Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 167 (2020) 2297–2307

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2019)

Distributed Denial of Service (DDoS) Attacks Detection System for OpenStack-based Private Cloud

Karan B. Virupakshar [1], Manjunath Asundi [2], Kishor Channal [3], Pooja Shettar [4], Somashekar Patil[5] ,Narayan D. G. [6]

KLE Technological University, Hubballi, Karnataka, India
karanv25@gmail.com, poojashettar85@kletech.ac.in, narayan_dg@bvb.edu

Abstract

Cloud computing technique is commonly used by fast growing Internet-based applications. Moving to cloud computing results in reducing the cost of managing and maintaining IT infrastructure. Different organizations controls the cloud resources over internet using networking protocols and standards. This makes IT infrastructure distributed in nature but controlled centrally, which opens the door to the attackers for intrusions. Attacks as Distributed Denial of Service (DDoS) is one of the most popular intrusions in private cloud which cause a degradation of services being reduced or denial of services. In this work the sole focus is on DDoS attack which specifically targeted towards detection of bandwidth flooding and connection flooding. Such attacks target the network layer of the cloud set up with invalid requests and make it denial to legitimate requests. Thus, the entire cloud set up becomes vulnerable and can be disrupted by these DDoS attacks. To overcome this a cloud operating system which has an integrated firewall with DDoS detection system is required. Here a system with OpenStack integrated firewall and raw socket programming for monitoring the network traffic is proposed. Based on the dataset generated in controlled DDoS attack environment, algorithms such as Decision tree, K nearest neighbor (KNN), Naive Bayes and Deep Neural Network (DNN) algorithms are compared against the trained model. Eventually, DDoS attacks are detected, and administrator of the private cloud is notified.

Keywords: Cloud; Distributed Denial of Service (DDoS); OpenStack; Raw Sockets; K nearest neighbor; Deep Neural Network.

10.1016/j.procs.2020.03.282

1. Introduction

Cloud infrastructure holds its significance in relatively every field. The major advantages of using cloud infrastructure include cost-effectiveness, unlimited storage space, device diversity, scalability, backup and recovery, energy efficient, quick deployment, and agility. Cloud infrastructure has been changing the manner in which organizations work nowadays. Organizations of all kinds have been adjusting to this new innovation. Industry specialists trust that cloud infrastructure will keep on benefiting the average sized and extensive organizations in a coming couple of years.

Despite having so many advantages, security is the make or break factor when the organization is deciding if cloud infrastructure is the right choice for them. In recent years, there has been a surge in DDoS attacks because of the simplicity involved in initiating a DDoS attack. DDoS attacks can be broadly divided into bandwidth flooding and connection flooding. Technically, in DDoS attacks, there is one server and there are multiple systems trying to throw invalid requests simultaneously. Even after the server responds to this request, the system throws the same request repeatedly. This results in the crashing of the server [1]. Thus, the DDoS detection system is an essential tool in the overall development of an organization's statement by describing the rules and practices to deliver security. The research community has been working on detecting several different kinds of DDoS attacks occurring in the cloud such as ICMP flooding, HTTP flooding, TCP flooding [2].

In this work, the aim is to detect DDoS attacks in the private cloud deployed using OpenStack. Several classifier models were compared and the model with best accuracy and precision was selected. Network traffic from the cloud is captured and processed dynamically in order to find anomalies. In the final phase, the cloud administrator is notified about the system causing the attack. The cloud administrator can then choose to add the systems network address to the integrated OpenStack firewall.

2. Literature Review

In [3], the authors identify the characteristics of a DDoS attack and provide an Intrusion Detection System (IDS) tool based on Snort to detect DDoS. Here the system is proposed which will alert the network administrator regarding any attack for any possible resources and the nature of the attack. Also, it suspends the attacker for some time to allow the network admin to implement a fallback plan. The proposed tool helps minimize the effect of DDoS by detecting the attack at a very early stage and by altering with various parameters which facilitate to easily diagnose the problem.

