An Efficient Security Framework to Detect Intrusions at Virtual Network Layer of Cloud Computing

Kamatchi A
Computer Science and Engineering Department
National Institute of Technology Goa, India
kamatchinitg@gmail.com

Chirag N. Modi
Computer Science and Engineering Department
National Institute of Technology Goa, India
cnmodi@nitgoa.ac.in

***Abstract*—The major security concern in Cloud computing is to detect intrusions at virtual network layer. In this paper, we propose an efficient security framework to detect intrusions at the virtual network layer of Cloud. It combines signature and anomaly based techniques to detect both known and unknown attacks. It monitors multiple virtual machines at the host system of Cloud. It detects distributed attacks with the help of a correlation module at each Cloud cluster layer. In addition, a management module at the Cloud controller layer identifies collusion attacks in whole Cloud network. The accuracy of the intrusion detection is further improved using Dempster-Shafer theory (DST) for final decision making. We analyze the proposed security framework in terms of Cloud IDS requirements.**

Keywords—Cloud Computing, Virtual Network, Intrusion Detection System, Data Mining, Dempster-Shafer theory (DST)

I. INTRODUCTION

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1]. Cloud can be viewed as two ends viz; *front end and back end* (refer Fig. 1). Through *front end*, Cloud users can communicate with Cloud and access the offered services. *Back end* consists of the hardware and software resources which are designed for delivering the Cloud services. The host machines support multiple virtual machines (VMs) to run on it through hypervisor or virtual machine monitor (VMM). Such virtualization helps in sharing the physical hardware among many isolated machines, called as VMs. Here, VMs communicate with each other through the virtual network, which is created with the help of a virtual switch.

The problem with current virtualization technologies is that they have several vulnerabilities which allows an attacker to affect the security and privacy of Cloud resources and their services [2]. Hence, security and privacy are the major concerns in adopting the current virtualization technologies for Cloud computing [3]. Looking at the network layer of the Cloud, there are mainly two types of attackers viz; external or insider (refer Fig. 1). An external attacker (outside the Cloud network) often performs various attacks such as Denial of Service, port scan, probe, etc. to interrupt the Cloud services and resources. Distributed attacks such as large scale stealthy scans and distributed denial of service (DDoS) pose a serious threat to Cloud computing as they affect the Cloud resources on a large scale. Insiders may be Cloud user, client at the Cloud provider end or Cloud provider itself. Insider attacks are more serious problem since they have some privileges over the Cloud resources. They perform various attacks on other user's VMs in order to gain other's confidential information. They may affect the confidentiality, integrity and availability of the other user's resources as well as services over Cloud.

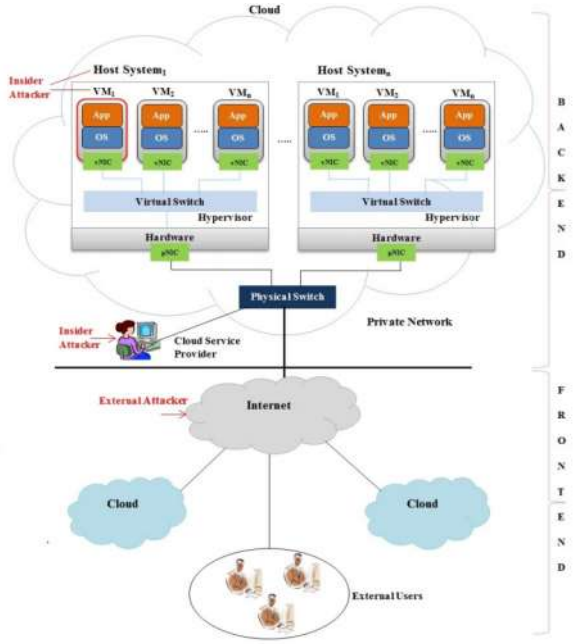

Fig. 1 Cloud computing architecture and threat model

Some of the reported incidents are as follows: In 2009, attackers performed distributed denial of service (DDoS) attack on the web based code hosting service *Bitbucket* and brought down the servers hosted on the Elastic Compute Cloud (EC2) for more than 19 hours [4]. In 2010, using "*Thunder Clap*" program, hackers carried out denial of service (DoS) attack on Amazon's EC2 with the investment of $6 on rented virtual machines [5]. In 2014, *Code Spaces* (a code hosting service) suffered from DDoS attack on their servers hosted on Amazon Cloud [6]. Here, the attacker deleted most of the data backups,

machine configurations and off-site backups which made the website completely down for 12 hours.

To detect such attacks, intrusion detection system (IDS) can be deployed on the Cloud. However, the shared, distributed and virtualized nature of Cloud brings additional challenges to traditional IDS frameworks. These challenges are discussed later.

In this paper, we design an efficient security framework to detect intrusions in virtual network of Cloud computing. It is a distributed framework with collaborative analysis of the intrusions. It combines both the signature and anomaly technique to improve its efficiency. In addition, it uses the Dempster-Shafer theory (DST) [7] for final decision making about the distributed intrusions. It can detect virtual network layer attacks with high accuracy, low false alarms, minimum communication and computation cost while fulfilling the Cloud IDS requirements for virtual network layer security.

