Intrusion Detection and Prevention in Cloud Computing systems using Machine Learning Techniques

November 5, 2020

Group Organization

1217031322 - Atit Gaonkar (L)

1217031218 - Kusumakumari Vanteru (DL)

1217504873 - Chandrasekhara Bharathi Narasimhan

1217390967 - Swarnim Sinha

1217212217 - Jasvinder Chugh

1213233305 - Smruti Sudha Dash

1217135543 - Akanksha Magod

1217136895 - Santhosh Bijinemula

Summary:

Cloud computing systems are being widely used to provide scalable , high availability and secure services and it has come out to be an efficient way to manage computing resources on the the fly. However , with the ease of access and scalability of the resources comes threats of cyber attacks to the cloud system. So security challenges becomes an important aspect of cloud computing in current scenario. In this report a study of different types of cyber attacks, which could adversely affect the cloud computing resources and its services along with its detection and prevention using machine learning have been done by diving the attacks into 4 major categories namely intrusion detection , botnets detection , distributed denial of service attacks and malware attacks.

Contents

1 Introduction **5**

1.1 Motivation and Background . . . 5

1.2 Goals and Scope of the Project . . . 5

2 Project Overview and Member Responsibilities **6**

2.1 Introduction to Cloud Computing . . . 7

2.2 Threats to Cloud Computing . . . 8

2.3 Supervised machine learning . . . 8

2.3.1 Classification . . . 8

2.3.1.1 Naive Bayes classification . . . 9

2.3.1.2 Logistic regression . . . 9

2.3.1.3 Support vector machines . . . 9

2.3.1.4 Decision trees . . . 10

2.3.1.5 Random forest . . . 10

2.3.2 Linear Regression . . . 10

2.4 Unsupervised Learning . . . 11

2.4.1 Clustering . . . 11

2.4.1.1 K-means . . . 11

2.4.1.2 DBSCAN . . . 11

2.4.1.3 FCM: Fuzzy C-means Clustering . . . 11

2.4.1.4 KNN:K-nearest neighbours . . . 11

2.4.2 Artificial Neural Networks . . . 12

2.4.3 Deep Neural Networks . . . 12

2.4.4 Anomaly Detection . . . 12

3 Intrusion Detection and Prevention **13**

3.1 Existing state . . . 14

3.1.1 KDD'99 data set . . . 15

3.1.2 NSL-KDD . . . 15

3.1.3 UNSW-NB15 . . . 15

3.2 State of the Art Techniques . . . 15

3.2.1 SVM based technique . . . 15

3.2.2 K-means and KNN based technique . . . 16
3.2.3 K-means and SVM based technique . . . 17
3.2.4 Genetic based feature algorithm and Fuzzy SVM based technique . . . 18
3.2.5 FCM and SVM based technique . . . 18
3.2.6 Neural Networks and Deep Learning based technique . . . 19

4 Botnet Detection **22**
4.1 Life cycle of Bots . . . 22
4.2 Modes and architecture of Botnets . . . 23
4.2.1 Direct . . . 23
4.2.2 Centralized Model . . . 24
4.2.3 P2P or Decentralised . . . 24
4.2.4 Hybrid . . . 25
4.3 Botnet Communication Protocols . . . 25
4.3.1 IRC based centralized Botnets . . . 25
4.3.2 HTTP based centralized botnets . . . 26
4.4 Botnet Counter Measures . . . 26
4.5 Botnet Detection and Machine learning . . . 27
4.5.1 Unsupervised Learning based Botnet Detection . . . 28
4.5.2 Supervised Learning based Botnet Detection . . . 30
4.6 State of the Art . . . 31

5 Distributed Denial-of-Service (DDoS) Attacks **32**
5.1 Introduction and Overview . . . 32
5.2 State of Art Techniques . . . 33
5.3 DDOS Attacks and Detection methodologies . . . 33
5.3.1 DDoS Intend and Launch methods . . . 34
5.3.2 Why DDoS: Attacker's Incentives . . . 34
5.3.3 Conventional DDoD Detection Mechanisms . . . 34
5.3.4 Using Machine Learning to detect DDoS Attacks . . . 34
5.3.4.1 Basic Machine Learning . . . 34
5.3.4.2 Incorporating C4.5, Naive Bayesian and K Means Algorithms . . . 35
5.3.4.3 Self learning methods . . . 36
5.3.4.4 Combining Conventional and Modern Detection Techniques . . . 37

5.3.4.5 DDoS Attack Detection on Source Side . . . 38
5.3.4.6 DoS attack detection on Cloud Provider's side . . . 41
5.3.4.7 Artifical Intelligence technique . . . 42

6 Malware Attacks **44**
6.1 What is a Malware . . . 44
6.2 Types of Malware . . . 44
6.3 Machine learning for Malware Analysis . . . 45
6.4 State-of-art ML Techniques for Malware Detection . . . 45
6.4.1 One-class SVM . . . 46
6.4.2 Random Forest and Naive Bayes . . . 48
6.4.3 Deep Reinforcement Learning . . . 49
6.4.4 Deep Learning . . . 51
6.4.5 Convolutional Neural Network . . . 52

7 Conclusions and Recommendations **55**
7.1 Intrusion Detection and Prevention . . . 55
7.2 Botnets . . . 56
7.3 Distributed Denial-of-Services Attacks . . . 59
7.4 Malware Attacks . . . 60

References **63**

1 Introduction

1.1 Motivation and Background

Cloud computing provides users with dynamic capabilities to store and process their data on (Public, Shared or Private) data centers. Different organizations use the cloud in a variety of different models. Some use PaaS (Platform as a service), SaaS (Software as a service), IaaS (Infrastructure as a service). Since the advent of cloud technologies and the booming advantages of cloud computing, there has been a huge paradigm shift taking place where business models are being changed from on-premise to cloud.This shift from on-premise to cloud have rendered an increase in attack surface, allowing attackers to take advantage of errors occur during this shift or vulnerabilities that arise later. Security concerns in cloud computing generally fall into two broad classes: security issues faced by cloud providers and that faced by the customers.

Hence the responsibility is shared. There are several benefits of cloud security such as Centralized security, Reduced costs, Reduced Administration, Reliability etc. Cloud security faces numerous challenges. Insider Threats – Privileges, Malware & External Attackers, Network Breaches, Misconfigurations to name a few. These attacks have become sophisticated and continue to grow at a rapid pace, so to detect and counter these attacks have become a challenging task [43]. We should employ every possible measure to secure the cloud environment. Having an outdated countermeasure system for detecting and preventing these attacks, would not be feasible. In-order to dynamically secure the cloud systems from such evolved attacks, it would be logical to use machine learning and its components to further strengthen our cloud systems by understanding the continuous changes in the cloud environment.

1.2 Goals and Scope of the Project

Our goal of this project is to study various techniques revolving around Intrusion detection and prevention. We aim to study various mechanism and techniques to detect and mitigate Distributed Denial-of-Service, Botnets and Malware attacks. We also compare and contrast these techniques to apprehend pros and cons of these mechanism along with further understanding of the state-of-art techniques. The scope of this project is limited to intrusion detection and prevention in cloud computing. The term 'Intrusion detection and prevention' encompasses various aspects of security in cloud computing. Hence our paper will revolve around four main topics. Intrusion Detection and Prevention(Distributed Denial-of-Service Attacks, Botnets and Malware). We also discuss on numerous ways to detect and prevent from these attacks.

2 Project Overview and Member Responsibilities

Name	Responsibilities
Atit Gaonkar	Threats to cloud computing. Need of Machine Learning in Intrusion Detection and Prevention. Study of Distributed Denial of Service attacks, detection methodologies, ways in which machine learning can improve the detection of Distributed Denial of Service attacks.
Kusumakumari Vanteru	Introduction to Cloud Computing concepts, Overview of types of malware detection approaches, In-depth research of SVM, Deep Reinforcement learning and Convolutional Neural Networks for detecting and classifying Malwares, Hybrid Cloud Malware Detection (MCD) methods, IoT-based techniques, Comparative analysis of the papers, challenges faced and scope for future work.
Jasvinder Chugh	Overview of various supervised machine learning algorithms used in cloud security systems. Introduction of threats posed by botnets in cloud computing systems. Review on Life cycles of bots, botnet architectures and communication protocols used on C&C architectures. In-Depth study of Botnet counter measures and machine learning (Supervised and unsupervised) based botnet detection mechanisms.
Swarnim Sinha	Summary of the project. Overview of the unsupervised learning algorithms along with clustering algorithms. Comprehensive analysis of current state being used in intrusion detection and the data set that could be used for further analysis. In depth review and analysis of state of the art machine learning techniques being used for intrusion detection along with study of challenges of using machine learning , recommendation for the same and conclusion
Akanksha Magod	Overview of machine learning algorithms used in botnet detection, and conclusions derived from the respective research paper.Analysis and review of future research and development needed in these fields for better botnet detection and prevention,as well as analysing current state of the art findings in botnet detection.

Chandrasekhara Bharathi Narasimhan	Overview of Cloud Computing, Analysis of supervised and unsupervised Malware detection techniques based on machine learning, In depth study of research papers proposing the use of one-class SVM, Naive Bayes and Random Forest classification algorithms for Malware detection.
Santhosh Bijinemula	Overview of Un-Supervised Machine learning algorithms and Intrusion detection and Prevention. Analysis of neural networks and deep learning techniques to detect and prevent intrusion detection and prevention in cloud computing networks.
Smruti Sudha Dash	Overview of cloud computing and different types of attacks on cloud. A comprehensive study of different types of Denial Of Service Attacks on Cloud. In-depth study of various machine learning models that can be leveraged to detect DoS and DDoS attacks on cloud. Analysis and comparison of various points of DoS and DDoS attacks on Cloud and Intrusion Detection systems that can be implemented to detect the attacks.

Table 1: Team Member responsibilities

2.1 Introduction to Cloud Computing

The definition of Cloud Computing differs from person to person based on how one uses it. Most people refer to it as the service you pay for which takes care of your hardware and sometimes software needs as well over the internet to support your systems and applications. From simple Google searches to email services, we make use of Cloud computing in the form of thousands of clustered PCs located at different parts of the world. The basic idea of Cloud computing is that the service you use is managed by someone else on your behalf. This takes away the burden of having to buy software licenses and keeping them up to date or the viruses that might affect your computer. Cloud services are provided on demand on a pay-as-you-go basis. It means we don't have to pay for massive computing power and not use it at all. It also comes in the form of public and private cloud services which gives the user complete control over the systems. Following are the different types of Cloud Computing Services

Software as a Service (SaaS) means the user can use a finished application provided by the cloud.

Platform as a Service (Paas) means the users are free to develop their own applications using the software tools and hardware from a separate provider.

Infrastructure as a Service (IaaS) means the user buys from the service provider everything from hardware to storage spaces and builds the entire application on top of that.

Cloud computing lets the users pay only for the services they want, at their convenience, cutting the upfront costs of computing power and hardware. This flexibility is what attracts more and more users and services towards this technology.

2.2 Threats to Cloud Computing

There are numerous threats and attacks to cloud computing that raise concerns and hence it is necessary to address these threats and attacks. In this section we will discuss major attacks that could be detected and prevented using machine learning.

Cloud malware injection attacks are done to take control of the system either the platform, software or the infrastructure. Generally the requests are rerouted to hackers resources. Malware attacks can also be a means to other larger attack scheme, such as using the machine as zombies to successfully conduct a large scale Distributed Denial of service attack. Once the machines are compromised these can be converted into bots. These bots can then be used to achieve a DDoS attack

Distributed Denial of Service are ones of the most frequent used attacks in today's date. DDoS inflict serious damage and affect the cloud performance. DDoS attackers usually uses compromised computers (zombies) by taking advantages of known or unknown bugs and vulnerabilities to send a large number of packets from these already captured / compromised servers, to render the service unavailable or to exhaust the resources and deny it to legitimate users[41]. Intrusion and Insider Attacks is a huge concern. This can sometimes lead to data loss and data breaches as the intruder is present in an unauthorized area.

2.3 Supervised machine learning

Supervised machine learning techniques are widely used in majority of practical systems.The idea of these algorithm is to learn a mapping function from input to output. This is called supervised learning because training data set contains the ground values of y, also called as labels, and using those input values we continuously predict the values of y and compare it with given ground truth to make them more alike the parameters for next iterations. This training ends when a desired accuracy level is achieved. Supervised algorithms are categorised into two sub groups:

2.3.1 Classification

A classification method is used when we need classify/categorise the input values in predefined labels/-categories. There are two main types of classification:

- *Binary*: This are used when we want only two output labels like true or false, spam/not spam etc.

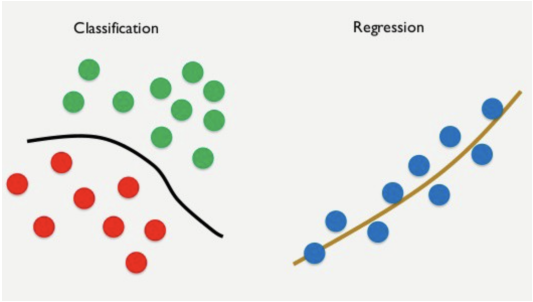

Figure 1: Classification vs Regression

- *Multi-class* : This is used when we want to classify input into multiple labels/classes like sentiments(happy, sad, angry and frustrated) etc.

Few of the popular classification algorithms are:

2.3.1.1 Naive Bayes classification Naive Bayes is a family of probabilistic classifier. It is based on Bayes theorem of probability and assumes that all features are independent with each other. These classifiers are highly scalable and are comparatively less computation hungry than other classification techniques. In simple Bayesian classifier we extract features from data set and find condition probability of input classified as a particular class given feature values of input. We do this for every class and based on our probability thresholds we decide the predicted class of the input.

2.3.1.2 Logistic regression One the most basic problems of classification is to separate classes which cannot be linearly classified. Logistic regression is a a statistical model that solves this problem using logistic function. It uses odds ratio of probability of an event occurring and a logit function which is nothing by logarithm of odds ratio. To convert logit function into probability and classify input data, the output of logit is passed through an activation function (sigmoid, tanh etc). Finally, the output is passed through a quantizer to determine predicted classes.

$$OddsRatio = \frac{P}{1-P} \qquad logit(P) = log\left(\frac{P}{1-P}\right)$$

$$Logit(P\,(y=1|x)) = W0X0 + ... + WmXm = Sum\,(WiXi) = W^T X$$

where P(y= 1—x) is the conditional probability of class 1 given features x.

2.3.1.3 Support vector machines Support vector machines/SVM are extension of perceptron algorithm. This algorithm is more of a optimization problem where we try to find the decision boundary which divides the two given classes in such a way that distance of closest points to decision boundary of each class is minimum in hyper plane. These points are called support vectors. SVM by default

can classify linearly separable data points. To use them for non linear classification we can use kernel trick/kernelizing through which we can map data points to multi dimensional feature space. The basic idea is to transform features into higher dimension by applying the projection, separating the classes in hyper-plane and then applying inverse projection to transform features back. Some of the widely used kernels are Polynomial Kernel, Laplace RBF Kernel, Gaussian Kernel, Sigmoid Kernel, Radial Basis Function (RBF), Anove RBF Kernel, etc.

2.3.1.4 Decision trees These are hierarchical classification algorithm where we design a question flow graph to predict the class of the input variables. These graphs are often called classification trees. Classification trees contains a root node, leaves representing the class labels, branches/conjunction points which lead to the class labels. Decision trees can also be used for continuous value prediction, those decision tress are called regression trees. Some of the common hyper parameters of decision trees are: 1. Maximum & Minimum Number of Samples : To avoid unproportional distribution of sample across leaves maximum number of samples is specified at start of the algorithm. 2. Maximum Depth: This is an essential hyper parameter which decides the maximum height of the decision tree,i.e., from root to leaves. To avoid over fitting, tree pruning is done to reduce depth of the decision tree.

2.3.1.5 Random forest This is considered an extension of decision trees. If a data set has lots of features/ decision tree is over fitting over a certain data set then random forest is the way to go. Random forest is an ensemble model where we develop a bunch of weaker models and combine them to get better accuracy and reducing the complicated models of decision trees. The idea is to segregate the features in batches, develop decision trees on a batch and finally applying majority voting to get final class label. Algorithm normally has 4 steps: 1. Drawing a random bootstrap of size n. 2. Growing decision tree by randomly selecting m features for maximizing information gain. 3. Repeat the above two points k times. 4. Aggregating the predicted values and predicting the final class using majority voting.

2.3.2 Linear Regression

Linear regression is a method we use when we predict a variable's projection(or score) from the projection(or score) of another variable.The variable we are predicting is called the response variable and the one we use to predict this criterion variable is called the predictor variable.These variables are usually continuous(quantitative) variables. This method is mostly used to study statistical relationships,where relationships between variables is not proper.Hence we find the most appropriate line, or the "best fitting line" across these variables.For this line the n prediction errors are as small as possible overall. Summarizing the conditions that overlay in the "simple regression model": (1)At each value of the predictor, xi,

the mean of the response variable, is a Linear function of xi. The errors at each value of the predictor are (i)independent,(ii) has equal variances and (iii)is normally distributed.

2.4 Unsupervised Learning

Unlike the supervised machine learning, unsupervised mahcine learning is a technique in which the algorithm performs classification of data sets which doesnt have any already existing classification or set of labels attached to it. Cluster analysis and principal component analysis are the two important methods used in unsupervised learning. Clustering involves grouping or categorizing a set of data into a common group based on some common feature the data posses. Principal component analysis determines the important features that contribute to the variance of the data set and then the top features that contribute the most can be selected for further analysis. This also helps in reduced dimensionality of the data which can help in better analysis of the data by the algorithms. The major techniques in unsupervised learning are grouped in the following ways:

2.4.1 Clustering

2.4.1.1 K-means It is a vector quantization method which partitions the data set into K clusters in which each data belongs to a mean in the cluster , which serves as a model or prototype of the cluster . The mean is called as the centre of cluster or centroid.