In [4], the authors discuss the importance of SDN-based cloud against DDoS attacks in cloud computing. It also highlights the importance of security to SDN itself from the attacks. This paper also discusses the new trends and characteristics of DDoS attacks in cloud computing.

In [5], the authors discuss the cloud architecture and its design. He also discusses the importance of detecting DDoS attacks at the network layer. This paper proposes machine and deep learning algorithms namely Decision tree, Naïve Bayes, DNN, KNN to detect DDoS attacks in SDN. Then these algorithms are evaluated using parameters such as detection rate and efficiency.

In [6], the authors propose a framework integrating network intrusion detection system (NIDS) in the Cloud. NIDS module consists of Snort and signature apriori algorithm. It generates new rules from captured packets. These new rules are appended in the Snort configuration file to improve efficiency of Snort. It aims to detect known attacks and derivative of known attacks in Cloud by monitoring network traffic, while ensuring low false positive rate with reasonable computational cost.

2.1 Distributed Denial of Service

A distributed denial-of-service (DDoS) attack is an intrusion attempt to interrupt normal traffic of a targeted website, server or any other network resources and leads to the denial of service to the legitimate users. DDoS attacks attain success by exploiting multiple compromised computer systems as sources of attack traffic. From a top view, a DDoS attack can be viewed like a traffic jam blocking up with highway, blocking fixed traffic from arriving at its preferred destination. Different types of DDoS attack include: Bandwidth flooding and connection flooding.

2.1. 1 Bandwidth Flooding

The bandwidth flooding exhausts all available bandwidth to a particular organization, main targets being outside links and edge routers. The goal of the attack is to overflow random ports on a remote machine. This causes the machine to indefinitely check for the application hearing at that port and (when no application is found to communicate) to

respond with an ICMP 'Destination Unreachable' packet [6]. This procedure unsettles host resources, which can, in the end, lead to inaccessibility [7] [8]. There are several variants of flooding algorithms. Most work approximately as follows: Each machine acts as both a transmitter and a receptor. Each machine attempt to forward every message to every one of its adjoining machines except the source machines. A variant of flooding called selective flooding fractionally addresses these issues by only sending packets to routers in the same direction.

2.1.2 Connection Flooding

Connection flooding (a.k.a. SYN flood) is a type of DDoS attack that utilizes part of the regular TCP three-way handshake to waste resources on the targeted machine and requite it unresponsive [9]. Essentially, with SYN flood DDoS, the attacker emits TCP connection requests faster than the targeted machine can process them, causing network overload. In this work, SYN flooding [10] [11] is going to be simulated. A SYN flood program works by producing SYN packets which need raw socket support [12]. Linux has raw socket support naturally and hence the program suggested in this work shall work only on a Linux system. This causes crowding of the networking route of a remote machine or host. This results in the server being unable to serve actual requests from legitimate users [13].

2.2. Open stack architecture

OpenStack is a communicative and open-source software model for cloud computing, mostly extended as an infrastructure-as-a-service (IaaS). The software model comprises correlative components that guide different, several vendor's hardware pools of processing, storage, and networking mean throughout a data center. Users either govern it through a web-based console, through command-line tools or using a set of RESTful APIs [14] [15]. The OpenStack community collaborates around a six-month, time-based release cycle with persistent growth milestones. During the design phase of each release, the community gathers for an OpenStack Design Summit to ease programmer working sessions and to convene schemes.

The Neutron, Controller, Compute and Horizon are the three main components of OpenStack architecture.

• *Neutron*: Neutron is the component that provides the software-defined networking stack for OpenStack. It gives the cloud tenants an API to build rich networking topologies and configure advanced network policies in the cloud.

• *Compute***:** Nova compute provides a platform on which instances are going to be launched. It's the instance provisioning and management module that defines drivers that communicate with underlying virtualization.

