A Novel Hybrid-Network Intrusion Detection System (H-NIDS) in Cloud Computing

Chirag N. Modi
NIT Surat, India
cnmodi.956@gmail.com

Prof. Dhiren Patel
NIT Surat, India
dhiren29p@gmail.com

***Abstract*— To detect and prevent network intrusions in Cloud computing environment, we propose a novel security framework hybrid-network intrusion detection system (H-NIDS). We use different classifiers (Bayesian, Associative and Decision tree) and Snort to implement this framework. This framework aims to detect network attacks in Cloud by monitoring network traffic, while ensuring performance and service quality. We evaluate the performance and detection efficiency of H-NIDS for ensuring its feasibility in Cloud. The results show that the proposed framework has higher detection rate and low false positives at an affordable computational cost.**

Keywords—Cloud computing; Virtualization; Network Intrusion detection; Classifier; Snort;

I. INTRODUCTION

Cloud computing delivers a convenient, on-demand, network access to a shared pool of configurable computing resources "As a Service" on the Internet for satisfying computing demand of users [1].

Fig. 1 depicts Cloud computing architecture, consisting of mainly two ends viz; the front end (Cloud users and Cloud controller) and the back end (Host machine, virtual network and virtual machines (VMs)). Using the front end, Cloud users request the instances of offered services through Internet. The Cloud controller manages Cloud applications through their entire life cycle, from provisioning to monitoring, metering, and billing. Host machine consists of computer hardware and software, which processes the user's query and executes it for allowing to access VM instances, where Cloud application is running. It queries and controls the system software on its node (E.g. Host operating system and Hypervisor) in response to queries and control request coming from the front end. Virtual network (Internal network) is designed for VM instance interconnectivity. The sensor can be firewall (or any defense system) to secure the Cloud.

A. Network Threat Model for Cloud

As per International Data Corporation (IDC) [2] and K. poovic *et al.* [3], security to Cloud resources and services is the greatest challenge of Cloud. In a survey [4], we have discussed various security issues and their existing solutions by providing some open challenges at each layer of Cloud.

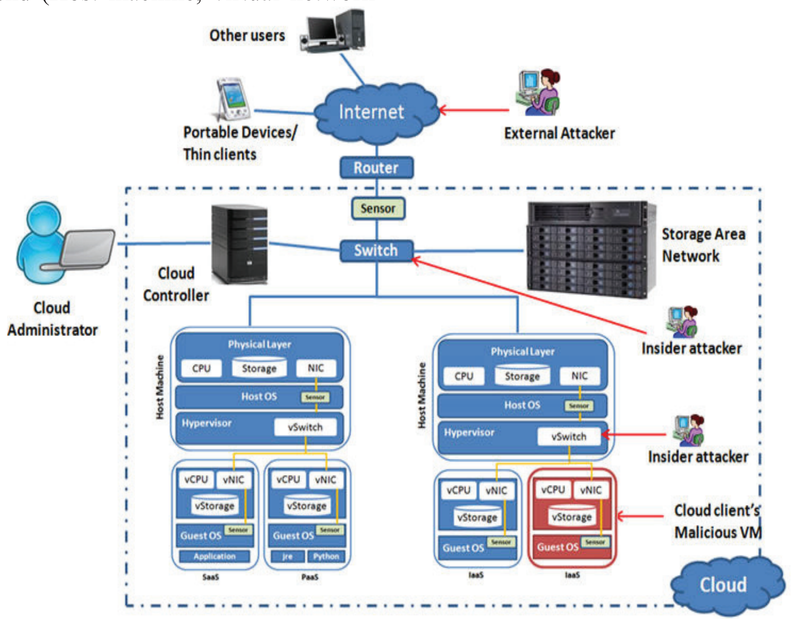

Fig. 1. Architecture of Cloud computing and attack model.

One of the major security issues in Cloud is to detect and prevent network intrusions since the network is the backbone of Cloud, and hence vulnerabilities in network directly affect the security of Cloud. L. Martin from Cyber Security division [5] stated that the major security concern after data security is an intrusion detection and prevention in the Cloud. As shown in Fig. 1, there are mainly two types of threats viz; insider (attackers within a Cloud network) and outsider (attackers outside the Cloud network) considered in Cloud network.

Insider attackers viz; malicious user at client side, malicious user at Cloud provider side and provider itself, can learn authentication information to gain access of the other's VMs. Malicious provider monitors network communication to gain information about client's behavior. Outsider attacker can be called as a network attacker who is able to perform various attacks viz; inserting malicious network traffic, man-in-the middle attack, Denial of Service (DoS)/Distributed Denial of Service (DDoS) attacks, phishing attack etc. In [6], we have presented several intrusions (insider attack, flooding attack, user to root (U2R) attack, attacks on a virtual machine or hypervisor and backdoor channel attack) to Cloud resources and services.

