Abstract

Distributed Denial of Service (DDoS) attacks are common these days [1]. So it is evident that current industry solutions such as completely relying on Internet Service Provider(ISP) or setting up DDoS defense infrastructure are not sufficient in detecting and mitigating DDoS attacks, hence consistent research is needed. Most of the current industry solutions involve setting up centralized expensive hardware system which can analyze the packets[1] [2] for probable DDoS attacks. Also each organization or ISP has different systems which are not compatible with other ISP's. In this paper we are going to discuss another a way to detect and mitigate DDoS attack using machine learning tools.

1 Existing Systems

Distributed Denial of Service (DDoS) attack is the way to jam host network or its resources with large number of data packets, so that host become disabled to serve. There are different types of DDoS attacks such as 1. Volume based e.g. SYN Flood Attacks, in which victim is flooded with hight volume of packets or connection. 2. Application based, in which application such as DNS, VOIP or HTTP where attacked. 3. Low rate DDoS attacks, in which attacker exploit the vulnerability in application design, e.g. Slowloris. [3]

The real challenge in detecting and defending DDoS attack is because of its dynamic nature. The source[2] is not a single node or system on the Internet but can be many, and often distributed over the Internet. Also the source of the packet is often spoofed[3] [4], which makes more harder to know the actual IP address of system from where attack is originated because they hide original attack source. On top of that, many times the source itself is not aware that it is compromised and has been used as bot [5] by attacker.

Detecting and mitigating attack at the destination[4] is not very useful as because 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, because source address can not be reliable way to know the attack source. To avoid this, firm which provide the networking devices have come up with solutions.

Many of the solutions available in the market or the research that was done is to collect network traffic flow [6] (we would call it just flow) samples at routers(gateways) and feed/send it on the central system to analyze. Central system is a hardware and software infrastructure which is capable of processing and analyzing large flow information.

[1] Messages that are sent on Internet are broken into shorter messages for transmission. These short messages are called packets. Term coined by Donald Watts Davies.

[2] It is a system/device on the Internet which has an IP address and which is involved in DDoS attack

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

[4] System under DDoS attack

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

E.g. Cisco netflow has flow Exporter, Collector and Analysis modules. Flow exporter is generally router who send flow information to collector module which acts as flow storage. Analysis module then try to find out different patterns in the flow.

These technologies scales well and sufficient to indicate trends in network traffic but they have limitations. 1) They are not cross platform, e.g. router with Sflow protocol would not be able to work with Cisco routers. 2) They involve setting up expensive hardware which would act as collector server. 3) In these technologies source address is used for flow analysis but as we know it is not reliable due to IP spoofing in the case of of DDoS attacks.

Now we know that 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 protocol or system which is compatible with other routers. Also we want to only rely on destination IP address for flow[5] analysis. So if we could come up with the way by which we could detect anomaly in the traffic at the routers independent of manufacturer, and create a protocol to communicate the attack parameters to allow routers to take decisions then we would be more efficient in detecting and mitigating DDoS attacks.

If we use the learning algorithm at the router and if router could learn the normal behavior of the flow then if there is any anomaly then it will be able to identify it from its previous learning and that change in behavior of the traffic can be communicate to destination network.

With the advance of electronic and the Internet of things, electronic devices are getting equipped with faster processor and fast memories. Router are also not left behind. Only thing that is missing is the storage space. If we use the learning techniques which don't need much storage then we don't have to store the packets instead we would learn from every flow and discard once learning is done. This is necessary because number of entries on the Internet routing table has steadily grown. Now that the table has passed 500,000 routes [9] so storing each and every flow information for these routes could be difficult.

[5]Hence forth In our paper we would consider flow only in sense of destination address

2 DDoS Detection and Mitigation

One of the way to detect the DDoS attack is to check for anomalous behavior in the network traffic. This can be done at different points. Either at the destination server or at the router. Once the attack is detected by observing the behavior of the traffic to mitigate it, we would have to block all the packets which are causing the attack. One of the way to identify those packets is by their source address.

3 Network Functioning

A switch creates a network and router connects networks. A router links computers 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 Internet has to go through router [10].Router knows where the packet is destine hence it could serve as first point of knowledge about the change in the flow information for a destine network. Each router has interfaces to which hosts or other network are connected. So router is aware to whom it is connected. Router uses protocol to communicate and by that they gather knowledge about other networks or router on the Internet. ICMP [11] is one of the most frequently use protocol by routers to communicate.