• *Horizon***:** Horizon is a web-based graphical interface that users and administrators to manage OpenStack storage, compute and networking services. For example, the administrators can use Horizon to launch virtual machine instances, view current state and size of their OpenStack cloud deployment, and manage networks.

• *Controller***:** Controller is the heart and brain of the OpenStack. All the major components like Neutron, Compute, and Horizon are present here and can be controlled using one of the commercial controllers.

The OpenStack architecture can be of two types which are single-node and multi-node architectures. The single-node architecture of the OpenStack is having a single computer that runs all the services. This architecture is good if the computer is of high resource capacity. The multi-node architecture of the OpenStack usually has three different running each node of OpenStack. Single-Node architecture is usually used for development purpose. But the multinode architecture is used in case of production purpose and is the one that is implemented in organizations [8].If the system has low resource capacity, there are chances of failure in the OpenStack operations. In this work single-node OpenStack has been deployed.

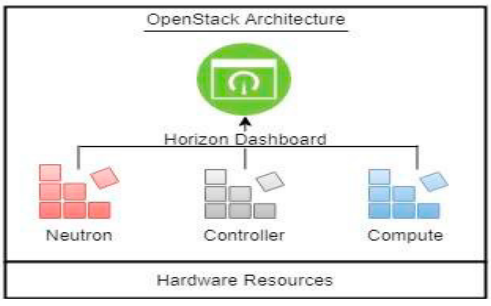


Figure 1: Open stack architecture

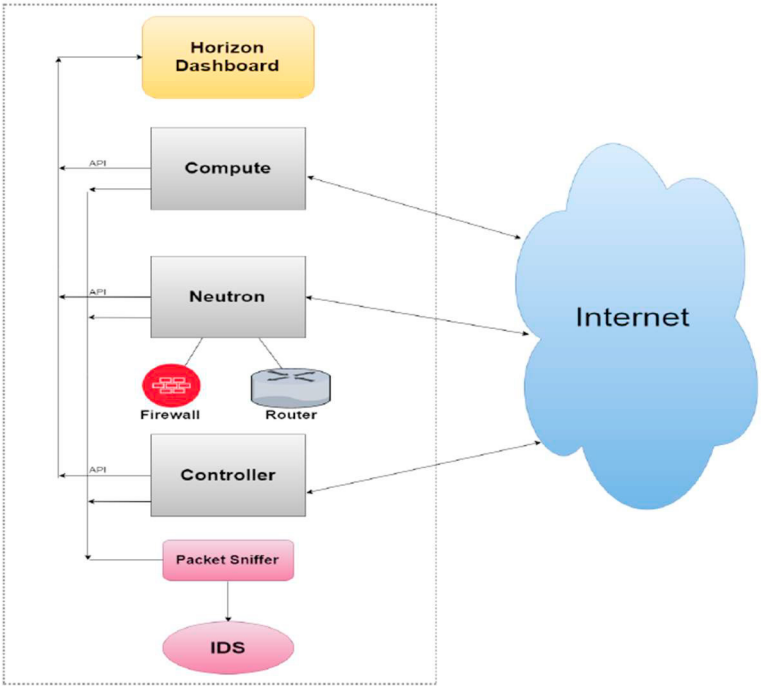
3. Proposed Work

This section explains the network traffic capture, preparation of training dataset and different machine learning algorithms and Deep Neural Network (DNN) algorithm proposed to detect the DDoS attacks in OpenStack based cloud. Socket programming has been used for network traffic capturing, three machine learning algorithms, namely Naive Bayes, Random Forest, Decision Tree and a DNN algorithm to implement and test the IDS in OpenStack based cloud.

3.1. System Model

Figure 4 shows the system model. It consists of 4 basic components when it comes to OpenStack. As one can see, the 4 components are; Horizon, Compute, Neutron, and Controller. The Horizon represents the Dashboard or the GUI interface of the whole OpenStack. Functionalists like creating user, project, viewing resource usage summary and many more can be managed in this dashboard. The next part is computed where all the managing happens. The computing section is the place that handles the process of redirecting and managing the whole file set. The third section is the neutron which deals with the networking part of the OpenStack. This consists of the router which is connected to multiple connections. The firewall sits on this.