To address above issues, network based intrusion detection system (NIDS) can be deployed in the Cloud. NIDS captures the network packets and applies intrusion detection technique on captured packets in order to detect network attacks. In [6], we have discussed existing NIDS techniques that can be classified into two categories viz; signature based detection and anomaly detection. Signature based detection technique defines a set of rules (or signatures) that are used to decide that a given pattern is that of an intruder or not. In the Cloud, the signature based detection can be used to detect external intrusions at the front end or to detect external/internal intrusions at the back end. Like traditional network, it fails to detect the unknown attacks. The anomaly detection involves the collection of data relating to the behavior of legitimate users over a period of time, and then applies statistical tests to the observed behavior, which determines whether that behavior is legitimate or not. It has the advantage of detecting unknown attacks. Anomaly detection can be used for Cloud to detect unknown attacks at different layers. The only limitation of anomaly detection is that it produces many false alerts.

B. Our Contribution

We propose a novel security framework that integrates hybrid-NIDS (H-NIDS) to Cloud. We deploy our H-NIDS sensor on each host machine to monitor and detect network intrusions in Cloud environment. We use both the techniques viz; signature based detection and anomaly detection. We use Snort as a signature based detection, while for detecting network anomaly, we use different classifiers viz; Bayesian, Associative and Decision tree, which are discussed later in this paper. Moreover, a suitable score function determines whether the intrusions predicted by different classifiers are actually intrusion or not. Also, it is used to detect distributed attack in Cloud. We have evaluated the performance results of H-NIDS.

The rest of this paper is organized as follows: Section II discusses the challenges to Cloud NIDS and existing NIDS approaches to Cloud followed by theoretical background used in designing H-NIDS. A detailed description of the proposed H-NIDS is given in section III. Experimental results related to the performance and quality are given in section IV. Section V concludes our research work with references at the end.

II. Related Work and Theoretical Background

Our main objective is to detect network attacks in traditional as well as a virtual network in a Cloud, while reducing false alerts at an affordable computational cost and higher detection accuracy.

A. Challenges to Cloud NIDS

Cloud NIDS has some major challenges that are as follows:

1) Attacks on virtual environment: As presented earlier, malicious user having a VM instance can perform various attacks on other's VM and its host machine. For Cloud NIDS, it is difficult to detect attacks in a virtual network. Cloud NIDS should be capable of monitoring and detecting intrusions from network traffic between VM and the host.

2) High network traffic: Recently, Cloud is a rapidly growing computing model that offers various benefits in business and economic aspects. Therefore, Cloud users are increasing at a very high rate. This produces heavy network traffic from a large number of Cloud users. Intrusion detection activity in such traffic should be very fast. Otherwise, it will be resulted into high probability of packet dropping.

3) NIDS deployment: In Cloud, the main challenge is to monitor internal and external network traffic for securing each component (front end and back end) of Cloud. This is due to the virtualized and distributed nature of the Cloud. Therefore, NIDS sensors should be deployed in such a way that they can detect external attacks, internal attacks and distributed attacks in the whole Cloud network.

In addition, traditional NIDS challenges viz; detection rate, detection accuracy; false positives and false negatives [7] should be considered before integrating NIDS to Cloud.

B. Related Work

To solve above presented problem and challenges, there have been several works to-date detecting intrusions in the Cloud. S. Roschke *et al.* [8] have deployed IDS sensors on each VM. As shown in Fig. 2, sensors are managed by remote controller. Each sensor detects and reports the malicious behavior and transmits triggered event to event gatherer that collects malicious behavior and stores in the event database for further analysis. This approach prevents the VMs from being compromised. However, it requires multiple instances of IDS.

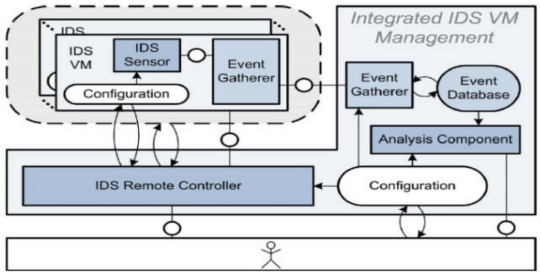

Fig. 2. VM compatible IDS architecture [8].

Mazzariello *et al.* [9] presented Snort based approach for Eucalyptus Cloud. Here, Snort is deployed at the Cloud controller as well as on host machines to detect intrusions. This approach solves the problem of deploying multiple instances of IDS, as in [8]. It is a fast and cost effective solution. However, it can detect only known attacks.

In [10][11], individual NIDS module is deployed in each Cloud region. In case of intrusion detection, it drops the attacker's packet, and then sends alert messages to other regions using agents. At each region, an alert produced by other regions are collected to determine whether it is true or false. This is done by calculating the severity of an alert. These approaches are suitable for preventing Cloud from DDoS attack. However, the computational and communication effort is increased.

Sandar *et al.* [12] proposed a solution framework that uses a firewall and puzzle server to prevent Economic Denial of Sustainability (EDoS) attack in the Cloud. A firewall is used to detect EDoS attack at entry point of Cloud, where as the puzzle server authenticates the user. In this work, EDoS attack is performed in the Amazon EC2 Cloud.