The rest of the paper is organized as follows: Section II investigates the existing Cloud IDS approaches with research scope for further improvements. Section III presents the proposed security framework in detail. Section IV analyzes the proposed security framework in terms of fulfilling Cloud IDS requirements. Section V concludes our research work with future work and references at the end.

II. Intrusion Detection System in Cloud: Literature Survey

Intrusion is any unauthorized activity which affects the confidentiality, integrity and availability of the resources as well as services. Intrusion detection system (IDS) is a security system or a device which monitors the system or network for malicious activities. It uses mainly two techniques viz; signature based and anomaly detection. Signature based technique matches the monitored packets with the attack rules stored in the signature database for any correlation. If any match found, then it is considered as an attack. However, it detects only known attacks. Anomaly detection technique identifies the possibility of an attack by observing the network behavior. It helps in detecting unknown attacks. However, low accuracy and high false alerts should be investigated.

In Cloud, IDS can be deployed on each VM, each host machine and on the external network. IDS deployed on each VM monitors only the corresponding VM related attacks and protects the VM. However, multiple instances of VM-IDS are required. If VM is compromised and then underlying IDS can be compromised. It makes the VM-IDS management more complex. IDS can be deployed on each host machine in order to monitor multiple virtual machines (through single IDS) connected by the virtual switch at that host. However, it should be very fast to handle high network traffic from the underlying VMs; otherwise severe packet dropping at IDS may occur. IDS can be deployed at the external network of Cloud, which helps in detecting external attacks. However, it cannot detect internal attacks.

A. Cloud IDS Frameworks

There have been several works-to-date to detect intrusions in Cloud networks.

Dastjerdi *et al.* [8] have designed a distributed IDS using mobile agents (refer Fig. 2 [8]). IDS is deployed at each VM. Here, static agent (SA) monitors the VM for intrusions and reports it to the IDS control center (IDSCC). IDSCC generates mobile agent (MA) and sends it to corresponding VM to investigate further. MA analyzes the VM for intrusion activities and confirms the intrusion to IDSCC. Then IDSCC updates all other VMs except the affected VM about the intrusion. IDSCC correlates the alerts received from all IDSs to detect distributed attack. Agency in each IDS checks the health of SA and sends "heart beat" message (HB) to IDSCC periodically. If there is no message from any agency then IDSCC reduces the trust score of the corresponding VM. It is difficult to manage mobile agents and static agents in a distributed environment.

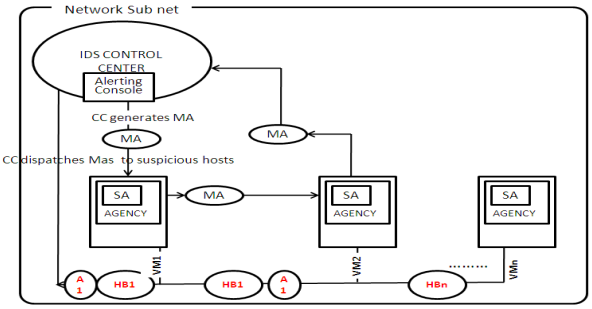

Fig.2 Mobile agent based Cloud IDS approach [8]

Lo *et al.* [9] have proposed a cooperative agent based IDS for Cloud. As shown in Fig. 3 [9], an individual IDS is deployed at each Cloud region. It analyzes the incoming network packets using *snort* [10]. Alert clustering and threshold check the severity of the detected intrusion. If it is serious, response and block module drops the malicious packets. Cooperative agents send/receive alerts from NIDS of other Cloud regions. Here, each sensor sends alert to the other NIDS sensor in the network. Hence, if there are *n* sensors detecting intrusion, then $n\times(n-1)$ are exchanged, and thus communication cost is high.

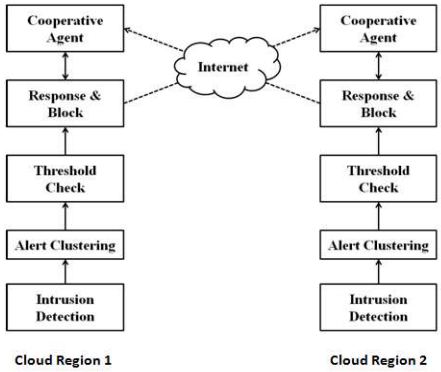

Fig. 3 Distributed Intrusion Detection Systems [9]

Dhage *et al.* [11] have presented a distributed IDPS architecture which combines signature and anomaly based techniques. Each Cloud service provider (CSP) has one IDS controller which stores the signature and anomaly status of all users in the network. To manage the workload, it creates an

instance of IDS whenever a Cloud user connects to the system. IDS instance monitors user's activities and send these data to IDS controller. IDS controller receives log data from various IDS instances and stores them in the database. When the user connects to the system next time, IDS instance gathers data about the user's previous usage from IDS controller and use it to detect intrusions.

Gupta *et al.* [12] have proposed a signature cum anomaly-based NIDS for Cloud. For each VM, profile database (DB) is created after monitoring the user's behavior over a period of time. The profile DB stores the attacks with ranks and it is easy for an early detection of critical intrusions. The DB is also updated with newly detected attacks and hence it is an adaptive. This model detects the attacks such as DoS, SYN flooding and TCP flooding. In this approach, there is no protection over NIDS. If NIDS is compromised then the security of all the VMs can be easily compromised.