The policies and rules to the firewall are added and thus incoming packets can be managed and detect if it's an anomaly or not. The fourth section is the controller section that controls the working of the whole OpenStack. Further DDoS detection module will use the captured data to check if the connection is normal or anomaly. It also handles notifying the administrator after detection.



. Figure 2: Connection flooding

3.2. Methodology

Naive Bayes, Random Forest, Decision Tree, Deep Neural Network have been used for building the model that are used in detecting the DDoS attacks.

Algorithm 1 Pre-processing and model generation

Require: Packet Sniffer using socket programming
Ensure: Model with maximum possible accuracy
1: Network traffic in the cloud is captured using raw socket programming to prepare the training dataset
2: DDoS attacks are simulated using LOIC tool
3: Important features pertaining to DDoS attacks are extracted
4: Out of 13 features captured as shown in Figure 5, dataset is pre-processed and converted to 7 features as shown in the Figure 6
5: Training models are constructed using Naive Bayes, Random Forest, Decision Tree algorithms, Deep Neural Network
6: Model with highest accuracy is selected for IDS in Openstack based cloud

Algorithm 2 Details for IDS using Deep Neural Network.

Require: Trained DNN model built using dynamic training dataset
Ensure: Correct detection of attacker
1: Dataset is created by capturing the network traffic from the neutron node of the cloud
2: Captured dataset is processed to the format of training dataset
3: The obtained values is classified by the DNN model
4: **if** *classifier classifies the request as anomaly* **then**
5: Labels it as 1 and identifies IP address of the attacker
6: **else**
7: Labels it as 0 and displays the connection as normal
end if

Algorithm 1 shows the algorithm for capturing and pre-processing of the cloud dataset. It also shows model generation based on the training dataset. Algorithm 2 shows the algorithm for anomaly-based IDS using DNN classifier. DNN classifier classifies the new incoming connections as anomaly or normal.

4. Experimental Setup

An OpenStack based cloud is set up in our system and a machine running Ubuntu is hosted in the cloud, for which intrusion detection is implemented. A protocol based packet sniffer is considered which will track packets by attributing itself to a socket in promiscuous mode, packets will be tracked according to the filtering standards set by the network administrator. The acceptance and rejection of packets will be completed based on these standards. It will also accumulate the log into a database for further analysis. An integrated intrusion detection system will lessen the efforts of the network administrator essential to manually analyze each packet. The automated system will decrease the time required to analyze each packet and efficiently interpret the contents of each packet. The DDoS attack will be simulated which aims to send the packets with large data, and from multiple connections to simulate both bandwidth flooding and connection flooding. The captured dataset is checked for the bandwidth flooding and connection flooding. The administrator receives the IP addresses of the systems performing the attack. The graph or charts are displayed based on the analysis so that the administrator can easily identify the attacker. These modules are implemented in the private cloud.Table1 shows experimental setup used.

Table 1. Configuration used for experiment

Sl.No	Node	Configuration			IP address
		RAM (GB)	MEMORY (GB)	PROCESSOR (No. Of core)	
1	Controller	1	50	2	192.168.120.2
2	Neutron	1	20	2	192.168.120.3
3	Compute-1	1	20	2	192.168.120.4
4	Compute-2	1	20	2	192.168.120.5

4.1. Dataset

Since Deep Neural Networks has been used, training dataset for the learning process is needed. The training dataset will be created in the test set-up cloud environment. The cloud with 5 virtual machines has been set-up each having their own IP addresses. Raw socket programming has been used to capture the 13 fields mentioned in Table 2.

Table 2: Attributes in the data-set.