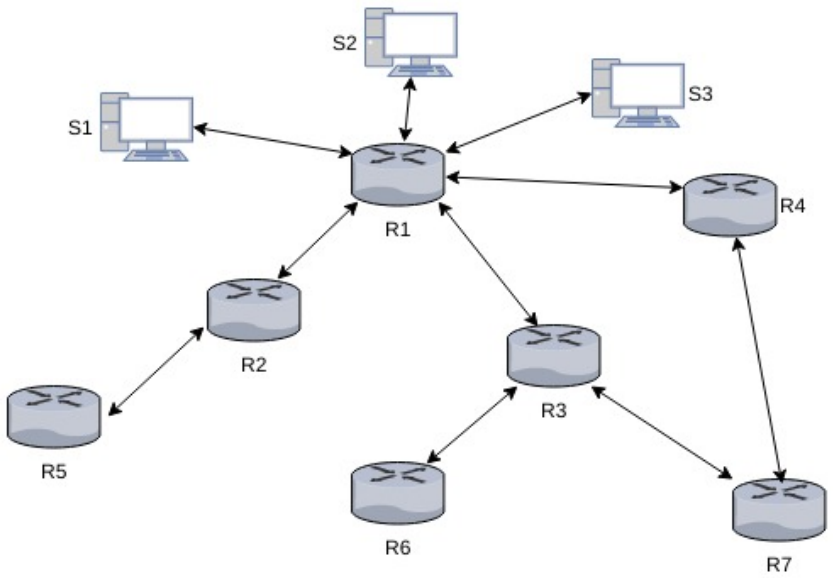

Figure 1: Network Example

Lets illustrated this using an example. In the above figure we can see that host S1, S2, S3 are connected to router R1. Router R1 is connected to Internet through router R2, R3, R4 ,

thus every packet that would be reaching to S2 would come from either of these three routers. All three routers are locates at 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 region it is meant for. E.g. Rutgers University website would be 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 find out how many hope away the destination is. Each hope is the router on the Internet. 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 there are about 14 routers (if we don't consider the home router 192.168.0.1) to reach to the camden.rutgers.edu. This trace route is taken from a location in the New Jersey State.

4 Suggested New Approach

From Figure 1 and 2 we know that routers are located at different geographical locations and also we know that there is pattern in which the particular destination website is getting accessed from the different regions. Some of the service providers such as GeoIP or Google can find out the location from where the traffic is coming in the network for a given destination, but that is approximate based on the source IP, which in the case of DDoS attack is unreliable information because packet source address are often spoofed. So it could be difficult to know the actual geographical location from which packet has come. Only routers can provide the correct geographical information about the source of the packet.

In the normal scenario there would be some definite pattern in which the website is getting accessed from different regions and this pattern can be learned over the time with learning algorithms at the routers. Thus finding this pattern in the flow at the router can form the basis

of our analysis. When ever there is deviation from the normal pattern of the flow for a particular destination then that change in pattern as well as the geographical region can be communicated to the destination network. Destination network on receiving that information can decide based on the region and how much is the deviation from the normal to decide whether it want the reporting router to discard or forward the traffic for the destination. This is selective process in which traffic from other routers remain unaffected