In [13], we have integrated a signature apriori based NIDS to Cloud. Signature apriori takes network packets and known attack signatures as an input and generates new derived rules that are updated in the Snort. Therefore, Snort is able to detect known attacks and derivative of known attacks in the Cloud. This approach improves the efficiency of Snort. However, it cannot detect unknown attacks. In [14][15], we have investigated performance of the Bayesian classifier and Decision tree classifier individually for detecting network anomaly. In these approaches, first we apply Snort to filter known attacks from the captured network traffic. Then, we apply a classifier to detect network anomaly. These approaches can efficiently detect known as well as unknown attacks.

In this paper, to improve detection accuracy, we combine three classifiers (Bayesian, Decision tree and Associative) rather than using them individually (as in [14][15]). Moreover, we improve efficiency of Snort (in our H-NIDS) by integrating signature apriori algorithm with it.

C. Theoretical Background

1) Hybrid-NIDS: We use a hybrid approach that combines two or more network intrusion detection techniques viz; signature based detection, anomaly detection, and soft computing techniques. The motivation for using the hybrid approach is to improve the accuracy of the intrusion detection system when compared to using individual approaches. In a survey [17], the authors compared various classifiers based anomaly detection techniques and concluded that no any individual classifier is performing well for detecting intrusions. Therefore, we combine different classifiers (in our H-NIDS) based on their performance. For signature based detection, we use Snort [18] and signature apriori algorithm [19].

2) Snort: It uses a signature based detection technique. It captures the network data packets and checks their contents with the predefined patterns for any correlation. The detection engine of Snort allows registering, alerting and responding to any known attack. It is an open source packet sniffer that is configurable, free, can run on multiple platforms (i.e. GNU/Linux, Window) and constantly updated. In inline mode of Snort, the functionality of Snort is extended, which provides active defense capability.

3) Signature Apriori: In signature apriori algorithm [19], known attack signatures and the captured network packets are given as an input. As an output, it generates new attack signatures that are frequent and satisfies the given threshold. These new signatures are used to detect derivative of known attacks in the Cloud. In our H-NIDS, such signatures are used to Snort to detect derivative of known attacks. This improves the efficiency of Snort.

For anomaly detection, we use different classifiers viz; Bayesian, Associative and Decision tree [20]. The aim of classifier is to predict the class label of the given unknown data based on previously observed data. In our proposal, different classifiers are used to detect network anomaly based on previously observed network packets.

4) Bayesian classifier: It is based on Bayes' theorem [20]. In our H-NIDS, it is used to predict the class label (normal or intrusion) of a given network event by calculating the probability of that event for each class. We use it as follows: let D is a training set of packets and their related class labels, and each packet is represented by a vector X = (x1, x2… xn). Suppose there are m different classes C1, C2… Cm. Bayesian classifier predicts X to class Ci, if the probability $P(Ci|X)$ is the highest among all the $P(Ck|X)$, for all the k classes. So, classification is to derive the maximal $P(Ci|X)$. It can be derived by (1). Since $P(X)$ is constant for all classes, we need to maximize (2).

$$P(Ci|X) = \frac{P(X|Ci)\,P(Ci)}{P(X)} \tag{1}$$

$$P(Ci|X) = P(X|Ci)P(Ci) \tag{2}$$

Thus, Bayesian classifier predicts class label (normal or intrusion) of the given network packet by observing previously stored network behavior.

5) Associative classifier: It explores highly confident associations among multiple features in given data. It consists of association rule mining (i.e. Apriori algorithm [20]) and classification technique. For our H-NIDS proposal, we use it as follows: Let *D* is a dataset of network packets that has n features (F1, F2,. . . , Fn) and one class feature *C* (c1, c2,..,cn). Each feature has a set of possible values. Each packet is in the form (f1, f2,. . . ,fm, c), where f1, f2,.., fm are the values of the corresponding packet features and c is the class label. Now, the task of an Associative classifier is to learn the confident association between a set of feature values to class and later use this relation to predict class labels for test packets from the values of their features only. We use an Apriori algorithm for mining rules. It generates rules in the form of *R*: *F*→*C*, where *F* represents features set of (F, value) pairs (f1^f2^f3^…^fn) and *C* is a class label. It generates the high number of rules. However, only few rules are interested. To solve interestingness problem, minimum support threshold (MST) and minimum confidence threshold (MCT) are applied to each rule. The confidence and support of rule *R*: *F*→*C* are derived by equation (3) and (4).

$$Confidence\ (R) = \frac{Support(R)}{Support(F)} \tag{3}$$

$$Support\ (R) = \frac{Number\ of\ packets\ having\ F}{Number\ of\ packets\ in\ D} \tag{4}$$

In our H-NIDS, associative classifier can detect DoS attacks efficiently since it uses the frequency of attack patterns to generate attack rules.

6) Decision tree classifier: It builds a decision tree, where each node represents a feature name, each leaf indicates a class label and each branch represents an outcome of the associated node. It can classify a large amount of data with faster learning speed than other classification techniques (i.e. ANN) [20]. In our H-NIDS, we use ID3 [20], a decision tree classifier as follows: Let a given dataset *D*={*F*, *C*}, where *F* is a set of features in *D* and *C* is the Class label. In step1, it checks that if all the data in *D* are in the same class, then it creates a node of that class and stops. Otherwise it selects a feature *F* from *D*, with values f1, f2,.. fn, and creates a decision node. In step2, it splits the training data *D* into subsets D1, D2,...,Dn according to the values of *F*. It applies step 1 and 2 recursively to each of the sets *Di*. As an output, it generates decision tree. After building tree, rules are generated by tracing each path from root to leaf. These rules are used to predict the class label of future data.