2.4.1.2 DBSCAN It is one of the most cited algorithm in unsupervised clustering in scientific literature. It is based on density clustering. If a set of data points are given , the data points that are close to each other are grouped or classified together. The points that lie in the low density region are considered as outliers because there are not much nearby points to be grouped together.

2.4.1.3 FCM: Fuzzy C-means Clustering It is a type of clustering in which a data point can belong to multiple cluster. Each data point is assigned points or score for each cluster it belongs to based on its similarities. As these scores are not probabilities so they don't add up to 1.

2.4.1.4 KNN:K-nearest neighbours KNN is used for clustering of data points where nearby points are considered to be similar. Based on the value of K , the number of data points are selected which are closest based on euclidean distance and outputs the class of the data

2.4.2 Artificial Neural Networks

A neural network or artificial neural network (ANN) is a group of artificial neurons which are interconnected and that uses computational models for processing the information based on cognitive/connective approaches for computation. Artificial neural networks (ANNs) were initially inspired by distribution and communication between the neurons in the biological system. Neural networks are non-linear decision making tools which can be used to model complex relationships between inputs and outputs and to find patterns in a data set.

2.4.3 Deep Neural Networks

A deep neural network (DNN) is Artificial Neural Network which has multiple layers between the input and output layers. Each mathematical computation as such is considered a layer, and a compound, complex DNN have many layers, hence this is called deep neural networks. DNNs can model more complex non-linear relationships than neural networks. The extra layers enable the feature composition from previous layers, potentially modeling much more complex data with fewer units than that of a similarly neural network. Deep learning algorithms can be applied to unsupervised learning tasks which is very important in real-life applications because unlabeled data are more abundant than the labeled data.

2.4.4 Anomaly Detection

Anomaly Detection deals with finding rare events, which occurs when significant amount of data is altered giving raise to suspicions. From the network intrusion detection point of view, the events are not rare events, but unexpected threats which might compromise the system. The pattern of these events is not compatible to the definition of a rare event outlier, unless it has been aggregated appropriately. Unsupervised anomaly detection techniques, in particular detect anomalies in an unlabeled test data assuming that most of the samples in the data are benign by looking, that seem to fit least to the rest of the data set.

Detailed Results

In the following sections, we will be discussing a detailed research study and results of each of the 4 subtopics for Intrusion Detection in Cloud using Machine Learning techniques.

3 Intrusion Detection and Prevention

Intrusion detection is the process of tracking the incidents occurring in a computer network or application and analyzing them for detecting any incidents, which are harmful or against the security policies or practices. A system which makes this automated is called intrusion detection system (IDS). A system that has the features of an Intrusion detection system and which also stops the intrusion incidents is called Intrusion Prevention System (IPS). In general, both the systems have many of the same features, and the admins can enable or disable features in IPS systems, making them to work as Intrusion Detection Systems. The term 'IDPS' is used to refer both IDS and IPS technologies.

IDPS can detect when an attacker has compromised a system by exploiting the vulnerabilities in the system. The IDPS logs the data on the incident and report the incident to the concerned security individuals, to reduce the damage by giving an immediate response action to the event. Also, IDPSs can also be configured to recognize events against the violations of standard policies.IDPSs can identify reconnaissance activity, which tells us the particular system characteristics that is of importance to the attackers. IDPS is also used to detect the threats, particularly the frequency and features of attacks, so that needed measures can be considered and performed.

IDPS technologies use various methods to detect attacks. The primary detection methods are signature based, anomaly based, and stateful protocol analysis. Signature is a pattern which pertains to a type of attack. Signature-based detection is the process of weighing up signatures with the previously detected incidents to find the attacks. This is a very efficient method to detect known attacks but very inefficient at detecting unknown attacks. Anomaly-based detection is the process of comparing the incidents which are considered regular with those previous incidents to identify significant variations.An IDPS using anomaly based detection has profiles which tells us the common behavior of things such as users, network connections. The major use of anomaly-based detection methods is that are very effective at detecting unknown attacks. Stateful protocol analysis is the process of comparing previous profiles of normally accepted definitions of benign protocol activity for each protocol state against previous incidents to identify variations. In the anomaly detection, it uses host or network profiles, here in the case of stateful protocol analysis it completely relies on business-developed profiles which tells us on how the protocols should and should not be used. The stateful term pertains to that, the IDPS can understand

and track the state of network, transport, and applications which have a state.

Intrusion Prevention technologies differ from Intrusion Detection technologies by one characteristic. i.e. IPS technologies can react to a detected threat by taking measure to prevent it from happening. They use several response methods, which can be divided into the following groups:

• The IPS stops the attack itself.

• The IPS changes the security environment. The IPS could change the configuration of other security controls to disrupt an attack.

• The IPS changes the attack's content. Some IPS technologies can remove malicious portions of an attack to make it benign. Some IPS sensors have a learning mechanism that reduces all prevention actions and instead indicates when a prevention action should have been performed. This allows us to monitor and change the required configurations of the prevention before enabling prevention actions, which reduces the frequency of inadvertently blocking benign/good activities.

3.1 Existing state

The traditional methods for intrusion detection is based on pattern detection and on intrusion rules. However , there were some issues in performance of these , when multi cloud scenario came into picture. Some of the ways used as discussed in [29] are:

Misuse Detection Techniques: In this technique , the data found in audit logs matches with the attack. The defence is very high for the type of data that the system has previously seen as it was already present in the audit data and rules were modified accordingly , but faces issue for data which the system has not seen previously.

Anomaly Detection Technique: This technique is based on observing and analyzing data that are legitimate and if there is a deviation from the observed pattern of network traffic , that traffic is treated as an anomaly and assumed as an attack on the system because its not matching the definition of a legitimate traffic for the system.

Hypervisor and virtual machine introspection: In this technique , the inspection of hypervisor is performed to check the detection and different methods are used to protect it like adding one more layer of virtualization or changing the design of the hypervisor.

Hybrid Technique: This technique is implemented through combining the above discussed method which helps in avoiding the shortcomings of the individual techniques and build an overall a more secure system.

For the analysis in the further sections, there are certain types of data set that were used to perform the experiment or analyze different methods and algorithms for machine learning. A brief overview of

those data sets are given below , which will help in better understanding of the analysis of this topic

3.1.1 KDD'99 data set

The KDD cup was an International Knowledge Discovery and Data Mining Tools Competition.The goal of this competition was to collect all types of network traffic records. The competition errand was to construct an intrusion detection system, which was able in recognizing between "bad" traffic, called attacks or intrusion, and "good" typical traffic. As a result of this competition, a huge amount of records were collected and bundled into a information set called the KDD'99

3.1.2 NSL-KDD

The KDD99 data was cleaned up like removal of redundant data (the use is explained in later sections) and revisions were performed on it , which came to be known as the NSL-KDD data set.

3.1.3 UNSW-NB15

In the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) , network packets were created which resembled actual latest normal traffic activities and attack behaviours which came to be known as UNSW-NB15

3.2 State of the Art Techniques

There are various approaches and combination of machine learning algorithms that were combined to achieve maximum efficiency in detecting intrusion detection and to subsequently prevent it. The methods have been discussed below and is divided as per the techniques used for the analysis.

3.2.1 SVM based technique

In one of the study, [29] two feature selection algorithms were used to determine the optimal feature. One was univariate and the other one was principal component analysis. Univariate examines each feature's contribution in classifying the data. Principal component analysis helps in identifying important features in the form of dimensions which helps in reduction of training complexity , especially for huge datasets. It uses liner algebra for doing that.

The dataset was divided onto 5 different categories to get a better picture of its contribution in classification. SVM was trained using a penalty parameter C which was different for each category of data. The kernel functions used were also different like sigmoid, linear and polynomial. The general

purpose category with sigmoid function and C as 20, achieved the highest accuracy of 92 percent on the validation set and its SVM parameters.

SVM was trained for different values of K for univariate method of selection. K was varied from 2 to 28. The accuracy was highest when the K value was set as 4. The svm was retrained with optimal value of K, which is 4, but the SVM parameters were changed . With the sigmoid and polynomial function and different values of C , almost the same accuracy was achieved.

The SVM was trained using principal component analysis with different value of components ranging from 2 to 29. The highest accuracy was achieved with the component value of 8.

In general , the highest accuracy was achieved in the general purpose category with sigmoid function and C values as 20.

3.2.2 K-means and KNN based technique

In one of the novel approach discussed in [30] a combination of classification and clustering is used , so that a data is assigned to its closest cluster. the NSL-KDD data is normalized with standard deviation in the beginning. Then 20 best features were selected through principal component analysis , which were then clustered using K-means method. The next task of of finding the cluster centre was done through majority voting. With K as 1 in the KNN algorithms and running it through maximum number of iteration the final/best cluster was decided as the classifier. So , now the label of each data was decided through the nearest cluster it belonged to. The detection accuracy is calculated when it classifies the test data. When compared with the regular KNN , there was an increase in the accuracy in this model. This model can be achieved with lesser complexity and less amount of memory utilization as compared to the traditional machine learning approaches. Considering K as the number of neighbours , d as the dimension of data and N as the number of training samples the time complexity of KNN is OdN^2, where for the approach mentioned above , the time complexity is found to be OdN^2/k^2

In this approach , the NSL-KDD dataset was used instead of the KDD99 data , because of the advantages it offered like , removal of redundant data to avoid the bias while classifying. It also has reasonable amount of bifurcation for training and testing data as well. The clustering of data was done as follows : DoS(Denial of Service attacks), Probe (Probing attacks) in which the the system detects the attack as a recognizable fingerprint, R2L (Root to local attacks) in which local access is gained by the attacker on local machine, U2R (User to root attacks) in which a user access is gained by the attacker but later on the access is changed to root by the attacker

The experiment was performed in two parts. In the first part , only the training dataset is used for the proposed alogorithm and in the second experiment along with the training data, a part of the testing

data is also used. Similar dataset is also used for the tradational KNN algorithm as well. Same values of K was used for both the experiments for better analysis.

The accuracy of the proposed method was better than the regular KNN method. The accuracy was better when the K was large for both the approaches

3.2.3 K-means and SVM based technique

In one more hybrid method for intrusion detection [31] a combination of K-means and SVM is used to detect the anomalies. K -means was used for clustering and the learning model was build from SVM using the clusters.

The data set that was used for the experiment was UNSW-NB15. In the preprocessing stage , a traffic analysis was performed. It included the capture of packets between the network through wireshark, creating a dump file. Wireshark gives the origin and destination of each packets in the network. After the data was collected , the noisy data were removed. Also there was dimensionality reduction which helped in elimination of outliers. Principal component analysis was performed , so that only the important features were selected in the analysis. Because , removal of certain data , which doesnt contribute in the feature wont affect the final result much but the presence can affect the analysis in a negative way.Here , the top 10 features out of 16 were considered in PCA

First the data was fed to unsupervised clustering method , that is k-means clustering. Since the data packets has no special feature of their own , so in the clustering case , it was considered that the cluster which was large and dense was normal traffic and the one which was sparse or small were the anomalies These clusters or the output of K-means clustering was fed into SVM . As the testing of supervised learning method is faster than the unsupervised ones , so SVM was selected. 75 percent of the data set were used as training data and 25 percent were used as testing data. In phase one , after performing the PCA, 25 out of 36 features were selected to build the supervised model. First the model was trained with 5 features and then consequently , the features were increased for each training session. The first twenty five features had almost the same results as with less number of features , so 25 features were selected and the data was divided into training and test data as stated earlier For the clustering of the data , three parameters were defined namely , alpha , beta and gamma to label the cluster. All the values were between 0 to 1 , but the value of alpha was selected to be near 0, so that small cluster could be selected , while aplha and gamma were kept near 1 , so that large and sparse clusters could be selected respectively. In phase three the labelled cluster were fed to SVM. The SVM classifies 1 as normal activity and 0 as anomaly.

The overall performance of the proposed solution was 86.7 percent.

3.2.4 Genetic based feature algorithm and Fuzzy SVM based technique

A genetic based feature algorithm was also proposed in this paper [32] along with Fuzzy Support Vector Machine for better classification. Fuzzy SVM is a variant of SVM in which the dataset can belong to multiple class. It is similar to FCM as explained in the unsupervised learning section. The dataset used for this is KDD99 Genetic based selection algorithm is based on forming subset of suitable features that helps in better classification. It also leads to removal of redundant data which misclassifies the dataset and reduces the efficiency. It helps in better performance in globally varied data and its clustering. The feature reduction due to this algorithm helps in faster running of the algorithm and hence a better testing.

After the generation of subset from the genetic algorithm , the optimal subset is determined based on different criterion. But the optimal subset generated from one criteria may not be the optimal subset for another criteria. Further , a stopping criteria was also decided , based on which the algorithm needs to be stopped. The stopping criteria was decided when the maximum number of iteration was reached or when a good quality of subset was selected which meant that the fitness value of the previous iteration was less than the threshold value , in the current iteration.

The best way to detect the accuracy was to detect the original value and compare it when the final result. But in real world, availability of ground truth is not available , so in this approach , the detection of intrusion was detected based on changing the feature and the accuracy was studied. The accuracy was studied through both the methods and the results were within the same range, so it was concluded that the selection of feature could be used in improving the performance of this approach. The results of this approach were compared with Fuzzy SVM and SVM. It turned out that with the use of genetic feature selection the detection accuracy rate was around 98.5 percent and error rate was around 3.3 percent with 112 mili seconds of training time. In the regular SVM , the accuracy rate was 83.58 percent and the error rate was 6.5 percent with training time of 846 mili second. In the Fuzzy SVM , the accuracy rate was 92.4 percent and the error rate was 5.2 percent with training time of 223 mili second. It can be concluded that the introduction of genetic based feature selection algorithm reduces the training time and the error rate compared to other types of SVM and increases the accuracy of detection as well.

3.2.5 FCM and SVM based technique

The security of cloud computing environment can also be enhanced by using a combination of FCM and SVM as suggested in [33].The dataset used for this was NSL-KDD. The FCM was used before SVM to divide big data sets into small cluster for effective learning of SVM. The division of datsets allowed SVM to learn in a timely manner and in an efficient way. It can be said that FCM enhances the performance of SVM. The results of this approach was analyzed through accuracy , incorrect classification rate, false

negative rate , true positive rate , Precision , Recall and F1 score.The accuracy of this setup was found out to be 99.37 percent while the incorrect classification rate was as low as 2.6 percent. The false negative rate was 0.005 percent while the true positive rate was 97.9 percent. Furthermore the precision was 96.2 percent and the recall and F1 score was found out to be 97.9 percent and 97 percent respectively.

Talking about accuracy , neural networks can also have accuracy as high as 99.98 percent as per [34]. The neural networks have been discussed in a separate section in this report

3.2.6 Neural Networks and Deep Learning based technique

With the fast development of internet, attacks are rapidly changing, and the threats getting much more crucial in normal networks, smart grid networks and cloud-based computing networks. Originally, IDs require manual intervention and rules to find the intrusion behaviors. However, the adaptation of the rules is getting behind the new attacks happening in the systems. This is leading to failure of attacks which are unknown. This is where machine learning plays a major role for the data classification. As we have previously how various machine learning techniques like KNN, SVM etc use various data to train, and learn the insights of the data and improve the accuracy of intrusion detection. The best part of these techniques is the generalization capabilities which reduce the false negative rate and false positive rates.

However, these classifiers need to extract the features and select them to reduce the redundant features if any. Also, these techniques required a lot of human intervention and knowledge for data processing to find patterns in the data. This in turns might lead to error prone classification. Moreover, when it comes to cloud-based computing systems, we tend to deal with big data thus requiring a vast of training data to train, which can be very challenging as we need to cope up with the speed and computation overhead. Some of the machine learning techniques discussed so far build models which are based on the previously features data and parameters, which might not be suitable for a new attack. Thus, when a new attack comes a much efficient self-learning technique is required to prevent them. Deep learning has a better performance to classify through big data analysis and performs better compared to other methods. It enables a deeper and better analysis of the data across the network and quicker detection of any threats/anomalies. One of the recent researches in this area, is to build a stacked auto-encoder to extract the features of the actual data to classify later when required. However, such method would require high storage and computation power. Hence, these type of techniques can only be used in networks which are quicker.

To mitigate this drawback, in one of the study [37] a scheme was proposed which has 2 deep learning based parts, one part to do classification and the other part to perform feature selection. First, the selection part is done by De-noising autoencoder(DAE) where the process adds weights to loss function

which in-turn makes the selection results better by placing the focus more on the attack data. i.e the attack datasamples are selected using a strategy for good performance. Second, the classification is then done by Multi-layer perceptron(MLP) to do a high level performance by using minimal parameters. Thus, using minimal resources i.e the data samples and minimal parameters, the intrusion detection is performed which makes it practically useful for any network. One other approach proposed in the study [38], is a combination of deep auto-encoder and the Random Forest (RF). Unlike the conventional auto-encoder here a non-symmetric approach data dimensionality reduction is done. This in-turn helps in reducing the overhead because of the high performance strength of RF algorithm.

Coming to Cloud computing systems, there are multiple systems at which the attack can happen. It becomes very difficult for a single Intrusion detection system to detect all the possible attacks at the site at which the threat happens. So, multiple IDs which are of different providers can cooperate with each other the feed backs and knowledge and thus they can improve their respective skills of detecting a threat if any. However, the feedback being shared might have significant delays as the computation that takes place at each site is very high. Also, this is very costly in terms of computation because of the number of sites in the network. Thus, if suspicious events are delayed and the feedback is missed, this co-operative architecture would fail. To make this work, the authors in the study [39], come up with a pro-active cooperation strategy. In this method, IDSs need not wait for the feedback from others to perform or make a decision if there is a suspicion activity. This makes it feasible in real time because the decisions can be made immediately and required measures can be taken. In this case, a stacked de-noising auto-encoders are used as before, however the input data in to it would be the partial feedback that the site gets from different sites and the feedback would be re-constructed from partial feedback. Now, this would learn and extract features that are important and robust to incomplete IDS's feedback. This enables us to take decisions without using any aggregation strategy on the partial feedback, and in acceleration of the decision process.

