A Collaborative Intrusion Detection System for Cloud Computing

Ms. Riddhi Mistry[1] Mr. Krunal Kantharia[2] Mr. Sandip Chauhan[3]
[1]Computer Engineering [2]Network Engineer [3]Assistant Professor
[1]GTU, Ahmedabad [2]KPTIT, Viramgam [3]KITRC, Kalol, Gujarat

Abstract— Cloud computing is a computing paradigm that shifts drastically from traditional computing architecture. Although this new computing paradigm brings many advantages like utility computing model but the design in not flawless and hence suffers from not only many known computer vulnerabilities but also introduces unique information confidentiality, integrity and availability risks as well due its inherent design paradigm. To provide secure and reliable services in cloud computing environment is an important issue. To counter a variety of attacks, especially large-scale coordinated attacks, a framework of Collaborative Intrusion Detection System (IDS) is proposed. The proposed system could reduce the impact of these kinds of attacks through providing timely notifications about new intrusions to Cloud users' systems. To provide such ability, IDSs in the cloud computing regions both correlate alerts from multiple elementary detectors and exchange knowledge of interconnected Clouds with each other.

Keywords: Intrusion Detection System, Cloud Computing, Collaborative IDS, Collaborative IDS for Cloud

I. INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet)[1]. As per NIST definition, cloud model is composed of five essential characteristics as On-demand self-service, Broad network access, Resource pooling, Rapid elasticity, Measured service**,** three service models like Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and four deployment models like Private cloud, Community cloud, Public cloud, Hybrid cloud[2].

Intrusion Detection System [5] is software that automates the process of monitoring the events occurring in a computer system or network, analyzing them for signs of possible incidents, which are violations of security policies.
For, this research work we are mainly concentrating on Large-scale coordinated attacks, such as stealthy scans, worms and DDoS are powerful tools to assist attackers to achieve monetary gain. These attacks can occur in multiple network domains simultaneously, which makes prompt detection an extremely difficult task [6].

During the large-scale stealthy scans, there is one source (attacking host) that is responsible for numerous scans. Similarly, there is one source (infected host) that begins to connect to numerous hosts in order to spread itself during the worm outbreak. In contrast, the attack topology of a DDoS attack is many to one, namely, all the attack traffic is forwarded to one destination (the target system), although in a distributed reflector DDoS attack, part of the attack topology may appear as one to many [5]. Therefore, in order to detect the source address of a stealthy scan or worm outbreak, we need to correlate suspicious source addresses from incoming traffic across multiple network domains. Similarly, to detect and filter DDoS traffic we either need to correlate traffic at its source based on a common destination address, or correlate traffic at the reflectors based on a common source address. Moreover, given that the attack rate is high; this correlation of attack evidence must be done in a timely manner. The combination of complementary IDSs to build a Collaborative IDS (CIDS) is a promising technique that can be used to obtain a precise and comprehensive view of suspicious events.

A CIDS framework [4] is a mechanism to solve these issues as by correlating suspicious evidence and attack signatures from different sources of IDS. CIDSs provide the efficiency of detecting intrusions over a large-scale environment is improved. They have the potential to reduce computational costs by sharing ID resources between networks. The number of false alarms and irrelevant alerts that would be generated by individual IDSs can be reduced. The alarms raised by different IDSs produce more comprehensive information about intrusion attempts than that using a single IDS technique. The knowledge synthesis from distributed IDSs in all interconnect Cloud regions about intrusions, suspicious behaviors, blacklisted attackers or compromised VMs to enhance the efficiency and rate of intrusion detection.

II. LITERATURE SURVEY

A. Need for Cloud Log management

Now, when we have confronted various problems[7] and limitations[8] of cloud computing, have come to know that in each service model basic challenge occurred is Cloud log management. Cloud Log Management can manage any volume of data over any span of time. To Solve the cloud logging problems, a log management solution or architecture to support the following list of features[9] needs to be addressed: Centralization of all logs, Scalable log storage, Fast data access and retrieval, Support for any log format, Running data analysis jobs, Retention of log records, Archival of old logs and restoring on demand, Segregated data access through access control, Preservation of log integrity, Audit trail for access to logs.