Modi *et al.* [13] have used *snort* and anomaly based techniques such as Bayesian, associative and decision tree classifier. *Snort* detects known attacks, while classifiers detect network anomaly. Then the output from each classifier is applied to a weighted average method for score calculation in order to make a final decision on intrusion. Here, a majority voting is applied to the central log to detect distributed or correlated attack in the Cloud. It reduces computation cost and detection time, while improving detection accuracy. However, central log may be a single point of failure.

Shamsolmoali *et al.* [14] have proposed statistical-based Cloud confidence DDoS filtering (C2DF) for Cloud. It analyzes each network packet and extracts the Time-to-Live (TTL) value. It matches the TTL value with hop count, and if there is a mismatch, then the IP address is considered as spoofed IP and all the packets from that IP address are dropped. The rest of the packets are forwarded to the DDoS detection system. It compares the current header information with the profile database to determine information divergence between them. If it crosses the given threshold, packets from the corresponding IP address are dropped and the IP is added to the black list.

Cao *et al.* [15] have presented an entropy-based IDS to detect DoS attacks in the Cloud data center. Cloud infrastructure tier is monitored here. Cloud infrastructure management tier manages various functions such as system images, VMs and Cloud network management. Attack detection tier collects network, CPU, and I/O usage of each VM and stores it in the DB. It queries the VM status from DB and computes entropy based on the collected information to detect intrusions. It queries the attack detection module and applies policies based on the Cloud data center and notifies the administrator management tier. It can detect only DoS attacks.

Toumi *et al.* [16] have built a cooperative hybrid IDS in which mobile agents are used to identify intrusions from virtual environment. Here, signature database is updated after detecting new attacks at local IDS. Then signatures are updated to all other IDSs in the Cloud clusters to improve the detection accuracy in the overall network. It enables the system to detect the correlated or distributed attacks. However, the computational complexity is high and managing the mobile agents is also difficult. It detects only known attacks.

Singh *et al.* [17] have proposed a collaborative IDS framework for Cloud. In this model, NIDS is installed on the virtual bridge of each host machine to monitor the network packets from all VMs. They have deployed a correlation unit (CU) at one of the Cloud clusters based on the load on the clusters. Here, *snort* is used to detect known attacks and the combination of decision tree (DT) and support vector machine (SVM) is applied to detect network anomaly. Whenever attack frequency of an unknown attack crosses the threshold, new signature is created for that attack and updated in the local database. Here, the combination of DT and SVM helps in improving detection accuracy. However, if there is a high network traffic, a correlation unit may lead to a single point of failure.

B. Cloud IDS Challenges and Requirements: Research Scope

To design an efficient intrusion detection system for Cloud, the following challenges and requirements should be considered.

Dealing with large-scale computing system: Cloud is considered as a large scale interconnected systems. Hence, IDS should be able to handle large dynamic virtual network traffic and to manage itself with least or no human intervention in order to monitor and control systems in *real-time*.

Identify variety of attacks: Vulnerabilities in virtualization technology attract a number of attacks such as VM escape, external modification of host or VMs, zero day attacks and traditional attacks such as DoS/DDoS, Scanning, Spoofing etc., on Cloud resources. Hence there is a need for IDS to detect known as well as unknown attacks with minimal false alert rate for an efficient detection. It should be capable of self-improving the detection accuracy by learning the various attack patterns over a period of time. It should be capable of monitoring both physical as well as the virtual network. In addition, as Cloud has distributed environment, intrusion alerts should be correlated in order to identify distributed attacks.

Fast detection: Due to the technological growth in high speed networks, the number of network packets per second in Cloud is very high. Hence IDS should be capable of handling such huge network traffic without dropping the packets at the busy network in order to provide uninterrupted service to the users. It should be able to capture *real time* events and to detect malicious activities. The learning process of IDS should be incremental to reduce the re-training time.

Automated self-adaptive capability: Cloud has a dynamic nature as the resources to Cloud users are allocated on demand based on their requirements. Due to the dynamic nature of Cloud, signature database and behavior database should always be relevant to each VM user and updated frequently. Most of the existing approaches are unable to satisfy this requirement as they cannot handle dynamic load. Hence, IDS should be able to configure itself and be adaptive to these configuration changes automatically.

Scalability: As VMs are added and removed dynamically in Cloud, IDS should be scalable to monitor a large number of dynamic VMs and Cloud nodes.

Synchronization of IDS sensors: In distributed IDS, the alerts detected by various nodes are exchanged with each other or with a centralized server to detect correlated or distributed attacks. For an efficient and accurate detection, all the IDS sensors should be synchronized to the same clock automatically. If the intrusion detection operation is not synchronized, collusion attacks cannot be identified by IDS. Utilization of the network bandwidth for exchanging alerts between nodes should be minimized to reduce the data transfer cost.

Resistance to compromise: If an attacker is successful in compromising a system where IDS is residing, then his first motive would be disabling the functions of IDS. Hence IDS should be isolated from the monitoring machine to protect it from possible compromise or to prevent it from other unauthorized access. On successful IDS compromise, an attacker can disable it or prevent it from producing intrusion alerts.