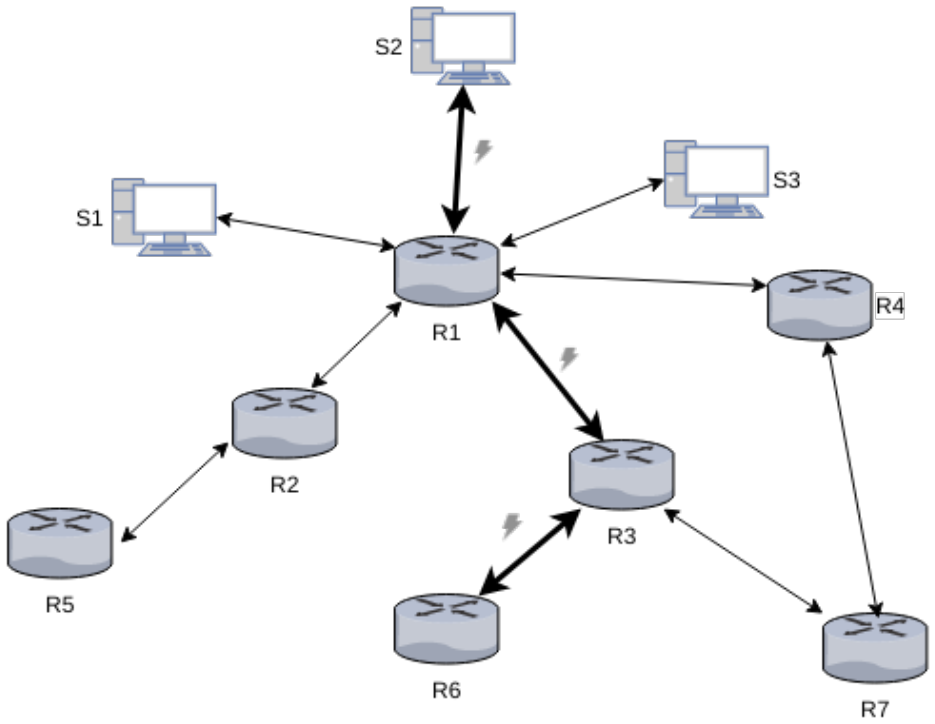

Figure 3: DDoS Attack path

In the above figure attack initiated from the region where router R6 is located and from R6 data packets reach to victim[6] from R3 and R1. If we could detect attack at R6 itself then we can discard packets at R6 which are heading towards S2, while traffic from other routers remain unaffected.

To achieve this we would measure the flow during a time window (e.g 300 sec) whose size would be fixed at the beginning. These time windows can be combined to form a period which is a portion of a day during which traffic is measured. Once we have flow information we can apply learning techniques on each flow iteratively to gain deeper knowledge about normal behavior of the flow.

In the proposed system, each router will itself act as a analyzer. Each packet will be analyzed and flow statistic would be created based on the destination IP address. Based on the statistic

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

we would cluster destination IP address using input feature vector. Clustering is the process of examining a collection of points, and grouping the points into clusters according to some distance measure.The goal is that points in the same cluster have a small distance from one another, while points in different clusters are at a large distance from one another [12].

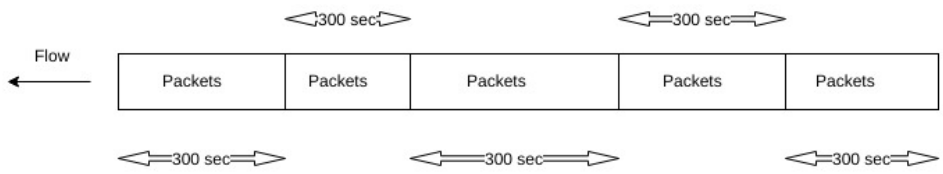

Figure 4: Data Flow Segments

Once cluster is learned it would be used as bench mark for all future flow. Router would constantly keep clustering destination IP and if there is deviation in the normal traffic at router for a destination then that would affect the clustering and would cause destination to be placed in different cluster, cluster with more packet frequency. This change in clustering would be reported to destination network, which then decide on regulating the traffic

5 Implementation

There are different types of DDoS attacks such as Volume based, Application based and also Low rate DDoS attacks. Among which the volume based attacks are most common. In the volume base attack victim is flooded with high volume of Internet packets (TCP, UDP, HTTP or ICMP)

We would stick to the volume based attack for the demonstration of our approach and we would try to simulate Bot attack. Bots are the compromised system controlled by attacker for launching attack. They are not bounded by geographical boundaries so can be anywhere in the Internet. Botnet(Network of bots) are employed by attackers to launch DDoS attack. As we know that Internet is connection of different computer system which can communicate with each other. This communication can occur trough cables, satellite or radio device called communication channels, and these communication channels run through out our world connecting different computer systems at different locations.

We are going to use Wireshark, an open source tool, for capturing Internet packet. Wireshark can capture all digital information received or send through devices such as Ethernet devices, which connects computer to the Internet. It also helps to segregate data in terms of packets pertaining different protocols (e.g. TCP, UDP).