A log management system[10] is the basis for enabling log analysis and solving the goals introduced in the previous sections. For Cloud Log Management, some Architecture and guidelines should be followed. Setting up a logging framework involves the following steps:

1) Enable logging in each infrastructure and application component
2) Setup and configure log transport

3) Tune logging configurations

Guidelines for log management provide information about when to log, what to log, how to log, etc.

Making the decision when to write log records needs to be driven by use-cases. These use-cases in cloud applications surface in four areas: Business relevant logging, Operations based logging, Security related logging, Regulatory and standards mandates

What to log defines which parameters must be logged for detailed log management. At a minimum, the following fields need to be present in every log record: Timestamp, Application, User, Session ID, Severity, Reason, and Categorization.

The following is a syntax recommendation that is based on standard work and the study of many existing logging standards which is Common event expression (CEE). This information gives answer to how to log.

Time = 2010-05-13 13:03:47.123231PDT, session_id = 08BaswoAAQgAADVDG3IAAAAD, severity = ERROR, user = pixlcloud_zrlram, object = customer, action = delete, status = failure, reason = does not exist

B. Eucalyptus Logs for Forensics

In [11], http based DoS/DDoS attack had been implemented where attacker has used VB script on host physical Win XP machine and Bash script on Linux machine to start multiple Firefox with multiple tabs to send stream of randomized http requests to Eucalyptus Cloud Controller node to exhaust its communication channel or bandwidth resources.

The virtual Eucalyptus Cloud Controller's (CC) Bandwidth usage under normal conditions shows that during normal conditions normal total traffic in/out rate is 39.1 Kbps. Similarly processor usage under normal condition is 5.6% After executing VB script and Bash script means after implementing http based DOS attack, Virtual Eucalyptus Cloud Controller's (CC) total bandwidth usage has risen to 6 Mbps during attack condition which is far above under normal attack conditions rate which was 39.1 Kbps. The bandwidth resources can be sharply consumed by opening more attack windows on client side and it reached as high as 16 Mbps when one more Firefox window with 50 tabs was opened in physical host WinXP client machine of virtual Eucalyptus cloud. Similarly Eucalyptus Cloud Controller's (CC) processor usage has risen to 83.1% under attack conditions compared to 5.6% under normal conditions.

Finally the relevant logs were identified in "/var/eucalyptus/jetty-request-05-09-xx" file on Cloud Controller (CC) machine which shows attacking machine IP, browser type and content requested. Now, here I have observed that logs can be maintained in a systematic manner by detailed procedure. But we need to differentiate them, correlate them and use them in a specific order to implement CIDS is the most important issue.

C. Architecture of Collaborative IDS Framework

In [4], each Cloud Provider (CP)'s infrastructure in a Collaborative Cloud Computing model is considered as a Cloud region while each physical machine residing at a Cloud region is called a node for convenience. This framework consists of three main components; namely, IDS Manager, which resides at the management region of a Collaborative Cloud, IDS Dispatcher, which is built inside each Cloud region, and Elementary Detector, which is distributed to monitor each VM and generates alarms for a detected anomaly. For communication among components, messages containing data and necessary information are created and encrypted at each component before being exchanged. Messages use TCP as the data transport. Two kinds of Database servers, Global and Local, reside at Management region and each element region, respectively.