In addition, traditional IDS requirements such as high accuracy, low false alerts, and low computational and communication cost are applicable to Cloud IDS.

Existing approaches such as [8][9][11] and [13]-[17] are not capable of monitoring both physical as well as the virtual network. Existing works [8][9][14]-[17] are unable to detect both known and unknown attacks in Cloud. Approaches like [8][11][17] need fast detection capability. Existing IDS such as [13]-[17] are not self-adaptive to dynamic changes in Cloud. Some approaches [12]-[17] are not capable of handling a large number of systems. Few distributed IDS approaches [11][13] [15][17] utilize high network bandwidth for data transfer and hence increases the data transfer cost. Proposals such as [8] [12]-[14][17] are not capable of protecting IDS from external and insider attacks.

III. Proposed Security Framework

A. Design Goals

The objective of the proposed security framework is to detect virtual network layer attacks in Cloud with high accuracy, low false alarms, minimum communication and computation cost while fulfilling Cloud IDS requirements. The derived design goals of the proposed security framework are as follows: handling large-scale dynamic systems, identifying a variety of attacks, fast detection, scalability, synchronization of IDS sensors and resistance to compromise (as discussed earlier).

B. Design of the Proposed Security Framework

A high level design of the proposed security framework is shown in Fig. 4. It consists of modules viz; *management module* (*MM*), *correlation module* (*CM*) and NIDS. It uses hierarchical based alert management structure. In a hierarchical structure, an entire system is divided into multiple groups based on the geography, software platforms and administrative control. The alerts from lower level are sent to a higher level of detection. The bottom most layer has only detection component and all other layers have both detection and correlation components. Modules of the proposed security framework are clubbed with the Cloud computing layers viz; *Cloud layer, Cloud cluster layer, Host layer and VM layer*.

Cloud Layer and Management Module: It is the top most layer of the architecture which consists of the *Cloud Cluster Controller* (*CCC*) which manages and controls the Cloud clusters. We deploy *management module* (*MM*) of the proposed security framework at *CCC*. *MM* receives intrusion alerts from Cloud clusters across the network and identifies distributed or collusion attacks in whole Cloud network by applying Dempster-Shafer Theory (DST) combination rule. Here, access control is used to allow only privileged clusters to send alerts to *CCC*.

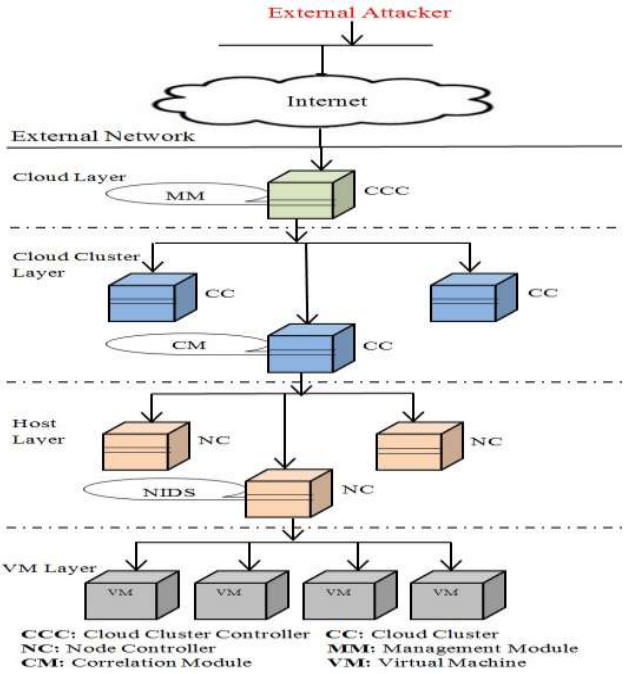

Fig. 4 Overall architecture of the proposed framework

Cloud Cluster Layer and Correlation Module: It consists of multiple *Cloud clusters* (*CC*) connected to the network. *CC* manages and controls all the underlying host systems. We deploy *correlation module* (*CM*) at *CC*. *CM* collects the attack evidences from the NIDSs which are deployed at underlying host systems. It applies DST combination rule on these attack evidences to identify distributed attacks at that cluster. *Cloud cluster layer* forms the next level of hierarchy, it has both detection and correlation units.

Host Layer and NIDS: It comprises multiple *node controllers* (*NC*) which hosts the multiple virtual machines (VMs). Here, NIDS monitors virtual network traffic (belonging to VMs at bottom layer) from virtual switch at each host machine. It sends the alerts to the *CM* of the corresponding cluster node. The advantages of this deployment are: multiple VMs can be monitored through single instance of an NIDS and virtual network can be secured from both insider and external network attacks.

VM Layer: It forms the core component of the Cloud network and consists of the VMs allocated to the Cloud users. Hence it forms the bottom most layer of the hierarchy for which intrusions are detected.

A detailed view of the proposed security framework in given in Fig. 5. For intrusion detection, we combine firewall, signature-based and anomaly-based detection.

