Protecting Host-Based Intrusion Detectors Through Virtual Machines

D. Satish
Department of Computer Science & Engineering, GIET, JNTU Kakinada (A.P), India.

Dr. B. Sujatha
Professor, Department of Computer Science & Engineering, GIET, JNTU Kakinada (A.P), India.

Abstract – Virtually every industry and even some parts of the public sector are taking on cloud computing today, either as a provider or as a consumer. Despite being young it has not been kept untouched by hackers, criminals and other "bad guys" to break into the web servers. Once weakened these web servers can serve as a launching point for conducting further attacks against users in the cloud. One such attack is the DoS or its version DDoS attack. Particularly, attackers can explore vulnerabilities of a cloud system and compromise virtual machines to deploy further large-scale Distributed Denial-of-Service (DDoS). DDoS attacks usually involve early stage actions such as multi-step exploitation, low frequency vulnerability scanning, and compromising identified vulnerable virtual machines as zombies, and finally DDoS attacks through the compromised zombies. Within the cloud system, especially the Infrastructure-as-a-Service (IaaS) clouds, the detection of zombie exploration attacks is extremely difficult. To prevent vulnerable virtual machines from being compromised in the cloud, we propose a multi-phase distributed vulnerability detection, measurement, countermeasures. The system and security evaluations demonstrate the efficiency and effectiveness of the proposed solution.

Index Terms – Intrusion Detection, Network Security, Cloud Computing, Attack Graph, Zombie Detection.

1. INTRODUCTION

A recent Cloud Security Alliance (CSA) survey shows that among all security problems, abuse and nefarious use of cloud computing is considered as the top security threat, in which attackers can make use of vulnerabilities in clouds and utilize cloud system resources to make attacks. In traditional data centers, where system administrators have full control over the host machines, vulnerabilities can be detected and fixed by the system administrator in a centralized manner. However, fixing known security holes in cloud data centers, where cloud users usually have the rights to control software installed on their managed VMs, may not work efficiently and can violate the Service Level Agreement (SLA). Furthermore, cloud users can install vulnerable software on their VMs, which basically contributes to loopholes in cloud security. The challenge is to set up an effective vulnerability/attack detection and response system for accurately identifying attacks and minimizing the impact of security breach to cloud users. In a cloud environment where the infrastructure is shared by potentially millions of users, abuse and illegal use of the shared infrastructure benefits attackers to make use of vulnerabilities of the cloud and use its resource to deploy attacks in more efficient ways. Such attacks are very effective in the cloud environment since cloud users usually share computing resources, e.g. being interconnected through the same switch, file systems and sharing with the same data storage, even with potential attackers. The similar setup for Virtual Machines in the cloud, e.g. VM OS, virtualization techniques, installed vulnerable software etc. attracts attackers to compromise multiple VMs.

2. LITERATURE SURVEY

H. Takabi, J.B. Joshi, and G. Ahn explained the security and privacy challenges in cloud computing environment [1]. Cloud computing has generated significant interest in both the academy and industry, but it is yet an evolving paradigm. Essentially, it aims to combine the economic utility model with the evolutionary development of lots of existing approaches and computing technologies, including services, distributed applications and information infrastructures consisting of pools of networks, computers and storage resources. Many doubt exists in IT communities about how a cloud differs from existing models and how these differences affect its adoption. Some see a cloud as a new technical revolution, while others consider it a natural evolution of technology, economy and culture.

1) However, cloud computing is an important paradigm, with the potential to significantly reduce costs through optimization and increased operating and economic efficiencies.

2) Furthermore, cloud computing could significantly enhance collaboration, agility, and scale, thus allowing a truly global computing model over the Internet infrastructure. However, without proper security and privacy solutions designed for clouds, this computing paradigm could become a big failure, much number of surveys of potential cloud adopters indicates that security and privacy is the primary concern constraining its adoption.

3) This article explains the issues of cloud computing that explains security and privacy challenges in clouds.

4) This article discuss various approaches to address these challenges and explore the future work needed to provide a trustworthy cloud computing environment.