A traditional auto-encoder is unsupervised which is used to learn data codings. This is used as a building block for deep learning networks. We use this before the layers of deep learning, which can help to improve the performance by intializing to proper suitable weights rather than using random initialization. In this case of unsupervised learning, the loss function is on how different the re-constructed input data is to the actual input data. Now, to make this auto-encoder suitable or robust for learning in-complete ID's input feedback, we add a noise to the actual input data let it be z. Now, this encoder which uses this noised input data is called the de-noising encoder. Now, using the encoder we try to re-construct x which is the input data without noise. Basically in here, the auto-encoders are used as a building block to help in the pre-training process of deep learning. We have a stacked layer versions of these

auto-encoder layers, which would help us in minimizing the reconstruction error and find the best input parameters, where the output of the previous layers are used as input of the next layers. When the last layer learns the parameters to be used to initialize the deep networks. Now, a predictor is added. Here, in the study [39] they have used logistic regression at the end to generate a deep neural network which lead to better results for binary classification. This approach gives a better accuracy than traditional is that it uses de-noising auto encoders which allow deep neural networks to extract robust features though having incomplete feedback as input.

In one of the study [40], they followed the approach of distributed IDS architecture. Basically, this consists of systems in the network using a Artificial Neural Network based approach where each of the system in the network act as a single node. In this study, they used Ubuntu Enterprise cloud platform and the Virtual Machines in the network form the nodes. They will be one manager node and other worker nodes. The monitor nodes handles the interaction between the nodes in the network. The worker nodes form the nodes of the ANN at different layers i.e input, hidden and output layers. They will assigned to each of the layers' based on the resources each of the nodes have. Initially all the request data would be sent through the network for any detection of malicious activity. The hidden get the data from the input layer and the weights get modified at each iteration. The manager VM handles the connections of the ANN, which are added/removed on the basis of the resource usages. It sends messages to the nodes to add remove the connections. Also, in case any of the nodes get inactivate the manager node will be notified at regular intervals so that the IDS (distributed) structure is maintained, trained and the models are deployed. Also, measures are taken to improve the system flexibility, which helps in providing robustness to the system. So, when one of the node in the network fails, the IDS adjusts to create a network/architecture. An experiment was performed by the authors, where the first stet was to train the ANN. As we need a flexible IDS, we use 3 trained models. The number can change according to the specific application. A 5-node architecture was chosen as an example for the IDS, where 2 of them are input layers, 2 are hidden layers and one output layer. The weight values gets updated to adapt the training data, thus improving the performance of the model. As in the KDD dataset they used, it had 41 features thus 41 weights were saved in the layer, which are ready to detection of tasks. In the last layer, the number of iterations continue until the error is no longer considerable. Now, the ANN is ready to used as an IDS. This approach fails when various system within the cloud environment are compromised simultaneously and moreover the attacks which are unknown are sometimes difficult to find. Thus, there is much scope of improvement for this approach.

4 Botnet Detection

Botnets pose serious risk to Cloud computing platforms as it is really difficult to detect bots from millions of web requests that they get everyday. Botnets are networks of zombie/infected computing devices(Bots) which is controlled remotely by a bot master through command-and-control(C&C) infrastructure. These bots are individual programming devices and has the ability to broadcast malicious code to add more bots in network or to carry out attacks like Distributed denial-of-service (DDoS), phishing, spamming etc. These bots generally listen to a defined channel based on IRC (Internet Relay Chat) and/or HTTP(Hypertext transfer protocol) to receive instructions from bot master. All bots receive and execute same chain of commands/instructions and respond back to same C&C infrastructure with execution result. Most of the times these C&C networks are compromised of communication channels controlled by bot master. These networks are very hard to destroy because even if some detection technique managed to detect and block C&C networks, the bot master compromises a new C&C network using Dynamic Domain Name System (DDNS) moving his domain to a new C&C server. We need a detection technique which can adapt to these changes on the fly and machine learning based detection mechanism fit perfectly for these scenarios. There are several different types of methods like honey nets, intrusion detection system based on DNS traffic, signature, and anomaly detection which have proven to have highest accuracy. These detection systems are mostly machine learning driven approaches which adapt themselves with change in botnet architectures.

4.1 Life cycle of Bots

Typically each bot in a botnet have 4 generic stages: 1. Formation/Doping 2.Recruitment 3. Attack 4. Reporting and synchronization.

- **Formation/Doping**: This is the first stage where the bot master infuses the network with malicious code to exploit known vulnerabilities of software or network configurations. Once machines connected to the network get infused with this code they turn into zombie machines, whereby can be controlled remotely by bot master. There are 2 ways of executing this stage, either by active procedures like flooding, scanning, physical infusion, etc. or by passive procedures like click baits, spams, ads, emails, etc.

- **Recruitment** : This stage is the most important stage for a botnet as strength of a botnet is determined by bot-army strength. This stage involves finding newer target with similar vulnerabilities that can be recruited to the botnet.

- **Attack**: This stage is when bot master decides to carry out an attack to target machine/server. The attack can be real-time or can be scheduled (which is often used in botnets with higher latency). Bots carry out their instructions and a large influx of requests are sent from the bot-army pretending to be legitimate users, compromising target machine.

- **Reporting and synchronization**: This is the last stage which is carried out frequently when a bot is recruited or after an attack. In this the stage every bot that is recruited get synchronized with the C&C centre to receive command/directives for actions. When carried out after an attack, this stage involves regrouping and checking how many bots were compromised or detected by the system and replacing them with new ones. Bot master also analyse the whole attack to understand what went wrong and what he can do in future to not get detected in the future. As a result, botnets tend to become harder and harder to detect over time as they evolve at much faster rate then detection mechanisms.

4.2 Modes and architecture of Botnets

Botnets engage in two modes of attacks - synchronous and asynchronous. Synchronous is the most prevalent mode in which bots continuously listen to the host server for instructions which makes synchronous attack possible. These instructions/commands are coordinated and can be for immediate or post-date actions. Asynchronous mode on the other hand doesn't require listening to C&C server all the time. These botnets are self sufficient in their binaries and activities that they can attack without a C&C server. They are mostly distributed in nature and are very difficult to track and monitor. The most feasible way of detecting these botnets is reverse engineering where you track the footprint and activities of botnet. Stuxnet[6] is one such botnet using which Salamatian et al. in[6] demonstrated that asynchronous botnet pose same amount of risk as synchronous if not higher.
However, recent evidences show that new high profile botnet are able to switch between attack modes based on situation. These botnet switch to synchronous mode when they have internet; then switch to asynchronous mode when internet connectivity is not available. In synchronous mode of attack is these botnets have 4 basic method/architectures in which attackers deploy C&C servers. These are: 1. Direct 2. Centralized 3. P2P or Decentralised, and 4. Hybrid.

4.2.1 Direct

This architecture is shown in 2. In this, bot master has direct control over botnets and individual bots. The Bot master is able to recruit, interact and give commands to bots, either towards achieving same or different goals. In this, bot master has all the information and there is no interaction between bots. This

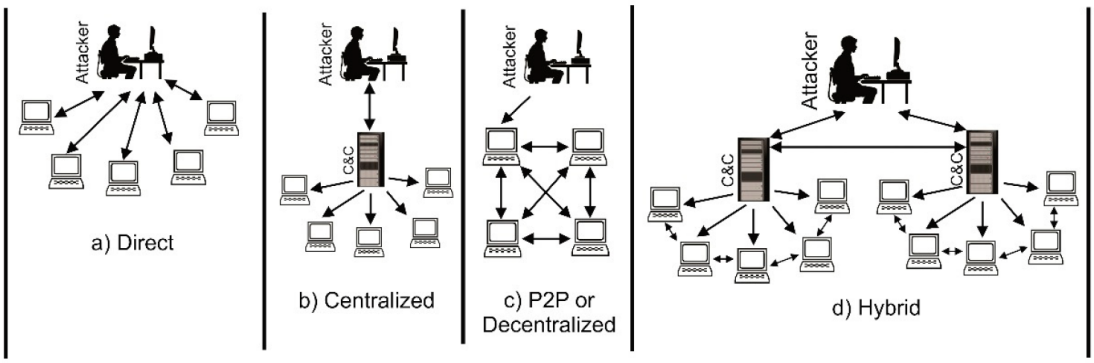

Figure 2: Common Botnet Architectures

architecture lost popularity because tracing back to bot master is possible and with more stringent laws in cyber security using this was a big risk. With the rise of sophisticated machine identity obfuscation techniques, this architecture is again gaining popularity in cyber security domain.

4.2.2 Centralized Model

In this model, there is central computer/host chosen by the bot master (usually high bandwidth computer) which connects and passes message to all bots. The communication is normally done using either IRC or HTTP. The main advantage of this model is small message latency which help bot master to arrange botnet and launch attacks in not time. In these networks there is a hard dependency on this host. In other words, host for these networks becomes single point of failure. If some detection mechanism is able to detect and eliminate this host the entire botnet network becomes useless and ineffective. However, to make this architecture resistive to failures bot masters tend to make deep distributed hierarchy of bots using layering and other sophisticated methods.

4.2.3 P2P or Decentralised

This is shown in third partition of 2. In this architecture, each individual bot is capable of serving as a bot and a C&C server to one other bot. There is no need of forward communication in this model as each bot is self sufficient to recruit new bots and give them the ability to turn into C&C server by itself. However, there is a need of reverse communication to provide operation reports and status which is normally done using various external layers to achieve anonymity. Since this model has no single point of failure, most of detection methods fail to detect and destroy such botnets.

4.2.4 Hybrid

This approach takes best of centralized and P2P architectures to make more resilient hybrid model. Most of the current botnet uses this kind of model where they deploy multiple C&C server with P2P network among the bots which makes these botnets very hard to detect and eradicate.
Below table shows comparison between above mentioned architecture based on their features and characteristics.

Features/Process	**Direct**	**Centralized**	**P2P**	**Hybrid**
Setup	Easy	Easy	Difficult	Difficult
Detection	Easy	Easy	Difficult	Very Difficult
Resilience	Least	Moderate	Moderate	High
Possibility of Failure	Easy	Easy	Moderate	Difficult
Latency	Instant	Fast	Slow	Slow(depends on layers)

Table 2: Common botnet Architecture Comparison

4.3 Botnet Communication Protocols

Since main communication protocols used are IRC and HTTP, this architecture can be further divided into two: IRC based Botnet and HTTP based Botnets.

4.3.1 IRC based centralized Botnets

IRC(Internet Relay Chat) is a protocol for real time text messaging or synchronous conferencing through internet [4]. These are typically based on clientserver model, which is mainly used in distributed networks. Some of the features that make this protocol popular for bot nets are:

- *Low Latency*: These networks have very low latency which enable client to send message to one or more server almost instantaneously.
- *Anonymous real-time communication* : IRC gives user ability to communicate through channel which can be used for preserving identity of the user and also communicate in real-time through direct channels with only those servers which subscribed to that channel.
- *Ability to communicate with group and private*: IRC communication is channel based so bot master has the flexibility to command a selective/few or whole bots at a time.

- *Simple setup & operation*: IRC is very easy to setup. There are only three basic commands needed for IRC communication are - connect to server - join channels - post messages in channels.

The flexibility provided by all above features really help bot master to efficiently instruct all bots to do Bot Master's bidding. To hide botnet request normally C&C servers use standard IRC services. The main downside of IRC is,since these work on channels a firewall or other detection mechanism can easily segregate HTTP request and IRC request, hence the IRC request can be blocked avoiding potential attacks.

4.3.2 HTTP based centralized botnets

HTTP is also a popular protocol used in botnet creation. Since IRC was very popular and came before HTTP every detection mechanism/technique gave a lot of attention to IRC traffic. Also, it was very easy to detect IRC traffic from HTTP which made most of the systems botnet safe. So adversaries started using HTTP protocol for C&C communication, hiding botnet traffic in normal web traffic, making it very hard to detect bots. HTTP request through these networks were easily able to bypass firewalls with IRC port filtering mechanism and other IDS detection. The common known HTTP bots like ClickBot [4] and Rustock [5] has referred HTTP based bots as "pull" style and IRC based as "Push" style. However, the architecture/topology of both the networks is same.

4.4 Botnet Counter Measures

Botnets keep on evolving to evade detection mechanisms over time. Bot Masters methods like surrogacy, stepping stones,etc., bot methods like anti-analysis, rootkit technology, security suppression,e etc., botnet traffic methods like protocol and traffic tweaking, encryption, etc. and C&C server methods like anonymisation, rogue DNS servers, etc. make botnets' counter measures very challenging. [7]
To counter botnets in an efficient way defining taxonomies and categorising botnets based on their behaviours is required. Plohmann, et al. [8] and Khattak et al. [7] in their paper presented well structured taxonomies and gave techniques for not only detecting these botnets but also for measuring impact and making your system resilient from them in the future. The knowledge provided by these studies does not exactly contain the systematic goals to counter present botnet attack because of continuously evolving threat landscape, but it is essential to learn this for understanding how this technology is evolved and what other aspects can be explored in the future.
There are many studies like Honeynet Project and Research Alliance (2005) [9] etc. which link botnet directly to various cyber crimes like Ransomware, Bitcoin mining, Denial of Service, etc. so having a efficient counter measure is really important to keep essential systems safe from these records. These

countermeasures should be deployed either on host or in the network to work properly. These counter measures can be divided into 5 different categories:

- **Prevention** : These measure are deployed at entry/exit point of network/environment to filter out malicious and non legitimate requests/user from the incoming traffic. One such preventive measure was proposed by Luo et al. [10] in 2017.

- **Offensive** : The goal of this measure is to take down botnet or disinfecting bots to turn against C&C server. One of the popular ways of doing this is sinkholing. This essentially takes advantage of information received from analysis of attack/ previously acquired knowledge and relates them to botnet under investigation. These measures are often very specific to the botnets which we are working on at the moment. These are can be categorised into *direct* where we go out against the whole C&C server and *indirect* where we target a part of botnet through surrogate points of network.

- **Recon** : This is the most passive counter measure where we analyse and gather information about already detected botnets. The goal is to monitor botnet for its strength, weaknesses, mode of operation, architecture, capabilities etc. Most the offensive techniques rely on this measure to be successful.

- **Mitigation** : This measure involves coping up from an attack. The goals are to do damage control, disinfect bots/services, analyse the attack to reinforce firewalls and protective mechanisms.

- **Detection** : The goal of this measure is to identify/detect botnets in the production network. This is the most challenging counter measure because the threat landscape and bot master C&C server controlling techniques keep on evolving over time. These techniques may not aim to prevent botnet attack rather to identify the presence of botnet. The detection can further be made difficult because of flash crowds. Flash crowds is the situation when large number of legitimate users send multiple requests to gain access. Flash crowds generally occur during peak times/dates and may lead to false positive in the detection algorithm. Machine learning based techniques perform best in detection domain because they can adapt to the changes in botnets and can give a high accuracy of detection for longer period of time.

4.5 Botnet Detection and Machine learning

As discussed in this paper above, there is increasing need of botnet detection and prevention in cloud computing domain. It is more critical to detect bots/infected machines before they attack our network.

Most of these detection techniques are based on network and are deployed on either host or network host environment to monitor entry/exit points. One of the traditional and effective techniques was honeynet which were functional to understand how botnet works but it got really ineffective with change in botnet architectures over the years. Few other traditional techniques based on Direct and centralized model like DNS similarity based detection [2] etc. worked fine till now but with increase in hybrid and P2P botnet architecture, these techniques are slowly losing there popularity.

Botnet detection is normally considered a part of intrusion detection mechanism and is normally classified differently depending on environment and techniques used. These approaches can be classified as - anomaly-based, specification-based, signature-based, DNS-based and data mining techniques [11]. Another research classified them into more generic types - Data mining based, machine learning based, network behaviour and traffic based. [12]. In terms of installation they can be classified as host-based, network-based and hybrid systems. Lu et al.[13] took a different approach and categorised botnet detection techniques based on machine learning type- supervised and unsupervised.

4.5.1 Unsupervised Learning based Botnet Detection

Unsupervised technique/clustering method is a type of machine learning technique where we don't have ground truth labels during training and we cluster the similar input together on the basis of various features. Zhao et al. [20] used ISOT data set in Reptree algorithm to detect botnets. They correlated the identification rate, data set size and time windows. They were able to achieve 98.5 percent identification rate with 300 sec time windows. Additionally, they deployed a real time botnet detection system to do experiments with centralized botnet (black energy and weasel) which gave 100 percent identification rate. These techniques were still not able to detect botnets based on network flows. Haddadi and Morgan [21] explored this area and gave an efficient solution using naïve Bayes as well as C4.5 machine learning algorithms. They used unsupervised algorithm to detect HTTP based botnet activity by collecting network flows and applying domain fluxing methods. Beigi et al. [22] used a greedy algorithm called step wise for feature collection along with c4.5. They used 3 datasets(ISCX, ISOT and malware capture facility project dataset) to get more data for clustering. They clustered data set based on time, behaviour,bytes and packages. Beigi et al. were able to get 99 percent of identification rate from the final feature set for limited number of botnet and 75 percent of identification rate for totally new botnets in testing phase. Huseynov et al.[23] provided a comparison between K-means and ant colony algorithm using ISOT data set. They used similar features for both algorithm and proved that K means performs far better than ant colony algorithm for botnet detection. K-means was able to get 82.1 % identification rate with 2.4% false positives where as Ant Colony System algorithm was able to get 67.8 % identification rate with 23.5

% false positives.