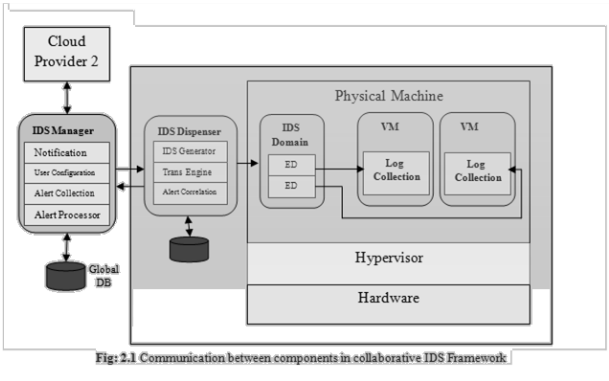

Fig (1): Communication between components in collaborative IDS Framework

1) Elementary Detector

Elementary Detector (ED) is a specialized IDS distributed to monitor each VM in the system. Based on the service models that Cloud users chose from the initialization phase, the default functions are assigned to each EDs to collect and analyze data about network traffic, memory, file systems, logs, etc. to find potential intrusions in the monitored hosts. Alerts generated by EDs are called raw alerts and sent to IDS Dispatcher to alert aggregation and correlation from other EDs for reducing the number of false raw alerts and generating higher level alerts about large-scale coordinated or multi-step attacks.

2) IDS Dispatcher

IDS Dispatcher is built in a secured independent node at each Cloud region and responsible for either generating distributed EDs or processing raw alerts which are sent from all EDs. It's considered as the parent node in the hierarchical CIDS model for aggregating and correlating all raw alerts from EDs into hyper alerts and analyzing them to detect large-scale coordinated attacks. IDS Dispatcher consists of three modules: IDS Generator, Translation Engine and Alert Correlation.

a) IDS Generator

IDS Generator (IDSGen) is responsible for generating and configuring EDs to monitor each virtual host. First, IDSGen receives user's information that is specified by Cloud users from IDS Manager. In addition, a blacklist from IDS Manager, which consists of information about suspicious attackers, intrusions detected by IDSs of other Cloud regions, is also used to update IDS configurations.

b) Translation Engine

After receiving raw alerts which are generated by all EDs located in all nodes of a cloud region or hyper alerts from Alert Correlation module, Translation Engine (TransEng)

takes charge of storing them in Local Database. Due to the diversification of alerts which are built in different formats, TransEng translates received alerts into a common format, IDMEF [13], before extracting necessary data and storing them into Local Database.

c) Alert Correlation

Alert Correlation is used to correlate alerts based on logical relationships among the alerts. This function will provide the system security operator with great insight into where the initial attacks come from and where they actually end up. It can also be used to find patterns among series of attacks. After the alert correlation, high-level alerts providing an overall view of the attacks will be presented to the system security operators and Cloud users. In addition, correlating raw alerts from different function of IDSs also helps to verify whether a certain attack is successful or failed to have appropriate responses.

Three key steps to correlate raw alerts into hyper alerts are alert aggregation, alert verification and alert correlation. **Alert aggregating** is the grouping of alerts that both are close in time and have similar features. **Alert verification** is to take a single alert and determine the success of the attack that corresponds to this alert. Finally, A**lert correlation** discovers the relationships between individual alerts raised by security incident detection systems and other security systems. In particular, when a new raw alert is stored into the Local Database by TransEng, Alert Correlation is simultaneously notified.

3) IDS Manager

IDS Manager is considered as the central management component of the CIDS framework and an intermediate to exchange information between Cloud's users and EDs. It also takes charge of gathering all events related to intrusions and sending notifications to users via one single interface for all cloud regions. There are four modules in IDS Manager; namely, User Configuration, Notification, Alert Collector and Alert Processor.

a) User Configuration

User Configuration (UserCfg) is built to collect users' IDS configurations and transfer them to other related components in our proposed CIDS framework. Through a single web-based user interface, Cloud users can specify monitoring functions, alert settings and thresholds which are considered as parameters for building and configuring their EDs. This interface is only sent to Cloud users after they are verified as legitimate users of Cloud. Based on the list of Cloud regions and VM locations, UserCfg relays messages containing users' IDS configurations to IDS Generator module of corresponding IDS Dispatchers.