In 2012 B. Joshi, A. Vijayan, and B. Joshi proposed a method for Securing cloud computing environment against DDoS attacks. In this paper they explained Cloud Computing is the newly emerged technology of Distributed Computing System. Cloud Computing user concentrate on API security & provide services to its consumers in virtual environment into three layers namely, Software as a service, Infrastructure as a service and Platform as a service with the help of web services .It provides service facilities to its consumers on demand. These service provided can simply invites attacker to attack by Saas, Iaas and Paas. Since the resources are gathered at one place in data centers in cloud computing, the DDOS attacks such as HTTP & XML in this environment is dangerous & provides harmful effects and also all consumers will be affected at the same time. These attacks can be resolved & detected by a proposed methodology. In this methodology, this problem can be overcome by using proposed system. The different types of vulnerabilities are detected in proposed system. The SOAP request form the communication between the client and the service provider. Through the Service Oriented Trace back Architecture the SOAP request is send to the cloud. In this architecture service oriented trace back mark is present which contain proxy within it the proxy that marks the incoming packets with source message identification to identify the real client. Then the SOAP message is travelled via XDetector. The XDetectors used to monitors and filters the DDoS attacks such as HTTP and XML DDoS attack and finally the filtered real client message is transferred to the cloud service provider and the corresponding services are given to the client in secured method. The paper [3] focuses on the detection of the compromised machines in a network that are used for sending spam messages, which are commonly known as spam zombies. Given that spamming provides a critical economic incentive for the controllers of the compromised machines to recruit these machines, it has been seen that many compromised machines are involved in spamming. A number of latest research efforts have studied the aggregate global characteristics of spamming botnets (networks of compromised machines involved in spamming) such as the size of bonnets and the spamming patterns of botnets, based on the sampled spam messages received at a large email service provider.

G. Gu, P. Porras, V. Yegneswaran , M. Fong, and W. Lee proposed a method for Detecting Malware Infection through IDS-driven Dialog Correlation**.** The paper [4] focus on a new kind of network perimeter monitoring strategy, which concentrates on recognizing the infection and coordination dialog that occurs during a successful malware infection. BotHunter is application designed to track the two-way communication flows among internal assets and external entities, developing an evidence trail of data interchange that match a state-based infection sequence model. BotHunter made up of a correlation engine that is driven by three malware-focused network packet sensors, each charged with detecting stages of the malware infection process, including exploit usage, inbound scanning, egg downloading, outbound bot coordination dialog and outbound attack propagation. The BotHunter correlate then bind together the dialog trail of inbound intrusion alarms with those outbound communication patterns that are highly indicative of successful local host infection. When a sequence of evidence is found to match BotHunter's infection dialog model, the overall report is produced to capture all the relevant events and event sources that played a role during the infection process. We refer to this strategy of matching the dialog flows between internal assets and the broader Internet as dialog-based correlation, and against this strategy to other intrusion detection and alert correlation methods. They presented experimental results using BotHunter in both virtual and live testing environments. BotHunter is made available both for operational use and to help stimulate research in understanding the life cycle of malware infections. In 2012 Z. Duan, P. Chen, F. Sanchez, Y. Dong, M. Stephenson, and J.Barker proposed a method for Detection of Spam Zombies by Monitoring Outgoing Messages [5] In this paper we have seen Compromised machines are one of the key security threats on the Internet; they are often used to launch various security attacks such as spamming and spreading malware, identity theft and DDoS. Given that spamming allows a key economic incentive for attackers to recruit the large number of compromised machines, we are concentrating on the detection of the compromised machines in a network that are involved in the spamming activities, generally known as spam zombies. An effective spam zombie detection system named SPOT is developed by monitoring outgoing messages of a network. SPOT is developed based on a powerful statistical tool named Sequential Probability Ratio Test, which has bounded false negative error and false positive rates. The evaluation study based on email trace collected in a large campus network show that SPOT is an effective and efficient system in automatically detecting compromised machines in a network. In addition, the performance of SPOT is compared with two other spam zombie detection algorithms based on the number and percentage of spam messages originated or forwarded by internal machines, and show that SPOT performs good than these two detection algorithms.