Garg et al. [24] presented a contrast in P2P botnet detection using naive bayes, J48 and nearest neighbour algorithm. They used various network traffic features and proved that accuracy of nearest neighbour and J48 is good but detection of legitimate traffic is very weak. Jiang and Shao [25] extended the previous approach and developed a detection mechanism which focuses only on C&C traffic and not on how they perform malicious activities. This exploits the dependency of bots to connect with each other in P2P network. They used single linkage hierarchical clustering mechanism to differentiate normal hosts from botnet hosts. This method depends on similarity of botnet traffic, hence it failed to detect botnets with varying traffic flow like storm bots. Furthermore, this mechanism had a limitation to identify individual bot behaviours.

Liao et al.[26] presented a interesting correlation between packet size in legitimate host and bots. They explored the fact that P2P bot does not stay idle and tries to update information of bots and also that bots uses minimum data connections to transmit data. They were able to achieve 98% of accuracy using J48 algorithm but later they found out that the packet size of botnet is smaller than botnet application which made this approach less effective. In contrast to dependency on packet size Zhang et al.[27] introduced a detection system for stealthy P2P botnets. This approach focuses on monitoring C&C network. They extracted 4 features based on number of packets and bytes received and sent in the network. This approach was not dependent on payload signatures and was able to cluster legitimate and botnet hosts with 100% true positive rate. They essentially used Hierarchical clustering and BIRCH algorithm to cluster network traffic. Although, this system was able to detect bots with high accuracy but it was limited to P2P botnets and could not detect HTTP/IRC bots. It can also be fooled by using high level privacy mechanisms in C&C like DGA and fast-flux algorithms for communication. Zhang et al. [28] further improves the above approach in 2014 to improve efficiency and scalability of system. They did this in two phases- 1. recognizing all machine using P2P traffic and extracting statistical fingerprints from the P2P traffic. 2. Analysing the traffic to differentiate legitimate and botnet hosts. For testing environment they used Limewire, skype, bittorrent etc. P2P applications to generate legitimate traffic and Storm and Waledac(installed on controlled environment) to generate botnet traffic. They were able to achieve same 100% detection rate and 0.2% false positive rate using hierarchical clustering approach. The significant improvement was that this system was able to detect botnets even when there is overlapping of legitimate P2P traffic and botnet traffic.

4.5.2 Supervised Learning based Botnet Detection

Supervised learning/classification 2.3 is type of machine learning technique where we know the labels in training data ahead of the time and our aim is to predict these label correctly for testing set. It includes a set of variables which are statically improved over iterations to map input values to output labels. These variable values are improved until a desired accuracy is achieved. Some of the examples of supervised techniques- Random Forest, Decision trees, Logistic Regression and so on. Strayer et al.[14] devised one of the first established supervised learning method for botnet detection. They used C4.5 decision tree, Naive Bayesian and Bayesian network to classify traffic of TCP. This model was fairly accurate with false positive and false negative less than 3%. This method was restrictive to IRC based botnets and TCP traffic classification which lead to downfall of this method in P2P botnet world. Another improved approach for IRC bots was developed by Masud et al.[15] which instead of classifying traffic relied on bot and human reaction patterns, they monitored and correlated incoming packets with outgoing packets, application startups in host and new outgoing connections. They used 5 algorithms - C4.5 decision tree, Naive Bayes, support vector machine, boosted decision tree and Bayes network classifier, side by side and compared to get around 95% detection rate, 5% false negative rate and 3% false positive rate. Boosted decision tree performed the best among the 5 machine learning algorithms but since it requires to access content of payload for classification, it does not work for encrypted communication channels. Also, this methods was only tested for IRC based botnets so its performance against P2P and hybrid botnets is still unknown. Matija and Jens [16] gave an interesting flow network based machine learning solution to detect botnets. They build a system which analyze flow network to detect patterns of malicious activities/traffic. They were able to accurately classify botnet traffic using simple flow features in Random tree classifier. The advantage of this method was we can accurately identify botnet traffic with high accuracy just by monitoring the network of partial amount of time and analyzing a partial amount of packets. They were able to give correct identification using just 10 packets and 60 seconds of monitoring per flow. However, there is scope of optimization in traffic analysis and deployment of this system which they left for further research.

In contrast to centralized, peer to peer botnet detection algorithms were far more challenging. Barthakur ,et al. [17] offered a rule induction algorithm using decision tree, Bayesian network and support vector machines to detect P2P botnets. They judged their algorithm using precision, accuracy and f1 measure. This system was good but it had 2 inherent assumptions for this algorithm to work- 1. Interaction between bots is done by a unique set of commands in a botnet. 2. Data flow happens in two way in every botnet sessions.

Abu-Alia [18] and Pei, X., Tian, S., Yu, L. et al.[19] presented a very effective system to detect botnet

using Domain Name Generation Algorithms(DGA) for communication malware in C&C server. They both used variations of neural networks to generate a replica of DGA and predict the domain names generated to block them in host network. They used alpha numeric features generated by training neural network using reverse DNS lookup. Abu-Alia took linguistic features in domain names like 1. vowel number in domain names - they proved that vowel number in an legit domain is higher than in malicious domain. 2. Alphanumeric characters - random domain names will have more unique alphanumeric characters. 3. dictionary words: they found a correlation between dictionary words and domain lengths which they used as a feature in neural network. Abu-Alia were able to get 6.1 percent false negative rate and 2.7 percent false positive rate using neural network. Pei, X., Tian, S., Yu, L. et al. on the other hand used visual features to classify domain name. They proved that visual features are spatially rich features which were able to encode visual aspects of domain a lot better than linguistic and traffic features. They were able to outperform current state of the art using Attention Sliced Recurrent Neural Network (to learn semantics) and Capsule Network (CapsNet)(to model high level visual information). This is one of the first deep learning system presented till date in DGA botnet domain and is considered current state of the art because they were able to get almost 99% detection rate even for few of the most sophisticated botnets known to researchers like Aleax, Corebot, Ramdo, Volatile etc.

4.6 State of the Art

Botnets have become an absolute threat to computers on the Internet. Most bots are now using Domain Generation Algorithm (DGAs)to generate pseudo-domain names in Domain Name Server(DNS) fluxing. In a recent paper, a Two-Stream network based deep learning framework(TS-ASRCaps) is being used now to reflect properties of DGAs.Moreover,Attention Sliced Recurrent Neural Networks are being used to mine the underlying semantics [19] . TS-ASRCaps is a multimodal-based model and is noted to outperform other classification methods for domain names.

Often malware commands are noticed being embedded in DNS data.These are mostly being allowed to pass through most new security configuration.These botnets use DGAs to generate large number of pseudo-random domain names which are called AGDs (algorithmically - generated domains).These AGDs need to be intercepted by defenders in order to be shut down.DGA domain names consist of random characters and words, and not legitimate domain names.We therefore investigate visual features in order to differentiate between normal and DGA domain names.

Nowadays, deep learning algorithms are also used for DGA domain name detection.To address this, Two Stream network based deep learning framework was proposed which enforces learning the correlation between textual and visual concepts.The semantic distribution and spacial context information contained

in DGA domain names, ignoring other complex/expert features.Although TS-ASRCaps performed extremely well in the experiments, there is more room for improvising.The multi-modal vectors were of the following two types: semantic-based and visual-based vectors.

- *Semantic Features Extraction*: Semantic-based vectors have been produced and also classified as short strings using deep learning methods.The DGA domain name characters are extracted as semantic features.Each character in a domain name is represented by a number.

- *Visual Features Extraction*: The visual features are extracted from the domain names.Malware infected binary files are represented as singular-channel gray-scale images and each bit is turned into an image pixel.These converted binary images were analyzed and a lightweight Convolutional Neural Network was applied on it for detecting malware.

This ideology inspired to extract gray-scale images from domain names and classify them according to similarity of image texture.These were useful for classifying the legitimate and malicious DGA domain names.The Attention Sliced Recurrent Neural Network (ATTSRNN) and CapsNet-based Two-stream network was designed to model this semantic and visual knowledge.These features are fed into two streams of the Two-Stream network.The Two-Stream network is evaluated on the fused classification score.The accuracy directly depends on the quality of extraction of the features.

5 Distributed Denial-of-Service (DDoS) Attacks

5.1 Introduction and Overview

Distributed computing has transformated in IT innovation. It gives versatile, virtualized on-request assets to the end clients with more notable adaptability, less upkeep and decreased foundation cost. Since these assets are provided over Internet through various known networking protocols, standards and formats, along with being managed by various organizations, there is a need for securing these assets as well as the medium it uses to transmit data [41].

However, the protection of these networks against various attacks such as Distributed Denial-of - Service (DDoS) attacks is a concern [42]. DDoS might result into either consumption of network bandwidth or in exhaustion of resources both resulting in unavailability of resources to the legitimate users. DDoS has the ability to occupy a major portion of network bandwidth. It may also consume much of the server time of the victim cloud infrastructures and network[41].

5.2 State of Art Techniques

Work has been done on detecting DDoS attacks based on Support Vector Machine (SVM). The detection is primarily based on similarity of bots while accessing the web pages. In order to understand the distribution of the frequency, they used request frequency sequence to record the request patterns of users. Detection is possible on applying rhythm-matching algorithm to identify similar patterns. Detection can also be done using the flow generated by a software in the network. K-nearest neighbours can be used to categorize the flow into normal or suspicious packets [41].

Analyzing various features of a packets and then using a neural network, especially Radial Basis Function (RBF) neural network can detect DDoS using anomaly-based detection method. This method should be applied at the edge router of victim networks to make the better use of the algorithm[41].

Rajendran, R et, al. [43] have discussed the use of Feature Selection Algorithm using Scoring and Ranking (FSASR) and Rule based Classification Algorithm for detecting DoS Attacks (RCADA) as an intelligent rule based classification algorith to detect DDoS attacks [44].

There is a high usage of techniques such as Artificial Neural Networks (ANN), Decision Tree, Entropy and Bayesian methods. Current mechanism to detect and prevent Distributed Denial-of-Service attacks use complex techniques such as Distributed Time Delay Neural Network (DTDNN), Radial Basis Function Neural Network (RBFNN), Learning Vector Quantization Neural Networks (LVQNN)[41], Fault Tree Analysis (FTA) [46], Decision Trees (Especially C4.5) [45], Multilevel Thrust Filtration (MTF)[46], Generalized Autoregressive Conditional Heteroskedasticity (GARCH) Model [47], etc. All these techniques can be used along with long used network intrusion detection system and intrusion prevention system, SNORT.

5.3 DDOS Attacks and Detection methodologies

In this section we discuss the why DDoS attacks are launched and the launching methods that could be used to facilitate DDoS attacks, as well as we review intrusion Detection Methodologies and defense strategies for DDoS attacks. As discussed in section 2.2, it is important to address DDoS. The DDoS attacks can be classified on the basis of network protocol and application used[43]. Some of the commonly used DDoS attacks are user datagram protocol (UDP) Flood, SYN Flood, Ping of Death, Slowloris, HTTP attack, internet control message protocol (ICMP) Flood etc[44].

UDP, ICMP and TCP protocols facilitate network or transport level attacks. These attacks can be directed based on high / low attack rates, based on various parameters, each having different consequences on the victim[43].

5.3.1 DDoS Intend and Launch methods

How: There are two ways to achieve DDoS Attack. Either disrupting a legitimate users connectivity by making sure that the bandwidth, router processing capacity or network resources are exhausted or by disrupting legitimate users by denying services of the server resources (e.g., sockets, CPU, memory, disk/database bandwidth, and I/O bandwidth).

We can also achieve DDoS attacks by disrupting a legitimate user are essentially network/transport-level flooding attacks (i.e., flooding attacks), whereas disrupting legitimate services essentially include application-level flooding attacks.

5.3.2 Why DDoS: Attacker's Incentives

Why: There are numerous reasons that lead to people committing DDoS attacks. Some of the reasons that might lead to a DDoS attacks are Economical/Financial Gain, Intellectual Challenge, Cyber warfare, Ideological belief, etc.

5.3.3 Conventional DDoD Detection Mechanisms

Signature-based Detection (SD) which uses signature to identify, previously known attacks. Anoymaly-based Detection (AD) relies on network behaviour to identify anything suspicious and Stateful Protocol Analysis (SPA) analyzes the packet state to reveal if there is anything out of the usual. These are conventional DDoS and Intrusion Detection mechanism.

5.3.4 Using Machine Learning to detect DDoS Attacks

In this subsection we will discuss numerous approaches taken to detect and prevent DDoS attacks in cloud environment.

5.3.4.1 Basic Machine Learning Marwane Zekri et al., [41] proposes the use of non-linear time series model (i.e. Generalized Autoregressive Conditional Heteroskedasticity (GARCH). This method can be used to correctly predict the traffic state, since it can captures long range dependence (LRD) and long-tail distribution. These two features are the property of general network traffic. Filtering is done with the help of back propagation artificial neural network (ANN), based on certain threshold.

Once the dataset was divided into various bins, the variance is calculated by aggregating the entropy of each bins. Threshold is then calculated to filter network traffic. Considering 'M' and 'σ' as mean and

standard deviation of prediction error respectively. Let 'α' be threshold parameter which is modified as per the needed by specific environment. Then threshold can be given as given in Equation 1

$$T = M + \alpha * (\sigma) \tag{1}$$

To gain maximum efficiency out of this method, it is recommended that this detection method be deployed at the edge routers of the Cloud network of the victim as shown in Figure 3

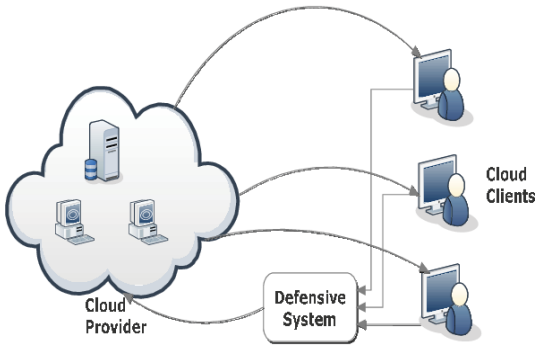

Figure 3: Defensive measures at Edge-routers

After performing various experiments and considering different values of size of bin, size of forecasted variance, along with the value of bin size as 14, $\alpha = -1.5$ Omkar P. Badve et al. [47] recorded a detection rate of 94.06%. This detection rate could easily be increased by classifying the packets using Artificial Neural Networks. Performing ANN along with this method increased the true detection rate to 99.4%.

5.3.4.2 Incorporating C4.5, Naive Bayesian and K Means Algorithms Marwane Zekri et al. [41] and Md Tanzim Khorshed et al. [45] envision to achieve the following goals of Low computational cost, Faster detection rate, Scalability, High accuracy along with Low false negatives and false positives. In order to compare and contrast a good algorithm to achieve the above said goals, M Zekri, et al. [41] and Khorshed, M.T et al. [45], used Naive Bayes classifier for anomaly detection. To capture signature based detection, SNORT was used. C4.5 algorithm was incorporated to construct the decision tree and k-means to cluster packets / resources into different categories.

C4.5 identifies and chooses the attribute as the splitting criterion based on the entropy-based gain ratio. It does so to overcome the overfitting problem. The attribute with the largest gain ratio is selected as the splitting attribute.

The training data set is then divided into various subsets, after selecting the splitting attrubute. This process of splitting the dataset using attributes is performed until until all the data in a subset belong to one paarticular class or the gain ratios of all the splitting attributes are remain the same.

Method	Correct Classification (%)	Detection Time (s)	F-Measure
Naive Bayes	91.4	1.25	0.914
C4.5	98.8	0.58	0.988
K-Means	95.9	1.12	0.959

Table 3: DDoS Attack Detection Result

From the obtained results Table 3, Marwane Zekri et al. [41] concluded that their proposed approach, in which C4.5 algorithm is adopted to detect DDoS attacks, gives more accurate results in less time compared to others machine learning algorithms for the given data set / environment. This method could further be improved using cost matrix associated to the confusion matrix. Having domain knowledge is crucial to assign a good cost matrix.

5.3.4.3 Self learning methods Over the past few decades, the development of protection method based on data mining has attracted many researchers because of its effectiveness and practical significance.

The detection mechanism using data mining are most commonly employ prelearned models or models based on rules. Due to thism the detection method often fails to perform well in dynamically and continuous changes in cloud networks. Andrey Rukavitsyn et al. [42], proposed a selflearning method that allows to the detection mechanism to adapt with network changes. This paradigm change might result into minimized false detection rate and decreases the possibility of indicating legitimate users as malicious ones and vice versa. The relearning algorithm allows dynamic monitoring to change of traffic in cloud computing, adapt to new traffic by dynamically changing the threshold values. They dataset used was generated while modeling and classifying legitimate users and DDoS attacks using OpenStack frameworks of RSVNET and IDACF.

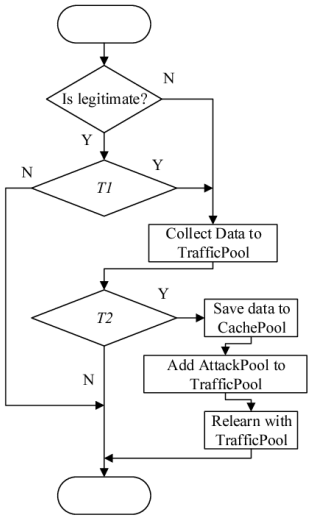

Figure 4: Self Learning Algorithm

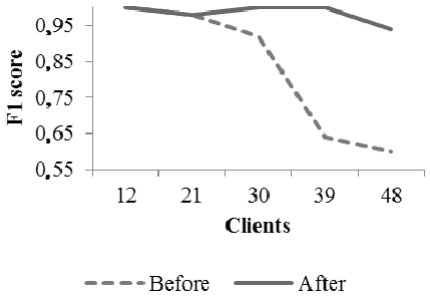

Figure 5: Self Learning Algorithm F1-Score Curve

We can provide a change in some attribute, lets say 'X' of a legitimate user, where the a number of intervals of ti (i = 1, n) in which the change of traffic is minimum. It is possible to carry out the model relearning effectively to adapt the model under changes for each interval of ti in traffic as depicted in Figure 4

Clients	Accuracy		Precision		F1-Score	
	Before	After	Before	After	Before	After
12	1.0	1.0	1.0	1.0	1.0	1.0
21	0.98	0.98	0.97	0.97	0.98	0.98
30	0.99	1.0	0.86	1.0	0.92	1.0
39	0.57	1.0	0.47	1.0	0.64	1.0
48	0.44	0.95	0.43	0.90	0.60	0.94

Table 4: Classification Assessment of Relearning Algorithm

Based on the data provided in Table 4 and Figure 5, it is evident that the use of a relearning algorithm significantly increases the resistance of detecting model to change of quantitative traffic characteristics and allows to lower false positive errors, hence increasing the F1-Score of the algorithm. Hence [42] recommends the use of relearning algorithm to effectively increase F1 score of underlying algorithm.