b) Notification

Notification directly interacts with Cloud users to notify detected intrusions which affect their own resources allocated from CPs. It does queries to Global Database to get new alerts which are stored into Database by Alert Collector and alarms to Cloud users. In this context, Global Database stores information about intrusions to resources of all users of all Cloud regions belonging to the Collaborative Cloud environment.

c) Alert Collector

Alert Collector takes charge of receiving hyper alerts from Cloud regions and updating them to Global Database for being processed by Alert Processor later. Because of the Internet-based nature of Cloud Computing, handling services and allocated resources of Cloud users is processed through a request–response model like an ordinary web client–server. Therefore, Cloud users' requests to Access Control component also needs to be monitored by the Cloud IDS framework as a source for anomaly detection. As usual, Access Control will assess the validation of all requests from Cloud users before determining to reject these requests or forward them to appropriate processing components in Cloud systems.

d) Alert Processor

Alert Processor is the module which processes hyper alerts at the highest level in the hierarchical Collaborative IDS framework. The main goal of Alert Processor is to analyze lower-level alerts stored in the Global Database, extract information and generate a blacklist of compromised VMs, identification of suspicious attackers, and details of recognized attacks.

4) Collaborative IDS Framework Workload

Step 1.1: After being authenticated and choosing appropriate services, users send IDS configurations to proposed IDS framework for generating new IDSs for their allocated systems.

Step 1.2: These users' IDS configurations are stored in Global Database before having been transferred to IDS Dispatcher in each Cloud region.

Step 1.3: At each Cloud region, all configurations are used to generate and configure new IDSs which are used to monitor user's virtual hosts.

Step 2.1: As an intrusion is detected by EDs, a raw alert is generated and sent to IDS Dispatcher node on the same Cloud region.

Step 2.2: This alert is converted into the common format (IDMEF) and stored in the Local Database.

Step 2.3: This alert is aggregated with other alerts to create hyper alerts.

Step 2.4: Hyper alerts are forwarded to IDS Manager on the Management region of this federated Cloud.

Step 2.5: After receiving new alerts, through a user interface, IDS framework notifies users about a threat to their system and requires a response.

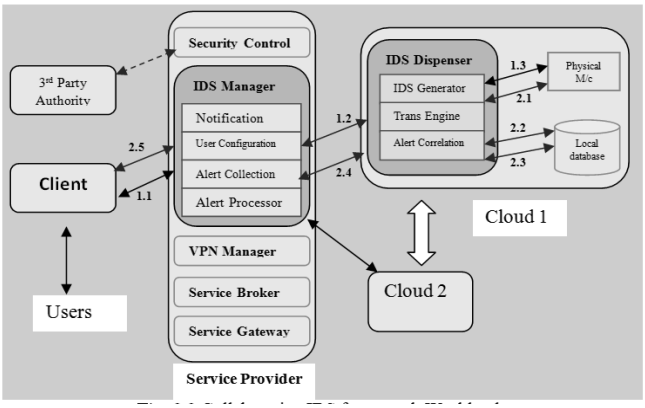

Fig (2): Collaborative IDS framework Workload

III. PROPOSED SYSTEM

Step: 1 Initially Logs from the all nodes will be generated.
Step: 2 These Logs will be sent to ED and ED will generate raw alert.
Step: 3 Raw alert will be sent to IDS Dispatcher where IDS generator will checks user information that user has got ED or Not. If new user has arrived, IDS Gen will verify user and then allocate ED to new user.
Step: 4 Raw alert will be then sent to alert correlation, where alerts will be co-related by queries from the Local DB. Here, it will check alert type whether alert is successful or failed.
Step: 5 Successful alerts which are known as hyper alert will be then sent to TransEng where alerts will be converted in IDMEF format and stored in local DB.
Step: 6 Hyper alert will be sent to IDS Manager, where it user configuration will be determined, alerts are collected, processed for further use and stored in Global DB.
Step: 7 IDS Manager will notify Cloud Controller as well as user for processed alerts.