Serial No.	Features
1	Source MAC
2	Destination MAC
3	Protocol
4	Service
5	Source Address
6	Destination Address
7	Sequence Number
8	Acknowledgment number
9	SYN Bit
10	ACK Bit
11	FIN Bit
12	Src Bytes
13	Duration

Further, the dataset is processed, and attributes shown in Table 3 are obtained. Low Orbit Ion Cannon (LOIC), a DDoS attack simulation tool, has been used to simulate DDoS attacks in a cloud environment. These DDoS attacks include ICMP flooding, TCP flooding, and HTTP flooding. There are several attributes in the data-set. The system will preprocess and transform the captured dataset into 7 fields so that it's in the format that is required for training. For example, the "src bytes" is calculated by adding all the data length from a unique source IP address with the different destination IP addresses in a 2-second frame. Similarly, "dst bytes" is calculated by adding all the data length from a unique destination IP address with different source IP addresses in the 2-second frame

Table 3: Protocol types and their respective value

Serial No.	Features
1	Protocol_type
2	Service
3	Src_bytes
4	Dst_bytes
5	Count
6	Srv_count
7	Same_srv_rate

Figure 3 shows the raw dataset which contains inconsistent data types. So, processing of the data to meet the requirements which will provide with efficient results. The processed data is shown in Figure 4.