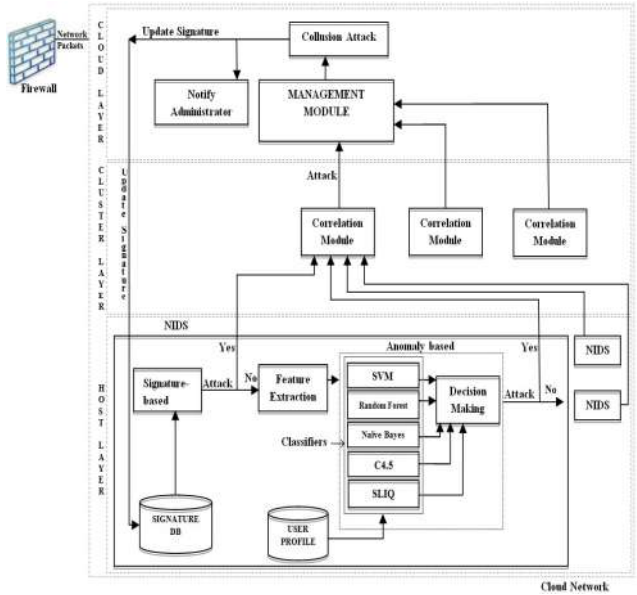

Fig. 5 Detailed view of the proposed framework

1) Firewall: It is a software or hardware system to prevent unauthorized access to/from a network based on predefined set of policy rules. It acts as a first line of defense in the network. It filters out the malicious IP addresses mentioned in the rules. Hence, it reduces the number of unauthorized request (packets) from external network to be inspected by the NIDS. Thus, the task of an NIDS is to detect only internal attacks.

2) NIDS: It is designed to be lightweight and installed in the virtual network (Virtual switch) at each host to monitor the network traffic from the underlying VMs. It combines signature and anomaly based detection (refer to Fig. 6).

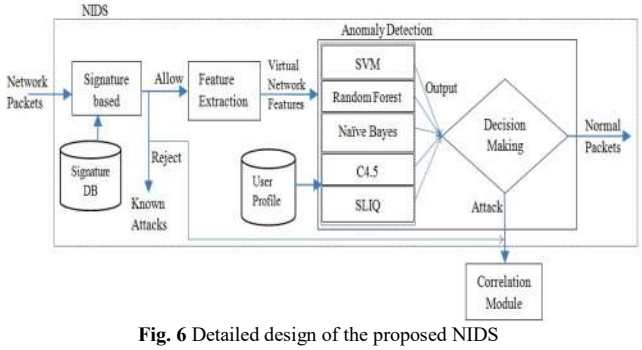

Fig. 6 Detailed design of the proposed NIDS

a) Signature-based detection: It defines a set of signatures or rules and uses them to detect if the given pattern leads to possible intrusion. Hence this technique detects all the known attacks present in the signature database. It passes the network packets classified as possible intrusions to the *correlation module* in the cluster node. It leads to high accuracy, minimal false positive rate and fast in detecting known attacks. The signature database has to be updated timely. For detecting known attacks, we are using *snort*, a well known signature-based IDS. It uses fast multi-pattern matching algorithm to detect attacks. It is easy to come up with new rules in *snort*.

b) Feature Extraction: This module takes the network packets (as input) which are not classified as intrusions by signature-based detection system. In addition to the traditional features (like protocol, port numbers, traffic related statistics), it extracts virtual network related features viz; virtual private IP address, virtual local area network identifier (VLAN ID), promiscuous mode set etc., and passes them to the *anomaly detection module*. Later number of features will be reduced based on correlation with intrusion, and thus it will help to reduce the training and testing time of the anomaly detection. Consideration of only relevant features in anomaly detection helps in improving the overall detection accuracy.

c) Anomaly detection: It collects the network behavioral data over a period of time. Statistical tests are then carried out on the observed behavior to determine whether that behavior is legitimate or not. It helps in detecting unknown attacks. Several techniques including data mining, statistical modeling and machine learning have been explored as anomaly detection in the literature. In our security framework, we combine multiple classifiers such as support vector machine (SVM), random forest, Naïve Bayes, C4.5 (decision tree) and supervised learning in quest (SLIQ). As per our observation, these techniques have low false alerts, better accuracy and low computation cost.

Support Vector Machine (SVM) [21]*:* It works based on structural risk minimization principle and statistical learning theory. The main aim of SVM is to construct a nonlinear mapping from the input space to high dimensional space. Then the classification is done by constructing a hyper plane which separates the training data by a maximal margin. It can classify intrusions with limited sample data and can handle a large number of features.

Random Forest [22]*:* It involves multiple classification trees for classifying the unknown data. For classification, unknown input vector is given to each of the trees in the forest. Each tree votes for a particular class for the given input. Then the forest chooses the class having maximum number of votes as the classification label. It can handle large dataset with many features and provides estimation of important variables in classification which makes it suitable for Cloud IDS.

Naïve Bayes [23]*:* It is a supervised learning method and based on the Bayes rule. This classifier has two assumptions: there is strong independence between the attributes in the dataset and the probability of each attribute is independent of the others. Despite these assumptions, it exhibits high speed and accuracy when applied to large datasets.