5.3.4.4 Combining Conventional and Modern Detection Techniques Abdul Raoof Wani et al [43] proposes a model to detect DDoD attacks using conventional method of using SNORT and overlaying

it with machine learning techniques. SNORT is an open source rule based tool to detect Intrusion detection including DDoS attacks. SNORT uses a rule based method similar to the rules.

The data points collected from SNORT is then reduced to few important features such as Duration of the flow, Type of Protocol, Internet Protocol Address (Source, Destination), Port (Source, Destination), Transmitted packets, Attack classification Labels, Number of transmitted bytes, etc. Abdul Raoof Wani et al [43] have experimented with three different machine learning approaches, namely Random, Forest, Naïve Bayes and Support Vector Machine for the classification of data.

These algorithms (Random Forest, Naïve Bayes and Support Vector Machine) were chosen on the basis of their efficient performance and application in the field of network security. We performed a comparison and Analysis of Accuracy, Precision, Recall on DDoS packets and Normal packets. [43]

Measure	**SVM**	**Random Forest**	**Naïve Bayes**
Recall	0.998	0.993	0.860
Precision	0.998	0.992	0.881
Accuracy	0.997	0.976	0.980
Specificity	0.996	0.995	0.505
F-Measure	0.998	0.996	0.826

Table 5: Comparing and Contrasting SVM, Naïve Bayes and Random Forest

After comparing and contrasting these three machine learning algorithms, Abdul Raoof Wani et al [43] arrived at the conclusion that SVM and Random Forest outperformed Naïve Bayes as seen through the results obtained in their experiment depicted in Table 5

5.3.4.5 DDoS Attack Detection on Source Side Zecheng He, Tianwei Zhang and Ruby B. Lee[48] have proposed DOS attack detection on the source side in cloud rather than destination side. The proposed system is designed to leverage and analyze statistical information from the virtual machines and the hypervisor of the cloud server, in order to prevent the network packages from being sent out to the outside network. Before launching DDoS, other attacks(like brute-force password guessing attacks) may be used to install bot code in order to intrude into a secondary victim machine. The system is tested on four very common network attacks: SSH brute-force attacks, ICMP flooding attacks, DNS reflection attacks and TCP SYN attacks.The VMM is responsible for monitoring the status of virtual machines and gathering statistical network information, which is then sent to a machine learning engine that feeds back if a suspicion action is detected.

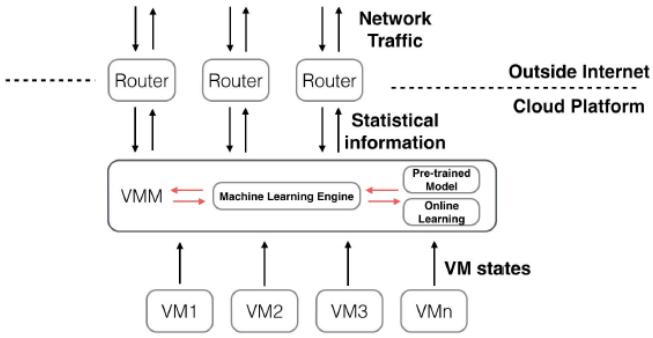

Figure 6: Architecture of proposed system

The machine learning engine has two parts, the pre-trained module is trained to determine whether the virtual machine's actions are suspicious and the online learning module is trained to update the parameters in the pre-trained module. Features used for the four types of DDos attacks:

1. SSH Brute-force Attack Feature:

   The SSH brute-force attack, makes remote brute-force guesses of passwords of legitimate users. In the proposed system, the rate of Diffie-Hellman key exchange packages is used to detect SSH brute-force attacks. Every time while establishing SSH, a pair of Diffie-Hellman keys is generated and exchanged between the two parties and used to encrypt the following sessions. If authentication fails three times, then the SSH channel is disconnected. Therefore, the total number and rate of Dillie-Hellman key exchange packages could be much higher than usual during an attack.

2. DNS Reflection Attack Feature:

   DNS (Domain Name Server) is responsible for translating domain names to IP addresses. The inbound traffic is approximately proportional to outbound traffic for normal DNS requests, which is not the case during DNS reflection attack.As the source IP address is spoofed to redirect the response to the victim in a DNS reflection attack, no response is returned to the requester which results in more request packages than response packages. So, the inbound/outbound DNS packages ratio is used to detect this attack.

3. ICMP Flood Features:

   In an ICMP(Internet Control Message Protocol) flood attack, the attacker sends a huge amount of ICMP packages and tries to overwhelms the victim. Under normal situations, there are not many ICMP packages. So, the system uses ICMP package rate as an indicator of an ICMP flood attack.

4. TCP SYN Attack Features:

   The TCP SYN attack is a stateful protocol attack that leverages the three handshakes feature of

TCP protocols. The TCP SYN attack is detected using the ratio of TCP packages with ACK tags and SYN tags, as it is very abnormal if the SYN/ACK ratio reaches a high ratio.In normal situation, packages with ACK tags are much more than packages with SYN tags.

Another source detection algorithm framework by Raneel Kumar, Sunil Pranit Lal and Alok Sharma[49], is an an entropy-based detection method using a supervised machine learning model. The framework can be implemented to detect source devices being used in DDoS attacks, which analyses outgoing network traffic from a device to identify abnormal behavior. The main focus of the paper is to detect TCP flood attack. In TCP flood, an attacker continuously sends TCP SYN packets to the victim. The victim or target server responses to the attacker or sender by sending SYN+ACK packet and waits for the ACK packet to establish final handshake, which will eventually exhaust the server's resources due to half-open connections. There are three possible DDoS senarios that the proposed system tries to detect: Constant Rate Attack, Increasing Rate Attack and Pulsing Attack.

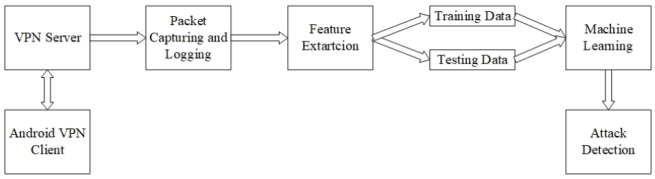

Figure 7: Proposed attack detection framework

Figure 7 shows the proposed framework. The important sections of the proposed framework are:

- Packet Capturing:

  The outgoing Internet traffic is captured and the packet header information is logged. The logged packet details are Timestamp of the day (in seconds), Destination IP, Destination Port, Transmission Protocol and Flag and Packet Length

- Feature Extraction:

  The feature extraction module provides the training data to the machine learning component to train its detection model. It aggregates packets using a fixed time window, T, to extract the packet information. The incoming packets are classified based on their protocol type and flags. For the analysis of devices network behavior, the Shannon entropy and Normalized entropy of the packet behavior is used.

- Machine Learning:

  Support Vector Machine (SVM) is used to create a detection model based on the training data

received from feature extraction stage. To train a machine learning model, the data is classified into two categories: Normal data set and attack dataset.

To experiment the proposed model, simulation of the attack is done by creating an Android application called "Packet Sender". The attack scenarios are implements by continuously sending TCP SYN packets to the target, where the rate can vary in the range of [25, 50] pps based on the scenario. If the most requested IP is the selected target IP in "Packet Sender" application, then it is labelled as an attack. In the machine learning modules, two approaches are employed. In one of the approach, all three features of the dataset(normalized entropy of destination IP, normalized entropy of destination port and transmission rate) are considered as the input features and in the other approach,Principal Component Algorithm (PCA) is used to reduce the number of features to two.The classifier model performance is evaluated using a confusion matrix for testing data.(See table 8).

Model	*Accuracy Rate*	*Precision Rate*	*Error Rate*
3D SVM	0.985	0.971	0.014
2D SVM	0.988	0.984	0.011

Figure 8: Confusion Matrix

5.3.4.6 DoS attack detection on Cloud Provider's side One of the papers[50] has proposed a model for (Intrusion Detection system) IDS, which incorporates a packet sniffer, feature extractor and a classifier to detect Dos and DDoS attacks on the cloud provider's side.The IDS is placed in the physical servers or the nodes of the cloud, where it can monitor all the incoming and outgoing traffic to be able to detect anomalies. The main focus of the paper is to detect DoS attack on VMs in the cloud, using one class machine learning classifier.

For the purpose of experimentation, an IaaS Cloud is created using An open source cloud computing platform – HP Helion Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems(Eucalyptus) Cloud. the cloud has six stand-alone components: Cloud Controller, Walrus, Cluster Controller, Storage Controller, Node Controller and VMware Broker. The IDS has three important parts:

- Sniffer and Feature Extraction:

  Sniffer is used to capture packet information which can be used for feature extraction. To secure the private information of the tenant, the data capture is limited to only four IP header: Source IP Address, Destination IP Address, Bytes of Data Transferred and Protocol.

- Machine Learning classifier:

  For machine learning, one class support vector machine (SVM) classifier is used. In contrast to the

traditional SVM, one class SVM tries to learn the decision boundary or hyperplane that achieves maximum separation between the points and the origin. This type of classifier focuses on learning what the normal data is, and based on that differentiates between normal and abnormal data.

- Alert Mechanism:

  The IDS has an alert component which notifies the cloud administrator of DoS attack on the cloud VMs. The network traffic is captured at 5 seconds intervals and based on the traffic flow features the classifier is trained to identify legitimate and malicious IP addresses.

The effectiveness of the classifier model is determined based on 3 characteristics: sensitivity, specificity and classification accuracy. Sensitivity and specificity is defined as the proportion of true positives and true negatives, respectively, in the testing data set that are correctly identified by the classifier.

Attacks	Sensitivity	Specificity	Accuracy
ICMP Flood	1	1	1
Ping of Death	1	0.87	0.94
UDP Flood	0.97	0.97	0.97
TCP SYN Flood	1	0.92	0.96
TCP Land Attack	1	0.95	0.98
DNS Flood	1	0.97	0.99
Slowloris	0.43	0.92	0.68

Table 6: Sensitivity, Specificity and Accuracy of the proposed model

The table 6 shows the sensitivity, specificity and accuracy of the proposed model for DoS and DDoS attacks. The classifier is able to classify both legitimate and malicious IPs accurately with high accuracy for all the attacks, except for Slowloris.

5.3.4.7 Artifical Intelligence technique The paper[51] proposed a system to detect known and unknown DDoS attacks, using two different intrusion detection approaches: anomaly-based distributed artificial neural networks(ANNs) and signature-based approach.

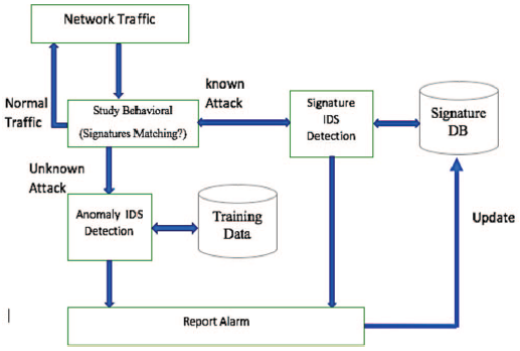

Figure 9: Proposed hybrid IDS

Figure 9 shows the architecture of the proposed system. The network traffic is first checked for know signature patterns. If the network traffic pattern matches the signature of an existing attack,it is tagged as an attack based on the match. A signature pattern consists of the action, header and the rule-option. If the attack is unknown, then it goes through anomaly-based detector, which uses neural network. The detector uses back propagation, where the parameters of the neural network are optimized to minimize the reconstruction error, i.e trying to set the output same as the input. When the reconstruction error exceeds the set threshold, it is considered as an attack. Once an attack has been detected, the signature is stored and used for future signature-based detection.

Metric	Signature -based detector	Neural based Anomaly detector	The proposed Hybrid System
Accuracy	97.52%	98.10%	99.98%
Detection Rate	76.10%	80.78%	98.15%
False Positive Rate	2.30%	1.70%	0.00%

Figure 10: Evaluation metrics

For the experiment, BigDL deep learning library is used over Apache Spar as a neural network for anomaly-based detection. The hybrid anomaly based and signature based detector is developed on Amazon Cloud AWS EMR.The evaluation metrics for signature-based detector, anomaly-based detector and the proposed hybrid system is shown in table 10. The proposed hybrid system perform better that the signature-based or anomaly-based detector in terms of efficiency and accuracy.

6 Malware Attacks

6.1 What is a Malware

Malware is used as the blanket term which is used to denote a wide range of malicious and intrusive pieces of code. Malware is getting stronger day by day and the obfuscation techniques are getting better as well. As a result, the security algorithms we deploy have a hard time detecting the malware. There are various approaches to solving the problem and we would look at the solutions which are based on Machine Learning to help protect Cloud Computing applications.

6.2 Types of Malware

Let's look at the most common variations of malware differentiated by their functions and capability.

- Trojan Horse: These are deceptive malicious pieces of code hidden inside programs which appear to be safe but are programmed to eventually gain control of the designated system. Examples of Trojan Horses are Game-thief, Mailifinder, banker, etc
- Worm: A worm is an isolated program that uses network vulnerabilities and failures to replicate itself and spread across networks of computers. The fact that they are standalone without the trigger from a host makes them very dangerous. Examples are MyDoom, Sasser etc
- Time or Logic Bomb: A logic bomb is a type of malware that gets activated once the system meets a logical condition. The time bomb is a type of logic bomb that stays put for a specific amount of time and is activated and causes computer programs to freeze or crash. Examples of this malware are the famous Jerusalem Virus and Michelangelo.
- Armored Virus: Armored viruses contain specific mechanisms to make their detection and reverse engineering very difficult so that their effects are not reversed. The Whale Virus is a very widely known armored virus.
- Polymorphic Virus: Polymorphic viruses are smart malware which mutate themselves to escape detection techniques while keeping their underlying functions unchanged. Some examples of this type of malware are the URSNIF, UPolyX, VOBFUS etc
- Macro Virus: A macro virus infects a software program and sets off a chain of actions once the application is opened by the user. Microsoft Excel and Microsoft Word are two of the most affected applications because of Macro Viruses. An example is the Melissa Virus that infected emails once users opened theirs.

- Rootkit: Rootkits are used to provide attackers privileged access to other people's computers and let them maintain it without being detected. NTRootkit and HackerDefender are some examples of rootkits.

- Ransomware: Hackers use ransomware to demand a ransom to let users access their data and applications. Once infected by this program, extensions are added to the system in attempts of making it hostage until the ransom the hacker demands is paid. Examples are Wannacry, Petya, Locky etc

6.3 Machine learning for Malware Analysis

Machine Learning methods to detect malware consist of a learning based system to find out the various categories of attacks by studying data of previous attack and normal behaviours. This is achieved by calculating dependencies between various features as well as irregularities within the data. Given the increasing complexity of Malware it is not possible to design one algorithm to detect all of its variants. So, the different algorithms outperform each other on various occasions. Malware detection happens in 3 stages: Malware Analysis, Malware Feature extraction and Malware Classification [64]

- Malware Analysis: This step attempts to answer the following questions about the malware. How it works, what programs and machines does it affect, what data is compromised, what is the scale of damage, etc.

- Malware Feature Extraction: In this step, we try to find features from the malware data we have to map and match them with incoming malware.

- Malware Classification: Using the extracted features and mappings we try to determine whether the file under suspicion is malicious or not. The most commonly used algorithms for classification are SVM, Naive Bayes, Random Forest etc. The algorithms outperform one another depending on the distribution of data and the dependency among the features.

6.4 State-of-art ML Techniques for Malware Detection

In this section, let us take a look at various classification algorithms used in the malware detection process.

According to [64], Malware detection approaches consist of 8 categories - Signature-based, Behaviour-based, Heuristic-based, Model checking-based, Deep learning-based, Cloud-based, Mobile-based and IoT-based. A detailed summary of all the research under each category can be found in [64]. Signature-based malware detection uniquely identifies each kind of malware depending on program structure. Similarly,

there is also Behaviour-based and Heuristic-based approaches which helps to identify new forms of malware. The scope of the papers discussed in this report are based on Mobile (Android) devices, SVM, Deep-learning, CNN in cloud.