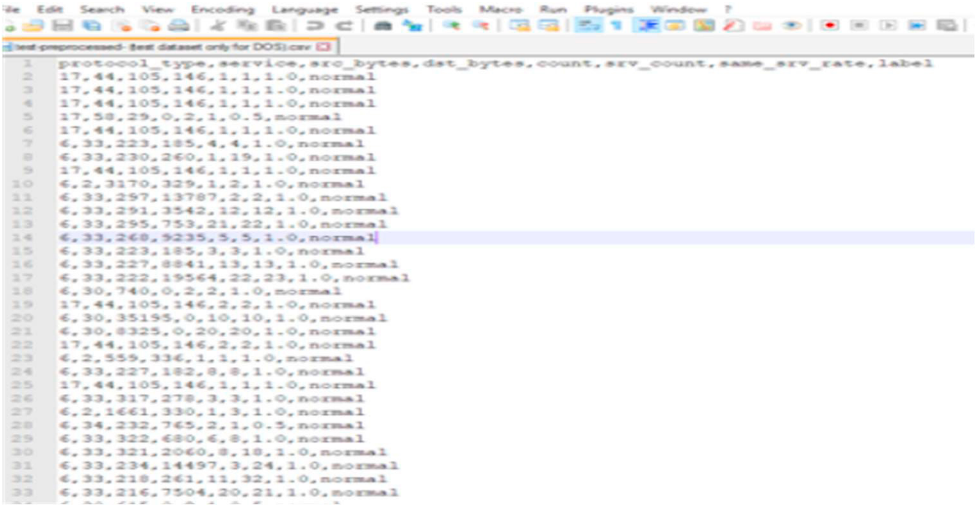


Figure 3: Raw data

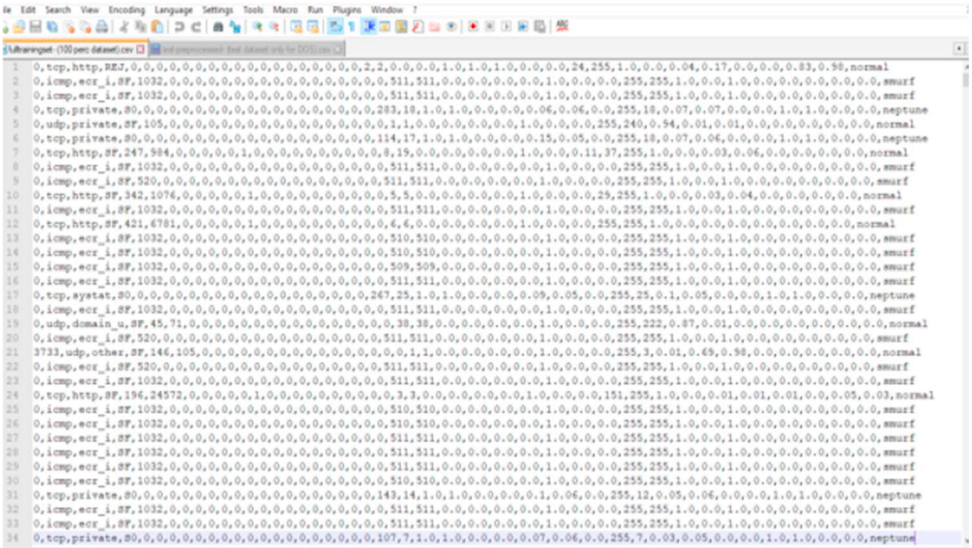


Figure 4: Processed data.

4.2. Topology

Figure 5 shows a simple representation of the topology used. It depicts the network infrastructure for the network in OpenStack. As depicted in figure three networks have been created, two are private networks and another one is the public network. Routers were configured between the networks. And two instances were deployed based on Ubuntu Xenial image.

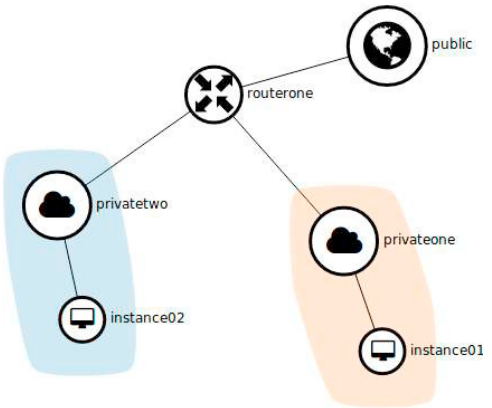


Figure 5: Network Topology

5. Results and Discussions

This section discusses the results of the simulated proposed IDS for OpenStack based private cloud. OpenStack is used to create hosts, switches, controllers, and links on a single Linux kernel virtually. It makes a single system appear as a complete network using virtualization.

Precision is defined as actual correct classification i.e., from all the instances classified as normal, how many are really normal. The obtained value explains how precise the machine learning algorithm is in classifying the requests/instances as normal. It is also said to be a positive predictive value. This implies that almost every item labeled as normal using DNN are actually normal and thus it has classified correctly. The reason why the algorithm with higher precision should be chosen because there shouldn't be any false positives in classification.

Table 4. Decision Tree Classifier Parameters

Dataset	Recall	F1-Score	Support
KDD Cup	0.94	0.93	2170
LAN	0.96	0.96	2180
Cloud	0.97	0.97	2182

Table 5. KNN Classifier Parameters

Dataset	Recall	F1-Score	Support
KDD Cup	0.98	0.98	2186
LAN	0.91	0.91	2146
Cloud	0.91	0.91	2140

Recall is defined as from all normal instances; how many are correctly classified. The algorithm with higher recall values is chosen because there should be minimal false negatives.

Table 4 shows the values of different parameters of each type of dataset in terms of Decision tree classifier. DCT classifier algorithm gives a precision of 92 percent for cloud dataset. This means that the DCT algorithm is 92 percent precise in classifying the requests into normal class. The DCT algorithm is comparatively more precise for KDD Cup dataset than any other dataset. The algorithm provides better recall, F1-Score, and Support values for KDD Cup dataset compared to other datasets.

Table 5 shows the values of different parameters of each type of dataset in terms of KNN classifier. KNN classifier algorithm gives a precision of 91 percent for cloud dataset. This means that the KNN algorithm is 91 percent precise in classifying the requests into normal class. The KNN algorithm is comparatively more precise for KDD Cup dataset

than any other dataset. The algorithm provides better recall, F1-Score, and Support values for KDD Cup dataset compared to other datasets.