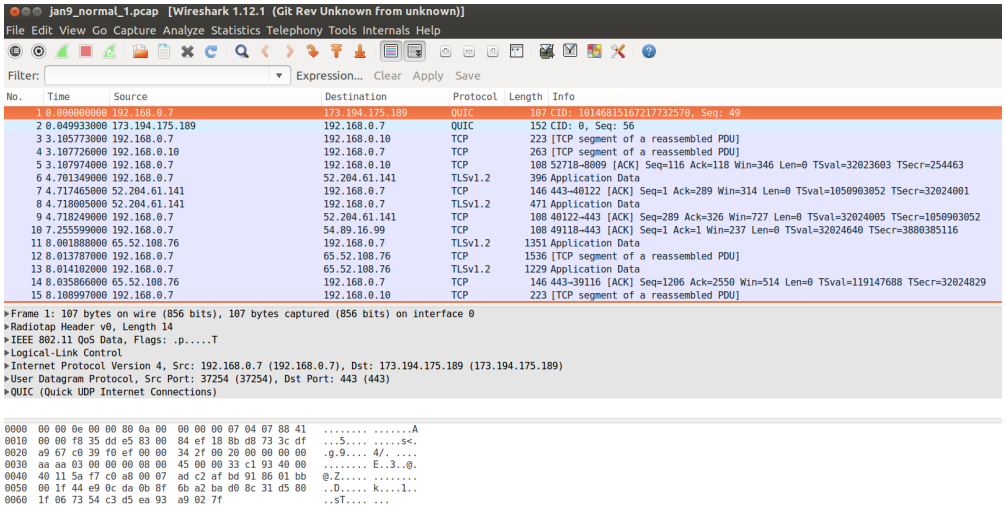

Figure 5: Wireshark Tool: snippet of captured packets

To gather data for the demonstration we have created a small network with one router and we designated a host machine in that network as a target for the DDoS attack. We have also installed Wireshark on one of the system in the same network. For capturing traffic in the network we are using the Promiscuous mode in Wireshark. Using this mode network interface can record not only the traffic that is intended to its self but all of the traffic on the network. This setup is similar to capturing traffic on a router which is connected to few host machines.

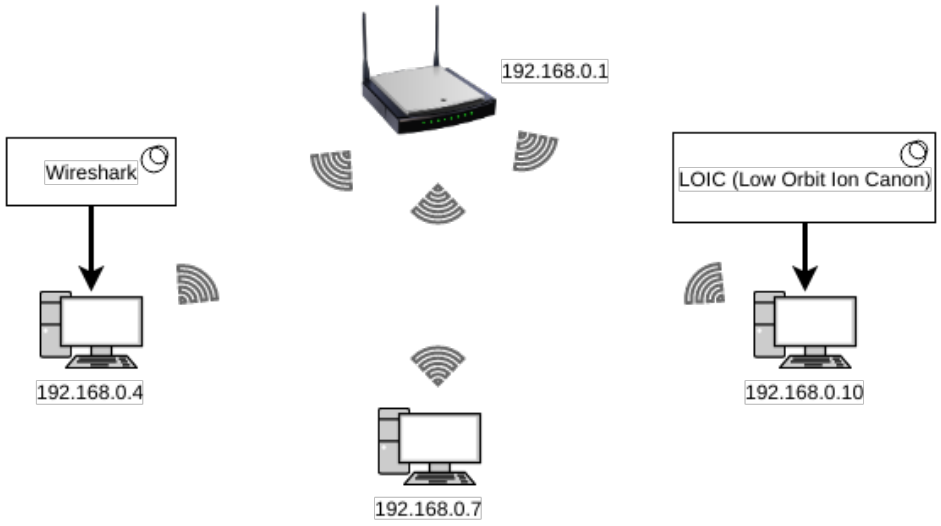

Figure 6: Demo Network: for purpose of demonstration

Once we start Wireshark we did let it run for a while and then we orchestrate an attack on one of the host (e.g. 192.168.0.4 in Figure 6) in the network from another host(e.g. 192.168.0.10 in Figure 6) from the same network.

For the sake of simplicity we would be dealing with TCP and UPD flood attacks.

Attack is engineered using Low Orbit Ion Canon(LOIC, a free tool which is even used by attackers in the real world DDoS attacks. This tool allows us to launch TCP and UDP flood attack on any destination. In this paper we would be 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 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 flood 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 in our network (e.g. 192.168.0.7) and let it continue for few minuets. We launched such attacks few times in between our packet capturing session.

All the traffic including the normal and the attack traffic will get captured in the 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 while attack packets flow information would be used for detecting the attack (Which would be explained further in the paper). Wireshark capture every detail of the packets but we don't need all of the information, we would only be interested 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 except than the performance that other layers of the packet are not analyzed.

To extract IP layer information from the captured packets data extraction program *data_extraction* is written in Python. This program would extract address, port and time information from each of the captured IP packets[7]. It will also divide the captured data into 300 seconds capture window thus creating a sample data which is collection of IP packets captured over the time of 300 seconds. It then write each sample in separate file for further processing. Every sample file represent a one flow information where flow is logically treated as the IP packets captured over 300 second time interval as explained before. This sampling of the packets in the form of flow is the continuous process. Will store the flow information in the form of sample files, we will train learning algorithms using those sample file, then those sample files will be discarded and new samples will be used 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. What ever the new information is learning is augmented with the previous learning to have the correct understanding of the flow.