C4.5 [21]*:* It is a decision tree technique which produces the classification rules to categorize the unknown data into their corresponding classes. The information gain ratio is used as a criteria to choose splitting attributes as it avoids the possibility of choosing attributes with different values (increases both training time and space complexity). It reduces the computational complexity.

SLIQ [24]*:* It aims to generate a decision tree for prediction. It uses gini index as a split measure to decide the splitting attribute and its splitting point. At each split, the gini index value is minimized and hence as we progress in generating the tree, the tree becomes less diverse. For every successive pairs of attribute values, class histograms are constructed. The histogram with the least gini index value is considered to be the split point for that particular node. It has high accuracy.

Our proposed NIDS can detect intrusions without any interruptions in *real time* mode even when the training of the classifiers is carried out in *offline* mode. Then the decision making module takes the output from all these classifiers and applies the DST combination rule to decide the unknown attacks. The network packets classified as possible intrusions are passed to the *correlation module* in the cluster node.

d) Decision Module: It applies the DST combination rule to make the final decision on unknown attacks. The background and working of the DST combination rules is as follows:

Bayesian Inference Rule [18]*:* It has been commonly used method for correlating alerts from multiple sources and to make a decision. It is based on Bayes' theorem which states as follows:

$$P\left(\frac{H}{E}\right) = \frac{P\left(\frac{E}{H}\right)P(H)}{P(E)}$$

where, *P(H|E)* represents the posteriori probability which is a measure of belief about a hypothesis or proposition *H* in response to the evidence *E*. *P(H)* represents the prior probability in terms of belief and *P(E)* represents the probability of evidence. The prior probability represents the belief about *H* in the absence of evidence. The main drawback of this method is to come up with the prior and conditional probabilities.

Dempster-Shafer Theory [7]*:* It is considered to be an extended Bayesian inference. DST in alert fusion has solved the problem of analyzing the uncertainty in a quantitative way by representing them using belief functions. The following example clearly illustrates the difference between probability theory and DST. If there is an experiment which produces result as intrusion or normal with an unknown bias, probability theory will assign 0.5 for intrusion and 0.5 for normal by the principle of indifference. It states that all unknown states of probability should be given equal probability. On the other hand, DST assigns 0 to {intrusion} and {normal} and 1 to the set {intrusion, normal} meaning "either true or false". When there is no basis to assign probabilities, DST does not force to assign probability. In general, it allows to decide three solutions: Intrusion, Normal, or Unknown. The last option "Unknown" allows ignorance which makes a big difference in evidential reasoning.

Frame of discernment [19]*:* It is a set of all possible mutually exclusive and exhaustive states of a system which is similar to a state space in probability theory and is denoted by **Ω**. A hypothesis represents a subset of **Ω** for which evidence can be presented. The power set 2^{Ω} represents all the possible hypotheses. A basic probability assignment (bpa) is a mass function which assigns belief values to all the possible subsets in contrast to Bayesian theory, where the assignments are done only to single elements and not on the subsets. The value of bpa indicates the degree of supporting or refuting the evidence and is denoted as *m(A)*. The value of bpa ranges between 0 to 1 such that bpa for null set ∅ is $m(\emptyset) = 0$ and the sum $m(A_1) + \ldots + m(A_n) = 1$.

Belief Function [19]*:* The belief function is an assignment which maps each hypothesis *B* to a value *Bel(B)* which ranges from 0 and 1. It represents the weight of evidence supporting B's provability. It is defined as

$$Bel(B) = \sum_{j:A_j \subset B} m(A_j)$$

Plausibility Function [20]*:* A plausibility function is an assignment which maps each hypothesis *B* to a value between 0 and 1 and is defined as

$$Pl(B) = \sum_{j:A_j \cap B \neq \emptyset} m(A_j)$$

The belief and plausibility functions are related as follows

$$Pl(B) = 1 - Bel(\bar{B})$$

where $\bar{B}$ refers to "not B". Shafer proved that there is one-to-one correspondence exists between bpa, belief and plausibility functions.

DST Combination Rule [7]*:* Dempster's rule utilizes the orthogonal sum of basic probability assignments from independent observers. For example, if there are two independent observers having bpa as $m_1(A)$ and $m_2(A)$ then Dempster's combination rule is as follows:

$$m(B) = m_1(B) \oplus m_2(B) = \frac{\sum_{i,j:A_i \cap A_j = B} m_1(A_i) m_2(A_j)}{\sum_{i,j:A_i \cap A_j = \emptyset} m_1(A_i) m_2(A_j)}$$

This combination rule can be extended for more than two belief functions pairwise in any order. DST does not need priori knowledge, and a value associated with ignorance makes it suitable for anomaly detection of previously unseen intrusions.

3) Correlation Module (CM): CM at each cluster node is responsible for detecting distributed attacks at the corresponding cluster node level (refer to Fig. 7). It gathers intrusion evidences from all the NIDS residing at that cluster. It uses access control policies to allow only the alerts from NIDSs residing in the same cluster and restrict alerts from unauthorized NIDSs or from attackers. Decision making module in *CM* applies the DST combination rule on the evidences from multiple NIDSs and detects collusion attacks.

If there is a distributed attack detected by *CM* then the evidence of that intrusion is sent to the *management module* to take further action.