In 2005, X. Ou, S. Govindavajhala, and A.W. Appel proposed a logic based security analyser [6] .To determine the security impact software vulnerabilities have on a particular network, one must consider interactions between multiple network elements. For a vulnerability analysis an application is useful in practice, two features are important. First, the model used in the analysis must be able to automatically integrate formal vulnerability specifications from the bug-reporting community.

Second, the analysis must be able to scale to networks with thousands of machines. To achieve these two goals by presenting MulVAL, an end-to-end framework and reasoning system that conducts multistage ,multihost, vulnerability analysis on a network. MulVAL adopts Data log as the modelling language for the elements in the analysis (Configuration description and bug specification, Operating-system permission, and privilege model and reasoning rules etc.). We easily leverage existing vulnerability-database and scanning tools by expressing their output in Datalog and feeding it to ourMulVAL reasoning engine. Once the information is gathered, the analysis can be done in seconds for networks with thousands of machines.

3. EXISTING SYSTEM

In traditional data centers, where system administrators have full control over the host machines, vulnerabilities can be detected and fixed by the system administrator in a centralized manner. However, patching known security holes in cloud data centers, where cloud users usually have the rights to control software installed on their managed VMs, may not work efficiently and can violate the Service Level Agreement (SLA).

3.1 Detecting Malicious Behavior

1) Duan et al. focused on the detection of compromised machines that have been recruited to serve as spam zombies. Their approach, SPOT, is based on sequentially scanning outgoing messages while employing a statistical method Sequential Probability Ratio Test (SPRT), to quickly determine whether or not a host has been compromised.

2) BotHunter detected compromised machines based on the fact that a thorough malware infection process has a number of well-defined stages that allow correlating the intrusion alarms triggered by inbound traffic with resulting outgoing communication patterns.

3) BotSniffer exploited uniform spatial-temporal behavior characteristics of compromised machines to detect zombies by grouping flows according to server connections and searching for similar behavior in the flow. An attack graph is able to represent a series of exploits, called atomic attacks, that lead to an adverse state, for example a state where an attacker has obtained administrative access to a machine. There are many automation tools to build attack graph.

3.2Binary Decision Diagrams (BDDs)

O. Sheyner et al. proposed a technique based on a modified symbolic model checking NuSMV and Binary Decision Diagrams (BDDs) to construct attack graph. Drawbacks Their model can generate all possible attack paths, however, the scalability is a big problem for this solution.

3.3 Intrusion Detection System IDS and firewall are widely used to monitor and detect suspicious events in the network. Drawbacks the false alarms and the large volume of raw alerts from IDS are two major problems for any Intrusion Detection system implementations. Many attack graph based alert correlation techniques have been proposed recently. L. Wang et al. devised an in-memory structure, called queue graph (QG), to trace alerts matching each exploit in the attack graph. Drawbacks The implicit correlations in this design make it difficult to use the correlated alerts in the graph for analysis of similar attack scenarios. Roschke et al. proposed a modified attack-graph-based correlation algorithm to create explicit correlations only by matching alerts to specific exploitation nodes in the attack graph with multiple mapping functions, and devised an alert dependencies graph (DG) to group correlated alerts with multiple correlation criteria.

3.4 Attack countermeasure tree Roy et al. proposed an attack countermeasure tree (ACT) to consider attacks and countermeasures together in an attack tree structure. They devised several objective functions based on greedy and branch and bound techniques to minimize the number of countermeasure, reduce investment cost, and maximize the benefit from implementing a certain countermeasure set. In their design, each countermeasure optimization problem could be solved with and without probability assignments to the model.

Drawbacks However, their solution focuses on a static attack scenario and predefined countermeasure for each attack. N. Poolsappasit et al. proposed a Bayesian attack graph (BAG) to address dynamic security risk management problem and applied a genetic algorithm to solve countermeasure optimization problem.

4. PROPOSED SYSTEM

1)NICE (Network Intrusion detection and Countermeasures Election in virtual network systems) is proposed to establish a defense-in-depth intrusion detection framework.

2)For better attack detection, NICE incorporates attack graph analytical procedures into the intrusion detection processes.