For selecting a feature to split in the decision tree, class entropy of dataset *D* and gain of each feature *F* is calculated by equation (5) (6) and (7). The feature having highest gain is selected for splitting.

$$Entropy\ (D) = -\sum_{i=1}^{m} Pi \times log2(Pi) \tag{5}$$

Where m is the different number of classes, *Pi* is the probability that a packet in *D* belongs to class *Ci*, calculated by |Ci, D|/|D|.

$$Entropy\_F(D) = -\sum_{j=1}^{f} \frac{|Dj|}{|D|} Entropy(Dj) \tag{6}$$

$$Gain(F) = Entropy(D) - Entropy\_F(D) \tag{7}$$

Equation (6) indicates that the information needed after using *F* to split *D* into *f* splits, whereas equation (7) represents information gained by branching on feature *F*.

III. H-NIDS in Cloud: Proposed Security Framework

A. Design Goals

Our H-NIDS has following goals:

1) Detection of network intrusions in the Cloud: Our H-NIDS detects network intrusions from external as well as internal network to secure each component (front end and back end) of Cloud. Moreover, it can detect distributed attacks in the whole Cloud environment.

2) Monitoring high traffic rate in the Cloud: Our H-NIDS ensures low computational cost and communication cost. Therefore, it is able to handle heavy traffic in the Cloud without dropping packets.

3) Securing each component in Cloud: We deploy our H-NIDS sensor on each host machine in a whole Cloud environment. So, our H-NIDS is capable of monitoring traffic from external network to VMs and VM to host. Such deployment strategy helps to secure host machine and VMs from network intrusions.

4) Higher detection rate and accuracy: Our H-NIDS is capable of detecting high number of intrusions and generates true alerts for detection.

5) Low false positive and negative: False positive can be defined as the number of false alerts of intrusions are generated by NIDS, while false negative can be defined as an inability of NIDS to detect the true intrusion. Our H-NIDS ensures very low false positives and negatives.

B. Architecture of Proposed H-NIDS

The architecture of the proposed H-NIDS is shown in Fig. 3. It consists of mainly seven modules viz; Packet capture, Signature based detection, Preprocessing, anomaly detection, Score function, Alert system and Central log.

1) Packet capture: From the host machine, it captures the in-bound and out-bound network traffic (packets) for auditing. To capture the network packets, packet sniffing tools (e.g Wireshrak) [21] can be used. These packets are inspected in real-time by signature based detection technique.

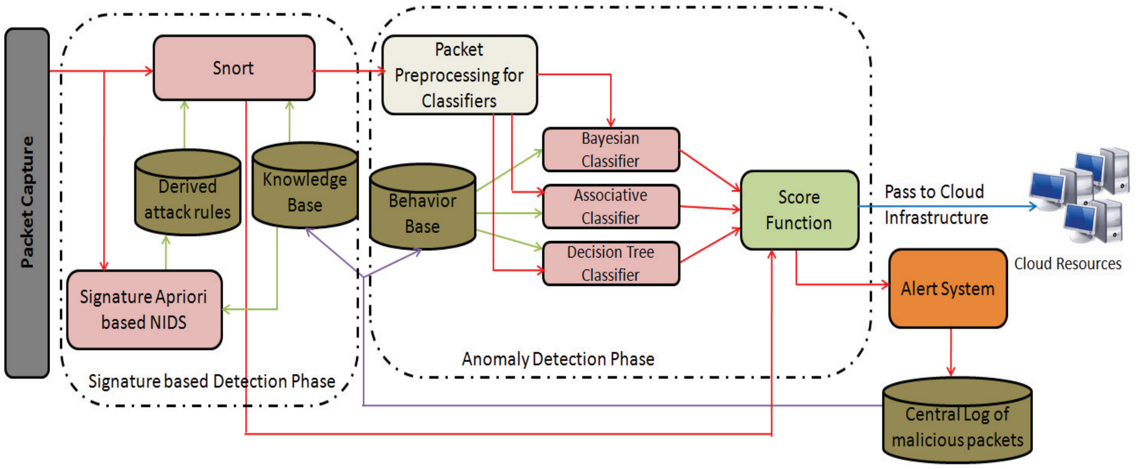

Fig. 3. Architecture of proposed H-NIDS.

2) Signature based detection: For signature based detection, we use knowledge base and derived attack rule base. The knowledge base is generated based on predefined network attack rules. In knowledge base, we store Cloud related known attack rules. To generate derived attack rules, we use signature apriori algorithm [19] that takes captured packets and known attack rules as an input and generates derived attack rules that are stored in the derived attack rule base. Now, Snort matches the captured packets with rules stored in the knowledge base and derived attack rule base to find any correlation. If it finds any intrusion, it determines the nature of the attack and sends alert message to score function. Other normal packets (specified by Snort) are applied to packet preprocessing module.