This learning can be done for the time during the day or given day during the week of year.

[7]packets containing IP information

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.

Destination IP	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: Flow snippet

The sample files created using the *data_extraction* program will be used to create the training set.

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

Creating training set is an intermediate step in which IP packet count for each destination for given protocol(e.g TCP, UDP) is calculated. The training set gives us the flow information for each destination (i.e. how many packets are recorded for a given destination IP address during a time window e.g. 300 seconds). We are using Python program to create training set from the sample files. Our training set look like following in which each row is one training example with destination IP address as label. A training example is $\mathbb{R}^3$ vector whose elements are the number of packets observed for a particular protocol during 300 second time.

	IP Packet count		
Destination IP	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 ran python program to convert each sample file to training set and testing sets. Each sample file has corresponding train/test set file. We store both train and test sets on the file systems so that they can be used for training and testing the algorithms.

There are around 150 protocols managed and assigned by Internet Assigned Numbers Authority (IANA) but most commonly used protocols in the DDoS attack are the ICMP, TCP and UDP protocols. For our 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.

5.1 Machine Learning

According to Tom Mitchell (1998) A computer program is said to learn from 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 build the hypothesis using training set as input then that hypothesis is used to perform predictions. Most common categorization of machine learning algorithm is Supervised and Unsupervised.

Let f be the function which we need to guessed from input vector $X = \{x^1, x^2, ..., x^m\}$. This X also called as training set where $x's$ represent an input variable/feature vector and m is number of feature vectors in the training set. Let h be the hypothesis about the function f. $h \in H$ and $f \in H$, where H is class of functions. Both f and h can be vectors. We select h based on a training set, Ξ, of m input vector examples. In Supervised leaning we know the values of f for the m samples in the training set Ξ. We assume that if we can find a hypothesis, h, that closely agrees with f for the members of Ξ, then this hypothesis will be a good guess for f when Ξ 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, Ξ_1, ,Ξ_R, in some appropriate way. [12]

Supervised algorithm such as One Class Support Vector Machine(One Class SVM) [13] could be efficient to identify the anomalies in the data but it is very process and memory intensive, so training the algorithm for each and every IP address is very costly. Because of the resource constraints of the router our approach is to first cluster the IP address 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. Unsupervised algorithms are fast and consume less resource which make them good to be used on the devices which have less cpu power and less memory.

5.1.1 Feature Scaling

Before feeding data to learning algorithm, we have to do feature scaling, also called Standardizing. Feature scaling is done by removing the mean and scaling the feature to a unit variance value. It is necessary because of the fact that, different features which are at the different scales could cause one feature dominating other 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 then it would be $(1-1)^2 + (2-3)^2 + (3000-2000)^2$. Form this, it is evident that the larger term would dominate the result.

First we will convert training examples into a vector. Each row of the vector is one training

example and each element in the vector is the feature. To standardize the input vector, mean and standard deviation is calculated for each feature in the input vector. Then new vector is created by subtracting the mean from every element of the feature vector and then dividing values of each feature vector by its standard deviation. The new vector created after this step is standardized vector, which is used as input to the learning algorithms.

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

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

5.1.2 Clustering

k-means clustering is one of the most efficient algorithm for creating clusters. The k-means problem is to find set of K points $\{\mu_1, \mu_2, ..., \mu_k\}$ called centroids, where $\mu_k \in \mathbb{R}^d$ and $K \in \mathbb{N}$, for training set $X = \{x^1, x^2, ..., x^m\}$, where $x^i \in \mathbb{R}^d$ and $i = 1, 2, ..m$ is a training example, such that the mean square distance from each training example x^i of training set, to its nearest center μ_k ($argmin_k ||x^i - \mu_k||^2$), is minimum. $m \in \mathbb{N}$, are the number of training examples. [14]

Lloyd's algorithm is most popular heuristic for k-means clustering, which we will be using for our analysis.