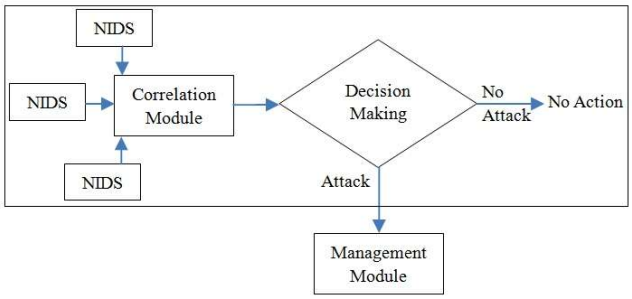

Fig. 7 Correlation module

4) Management Module (MM): It is the centralized system residing in the *Cloud layer*. It receives alerts from various cluster nodes across the Cloud network. These alerts are categorized as distributed attack with the help of DST's combination rule at *Cloud layer*. If there is an attack detected at *Cloud layer*, it is notified to the administrator and the rule for the newly identified attack is updated to the signature database of NIDSs residing in all the clusters (refer to Fig. 8). If the evidences are not identified to be distributed attack, *MM* concludes that the attack is confined to only those clusters reported the attack. Hence, *MM* updates the signature database of all NIDSs residing in only those clusters. The signature update helps in fast detection of the same attack in future.

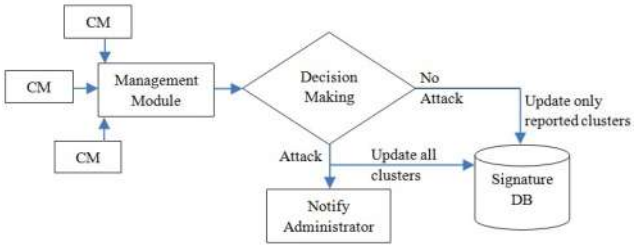

Fig. 8 Management module

IV. Analysis of the Proposed Security Framework

A. Analysis of the Derived Design Goals

In our proposed framework, NIDS is deployed at the virtual network of each host machine. It makes the NIDS to handle large-scale virtual machines at each host system end. It combines both signature and anomaly based techniques, and thus it is capable of detecting a variety of attacks (both known and unknown) at virtual network layer. In addition, it helps in detecting distributed attacks in whole Cloud environment. Known attack detection is fast due to only signature matching with the help of signature database. In case of unknown attacks, *correlation* and *management* modules use DST which provides fast intrusion detection.

Each NIDS is capable of handling all the VMs connected to particular virtual network within a host system. *Correlation* and *management* modules are capable of controlling the intrusion alerts from various NIDSs and Cloud cluster nodes respectively. Hence, it is scalable. NIDS sensors deployed at each host machine residing at the same network are synchronized to the same clock for efficient detection of distributed attack. Access control to NIDS helps in achieving resistance to compromise. In addition, the successful compromise of one NIDS does not affect the performance of other NIDSs in the network, and the compromised NIDS can be recovered later.

B. Analysis of the Proposed Design

As NIDS is deployed at the virtual network layer (virtual switch), it can monitor both internal virtual network traffic as well as the external physical traffic entering the host systems. Hence, it is capable of detecting the virtual network layer attacks. Signature-based detection provides high accuracy for known attack detection. For unknown attacks, it has three levels of detection systems such as anomaly detection through different classifiers, *correlation module* and *management module*. Thus, it improves the detection accuracy which leads to minimal false alerts.

Here, only distributed attacks are passed to higher layers. Thus, *correlation module* minimizes the number of intrusion alerts that are being sent to the *management module*. Hence, it reduces the overall computation and communication cost. As per analysis, it seems that the proposed security framework fulfills most of the Cloud IDS requirements for virtual network layer security.

C. Security Analysis of the Proposed Framework

The IDS should be resistant to compromise. Here, IDS is deployed at the virtual network of each host machine which minimizes the chance of attacking the IDS. In addition, access control is provided to access the NIDS. The proposed NIDS passes the evidences of the detected intrusion to the *correlation module* (*CM*). *CM* is responsible for detecting distributed and unknown attacks at cluster node level. If the *CM* is compromised then the attacker can send false decisions to all the NIDSs residing in the same cluster. It may avoid the legitimate users from getting the services provided by the service provider. To avoid this problem, the *CM* applies the access control to allow only authorized NIDSs to send and receive alerts.

Management module holds the responsibility to detect distributed attack at the Cloud network level. Successful compromise of a *management module* by any attacker may affect the performance of the intrusion detection. To overcome this issue, *MM* utilizes access control policies which restrict the unauthorized access of controlling the system. To ensure the authentic updates to the signature database at different levels, proper authentication mechanism is followed.

V. Conclusion and Future Work

The major security concern in the Cloud is to detect virtual network layer attacks. In this paper, we have proposed an efficient security framework to detect virtual network layer related attacks and distributed attacks. The combination of signature based and anomaly detection techniques helps in improving the efficiency and accuracy of the detection. Alert correlation at different layers of Cloud helps in detecting distributed attacks. It involves the less number of exchanges about alert evidences in order to detect distributed attacks and

thus communication and computation overhead is less. As per analysis, proposed security framework seems to be fit very well for Cloud network security, while satisfying Cloud IDS requirements. This analysis is very encouraging.