3) Packet preprocessing: From the given network packets, it removes redundant information (that has very less correlation with intrusion detection) from captured packets. The preprocessed packets are applied to classifiers for anomaly detection.

4) Anomaly detection: For anomaly detection, we use three classifiers viz; Bayesian, Associative and Decision Tree, as presented in Section II. These three classifiers are trained using previously observed network behavior (or packets) that are stored in behavior base. They individually predict the class label (intrusion or normal) of the given network packets and send the result to score function.

5) Score calculation: It is used to determine that whether the detected intrusion (predicted by any classifier) is actually intrusion or not, and to determine whether the detected intrusion is a type of distributed attack or not, by calculating its effect on whole Cloud. For each classifier, it sets score "1", if the given packet is predicted as "Intrusion" by classifier, otherwise it sets score "0". Then it uses weighted averaging method [22] to determine whether the predicted intrusion is really intrusion or not. The score of the packet is derived by equation (8).

$$Score\ (Packet) = \sum_{i=1}^{n} \frac{(Wi\ \times Oi)}{n} \tag{8}$$

Where, n is the number of classifier used in H-NIDS, *Wi* is the weight associated with the classifier. *Oi* is the outcome ("1" for Intrusion and "0" for Normal) predicted by the classifier. If the score of any packet satisfies defined threshold *T*, then that packet is considered as an Intrusion. Otherwise, it is considered as Normal and allowed to Cloud infrastructure. Moreover, it checks same intrusion alerts (determined either by Snort or score function) in central log where intrusion alerts from H-NIDS sensors at other regions are stored. Whether the given alert is of distributed attack or not, is determined by the majority vote method [10] and is derived by equation (9),

$$\frac{Number\ of\ sensors\ sends\ same\ alert\ to\ Central\ log}{Number\ of\ sensors\ in\ whole\ Cloud} \tag{9}$$

If this value ≥ *t* (where, *t* is the threshold defined by security administrator), that means if the number of H-NIDS sensors (deployed on whole Cloud) sending same alert of an intrusion to central log are greater than given threshold *t*, then the given packet is considered as an intrusion of type distributed attack and should be blocked in whole Cloud. Otherwise, it is considered as intrusion only for the relevant host and blocked for that host only. This method can secure whole Cloud from distributed attacks.

6) Alert system: It generates alerts about intrusion that is determined either by Snort or score function. It stores alerted intrusion in the central log database.

7) Central log: It is used for H-NIDS sensors deployed on other hosts. Other H-NIDS sensors update their bases (knowledge base and behavior base) with alerts logged in central log. So, the next time, such intrusion can be easily detected by using the signature based detection technique at other servers. This reduces computational cost. Moreover, it is used to determine distributed attacks, as discussed earlier.

In our H-NIDS, we use both the techniques (signature based and anomaly based) that are complimented of each other. This plays a major role to improve detection accuracy. Moreover, the signature based detection technique is applied prior to anomaly detection. As signature based technique detects known attacks and derived attacks first, and later anomaly detection technique detects fully unknown attacks, this reduces detection time. Central log and score function reduces detection time and helps to determine distributed attacks in Cloud. In [10][11], each NIDS sensor (deployed on each region) sends alert messages to each other to determine the severity of the detected intrusion. So, it requires n*(n-1) messages for exchanging an alert, where n is the number of NIDS sensors. While, in the case of our H-NIDS, each H-NIDS sensor sends an intrusion alert to the central log. Therefore, it requires only n messages to store an alert of distributed attack. Updating intrusion alert (as a rule) from the central log to the knowledge base and behavior base of other H-NIDS sensors helps to improve the detection rate and reduces the computational cost in the whole Cloud.

IV. Experimental evaluation of our H-NIDS

A. Experimental Setup

We have used Eucalyptus (an open source Cloud) [23] installed on the Ubuntu operating system. Cloud controller (CLC) and cluster controller (CC) are installed on the front end, whereas the node controller (NC) is installed at the back end. Our H-NIDS sensors are deployed on each NC. For testing purpose, we allow all types of traffic by opening all the ports in Eucalyptus. We use Scapy [24] for sending custom packets on the network. We install Wireshark (WS) [21] on the front end and back end of Cloud to monitor network traffic. MySql database (DB) is used to log intrusions.

For testing the feasibility of different classifiers in our H-NIDS, we have used KDD intrusion dataset [25] as training dataset for all three classifiers. As presented earlier, Cloud suffers from traditional network attacks and due to unavailability of other Cloud network related dataset, we have used this dataset to evaluate performance of the proposed H-NIDS. KDD (10%) contains 4,94,021 network records with 41 features and 24 different classes. Classification on 41 features of KDD dataset decreases detection accuracy and speed [26]. In [26], it is proved that only subset of the features from KDD dataset are relevant to each type of attack. If we use non relevant features in classification, they affect the overall detection accuracy. Therefore, we have used only 17 features (as shown in Table I) having better gain [20] to improve detection accuracy and classification speed. The gain of feature indicates the relevance of feature to detection. We have used 3,11,029 network records with all the classifiers for the predicting class label of them. In the preprocessing, the training data are converted into two distinct classes viz; Normal and Intrusion.