6.4.1 One-class SVM

As discussed earlier, attacks against cloud infrastructure are getting smarter by the day so it can be expected that in the future, our systems will have to resist attacks they have never faced. The paper [62], it is proposes the use of a new resilience system known as Cloud Resilience Managers (CRMs) to facilitate this. The functionality of CRM is divided into 4 parts, them being System Defense, Fault detection, Remidation against the detected faults and Recovery. The core of the detection methadology in [62] is based on the one-class SVM algorithm. According to Michael R. Watson et al., [62] For a given input matrix x, the SVM algorithm formulates a decision function that returns a class vector y that depends on the training data. The vector y classifies inputs into known and unknown instances for the model. The experiment referred to in [62] pans out pretty impressively. The CRM's working is tested on two malware types, Kelihos and Zeus. The results are shown in Figure 11 and Figure 12

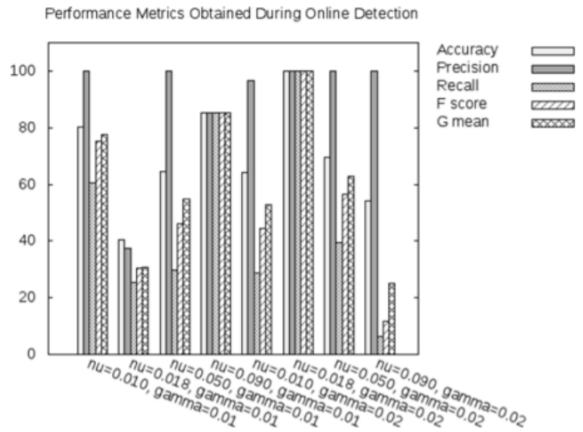

Figure 11: Results of detecting Kelihos-5 samples

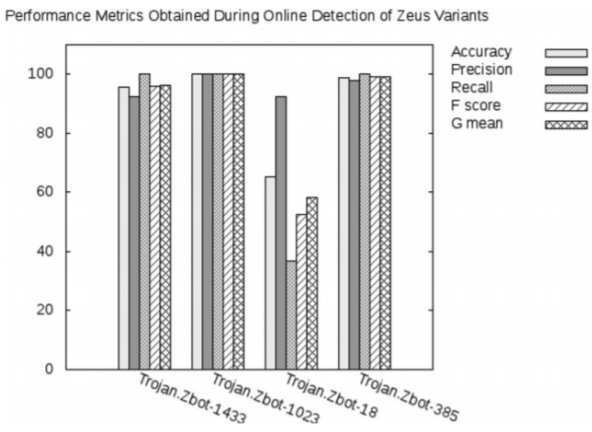

Figure 12: Results of detecting Zeus Samples

These experiments in [62] were carried out in on network-level detection of the attacks. The associated features of the network were utilized to create the model. The performance was pretty good on the basis of accuracy since the detection accuracy is comfortably above 90 percent. But the flip side of using only the network features were the low recall rates. Another utilization of the one-class SVM algorithm has been carried out by Trishita Tiwari et al., [63] in the paper [63]. This work is based on monitoring the usage of resources continuously and representing the same as time-series data. The algorithm analyzes the user's behaviour to identify outliers of any extremity and alerts the administrators. The way the

SVM has been used in [63] is as follows. The features deemed to be the most important in the time series were the average, percentiles, standard deviation, maximum, minimum, skew, kurtosis and the period. From the time series, the above mentioned features are extracted and the model is allowed to train on the features instead of raw data-points. This has proven to increase the classification accuracy and since the dataset is reduced to its most important attributes, the overhead is also reduced. For training the dataset, 500 instances of time-series data were used and testing was done using datasets containing both regular and anomalous data.

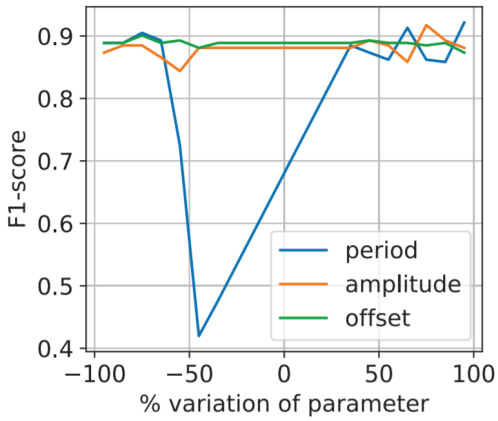

Figure 13: F1 score vs % Variation of parameter

Figure 13 represents the trends in the F1-score along with variations in each parameter. Another improvement in the usage of Support Vector Machines in defense against malware attacks has been given by Saket Kumar et al., [61].Zero day vulnerability is the name given to an exploit that attacks a system before the inherent security mechanism is ready for it. The paper [61] uses SVM extensively along with signature extraction as well as Sandboxing to tackle this issue. We use the Sandbox environment to execute our inputs and we collect data about all the lifecycle activities of the executable. Whenever we get an input executable, we calculate its hash and thus store it in a database using signature extraction techniques. This data is then passed on to the SVM where the algorithm finds the anomalies in the training data.

Each one of the samples sent with the input data labelled into one of the 2 classes, positive and negative are taken by the one-class SVM during training. A high dimensional feature space is created when the input sample matrix comes to play. Each of the samples is taken as a row in that. The dimensionality of the input space is provided by the number of attributes. Here a question of design arises where there is a conflict between the type of SVM we might want to use. A one-class SVM will only categorize the input into two classes. The problem with that is, this does not bode well with the classification metrics. Measuring the false negatives and false positives becomes a cumbersome process

with this approach especially when normal executables exhibit characters usually shown by malware. This is true the other way round as well. This is the reason why a multi-class SVM is preferred in [61]. As for the results of the experiment in [61], the time required to train the model exponentially increases with time. This is expected since the amount of data we read and accumulated is large and massive computations are required with the processing of the data.

[57] proposes a malware detection model called Significant Permission IDentification (SigPID), based on permission usage analysis. The authors employed a three levels of data pruning techniques to identify those Android permissions that are most effective in deciding benign and malicious apps. The authors identified that 22 out of 135 permissions were significant and a SVM classifier helped to get ¿90% accuracy, precision, recall and F-measure. The levels of pruning are Permission Ranking with Negative rate, Support-based permission ranking and finally, permission mining with association rules. This model can detect malware correctly with an accuracy of 93.62% and 91.4% for unknown/new malware.

6.4.2 Random Forest and Naive Bayes

Linda Joseph et al, [60], has worked on securing Cloud Systems, specifically IaaS services against malware and has come up wirh a self heal approach. This is accomplished using a large dataset of memory snapshots that belong to the virtual machines that run the respective services. The end goal is to classify the upcoming inputs which are in the form of API call sequences into malicious and normal memory snapshots. This classification task is done with the help of both supervised and unsupervised algorithms. The features such as anomaly patterns are extracted from the input data and are fed as input to the above mentioned algorithms. The paper [60] uses Naive Bayes algorithm for this but also takes into consideration SVM as well as Random Forest algorithm. Let us talk about the environment used for the experiment carried out in the paper [60]. So, for the test subjects, virtual machines were subjected to as well as not subjected to attacks by malware like CyberGate, Xtreme, DarkComet, Zeus and TeslaCrypt. The size of the dataset was 3500 successful API calls produced by the test VMs as memory snapshots. There is a possibility that the responses from the malware were not successfully received on the other end. Those responses were discarded. So, effectively the dataset contained snapshots from the VM which were of two types, malicious attacks and normal response API calls. These API calls belonged to various sections of the system like Registries, Information about the system, Networks as well as the file system access. The training to test data split was made to be 60-40 for reasons of better accuracy after cross validation. The way in which the API snapshots were selected for testing and training was with the use of an API tracer. This monitors the processes of the VMs and would alert the user on any delays and deviations in CPU response time. Non typical data from the tracer was taken into consideration for this

experiment.

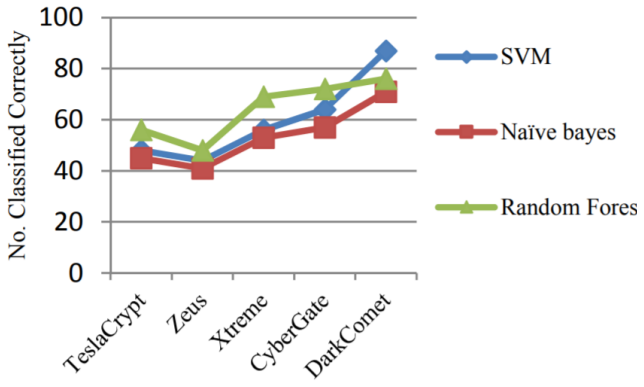

Figure 14: No of Infected files correctly classified for different algorithms

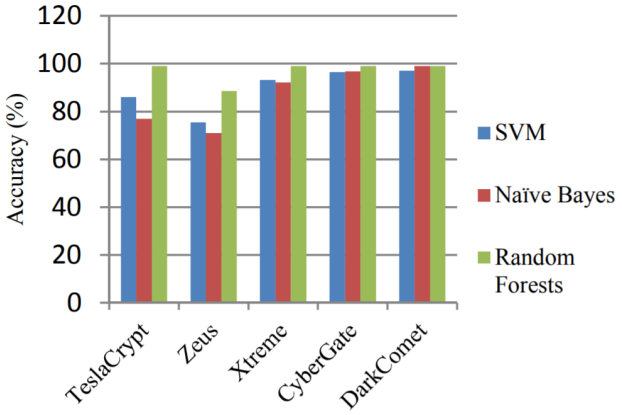

Figure 15: Classification Accuracies for different algorithms

Figure 14 shows the number of True Negatives from the classification process carried out on different malware infected files for different machine learning algorithms. Random forests achieved better results than the other two algorithms.

Figure 15 depicts the accuracy shown by the above mentioned algorithms in the overall classification process. We can see that Naive Bayes is the least impressive in helping classify correctly in the self heal approach.

6.4.3 Deep Reinforcement Learning

Nowadays, many companies are shifting to the cloud based approach where all their infrastructure can be placed off-premise, to increase their sales and reduce costs significantly as it is based on the 'pay-as-you-go' structure. This would help them maintain a distributed and scalable environment on the cloud and used as a foundation to establish multiple detectors to more effectively detect malware attacks. In [52], the authors use a deep reinforcement learning technique to selectively query a subset of detectors available. This paper addresses a very important challenge in this field which is the diversity of the threats, where it is really important to find the optimal number of detectors to place which is most cost effective. This study introduced the MDaaS (Malware Detection as a Service) algorithm [52], where the authors used a Deep Reinforcement learning (DRL) model to factor the monetary cost of each detector as well as the risk of misclassification. In order to do so, based on the studies, they realized that they need to select a subset of detectors that is less costly and not much threat.

The dataset used was portable executable (PE) files[52], consisted of 24,738 files on the whole, and had

equal proportion of both malicious and benign. Both these kinds of files were analysed with the help of VirusTotal tool, for at least 3 months after they were created [52] and labels generated by this tool were taken as the ground truth for the dataset.

In this paper, the primary factors considered while choosing the the detectors were Variance in Performance: there should be multiple kinds of accuracies for the detectors, Variance in Computational Cost: there should be a large variance in the cost of computation, No Dominated detectors: need to understand the performance of each detector on the files that were misclassified by the other detectors and Static and Dynamic analysis: need to emulate real-world scenarios as closely as possible.

So, based on these above factors, the detectors that were chosen finally were:

- Static[PEHeader] detector: Features are extracted from the PE Header of the executable file, and a decision tree algorithm is used to classify as malicious or benign
- Static[Byte] detector: Byte trigrams from the raw binary file are extracted and classification features are selected by choosing the top 300 most frequent items, which further uses Random Forest model (n = 100) for classification.
- Static[Opcode] detector: This one takes the sequences of opcodes from the disassembled PE file, selects the 300 most frequent items and Random Forest model for classifying, same as above.
- Dynamic[WinAPI] detector: This one, on the contrary to previous, applies dynamic analysis, where it extracts all the API calls that happen when we run the PE file in a sandbox environment. Then, features are obtained by taking the most frequent terms of the $\frac{3}{4}$-grams of API calls that are generated, and using LightGBM algorithm (1000 estimators) for classifying.

For conducting the experiment, the ChainerRL (a DRL library) framework and OpenAI Gym training environment were used. Both the Policy and Action-value networks above consist of the input layer with 4 neurons, a single hidden layer with 20 neurons and an output layer of 17 neurons. Throughout the network, ReLU is used as the activation function, except for the output layer where SoftMax algorithm is used. The optimizer used is RMSprop, with a learning rate of 7e-4 initially, exponential decay rate of 0.99 and a fuzz factor (epsilon) of 1e-2. The entire model was trained using 10-fold cross validation, balancing the class label ratios, until convergence.

The authors had set up 3 experiments, with the reward scheme as in 16 and the detailed accuracies and AWS cost obtained for each of the experiments are shown in 17 as below.

	True Positive	True Negative	False Positive	False Negative
Experiment #1	$\sqrt{TC}$	$\sqrt{TC}$	$-\sqrt{TC}$	$-\sqrt{TC}$
Experiment #2	$\sqrt{TC}$	$\sqrt{TC}$	$-10\sqrt{TC}$	$-10\sqrt{TC}$
Experiment #3	1	1	$-\sqrt{TC}$	$-\sqrt{TC}$

Figure 16: Reward setup based on accumulated Total Cost (TC)

Name	Accuracy	AWS estimated cost (US Dollars)		
		On-premise	EC2	Lambda
Baseline	96.843%	4186.8$	156.51$	33.23$
Experiment #1	**96.913%**	1390.1$	51.97$	16.69$
Experiment #2	96.629%	1360.6$	50.86$	16.11$
Experiment #3	96.244%	**426.2$**	**15.94$**	**4.17$**

Figure 17: Accuracy and Cost Estimation of Baseline and Experiments conducted

This MDaaS algorithm[52] was compared with the ensemble approach that is commonly implemented in commercial products. For the baseline, they implemented a stacking approach where the classification values of all the participant detectors were to train a Random Forest classifier, to get the final label for each file. The cost of running this algorithm was calculated using 3 platforms: on-premise, cloud (EC2) and serverless (Lambda) computing.

The results show that the agent proposed can be trained to perform well in various scenarios and sometimes outperform the baseline even though the resources are limited. As we can see, Experiment 1 achieved better accuracy than baseline while consuming only 30-50 % of the resources. The 3rd experiment took up only 10% of resources in an on-premise setting and achieved 0.5% lesser accuracy than baseline. The results also showed that the AWS calculator's estimations was highly accurate at 34.19$(estimated at 33.23$) for the baseline and 4.44$(estimated at 4.17$) for experiment 3, showing that the proposed method was approximately 8 times cheaper.

6.4.4 Deep Learning

New techniques to detect malware attacks are being developed everyday to handle the increased use of digital technologies. One of the recent Machine learning algorithms for this is Deep Learning, used to maintain the 3 aspects of data - Confidentiality, Integrity and Availability, discussed in detail in [53]. Please refer to this paper for further references. In early 2014, Yuan et al. were the first persons to develop DroidDetector, an online deep-learning based Android malware detection engine for android apps.
In 2017, Choi et al. generated images from the source files and then, they used Convolutional Neural Network with 3 convolutional, 1 pooling and 2 fully connected layers to detect malwares. Also, Zhu et al. proposed DeepFlow, a deep-learning based system for identifying malware from the data flows in an Android app. Later next year, Li et al. implemented a fine-grained automatic detection engine to detect the families of malicious applications on Android devices, using Deep neural network. The features were comprehensive static ones taken from the apps to train the detection models. Kan et al. developed a light-weight deep learning based method for malware detection based on disassembled

instructions from sample files. In order to reduce complexity, the authors did instruction analysis on the decompiled code for classification of the assembled instructions into various groups. Karbab et al. made use of sequences classification using deep learning techniques for automatic Android malware detection and family assignment in MalDozer. This identifies the malicious and benign patterns by learning from the actual samples and serves as a ubiquitous system that can be deployed on mobiles, IoT devices, etc. In 2019, Kim et al. proposed an Android malware detection framework where effective feature vector generation method was used. They used the multimodal deep learning approach, which was designed to handle different types of features. Liu et al. did the same using BLSTM deep learning method on API calls based on cuckoo sandbox, which is used to extract the malicious API call records. A combination of the contextual information from the API calls and natural language processing were used. ScaleMalNet, a highly scalable self-learning based framework was designed by Vinaykumar et al. to accurately detect, classify and categorize malware to their corresponding malware family. Then, DL-Droid, which used a state-based input generation approach for Android malware detection and enhanced code coverage was introduced by Alzayalee et al. For this, a random-based input generation was used as the baseline.

6.4.5 Convolutional Neural Network

Generally, malicious binary files may be deleted from the file system of the virtual machine or the container, after some time. So, some of the target files may be missing and we would need to take a dynamic analysis approach then. Since this introduces extra runtime overhead, this is not generally used in cloud environments. So, [54] has introduced a technique where the system analyzes the memory image periodically during malware execution, to eliminate any extra time added. In this paper[54], we can see the detection task as a process of 3 steps:

- Export the extracted the memory snapshot from the running virtual machines or containers
- Extract the grayscale image from the above memory image
- Classify the generated grayscale image using the above CNN model as having malware or not

There were 3 main components in the architecture of the system[54] as shown below: The extracting module is basically where the memory snapshot of the target VM is dumped and the learning module is mainly used to train the CNN algorithm using the grayscale images. In the monitoring part, CNN is used on real-time extracted images to identify malwares. The dataset used here is again taken from the VirusTotal tool, and it was a part of the Win32 type malware samples (over 10000) for CNN model training. The labels given by this website were stored as the ground truth.
The memory snapshot is automatically extracted from the running target VM using the Cuckoo Sandbox

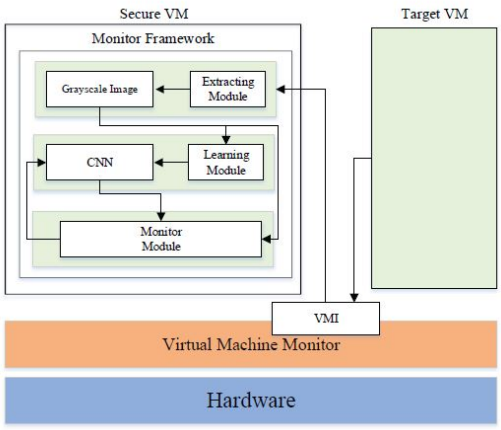

Figure 18: System Architecture

environment. It uses VMware Workstation to run Ubuntu as a host for Cuckoo, VirtualBox in Ubuntu to run Windows XP as guest for Cuckoo. Furthermore, a tool called 'Volatility' was used to analyze the extracted memory snapshot, to get a process list.[54] Next, based on the name of the malware sample, we get its process id, which is used in 'Volatility' to obtain the address space of the executable code, contained by the malware process. At the end, the executable code is stored on the host disk as a binary file.

After we obtain the binary file of the malware sample extracted from the memory snapshot, next we convert it to grayscale image. For this purpose, every time, the 8 bits of the binary file are read into a vector and the corresponding decimal value is calculated. According to the size of the binary file, the width of the grayscale image is decided. It can be seen in the paper that the grayscale images of the same malware family are very much similar and of different families have a lot of variance.