Algorithm 1 Lloyd's k-means algorithm

Arbitrarily initialize K cluster centroids $\mu_1, \mu_2, ..., \mu_k$
repeat
  for $i \leftarrow 1, m$ **do**
    $c^i = argmin_k ||x^i - \mu_k||^2$ ▷ Cluster Assignment
  end for
  for $k \leftarrow 1, K$ **do**
    $\mu_k = \frac{1}{n}[x^{(k_1)} + x^{(k_2)} + ... + x^{(k_n)}]$ ▷ Move Centroid
  end for
until convergence

For our thesis we will be using k-means++ algorithm [15] which is improvisation of kmeans, where arbitrarily initialization step is replaced by following simple, randomized seeding technique.

Let $D(x)$ denote the shortest distance from a data point to the closest center we have already chosen then following is the k-means++ algorithm.

Algorithm 2 kmeans++

1: Take one center μ_1, chosen uniformly at random from X.
2: Take a new center μ_k , choosing $x \in X$ with probability $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$
3: Repeat Step 2. until we have taken k centers altogether
4: Proceed as with the Lloyd's k-means algorithm.

For our DDoS attack detection program, we will be using the Scikit-learn libraries. Scikit-learn is most popular and rich open source machine learning software library for the Python

programming language and they have implementation for both kmeans and One Class SVM machine learning algorithms as well as data preprocessing programs.

We will be using training set in the format given in the figure 2 and we will be using k-means++ Scikit-learn library for clustering.

Before doing actual clustering we will first determine the number clusters and centroids[8] Deciding on the number of clusters is important as randomly choosing the number of cluster will not be useful to have correct clustering. We will use Elbow method to find the optimal number of clusters. Elbow method check the percent of variance explained as function of the number of clusters. Following are the Eblow Diagrams for four samples.

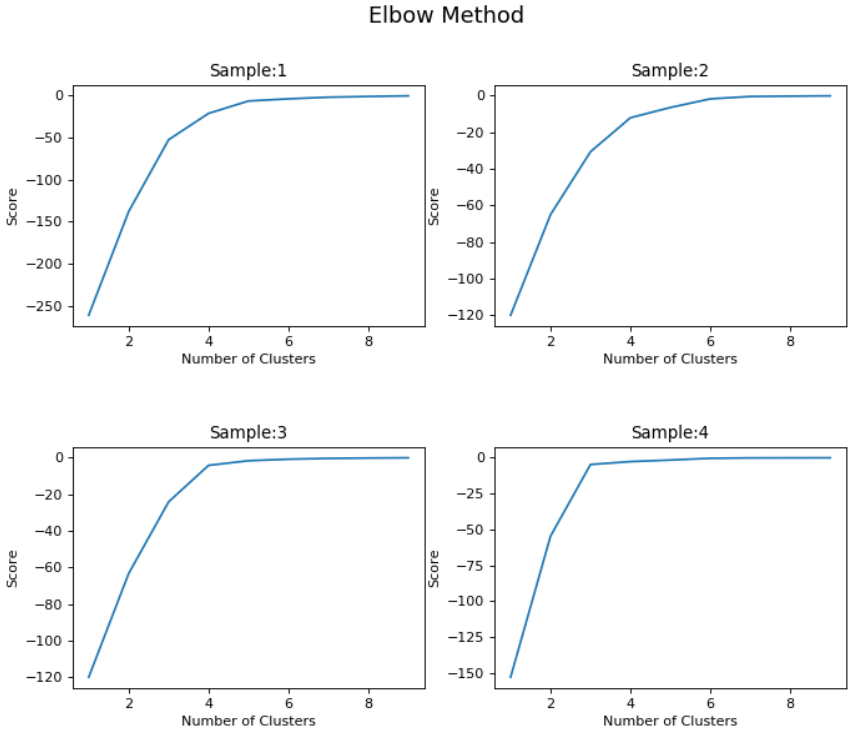

Figure 7: Elbow Method for cluster count detection

Using the elbow method, variance for each cluster number is calculated and the cluster number which produce less variance for the next cluster number is selected as best choice of cluster number for the give sample data. To come up with the best clustering scheme for all the sample we have

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 samples (each sample has file associated with it), we will have cluster centroids for each sample. We will save

[8]Centroids are the central vectors who presents the clusters.

all the centroids and then find the median of all the centroids removing the outliers so that we have good estimate of the correct centroids. This estimated centroids are used for clustering the samples. Following is the example of centroid vector where rows represents cluster label and columns represent the feature.