Table 6. Naïve Bayes Classifier Parameters

Dataset	Recall	F1-Score	Support
KDD Cup	0.98	0.98	2188
LAN	0.91	0.92	2140
Cloud	0.92	0.92	2142

Table 6 shows the values of different parameters of each type of dataset in terms of Naive Bayes classifier. NB classifier algorithm gives a precision of 91 percent for cloud dataset. This means that the NB algorithm is 91 percent precise in classifying the requests into normal class. The NB algorithm is comparatively more precise for KDD Cup dataset than any other dataset. The algorithm provides better recall, F1-Score, and Support values for KDD Cup dataset compared to other datasets.

Table 7. DNN Classifier Parameters

Dataset	Recall	F1-Score	Support
KDD Cup	0.99	0.98	2190
LAN	0.91	0.91	2140
Cloud	0.91	0.91	2138

Table 7 shows the values of different parameters of each type of dataset in terms of DNN classifier. DNN classifier algorithm gives a precision of 96 percent for cloud dataset. DNN classifier is best suited for the dataset generated dynamically in the cloud environment.

Figure 6 shows a comparative graphical representation of Decision tree algorithm, KNN algorithm, Naive Bayes algorithm, and DNN algorithm w.r.t accuracy. Accuracy is defined as how accurately the classifier classifies both positive and negative instances into positive class and negative class respectively. It can be observed that KNN has low values for accuracy for KDD Cup dataset, VM dataset, and Cloud dataset when compared to other algorithms.

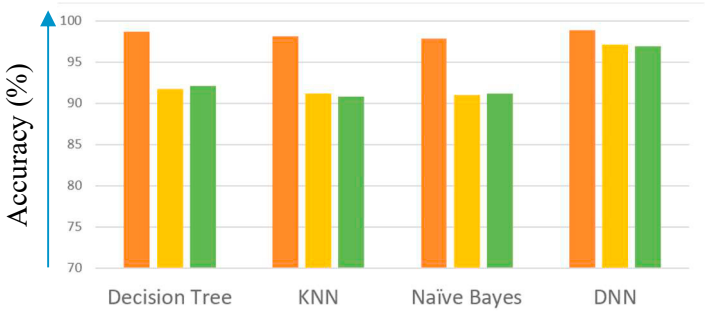


Figure 6: Accuracy Comparison of algorithms

Figure 7 shows a comparative graphical representation of Decision tree algorithm, KNN algorithm, Naive Bayes algorithm and DNN algorithm w.r.t precision.

From the results, it is clear that Deep Neural network has higher precision value and accuracy value compared to that of other classifiers when the dynamically generated dataset is being used. Thus, it can be said that DNN is better suited for the dataset that is generated dynamically from the cloud. It gives a better classification of requests as an anomaly or not since it has more significant values compared to other algorithms.

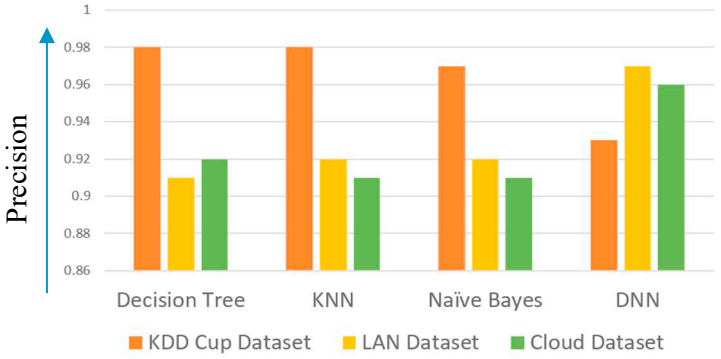


Figure 7: Precision Comparison of algorithms

5. Conclusion and Future Work

Cloud is advantageous because the data can be remotely maintained. Also, the cloud has gained importance due to its feature of 'unlimited storage'. The OpenStack cloud is another emerging domain. It can provide additional privileges to cloud set-up like administration of multiple users, dashboard interface, creating one's own topology. But the existing firewall in the OpenStack did not handle DDoS attacks specifically. Thus a DDoS detection module was required along with the firewall to have a proper security system in OpenStack. It would also be convenient for the network administrator to get a notification as and when a DDoS attack occurs. This experiment shows an overview of DDoS attacks, specifically bandwidth flooding and connection flooding, detection schemes and finally research issues and challenges have been presented. In addition, a comparison among current detection methods and a way to notify the administrator about the IP addresses causing DDoS attack has been provided.