As part of the the CNN detector for these images, the authors used vgg-16 as their training model, with a filter of size 3*3 in the convolutional network. Also, the dropout layer was added between the fully connected layers (with p=0.5), to address the issue of overfitting for families with smaller size. After the CNN detection model is trained, 'Libvmi' tool was used as the read/write memory tool to dump the target virtual machine memory flexibly and constantly.

The f-score for 5 different families are obtained as 1.0, 0.983, 0.928, 0.875 and 0.740. The sample data was used in 60-20-20 proportions for training, validation and testing. The accuracy obtained was 90.5% and the average precision, recall and f-score were 0.775, 0.772 and 0.745. The average time consumed for step-1 with a file size of 1GB, is 22.081s and we can see that it increases linearly with memory size. For the same conditions (including 157 processes), it took 2.694s for step-2. Also, the average time taken to classify a grayscale image is 1.486s. For the Lbvmi tool, the average memory latency is around 18.7ns in

normal and 20.2ns when exporting the memory snapshot. The overhead is about 8.02%.

In [55], the authors also make use of a standard 2d CNN by training the model using metadata for each of the processes in a Virtual Machine (VM) in a hypervisor. Further, they enhance the classifier for malware detection by using a novel 3d CNN, and this helps to reduce the mislabelled samples during data collection and training to a great extent. This paper mainly deals with Trojans and Rootkits and are selected randomly, to reduce the selection bias of known-to-be highly active malware for easy detection. For the 2d CNN, a learning rate of 1e-5 has been used, which helped to obtain the highest accuracy and lowest cross entropy loss. From the results, we can see that the highest accuracy obtained is 85.9%, for a mini-batch size of 20 whereas the lowest is seen for a batch size of 30. When this is upgraded to the 3d classifier which takes in a time-windowed input, with a window size of 20 and 30 seconds. We can see a significant increase in classification accuracy to 86% and 90%, respectively, as opposed to the 2d CNN classifier accuracy of 79

IoT environment based malware detection is talked about in [58] and [59]. Novel techniques for detecting DDoS malware in IoT data, using CNN method achieve 94.0% accuracy for the classification of benign and DDoS malware, and 81.8% accuracy for the classification of benign and two main malware families are discussed in [58]. Detecting cryptoransomware in IoT networks based on energy consumption footprint for Android devices is talked about in [59].

7 Conclusions and Recommendations

7.1 Intrusion Detection and Prevention

Using machine learning for intrusion detection in cloud computing has its own advantages but there are various challenges that can come across while deployment of machine learning in intrusion detection and prevention. Some of the issues discussed in [35] are worth mentioning in the subsequent sections.

The general basis of anomaly detection for intrusion detection is deviation from expected behaviour. This was done through observing it statistically. Since then this field evolved into application of machine learning. Although machine learning works really well in anomaly detection in other fields , there are some specific reasons in intrusion detection for which using machine learning becomes a bit challenging. The traditional machine learning is used for performing classification of data rather than finding the outliers. While anomaly detection requires better efficiency in detecting the outliers. The error cost is very high. Even if there is a small error in classifying the data as intrusion , the analyst will have to spend a lot of time examining it only to realize at the end that it was an error from the classification of the machine learning. However, if an actual intrusion is classified as normal traffic , then even one small misclassification can cause the organization's systems and network compromised leading to loss of millions of dollars. There are chances of semantic gap in the intrusion detection system. In the real world there are many different types of network traffic that is being generated based on bandwidth , duration of connection etc. , which can be an outlier for the machine learning algorithm but not for the analyst or system operator. The operator understands the difference between an attack and just a new type of network traffic , while machine learning doesn't. This creates a semantic gap between outliers detected by the algorithm and the attacks. As stated earlier , there are various types of network traffic that are generated in the real world which are not outliers or attacks , this makes it difficult for the machine learning algorithm to come to a stable definition of what is a normal traffic for the systems. Hence, diversity of network traffic can be an issue when thinking of deploying a machine learning algorithm.

Another study [36] says that when the security signals are being created , just identifying the data as anomalous is not enough. It should be explained as to why that data was anomalous. In one of the examples given , if an .exe file was executed and flagged as anomalous , the analyst should know why it was flagged as anomalous. It could be that it was executed form temp folder or anything odd behaviour of the .exe file. To achieve this the feature extracted for the data set should also be added in the report. But this can add information overhead in the report. It could be that the report contains all the features of a data , in this case it could be features of a .exe file. But analyzing all the feature of the data my be unnecessary or even the top 10 features could not be that important. It could be that only one or two

feature of the data is important in other cases more than two. But there are chances where information about more can required feature is present for the analyst to analyze the data which has been classified as anomaly.

In Neural networks, we focused on stacked denoising autoencoders, which is used to reproduce actual IDSs' feedback from incomplete feedback. This is very useful in making quick decisions though the full feedback from other IDSs is not available and also without any aggregation on the consulted feedbacks from the IDSs'. Also, we discussed a neural network-based IDS scheme where each system in the network is considered as a node of a ANN. These are in-turn assigned in to the input, hidden and output layers of the ANN. The ANN is trained using the data that it gets from all the systems and accordingly the weights are updated. In the output layer, the iterations continue until the error no longer significant. Now this ANN is used for intrusion detection. However, the drawback of this approach is that, it sometimes fails when the attack is unknown.

But we can say that the advantages outdo the challenges of deploying a machine learning technique here. Although the challenges should be kept in mind while deploying the machine learning techniques but that shouldn't stop hinder or reduce the usage of machine learning . Also if we do a comparative analysis on the kind of techniques being used , it should be noted from the discussion in section 3.2 that a combination of multiple machine learning techniques can enhance the accuracy and detection of intrusion and very high accuracy in detection can be achieved. For example , a combination of FCM and SVM was able to achieve accuracy as high as 99.37 % , While the use of only SVM was able to garner an accuracy of 92 % ,which is appreciable but not as high as the combination of FCM and SVM. But accuracy and detection for other combinations of techniques may not be that high and could go below 90 % as well in some combinations, as discussed in 3.2 as well. Although , the analyst responsible for information assurance should be aware that as the accuracy is not always 100 percent there can be false alarms as well, but on the other side , having such a high detection rate of intrusion detection in cloud computing through machine learning will definitely help in protecting the system from attacks really well and can be used as an added weapon in protecting the organization's security.

7.2 Botnets

We conclude our study of botnets by re-stating the fact that botnets have become a primary threat to cloud infrastructures.Botnets have become the primary facilitators of various types of attacks, e.g. DDoS attacks. Although in previous decades, many botnet-based solutions have been introduced to counter these attacks; there are still many challenges yet to be overcome.[1]

- *M*ost of the current defense methods are mostly useful for low-rate or high-rate attacks but not for

both at the same time.

- *T*he efficiency of performance of most of the methods depends on network conditions and other user parameters.
- *P*roper methods to evaluate the performance of these defense methods, without bias is lacking.
- *M*ost of the current preventive methods are only for existing and known attacks. More adaptive and dynamic prevention mechanism is needed.

Botnets are known to provide the foundation of cyber attacks namely DDoS attacks, malware and phishing as well.They provide anonymity with the help of the Control and Command(C&C)architecture. The botnet protocols and structures are humongous and this makes botnet detection a challenging task. Botnet detection techniques fall mainly into the four categories: signature-based, anomaly-based, DNS-based and mining-based. Most of these techniques used now work only on specific C&C communication protocols. The some of the botnet detection techniques are based from data mining and are DNS-based, which are pretty effective and detect botnets with a very high accuracy [11].

There have been many new Botnet Detection Mechanisms(BDMs) which detect botnets from the DNS traffic by comparing two blocks of hosts requesting the same domain name by checking for similarity of requests(Jaccard similarity coefficient).The average detection rate in this manner gave an accuracy of 89 %.However these are based only on three experiments and an 11 % false positive and false negative was generated[2].

In botnet detection using unsupervised machine learning, when flow data was encrypted the algorithm results were noticed to be affected. Defence approaches that constitute removal of C&C servers have yielded good results. However botnets still seem to find a way through and continue malicious activities. With this we can infer that investigating botnets needs to be done closer to the source. This along with ISP information can help tackle the botnet problem[3].

Honeynets help gather information about threats especially from botnets. Research from honeynets shows that most attackers are highly skilled and with the help of botnets can take down any website or network. With the increase of research in IRC based botnet detection, these attackers are now aiming to use P2P networks for C&C servers[9].

There has been extensive research in the P2P and IRC botnets.They leverage HTTP-based network communication as it is a web-based C&C communication protocols which is easier than implementing customized C&C communication protocols.Data Mining and Machine Learning techniques are applicable on network flow information.These flow data are structures and hence do not require a lot of preprocessing. They also sometimes consist of patterns inside, which make Data Mining algorithms more applicable.

Some online botnet detection are categorized to be of the following kinds:early detection; novelty detection and adaptability. Early detection involves traffic processing. Bot activity in C&C and attack phases in terms of its network flow characteristics are noted. Decision trees classification gave high accuracy for successful botnet detection.[20]

As the use of P2P botnets is increasing, Machine Learning algorithms are being investigated to compare their performances.IBk, J48 and NB are few algorithms used for P2P botnet detection. Performance of IBk and J48 is known to exceed NB.However J48 seems to have more training time than IBk and IBk seems to have more testing time than J48.Hence there is a trade-off of testing and training times[24].

There are some issues that need attention for further improvement of the defense towards intrusion, DDoS and other attacks facilitated by botnets. Some future work is required in these fields to generate a more robust defense system[1].

- *D*eveloping methods that are effective for both low-rate as well as high-rate attacks(DDoS) is still a problem that needs attention.
- *M*ethods free from limitations especially from the user end(such as network) must become a research initiative.
- *E*valuation frameworks(bias-free) for most defense frameworks is locking.This seems to be an important issue to be investigated.
- *A* generic defense solution for all types of DDoS attacks of all protocols is needed.Besides, more adaptive defense mechanisms are needed for unknown attacks.
- *A* traceback mechanism which integrates with customer support and which is also cost-effective (by reuse of information while detection) while not compromising QoS remains a research area.

Botnet Detection Mechanisms(BDMs) could need some future work in tracing infected hosts and enhancing detection to more than 89% by using statistical methods(chi-square) along with current similarity coefficients[2].

In botnet detection using unsupervised machine learning, flow data when encrypted gave inaccurate results, hence careful preprocessing of flow data can be of scope for future work[3].

As botnets are growing to be more sophisticated in the near future, further research is needed in areas such as the decentralized/ P2P control architecture. Technologies implementing cryptography as well as blockchain can help in these areas. Modern 'sinkholing' and 'darknet' implementations are providing valuable information about botnets, future research must make use of these and provide more robust insights into botnet protection. It has also become crucial to know and be aware of the nature of attackers

and botnet owners, in order to better secure the community and understand the characteristics of botnet attacks and threats[6].

ISPs are predicted to provide more insight into botnets towards the source. Detection, prevention and defense at source computer and routers will be a great future initiative. This accompanied with dismantling botnet C&C hubs can help tackle the botnet problem majorly[7].

We need and hope to have developments in advanced honeypots(which participate in networks by web-crawling and P2P networks) that can help us gather information about botnets or modify them so that we can capture these malware.Currently botnets are focusing on C&C servers using IRC but soon may use non-IRC for the C&C servers , hence research in this area is needed, and security must be provided[9].

There is some more research required into botnets, as the presence of these malicious botnets has been there for a long time but the research is still in its infancy[11].

TS-ASRCaps is a deep learning framework being used as the state-of-art but there is more room for improvement in it such as representation of higher level concepts needs to be promoted via introducing new multimodal features into the system[19].

HTTP-based network communication may use attribute analysis like timeslot, data calculation, mutual authentication and bot clustering analysis to mimic botnets. Web botnets although not very adequate, can be used to study behaviours of botnets, to enhance detection techniques.

Online botnet detection especially for early detection needs future work, in installing individual detectors or have dedicated network monitor traffic in the network, which may lead to better performance and especially useful for large networks(ISPs). As P2P protocols are growing, it is now more important to detect between malicious and non-malicious protocols with a high accuracy[20].

The recent known Machine Learning algorithms for P2P botnet detection, that is, IBk,J48 and NB are seen to have high accuracies however with a trade-off with testing and training time. Future algorithms must be implemented or current algorithms must be fixed to overcome these limitations[24].

Detecting cloud-based botnets is a noted future work as current work is only for P2P botnets[25].

7.3 Distributed Denial-of-Services Attacks

In section 5.3.4.1 we concluded that using a non-linear time series models outperform linear model time series models. GARCH model was able to predict the future network traffic states and further, prediction error was calculated. Due to the use of threshold and ANN; system performs faster and having high detection rate, lower false positive rate. Moving forward with incorporating C4.5, Naive Bayes and K-means algorithm in section 5.3.4.2, comparative analysis of various machine learning strategies and algorithms was presented with the conclusion that C4.5 outperforms Naive Bayes and K-Means algorithm.

Section 5.3.4.4 compared and contrasts SVM, Random Forests and Naive Bayes, along with the help of SNORT. SVM and Random Forest were favourable over Naive Bayes by a large margin.

We also discussed self learning method to accommodate the dynamic change of network in section 5.3.4.3 where it is clearly evident that incorporating a small change in existing machine learning to learn more about the change in the network provides a better result.

To maintain high anonymity and higher network bandwidth, DOS and DDoS attacks are mostly sourced from virtual machines in the cloud rather than attacker's own machine. Previous research that focuses on analysing traffic on the destination side with predefined thresholds. These methods have some disadvantages like these are only passive defenses which can be detected only when the attack is done and the only response to these attacks is to cut off the connection on the destination side. To detect DoS and DDoS attacks at the source side, [48] and [49] has proposed the algorithm which showed good accuracy in detecting Dos and DDoS attacks. Not only on the source side, it is also very important for cloud providers to detect and thwart such DoS attempts. [50] has proposed such an efficient algorithm to implement IDS on cloud provider's end. The main drawback is, the proposed IDS doesn't consider IP spoofing. So, if the source IP address is spoofed, it won't be able to detect the attack.

All the approaches we have seen so far, can detect DoS and DDoS attacks based on supervised learning, but might fail to detect unknown attacks. [51] deals with this problem by proposing an ANN model to detect unknown attacks.

Grouping together all the results and conclusion from section 5, it is recommended that we employ machine learning algorithms to effectively detect Distributed Denial-of-Service (DDoS) Attacks. To be specific, it is important to employ complex and efficient algorithms that provide a better F1 measure.

7.4 Malware Attacks

Since we cannot guarantee to rid the internet of the existence of malware, the best we can do is to detect it early and secure our systems and data from its ill effects. Also, the increasingly rapid growth of the usage of cloud computing services, security against malware isn't a luxury rather a must and machine learning seems to be a hopeful resort at this point of time. We had discussed the working and effectiveness of some of the Machine learning algorithms in the previous section. Since we keep track of most of the malware attacks that happen regularly we are not short of datasets to train our models to face the existing problems. Along that lines, according to the paper [64], signature based and heuristic based malware detection techniques perform better than others when it comes to malware we already know of. There were other papers like [60] which aimed at tackling this problem of novel malware and proposed some interesting methods. According to [60], a self heal approach was proposed and it was found that there

was a huge disparity between the performances of various algorithms like Naive Bayes, SVM and Random Forest (82 percent, 90 percent and 96.75 percent accuracy respectively). But this is not the case across all the approaches. So, we are to conclude that there does not exist a single machine learning algorithm that outperforms every other algorithm and detects every possible malware attack.

As seen in section 6, it can be seen that we have a highly cost-effective Malware Detection-As-A-Service (MDaaS)[52], that was evaluated on 3 different kinds of architectures, namely On-Premise, cloud-based and serverless. These consisted of 4 kinds of malware detectors and used Deep Reinforcement Learning to assess the accuracy and cost of various security policies on each. MDaaS[52] reduced the cost by orders of magnitude. One of the most important challenges in this paper is based on how to effectively process multiple files in parallel and the optimization of such processing could be taken up as a future work. Another one is queue management i.e. how to select which file to process first. A possible recommendation is to use a better queueing technique, using latest technologies like Kafka and also, to use a more advanced reward function. From [53], we can easily compare a wide variety of deep learning based Android Malware detection techniques. We can see that the MalDozer model (designed in 2018) has the highest accuracy at 99% whereas the one by Kan et al. (also 2018) had the least at 95%. We can conclude that a majority of the research was on Android devices because they are the ones that get most malware attacks[53]. Also, we can see from the papers that 80% of the methods were based on static or dynamic data. With the advent of large scale social media and images on the web and mobile phones, one of the important future works would be that authors need to think about using deep-learning methods in image processing, for detecting malwares. This is useful because malwares can also be sent by embedding in images and deep learning would be relevant in this area.

From [54], we can see that CNN based malware detection approach, whose accuracy varies a lot among the different malware families. Even though the accuracy is pretty good, a few challenges in this work are that for a minority of the families, the grayscale images of the same family are quite different. Also, these have less number of malware samples in general. So, we can see that there is a great imbalance between the different malware families in the dataset, however, few of them have very high precision. Also, another challenge is that some of the different malware have the same hashcode causing to form similar grayscale images, that might cause overfitting problems. A probable future recommendation for the above issues could be to include a better and bigger dataset that has balanced samples of the malware families. A balanced dataset and varied dataset may also help in reducing the overfitting problem and help us get a better model accuracy. But, this paper does have pros like more security, high accuracy and minimal runtime overhead, which is very important for malware detection.