For Associative classifier, we have set 60% as the minimum support threshold and 60% minimum confidence threshold. We have performed a series of experiments on different thresholds and found results better on 60% threshold. We have used evaluation parameters [7][16] viz; Precision, True Positive Rate (TPR)/Recall, False Positive Rate (FPR), Detection Accuracy and F_Score to evaluate performance and quality of classifiers in our H-NIDS.

TABLE I. Gain of the selected features in KDD dataset

Feature No.	Feature Name	Gain
2	protocol_type	0.3024
3	Service	0.5709
4	Flag	0.0630
5	src_bytes	0.6460
6	dst_bytes	0.5383
7	Land	0.0005
8	wrong_fragment	0.0008
10	Hot	0.0029
11	num_failed_logins	0.0008
14	root_shell	0.0004
22	is_guest_login	0.0007
23	Count	0.6193
24	srv_count	0.3457
28	srv_rerror_rate	0.0023
30	diff_srv_rate	0.0831
36	dst_host_same_src_port_rate	0.3847
39	dst_host_srv_serror_rate	0.0801

We have done following experiments for our H-NIDS.

Exp. 1: Evaluation of only Bayesian classifier.

Exp. 2: Evaluation of only Associative classifier.

Exp. 3: Evaluation of only Decision tree classifier.

Exp. 4: Evaluation of H-NIDS with case 1. In case 1, we use all three classifiers with score function for evaluation, where the score function determines any record as an intrusion, if all the classifiers predict it as an intrusion.

Exp. 5: Evaluation of H-NIDS with case 2. In case 2, score function determines any record as an intrusion, if at least any two classifiers predict it as an intrusion.

Exp. 6: In case 3, we apply different values to the score function in our H-NIDS: $0.0 \leq W1, W2, W3 \leq 1.0$, n=3 and $0.0 \leq T \leq 1.0$, where W1, W2, and W3 are the weights associated with Bayesian, Associative and Decision tree classifier respectively. In this case, score function generates different outputs (Intrusion or Normal) for one record based on different weights and thresholds. Among these outputs, maximum time coming result is taken as the final result.

B. Results and discussion

The performance and quality results of our proposed H-NIDS are shown in Table II.

1) Precision: It indicates the percentage of intrusions that have occurred, and NIDS detects them correctly. As shown in Fig. 4, our H-NIDS has higher precision value.

2) True positive rate (Recall): It indicates the probability of the correctly detected intrusions to the total number of intrusions in the network. As shown in Fig. 5, our H-NIDS has very high true positive rate. It indicates that it is capable of detecting high number of intrusions.

TABLE II. PERFORMANCE RESULTS OF PROPOSED H-NIDS

Experiment number	Approaches used in our H-NIDS	Precision (%)	True positive rate (%)	False positive rate (%)	Accuracy (%)	F_Score
1	Use of only Bayesian classifier	99.64	96.09	1.04	96.82	0.97
2	Use of only Associative classifier	95.90	96.93	12.23	94.61	0.96
3	Use of only Decision tree classifier	99.32	96.25	1.91	96.71	0.98
4	Use of all three classifiers with Case 1	99.97	91.99	0.09	93.99	0.95
5	Use of all three classifiers with Case 2	99.64	97.00	1.04	97.50	0.98
6	Use of all three classifiers with Case 3	99.60	97.14	1.17	97.57	0.98

3) False positive rate: It represents that the probability of an intrusion alert, while there is no intrusion. It should be as minimum as possible. As shown in Fig. 6, our H-NIDS has very low false positives. Only associative classifier generates a high number of false positives, while individually used it. While for other cases, our H-NIDS has a false positive rate below 1.5% at a higher detection rate. It indicates that our H-NIDS is an efficient solution for detecting network attacks in the Cloud.

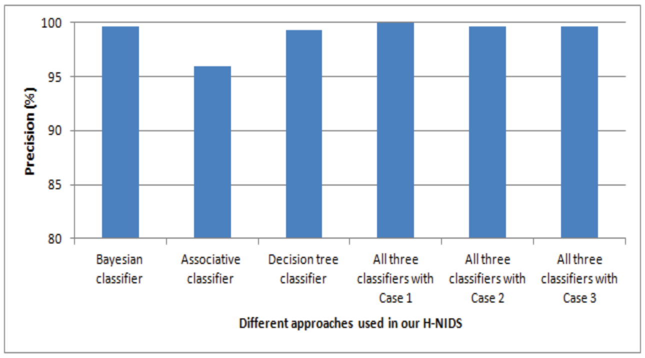

Fig. 4. Precision values for different approaches used in our H-NIDS.