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 3: k-means cluster centroids

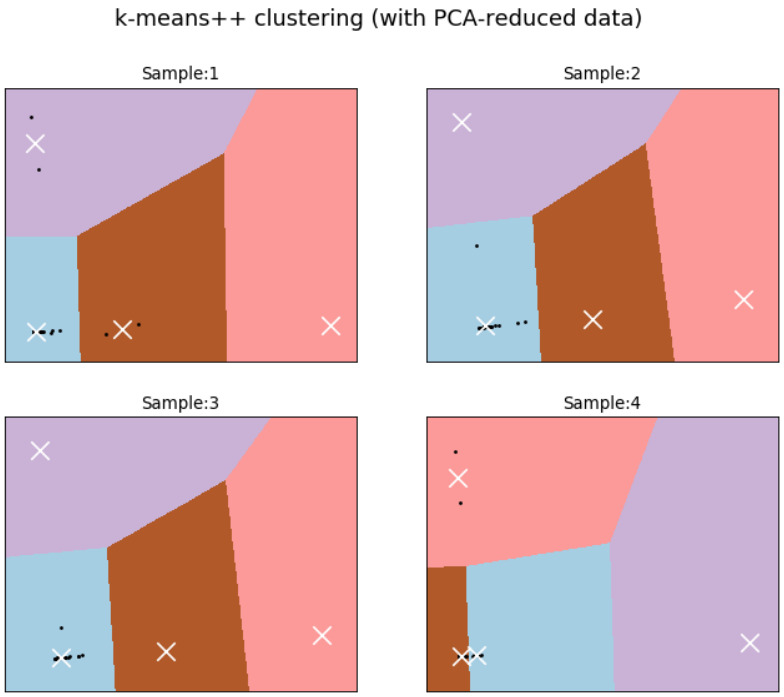

Figure 8: Clustering using k-means++ algorithm

To draw data points on 2 dimension space we had to reduce our 3 dimensional training example into 2 dimension without loosing 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. [16]. We have used Scikit-learn PCA module for this purpose.

Using the centroids obtained in he previous step we will run kmeans++ algorithm and then

test it for accuracy. kmean++ algorithm gives label to each training example. This label is the cluster number to which that training example has been assigned to. So our new labeled data look like following.

Destination IP	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 4: Labeled Training Set (with cluster number)

We will run kmeans++ to for both training and testing set and we produce labeled data for both. We then checked how similar the test set clustering is with train set clustering. For measuring this similarity Rand Index(RI) [17] is used. RI is a measure of how many percent does the test clustering matches with our trained model.

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

where TP, TN, TN and FN are the number of true positives, true negatives, false positives and false negatives respectively.

We have observed that with more number of training sets RI index improves.

Because of the reason that the training sets contain the flow information for different IP on a router during window of 300 seconds, there is very high possibility that same destination IP is captured in multiple training set. Our goal it to find out the correct cluster for destination IP so to achieve this goal we will check all the cluster assignment for a given destination IP and we will select the cluster which has the highest occurring frequency among all the training sets selected for the clustering. We will also count the average number of packets going to a given IP address which will help us reduce the error in deciding the DDoS attack. We will save this labeling and cluster count information for each IP destination on the file system. This information tell us the normal behavior of the packets traveling from the router to a given destination. As we have fixed the centroids for the clustering algorithm, every time in the future we should expect the an IP address to be found in the same cluster if flow of packets for a given destination is as per our our previous knowledge. If there is a DDoS attack on a any destination with flooding, then we can expect to see that destination assigned different cluster.

But from the experiments on different data sets it is found that the destination under attack is labeled with the same cluster number when it was not under attack. This happened because there is no other cluster it can be assigned to so it gets assigned to cluster whoes centroid is nearest.

To avoid such conditions we will have to create boundaries for the clusters. This scheme

will make sure that the attack will be detected even if destination is assigned to same cluster to which it was labeled when there was normal traffic for that destination.

for this we will be using One Class SVM which will help us to detect anomalies in the cluster. Compute and storage requirements of SVM increase rapidly with the number of training vectors, because of the fact that SVM is a quadratic programming problem (QP). So the approch presented in the paper is more efficient because the number of cluster are limited and always be far less than the number of destination IP address. Training one class SVM on clusters will be far less process and memory intensive than training on massive number of IP address.

For example the number of clusters we have create in our analysis are 4 in number while the number of IP address to which packets are flowing form router are 268 which is more than 60 times.