In future, we will validate the proposed security framework on the Cloud test-bed. We analyze the feasibility and comparative study of the proposed security framework for virtual network layer security requirements through *offline* as well as *real time* simulation of the Cloud specific attacks.

REFERENCES

[1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," Technical Report, NIST, 2011.

[2] C. Modi, P. Dhiren, B. Borisaniya, A. Patel, and M. Rajarajan, "A survey on security issues and solutions at different layers of Cloud computing," The Journal of Supercomputing, Vol. 63, No. 2, 2013, pp. 561-592.

[3] C. N. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in Cloud," Journal of Network and Computer Applications, Vol. 36, No. 1, 2012, pp. 42-57.

[4] "DDoS attack rains down on Amazon cloud", [Online]. Available: http://www.theregister.co.uk/2009/10/05/amazon_bitbucket_outage/

[5] "Thunder in the cloud: $6 cloud-based denial-of-service attack", [Online]. Available: http://www.computerworld.com/article/2468731/cloud-computing/thunder-in-the-cloud---6-cloud-based-denial-of-service-attack.html

[6] "Cyber Attack On 'Code Spaces' Puts Hosting Service Out of Business", [Online]. Available: http://thehackernews.com/2014/06/cyber-attack-on-code-spaces-puts.html

[7] D. Yu and D. Frincke, "Alert confidence fusion in intrusion detection systems with extended Dempster-Shafer theory," In Proceedings of the 43rd annual Southeast regional conference, Vol. 2, 2005, pp. 142-147.

[8] A. V. Dastjerdi, K. A. Bakar, and S. G. H. Tabatabaei, "Distributed intrusion detection in clouds using mobile agents," Third International Conference on Advanced Engineering Computing and Applications in Sciences, 2009, pp. 175-180.

[9] C. C. Lo, C. Huang, and J. Ku, "A Cooperative Intrusion Detection System Framework for Cloud Computing Networks," 39th International Conference on Parallel Processing Workshops, 2010, pp. 280-284.

[10] "Snort", [Online]. Available: https://www.snort.org/

[11] S. Dhage, B. Meshram, R. Rawat, S. Padawe, M. Paingaokar, and A. Misra, "Intrusion detection system in cloud computing environment," International Conference & Workshop on Emerging Trends in Technology, 2011, pp. 235-239.

[12] S. Gupta, P. Kumar, and A. Abraham, "A Profile Based Network Intrusion Detection and Prevention System for Securing Cloud Environment," International Journal of Distributed Sensor Networks, 2013.

[13] C. N. Modi and D. Patel, "A novel hybrid-network intrusion detection system (H-NIDS) in cloud computing," IEEE Symposium on Computational Intelligence in Cyber Security (CICS), 2013, pp. 23-30.

[14] P. Shamsolmoali and M. Zareapoor, "Statistical-based filtering system against DDOS attacks in cloud computing," International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2014, pp. 1234-1239.

[15] J. Cao, B. Yu, F. Dong, X. Zhu, and S. Xu, "Entropy based denial of service attack detection in cloud data center," Concurrency and Computation: Practice and Experience, 2015.

[16] H. Toumi, A. Talea, B. Marzak, A. Eddaoui, and M. Talea, "Cooperative Trust Framework for Cloud Computing Based on Mobile Agents," International Journal of Communication Networks and Information Security (IJCNIS), Vol. 7, No. 2, 2015.

[17] D. Singh, D. Patel, B. Bhavesh, and C. Modi, "Collaborative IDS Framework for Cloud," International Journal of Network Security, Vol. 18, No. 4, 2015, pp. 699-709.

[18] W. Hu, J. Li, and Q. Gao, "Intrusion detection engine based on Dempster-Shafer's theory of evidence," International Conference onCommunications, Circuits and Systems Proceedings, Vol. 3, 2006, pp. 1627-1631.

[19] A. M. Dermott, Q. Shi, and K. Kifayat, "Collaborative Intrusion Detection in Federated Cloud Environments," Journal of Computer Sciences and Applications, Vol. 3, No. 3A, 2015, pp. 10-20.

[20] T. M. Chen and V. Venkataramanan, "Dempster-shafer theory for intrusion detection in ad hoc networks," IEEE Internet Computing, Vol. 9, No. 6, 2005, pp. 35-41.

[21] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip, and Z. H. Zhou, "Top 10 algorithms in data mining," Knowledge and Information Systems, Vol. 14, No. 1, 2008, pp. 1-37.

[22] R. M. Elbasiony, E. A. Sallam, T. E. Eltobely, and M. M. Fahmy, "A hybrid network intrusion detection framework based on random forests and weighted k-means," Ain Shams Engineering Journal, Vol. 4, No. 4, 2013, pp. 753–762.

[23] X. Niuniu and L. Yuxun, "Review of decision trees," International Conference on Computer science and information technology (ICCSIT), 2010, pp. 105-109.

[24] M. Mehta and R. Agrawal, and J. Rissanen, "SLIQ: A fast scalable classifier for data mining," Advances in Database Technology—EDBT, 1996, pp. 18-32.