From [55], again we can see that the most common challenge is that we cannot differentiate between

labeling all the samples corresponding to the benign area(as that would pollute the data) and marking samples as malicious when the malware steals and sends data over the Internet. Another issue is the accuracy drop between validation and test set and the 3d CNN's increased need for data. From the experiment, we saw that a 40 secs time window decreased the accuracy a lot as there wasn't enough data for the model to learn and converge properly. As future work, CNN-based malware detection techniques with large scale data can be used to get better results. Another important recommendation is that the correlation between features need to be used to order the input matrices and may lead to better accuracy. A re-training step can be included to determine the effectiveness of ordering the processes and features in the input. One of the recommendation for [56] is we can research on developing malware detection and classification systems based on memory introspection and heuristic or statistical detection, as opposed to signature-based detection.[58] has the problem that it is vulnerable to complex code obfuscation techniques, and a recommendation to reduce this is by using more complex static features, such as Opcode sequences and API calls[58]. Many techniques are prone to False positive and negatives too and there is no well-known and accepted data set which can be used to evaluate the performance.

Overall, we can say that Signature-, behavior-, heuristic-, and model checking based approaches are being used for more than a decade [58]. Deep learning-, cloud-, mobile devices-, and IoT-based approaches are more recent [58]. Also, latest technologies like big data and block chain would give more chances to build a more effective detector. The below table gives a very good description of the pros and cons of every approach -

Malware Detection Approach	Pros	Cons
Signature-Based	Fast and efficient for known malware	Insufficient to detect new generation malware
	Used for many years	Prone to many *FPs*
	Effective to detect malware which belongs to the same family	Extracting signature takes time
		Vulnerable to obfuscation and polymorphic techniques
Behavior-Based	Determines the malware functionality	Produces high *FPs*
	Effective to detect new malware	Some behaviors are similar in malware and benign samples
	Effective to detect different variants of the same malware	Impossible to specify all behaviors
	Effective against obfuscation and polymorphic techniques	Difficult to group behavior as malicious and normal
Heuristic-Based	Can detect some previously unknown malware	Numerous rules and training phases
	Can use both static and dynamic features	Vulnerable to metamorphic techniques
Model Checking-Based	Effective to detect malware that belongs to the same family	Complex and resource-intensive technique
	Effective against obfuscation and polymorphic techniques	Obtains a limited view of the malware
		Cannot detect all new generation of malware
Deep Learning-Based	Powerful and effective	Not resistant to evasion attacks
	Reduces feature space drastically	Building a hidden layer takes time
Cloud-Based	Enhances the detection performance for PCs and mobile devices	Lacks real-time monitoring
	Bigger malware databases and intensive computational resources	Can disclose some sensitive data such as password, and location
	Easily accessible, manageable, and updates regularly	Over-head between client and server
Mobile devices-Based	Effective to detect traditional and new generation malware	Cannot detect complex malware
	Can use both static and dynamic features	Cannot scale large bundle of apps
IoT-Based	Can use both static and dynamic features	Cannot detect complex malware

Figure 19: Comparative analysis of Malware Detection approaches

References

[1] Hoque, Nazrul, Dhruba K Bhattacharyya, and Jugal K Kalita. “Botnet in DDoS Attacks: Trends and Challenges.” IEEE Communications surveys and tutorials 17.4 (2015): 2242–2270. Web.

[2] Manasrah, Ahmed M., Awsan Hasan, Omar Amer Abouabdalla and Sureswaran Ramadass. “Detecting Botnet Activities Based on Abnormal DNS traffic.” ArXiv abs/0911.0487 (2009): n. Pag.

[3] Wei Wu, Jaime Alvarez, Chengcheng Liu, Hung-Min Sun, “Bot detection using unsupervised machine learning”,©2016 IEEE DOI 10.1007/s00542-016-3237-0.

[4] Guofei Gu, Junjie Zhang, and Wenke Lee. ”BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic.” In Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS’08), San Diego, CA, February2008.

[5] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee. BotHunter: Detecting malware infection through ids-driven dialog correlation. In Proceedings of the 16th USENIX Security Symposium (Security’07), 2007.

[6] Ogu, Emmanuel & Ojesanmi, Olusegun & Oludele, Awodele & Kuyoro, Shade. (2019). A botnets circumspection: The current threat landscape & what we know so far. Information (Switzerland). 10. 337. 10.3390/info10110337.

[7] Khattak, S.; Ramay, N.R.; Khan, K.R.; Syed, A.A.; Khayam, S.A. A taxonomy of botnet behavior, detection, and defense. IEEE Commun. Surv. Tutor. 2013, 16, 898–924.

[8] Plohmann, D.; Gerhards-Padilla, E.; Leder, F. Botnets: Detection, Measurement, Disinfection & Defence; The European Network and Information Security Agency (ENISA): Heraklion, Greece, 2011.

[9] Honeynet Project and Research Alliance. Know Your Enemy: Tracking Botnets. Available online: http: //www.honeynet.org/papers/bots/ (accessed on 21 October 2020)

[10] Luo, H.; Chen, Z.; Li, J.; Vasilakos, A.V. Preventing distributed denial-of-service flooding attacks with dynamic path identifiers. IEEE Trans. Inf. Forensics Secur. 2017, 12, 1801–1815.

[11] Feily M, Shahrestani A, Ramadass S (2009) A survey of Botnet and Botnet detection. In: Third international conference on emerging security information, systems and technologies, SECURWARE ‘09, pp 268–273

[12] Han K-S, Im E (2012) A survey on P2P Botnet detection. In: Kim KJ the international conference on IT convergence and security 2011, vol 120. Springer, The Netherlands, pp 589–593

[13] Lu W, Rammidi G, Ghorbani AA (2011) Clustering botnet communication traffic based on n-gram feature selection. Comput. Commun 34:502–514

[14] Strayer, Lapsely, Walsh, and Livadas, Botnet detection based on network behaviour. in Botnet Detection, ser. Advances in Information Security, W. Lee,C. Wang, and D. Dagon, Eds. Springer, vol. 36, pp. 1–24, 2008.

[15] Masud MM, Al-khateeb T, Khan L, Thuraisingham B, Hamlen KW (2008) Flow-based identification of botnet traffic by mining multiple log files. Presented at the first international conference on distributed framework and applications, Penang

[16] Stevanovic and Pedersen, An efficient flow-based botnet detection using supervised machine learning, Department of Electronic Systems. Aalborg University. In Computing, Networking and Communications (ICNC), 2014 International Conference on p797–801 (IEEE), 2014.

[17] Barthakur, P., Dahal, M., Ghose, M.K., An efficient machine learning based classification scheme for detecting distributed command & control traffic of P2P botnets, I.J.Modern Education and Computer Science, p.9, 2013.

[18] Abu-Alia A, Detecting domain flux botnet using machine learning techniques, Qatar University, College of Engineering, 2015.

[19] Pei, X., Tian, S., Yu, L., Wang, H., & Peng, Y. (2020). A Two-Stream Network Based on Capsule Networks and Sliced Recurrent Neural Networks for DGA Botnet Detection. Journal of Network and Systems Management, 1 - 28.

[20] Zhao, Traore, Sayed, Lu, Saad, Ghorbani, and Garan, Botnet detection based on traffic behavior analysis and flow intervals, 27th IFIP International Information Security Conference, November, 2013.

[21] Haddadi, F. and Morgan, J., Botnet behaviour analysis using ip flows: with http filters using classifiers, 28th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 7–12, 2014.

[22] BEIGI, H. JAZI, N. STAKHANOVA AND A. GHORBANI, Towards Effective Feature Selection in Machine Learning-Based Botnet Detection Approaches, IEEE Conference on Communications and Network Security (CNS), 29-31, San Francisco, CA, 2014.

[23] Huseynov K, Kim K, Yoo PD, Semi-supervised botnet detection using ant colony clustering, SCIS 2014. In: The 31th symposium on cryptography and information security Kagoshima. The Institute of Electronics, Information and Communication Engineers, Japan, 2014.

[24] Garg S, Singh AK, Sarje AK, Peddoju SK (2013) Behaviour analysis of machine learning algorithms for detecting P2P botnets. In: 15th International Conference on advanced computing technologies (ICACT), pp 1–4

[25] Jiang H, Shao X (2014) Detecting P2P botnets by discovering flow dependency in C&C traffic. Peer-to-Peer Netw Appl 7(4):320–331

[26] Wen-Hwa L, Chia-Ching C (2010) Peer to Peer Botnet detection using data mining scheme. Presented at the the international conference on internet technology and applications, Wuhan

[27] Junjie Z, Perdisci R, Wenke L, Sarfraz U, Xiapu L (2011) Detecting stealthy P2P botnets using statistical traffic fingerprints. Presented at the IEEE/IFIP 41st international conference on dependable systems and networks (DSN), Hong Kong

[28] Zhang J, Perdisci R, Lee W, Luo X, Sarfraz U (2014) Building a scalable system for stealthy P2P-botnet detection. IEEE Trans Inf Forensics Secur 9:27–38

[29] N. Aboueata, S. Alrasbi, A. Erbad, A. Kassler and D. Bhamare, "Supervised Machine Learning Techniques for Efficient Network Intrusion Detection," 2019 28th International Conference on Computer Communication and Networks (ICCCN), Valencia, Spain, 2019, pp. 1-8, doi: 10.1109/ICCCN.2019.8847179.

[30] Sharifi, Aboosaleh and Amirgholipour Kasmani, Saeed and Pourebrahimi, Alireza. (2015). Intrusion Detection Based on Joint of K-Means and KNN. Journal of Convergence Information Technology(JCIT). 10. 42-51.

[31] I. Aljamal, A. Tekeoğlu, K. Bekiroglu and S. Sengupta, "Hybrid Intrusion Detection System Using Machine Learning Techniques in Cloud Computing Environments," 2019 IEEE 17th International Conference on Software Engineering Research, Management and Applications (SERA), Honolulu, HI, USA, 2019, pp. 84-89, doi: 10.1109/SERA.2019.8886794.

[32] A. Kannan, G. Q. Maguire Jr., A. Sharma and P. Schoo, "Genetic Algorithm Based Feature Selection Algorithm for Effective Intrusion Detection in Cloud Networks," 2012 IEEE 12th International Conference on Data Mining Workshops, Brussels, 2012, pp. 416-423, doi: 10.1109/ICDMW.2012.56.

[33] Jaber, A.N., Rehman, S.U. FCM–SVM based intrusion detection system for cloud computing environment. Cluster Comput (2020). https://doi.org/10.1007/s10586-020-03082-6

[34] A. Javadpour, S. Kazemi Abharian and G. Wang, "Feature Selection and Intrusion Detection in Cloud Environment Based on Machine Learning Algorithms," 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC), Guangzhou, 2017, pp. 1417-1421, doi: 10.1109/ISPA/IUCC.2017.00215.

[35] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," 2010 IEEE Symposium on Security and Privacy, Berkeley/Oakland, CA, 2010, pp. 305-316, doi: 10.1109/SP.2010.25.

[36] Kumar, Ram & Wicker, Andrew & Swann, Matt. (2017). Practical Machine Learning for Cloud Intrusion Detection: Challenges and the Way Forward. DOI: 10.1145/3128572.3140445

[37] H. Zhang, C. Q. Wu, S. Gao, Z. Wang, Y. Xu and Y. Liu, "An Effective Deep Learning Based Scheme for Network Intrusion Detection," 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, 2018, pp. 682-687, doi: 10.1109/ICPR.2018.8546162.

[38] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," in IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41-50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.

[39] Abusitta, Adel & Bellaiche, Martine & Dagenais, Michel & Halabi, Talal. (2019). A deep learning approach for proactive multi-cloud cooperative intrusion detection system. Future Generation Computer Systems. 98. 308-318. 10.1016/j.future.2019.03.043

[40] Z. Li, W. Sun and L. Wang, "A neural network based distributed intrusion detection system on cloud platform," 2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, Hangzhou, 2012, pp. 75-79, doi: 10.1109/CCIS.2012.6664371.

[41] M. Zekri, S. E. Kafhali, N. Aboutabit and Y. Saadi, "DDoS attack detection using machine learning techniques in cloud computing environments," 2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech), Rabat, 2017, pp. 1-7, doi: 10.1109/CloudTech.2017.8284731.

[42] A. Rukavitsyn, K. Borisenko and A. Shorov, "Self-learning method for DDoS detection model in cloud computing," 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), St. Petersburg, 2017, pp. 544-547, doi:10.1109/EIConRus.2017.7910612

[43] A. R. Wani, Q. P. Rana, U. Saxena and N. Pandey, "Analysis and Detection of DDoS Attacks on Cloud Computing Environment using Machine Learning Techniques," 2019 Amity International Conference on Artificial Intelligence (AICAI), Dubai, United Arab Emirates, 2019, pp. 870-875, doi:10.1109/AICAI.2019.8701238

[44] Rajendran, R., Santhosh Kumar, S.V.N., Palanichamy, Y. et al. Detection of DoS attacks in cloud networks using intelligent rule based classification system. Cluster Comput 22, 423–434 (2019). https://doi.org/10.1007/s10586-018-2181-4

[45] Khorshed, M.T., Ali, A., & Wasimi, S. Classifying different denial-of-service attacks in cloud computing using rule-based learning. Secur. Commun. Networks, 5, (2012) 1235-1247. https://doi.org/10.1002/sec.621

[46] Abusitta, A., Bellaiche, M. & Dagenais, M. An SVM-based framework for detecting DoS attacks in virtualized clouds under changing environment. J Cloud Comp 7, 9 (2018). https://doi.org/10.1186/s13677-018-0109-4

[47] O. P. Badve, B. B. Gupta, S. Yamaguchi and Z. Gou, "DDoS detection and filtering technique in cloud environment using GARCH model," 2015 IEEE 4th Global Conference on Consumer Electronics (GCCE), Osaka, 2015, pp. 584-586, doi:10.1109/GCCE.2015.7398603.

[48] Z. He, T. Zhang and R. B. Lee, "Machine Learning Based DDoS Attack Detection from Source Side in Cloud," 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), New York, NY, 2017, pp. 114-120, doi:10.1109/CSCloud.2017.58

[49] Y. Khosroshahi and E. Ozdemir, "Detection of Sources Being Used in DDoS Attacks," 2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/ 2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), Paris, France, 2019, pp. 163-168, doi: 10.1109/CSCloud/EdgeCom.2019.000-1.

[50] R. Kumar, S. P. Lal and A. Sharma, "Detecting Denial of Service Attacks in the Cloud,"2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), Auckland, 2016, pp. 309-316, doi: 10.1109/DASC-PICom-DataCom-CyberSciTec.2016.70.

[51] S. Alzahrani and L. Hong, "Detection of Distributed Denial of Service (DDoS) Attacks Using Artificial Intelligence on Cloud," 2018 IEEE World Congress on Services (SERVICES), San Francisco, CA, 2018, pp. 35-36, doi:10.1109/SERVICES.2018.00031.

[52] Y. Birman, S. Hindi, G. Katz, A. Shabtai, "Cost-Effective Malware Detection as a Service Over Serverless Cloud Using Deep Reinforcement Learning", 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), 978-1-7281-6095-5/20/ IEEE DOI 10.1109/CCGrid49817.2020.00-51

[53] P. Sreekumari, "Malware Detection Techniques Based on Deep Learning", 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), 978-1-7281-6873-9/20/ IEEE DOI 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00023

[54] H. Li, D. Zhan, T. Liu, L. Ye, "Using Deep-Learning-based Memory Analysis for Malware Detection in Cloud", 2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW), 978-1-7281-4121-3/19/ IEEE DOI 10.1109/MASSW.2019.00008

[55] M. Abdelsalam, R. Krishnan, Y. Huang and R. Sandhu, "Malware Detection in Cloud Infrastructures Using Convolutional Neural Networks," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, 2018, pp. 162-169, doi: 10.1109/CLOUD.2018.00028.

[56] M. M. Hatem, M. H. Wafy, M. M. El-Khouly (2014), "Malware Detection in Cloud Computing", International Journal of Advanced Computer Science and Applications, 5. 10.14569/IJACSA.2014.050427.

[57] z Li, J., Sun, L., Yan, Q., Li, Z., Srisa-an, W., & Ye, H. (2018), "Significant Permission Identification for Machine-Learning-Based Android Malware Detection", IEEE Transactions on Industrial Informatics, 14(7), 3216–3225, doi:10.1109/tii.2017.2789219

[58] J. Su, V. Danilo Vasconcellos, S. Prasad, S. Daniele, Y. Feng, and K. Sakurai, 'Lightweight classification of IoT malware based on image recognition', in Proc. IEEE 42nd Annu. Comput. Softw. Appl. Conf. (COMPSAC), vol. 2, Jul. 2018.

[59] A. Azmoodeh, A. Dehghantanha, M. Conti, and K.-K.-R. Choo, 'Detecting crypto-ransomware in IoT networks based on energy consumption footprint', J. Ambient Intell. Hum. Comput., vol. 9, no. 4, pp. 11411152, Aug. 2018.

[60] Self-Heal Approach of Virtual Machines in Cloud Computing," 2019 Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), Chennai, India, 2019, pp. 220-231, doi: 10.1109/ICONSTEM.2019.8918909.

[61] S.Kumar, C. B. B. Singh, "A zero-day resistant malware detection method for securing Cloud using SVM and Sandboxing Techniques", Proceedings of the 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT 2018) IEEE Xplore Compliant - Part Number: CFP18BAC-ART; ISBN:978-1-5386-1974-2

[62] M. R. Watson, N. Shirazi, A. K. Marnerides, A. Mauthe, D. Hutchison, "Malware Detection in Cloud Computing Infrastructures", IEEE Transactions on dependable and secure computing, Vol. 13, No. 2, March/April 2016, doi: 10.1109/TDSC.2015.2457918.

[63] T. Tiwari, A. Turk, A. Oprea, K. Olcoz and A. K. Coskun, "User-profile-based analytics for detecting cloud security breaches," 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, 2017, pp. 4529-4535, doi: 10.1109/BigData.2017.8258494.

[64] O. Aslan, R. Samet, "A Comprehensive Review on Malware Detection Approaches", Creative Commons Attribution 4.0 License, Vol. 8, DOI: 10.1109/ACCESS.2019.2963724