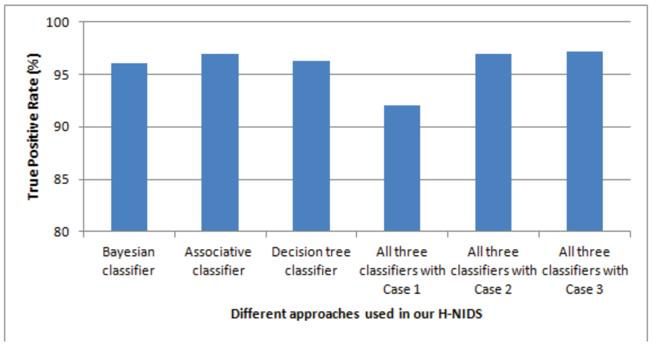

Fig. 5. True positive rates for different approaches used in our H-NIDS.

4) Accuracy: It indicates the percentage of predictions (that are done by NIDS) are true. Fig. 7 shows that our H-NIDS achieves higher accuracy (≅ 97%) in most of the applied cases. Our H-NIDS is accurate for detecting network anomaly.

5) F_score: It is considered as the harmonic mean of recall and precision. The higher value of F_score indicates the NIDS is performing better on recall (true positive rate) and precision. For an efficient NIDS, it should be maximum 1. As shown in Fig. 8, our H-NIDS achieves high F_score (> 0.95). This means, our H-NIDS is able to detect intrusions with high precision and recall.

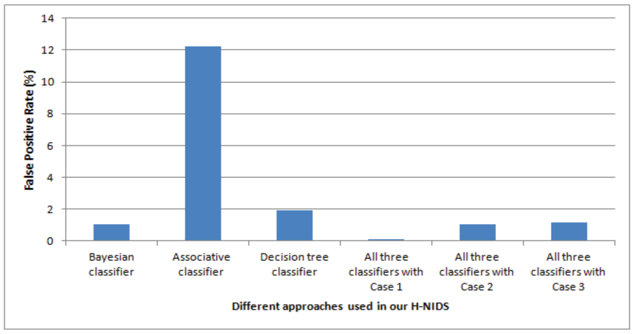

Fig. 6. False positive rates for different approaches used in our H-NIDS.

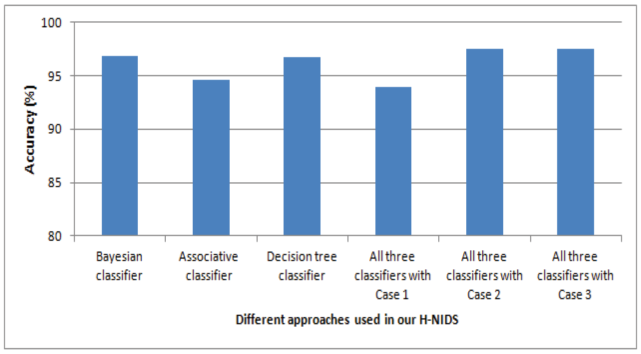

Fig. 7. Accuracy of different approaches used in our H-NIDS.

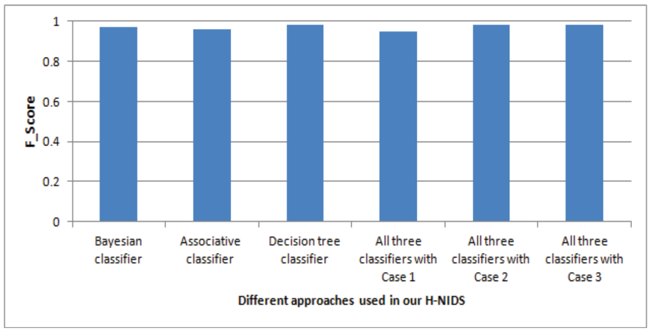

Fig. 8. F_score of different approaches used in our H-NIDS.

In general, proposed H-NIDS is capable of detecting higher number of intrusions with low false alerts and high accuracy at the network layer in Cloud, while satisfying network security needs. It has capability for detecting known and unknown attacks efficiently, while most of the other existing approaches [8][9][10][11][13] can detect only known attacks. Moreover, our H-NIDS has lower computational and communication overhead than agent based approaches [10][11][27]. Classifiers used in our H-NIDS have a lower complexity than other classifiers like artificial neural network (ANN), support vector machine (SVM) [20]. The combination of multiple classifiers in our H-NIDS improves detection accuracy.

V. CONCLUSIONS

In this paper, we proposed a security framework that integrates Hybrid-NIDS (H-NIDS) to Cloud. We deployed our H-NIDS on each host machine in Cloud. This has helped to detect internal and external network attacks. We have used both signature based and anomaly based techniques to improve detection accuracy. We have applied a signature based technique prior to anomaly technique, resulted in reducing the computational cost. In addition, the central log and the score function in our H-NIDS helps to detect distributed attacks in the Cloud. Experimental results for different possible cases show that the proposed H-NIDS has higher detection rate, higher accuracy, and low false alerts. This looks very promising and ensures the feasibility of this framework in Cloud.

REFERENCES

[1] P. Mell and T. Grance, "The nist definition of Cloud computing (draft)," 2011. [Online]. Available: http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_Cloud-definition.pdf

[2] F. Gens, "IT Cloud Services User Survey, pt.2: Top Benefits & Challenges," 2008. [Online]. Available: http://blogs.idc.com/ie/?p=210

[3] K. Popovic and Z. Hocenski, "Cloud computing security issues and challenges," In MIPRO, 2010 Proceedings of the 33rd International Convention, 2010, pp. 344–349.

