA, USA, 1999. MIT Press.

[15] Nathan S. Netanyahu Christine D. Piatko Ruth Silverman Tapas Kanungo, David M. Mount and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE*, 24(7):881–892, July 2002.

[16] David Arthur and Sergei Vassilvitskii. K-means++: the advantages of careful seeding. In *In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007.

[17] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[18] Jeffrey D. Ullman Jure Leskovec, Anand Rajaraman. *Mining of Massive Datasets*, chapter 12. Large-Scale Machine Learning. Stanford University, 2014.

[19] Jeffrey D. Ullman Jure Leskovec, Anand Rajaraman. *Mining of Massive Datasets*, chapter 11. Dimensionality Reduction. Stanford University, 2014.

[20] Cisco Systems. Cisco enterprise network functions virtualization. *Cisco Technical White Paper*, 2017.