Here in [4], Alert correlation algorithm is provided as shown below, but we modify it by adding parameters such as bandwidth usage, memory usage, CPU usage as we have seen earlier in [11] that by using certain programming, we can come to know that attack has happened in particular system.

Correlation Algorithm

```
A: list of raw alerts
r : Correlation threshold
s : Correlation sensitivity
for all each alert ai in A
        for all hyper alerts in H
                find an hyper alert hj containing an alert aj such that
                the correlation probability of ai and aj is maximum
m ← this maximum correlation probability
if m> r
        then for each alert ak in hj
                if m - (probability between ak and ai) < s
                        then connect ai with ak
                else
                        create a new hyper-alert
                        put ai in new hyper-alert
                initialize hyper alert list H
```

IV. CONCLUSION

Thus, we have seen that logs can be maintained at the single machine i.e Cloud Controller's machine. These Logs are randomly generated by any event occurred on the cloud infrastructure. Here, I have implemented scripts which can fetch this log files, collect those logs of the cloud infrastructure, analyze them, correlate them and provide alert in form of notification to the Cloud controller and the client if any unknown or unfaithful event occurs.

Thus, Collaborative IDS framework for a Collaborative Cloud Computing model builds multiple anomaly-based elementary detectors and a hierarchical architecture for combination of their alerts to make more accurate determination of intrusions. Here blacklist users have been tackled in a way that is considered as either notification to each interconnected Cloud or additional knowledge to increase efficiency of intrusion detection progress. So, I have implemented Collaborative IDS by using Collaborative IDS framework and got result from the same.

This framework needs to be concentrated on the security area as well as time duration taken. Here, analysis of some basic attacks mainly DDOS attack has been experienced. So, evaluation of the system by experimenting different attacks remains yet.

REFERENCES

[1] http://en.wikipedia.org/wiki/Cloud_computing
[2] Peter Mell, Timothy Grance, "The NIST Definition of Cloud Computing", Special Publication 800-145, National Institute of Standards and Technology, U.S. Department of Commerce.
[3] Furht B, Chapter 1, Handbook of cloud computing.
[4] N. D. Man (&) _ E.-N., "A Collaborative Intrusion Detection System Framework for Cloud Computing" in Proceedings of the International Conference on IT Convergence and Security 2011.
[5] Guide to Intrusion Detection and Prevention Systems (IDPS), Recommendations of the National Institute of Standards and Technology, NIST Special Publication 800-94, February 2007.
[6] Chenfeng Vincent Zhou, Shanika Karunasekera, and Christopher Leckie, "A survey of coordinated attacks and collaborative intrusion detection", Computer Science and Security, Volume 29, Issue 1, February 2010, Pages 124–140, available at www.sciencedirect.com.
[7] Scott Zimmerman and Dominick Glavach,, "Cyber Forensics in the Cloud", IAnewsletter. Vol.14-No 1, 2011.
[8] Mohsen Damshenas, Ali Dehghantanha, Ramlan Mahmoud, Solahuddin bin Shamsuddin, "Forensics Investigation Challenges in Cloud Computing Environments", in Cyber Security, Cyber Warfare and Digital Forensic (CyberSec), 2012 International Conference on 26-28 June 2012.
[9] Stephen Mason, Esther George, "Digital evidence and 'cloud' computing", computer law & security review 27(2011) 524 – 528, available at www.sciencedirect.com
[10] Raffael Marty, "Cloud Application Logging for Forensics", Loggly Inc.
[11] Zafarullah, Faiza Anwar, Zahid Anwar, "Digital Forensics for Eucalyptus", In IEEE, 2011 Frontiers of Information Technology.
[12] Tal Garfinkel Mendel Rosenblum, "A Virtual Machine Introspection Based Architecture for Intrusion Detection" Proceedings 10th symposium, Network and Distributed System Security (NDSS 03), Internet Society, pp 191–206
[13] Intrusion detection message exchange format available at http://www.ietf.org/rfc/rfc4765.txt