[4] C. Modi, D. Patel, B. Borisaniya, A. Patel and M. Rajarajan, "A survey on security issues and solutions at different layers of Cloud computing," The Journal of Supercomputing, Springer US, pp. 1-32, 2012.

[5] L. Martin, "Awareness, Trust and Security to Shape Government Cloud Adoption," White Paper, 2010. [Online]. Available: http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/Cloud-Computing-White-Paper.pdf

[6] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel and M. Rajarajan, "A Survey of Intrusion Detection Techniques in Cloud," A Journal of Network and Computer Applications (JNCA), Elsevier, Vol. 36, No. 1, pp. 42-57, 2012.

[7] S. Sen, J. A. Clark and J. E. Tapiador, "Power-Aware Intrusion Detection on Mobile Ad Hoc Neworks," In Proc. of the first International Conference on Ad hoc Networks, 2009.

[8] S. Roschke, C. Feng, C. Meinel, "An Extensible and Virtualization Compatible IDS Management Architecture," Fifth International Conference on Information Assurance and Security, 2009, pp. 130-134.

[9] C. Mazzariello, R. Bifulco and R. Canonoco, "Integrating a network IDS into an Open source Cloud computing," Sixth International conference on Information Assurance and Security (IAS), 2010, pp. 265-270.

[10] C. C. Lo, C. C. Huang and J. Ku, "Cooperative Intrusion Detection System Framework for Cloud Computing Networks," First IEEE International Conference on Ubi-Media Computing, 2008, pp. 280-284.

[11] S. Ram, "Secure Cloud computing based on mutual intrusion detection system," International journal of computer application, Vol. 2, no. 1, pp. 57-67, 2012.

[12] S. V. Sandar, S. Shenai, "Economic Denial of Sustainability (EDoS) in Cloud Services using HTTP and XML based DDoS Attacks," International Journal of Computer Applications, Vol. 41, No. 20, 2012, pp. 11-16.

[13] C. N. Modi, D. R. Patel, A. Patel and M. Rajarajan, "Integrating Signature Apriori based Network Intrusion Detection System (NIDS) in Cloud Computing," 2nd International Conference on Communication, Computing and Security (ICCCS-2012), Vol. 6, No. 0, 2012, pp. 905 – 912.

[14] C. N. Modi, D. R. Patel, A. Patel, and R. Muttukrishnan, "Bayesian Classifier and Snort based Network Intrusion Detection System in Cloud Computing," International conference on Computing, Communication and Networking technologies (ICCCNT-Coimbatore), IEEE, 2012.

[15] C. Modi, D. Patel, B. Borisanya, A. Patel, and M. Rajarajan, "A novel framework for intrusion detection in Cloud," Proceedings of the Fifth International Conference on Security of Information and Networks (SIN-2012), 2012, pp. 67—74.

[16] Thomas, C., Balakrishnan, N.: Performance enhancement of Intrusion Detection Systems using advances in sensor fusion, 11th International Conference on Information Fusion, 2008, pp. 1-7.

[17] H. A. Nguyen and D. Choi, "Application of Data Mining to Network Intrusion Detection: Classifier Selection Model", in Challenges for Next Generation Network Operations and Service Management, vol. 5297, Springer-Verlag, LNCS, 2008, pp.399-408.

[18] Snort-Home page. [Online]. Available: https://www.snort.org/

[19] H. Zhengbing, L. Zhitang & W. Jumgi, "A Novel Intrusion Detection System (NIDS) Based on Signature Search of Data Mining," In WKDD First International Workshop on Knowledge discovery and Data Mining, 2008, pp. 10–16.

[20] J. Han and M. Kamber, "Data Mining Concepts and Techniques 2nd edition, Morgan Kaufmann Publishers, 2006.

[21] Wireshark. [Online]. Available: http://www.wireshark.org/

[22] A. P. F. Chan, W. W. Y. Ng, D. S. Yeung, E. C. C. Tsang , "Comparison Of Different Fusion Approaches For Network Intrusion Detection Using Ensemble Of RBFNN," Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, pp. 18-21, August 2005.

[23] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff and D. Zagorodnov, "Eucalyptus: A Technical Report on an Elastic Utility Computing Architecture Linking Your Programs to Useful Systems," UCSB Computer Science Technical Report Number 2008-10, 2008.

[24] Scapy. [Online]. Available: http://www.secdev.org/projects/scapy/

[25] KDD Cup 1999. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[26] S.S. Sathya, R.G. Ramani and K. Sivaselvi, "Discriminant Analysis based Feature Selection in KDD Intrusion Dataset," International Journal of Computer Applications, vol. 31, no. 11, pp. 1-7, 2011.

[27] A.V. Dastjerdi, K.A. Bakar, S.G.H. Tabatabaei, S.G.H.; , "Distributed Intrusion Detection in Clouds Using Mobile Agents," Advanced Engineering Computing and Applications in Sciences, 2009. ADVCOMP '09. Third International Conference on , vol., no., pp.175-180, 11-16 Oct. 2009