The project can be extended to learn how several other DDoS attacks can overwhelm the controller in the cloud and thus cause problems to the OpenStack based private cloud. Algorithms can be modified to detect a wider range of DDoS attacks than already discussed. Optimized algorithms in Hadoop architecture can be used so as to get better efficiency and accuracy to further detect different kinds of DDoS attacks.

References

[1] Mukkamala, Srinivas, Guadalupe Janoski, and Andrew Sung. "Intrusion detection using neural networks and Support vector machines." Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on. Vol. 2. IEEE, 2002.

[2] Barki, Lohit, et al. "Detection of distributed denial of service attacks in software-defined networks." Advances in Computing, Communications and Informatics (ICACCI), 2016 International Conference on. IEEE, 2016.

[3] Bikram Khadka, Chandana Withana, Abeer Alsadoon, Amr Elchouemi, 2015. Distributed Denial of Service attack on Cloud Detection and Prevention. School of Computing and Mathematics, Charles Sturt University, Sydney, Australia Hewlett Packard. 2015 International Conference (pp. 1-5). IEEE.

[4] Qiao Yan, F. Richard Yu, Qingxiang Gong, Jianqiang Li, 2015. Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges. 2015 IEEE Communications Surveys & Tutorials (pp. 2-4). IEEE.

[5] Narayan D G, Mohammed Moin Mulla, Lohit Barki, Amrit Shidling, Nisharani Meti, 2016, September. Detection of Distributed Denial of Service Attacks in Software Defined Networks. 2016 Intl. Conference on Advances in Computing, Communications, and Informatics (pp. 1-3) (ICACCI).

[6] Chirag Modi,Dhiren Patel, Hiren Patel, Bhavesh Borisaniya, Avi Patel, Muttukrishnan Rajarajan, Integrating Signature Apriori based Network Intrusion Detection System (NIDS) in Cloud Computing, Procedia Technology, Volume 6,2012,Pages 905-912

[7] S. Umarani, D. Sharmila," Predicting Application Layer DDoS Attacks Using Machine Learning Algorithms" World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:8, No:10, 2014.

[8] P. S. Sheela and M. Choudhary, "Deploying an OpenStack cloud computing framework for a university campus," 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, 2017, pp. 819-824.

[9] Dimitrios Gkounis," Cross-domain DoS link-coding attack detection and mitigation using SDN principles" Master Thesis MA-2013-18 October 14, 2013, to April 13, 2014.

[10] Wang, Bing, et al. "DDoS attack protection in the era of cloud computing and software-defined networking." Computer Networks 81 (2015): 308-319.

[11] Gu, Yu, Andrew McCallum, and Don Towsley. "Detecting anomalies in network traffic using maximum entropy estimation." Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement. USENIX Association, 2005.

[12] Tsai, Chih-Fong, et al. "Intrusion detection by machine learning: A review." Expert Systems with Applications 36.10 (2009): 11994-12000.

[13] N. Meti, D. G. Narayan and V. P. Baligar, "Detection of distributed denial of service attacks using machine learning algorithms in software-defined networks," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, 2017, pp. 1366-1371.

[14] Bikram Khadka, Chandana Withana, Abeer Alsadoon, Amr Elchouemi, 2015. Distributed Denial of Service attack on Cloud Detection and Prevention. School of Computing and Mathematics, Charles Sturt University, Sydney, Australia Hewlett Packard. 2015 International Conference (pp. 1-5). IEEE.

[15] Qiao Yan, F. Richard Yu, Qingxiang Gong, Jianqiang Li, 2015. Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges. 2015 IEEE Communications Surveys & Tutorials (pp. 2-4). IEEE.