5.1.3 Anomaly Detection using One Class Support-Vector Machine

5.1.4 Support-Vector Machine

Support-Vector Machine is a supervised learning algorithm which tries to classify data. Classification is based on the label of the training set. Consider the training set $\{(x^1, y^1), (x^2, y^2), .., (x^m, y^m)\}$, where $x^i \in \mathbb{R}^d$ is the training example and $y^i \in \{-1, +1\}$ is the label which is the classification value for x^i,

$$f(x) = \begin{cases} \geq 0 \ y^i = +1 \\ < 0 \ y^i = -1 \end{cases}$$

SVM try to create non-linear separation boundary by projecting data points through a non-linear function ϕ to higher dimension space. The data points in space I which can not be separated by a line are projected to the feature space F where there can a hyperplane that separate data point of one class form another. If the hyperplane is projected back on original space I the we get non-linear curve.

For projecting point to the hyperplane a kernel function is used.

The hyperplane is represented with the equation $w^T x + b = 0$, where $w \in F$ and $b \in R$. This hyperplane separated the training example labeled with -1 and 1 into different classes. 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 happen because learned hypothesis fit training examples so well that it fails to generalize new examples. The constant C¿0 is the regularization parameter. It is choose large if we don't want to want to miss-classify training examples but if chosen small then we may miss classify few examples but margin would be large so that most of the points will be far away from the decision boundary. Thus SVM optimization problem is states as.

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

subject to:

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

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

If this minimization problem is solved using Lagrange multipliers then the classification function $f(x)$ can be stated as.

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

α^i here are the Lagrange multipliers and x^i with α^i are called the Support Vectors.

— When this minimization problem (with quadratic programming) is solved using Lagrange multipliers, it gets really interesting. The decision function (classification) rule for a data point x then becomes: f(x)=sgn(i=1niyiK(x,xi)+b) Here i are the Lagrange multipliers; every i¿0 is weighted in the decision function and thus supports the machine; hence the name Support Vector Machine. Since SVMs are considered to be sparse, there will be relatively few Lagrange multipliers with a non-zero value. —

5.1.5 One Class Support-Vector Machine

One Class Support-Vector Machine (One Class SVM) is the extension of SVM which detect boundaries of the training set so that every new training example will be classifies as belong to training set or not. It separate 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 which fit in the trained set region otherwise it will return -1

The minimization function of One Class SVM is slightly different than the SVM

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

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 Support Vector Method For Novelty Detection by Schlkopf et al. basically separates all the data points from the origin (in feature space F) and maximizes the distance from this hyperplane to the origin. This results in a binary function which captures regions in the input space where the probability density of the data lives. Thus the function returns +1 in a small region (capturing the training data points) and 1 elsewhere.

The quadratic programming minimization function is slightly different from the original stated above, but the similarity is still clear:

minw, i, 12w2+1ni=1ni subject to: (w(xi))ii0 for all i=1,,n for all i=1,,n In the previous formulation the parameter C decided the smoothness. In this formula it is the parameter that characterizes the solution;

it sets an upper bound on the fraction of outliers (training examples regarded out-of-class) and, it is a lower bound on the number of training examples used as Support Vector. Due to the importance of this parameter, this approach is often referred to as -SVM.

—-

Once the change in the normal behavior has been reported then router communicate to the destination network. Destination network collect that information to know the source of the attack. This information can be used to know the nature of the attack.

To Communicate the destination network about the change in behavior router can use the existing ICMP protocol. ICMP protocol has been used to provide feedback about problems in the communication environment. ICMP messages are sent in several situations: for example, when a datagram cannot reach its destination, when the gateway does not have the buffering capacity to forward a datagram, and when the gateway can direct the host to send traffic on a shorter route. [11] Similarly we can use ICMP protocol to inform destination router about the change in the flow. ICMP protocol has many unused type code (there can be 0-255 types but as of now only 0-41 are in use) available. We can create our own type to communicate the anomaly in the flow.

When depending on the type and severity of the situation destination host or network can decided whether to inform router to block the traffic.

6 Conclusion

Note: Following need more detailed explanation The benifit of this system is that if a destination detect DDoS attack it could identify the source attack routers and just can ask just to that router to hold on to the packets or descard but not to send them until further notices. This will allow legitimate traffic to come.

A novel way to identify and mitigate DDoS attack is discussed in the paper. With the advance of NVF(Network Virtual Functional) it would be easy to push the learning algorithms to the routers and thus making them efficient in detecting the attacks and then ICMP protocol can be used to communicate location and attack information in between the routers and network.

References

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

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

[3] 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.

[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] 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.

[10] 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.

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

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

[13] 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.

[14] 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.

[15] 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.

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

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