3)The design of NICE does not intend to improve any of the existing intrusion detection algorithms; indeed, NICE makes use of a reconfigurable virtual networking approach to detect and counter the attempts to compromise VMs, thus preventing zombie Virtual Machines.

4) Deploy a lightweight mirroring-based network intrusion detection agent (NICE-A) on each cloud server to capture and analyze cloud traffic. A NICE-A periodically inspects the virtual system Vulnerabilities within a cloud server to establish Scenario Attack Graph (SAGs), and then based on the severity of identified vulnerability towards the collaborative attack goals, NICE will decide whether or not to put a VM in network inspection state.

5) Once a Virtual Machine enters inspection state, Deep Packet Inspection is applied and/or virtual network reconfigurations can be deployed to the inspecting Virtual Machine to make the potential attack behaviors prominent. Does not need to block traffic flows of a suspicious VM in its early attack stage.

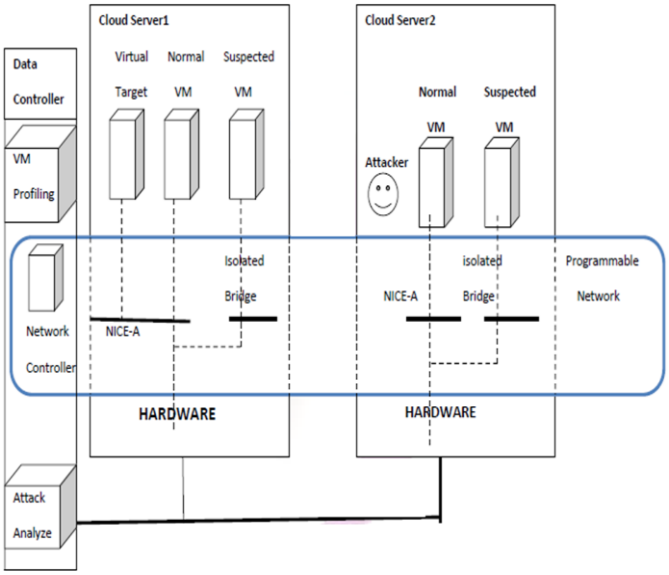

Figure 1: Nice Architecture with one cloud server

6)By using software switching techniques, NICE constructs a mirroring-based traffic capturing framework to minimize the interference on users traffic compared to traditional bump-in-the-wire (i.e., proxy-based) IDS/IPS.

7) NICE enables the cloud to establish inspection and quarantine modes for suspicious VMs according to their current vulnerability state in the current SAG.

8) Based on the collective behavior of VMs in the SAG, NICE can decide appropriate measures, for

4.1. Advantages

1)NICE significantly advances the current network IDS/IPS solutions by using programmable virtual networking approach that allows the system to construct a dynamic reconfigurable IDS system.

2)NICE, a new multi-phase distributed network intrusion detection and prevention framework in a virtual networking environment that captures and inspects suspicious cloud traffic without interrupting users' applications and cloud services.

3)NICE incorporates a software switching solution to quarantine and inspect suspicious VMs for further investigation and protection. Through programmable network approaches, NICE can improve the attack detection probability and improve the resiliency to VM exploitation attack without interrupting existing normal cloud services.

4)NICE employs a novel attack graph approach for attack detection and prevention by correlating attack behavior and also suggests effective countermeasures.

5)NICE optimizes the implementation on cloud servers to minimize resource consumption. NICE study shows that NICE consumes less computational overhead compared to proxy-based network intrusion detection solutions.

4.2. NICE System Architecture

The proposed system is designed to work in a cloud virtual networking environment. It consists of a cluster of cloud servers and their interconnections. We assume that the latest virtualization solutions are deployed on cloud servers. The virtual environment can be classified as Privilege Domains, e.g., the dom0 of XEN Servers and the host domain of KVM, and Unprivileged Domains, e.g., VMs. Cloud servers are connected through programmable networking switches, such as physical Open Flow Switches (and software-based Open v Switches deployed in the Privilege Domains. In this work, we refer Open Flow Switches and Open V Switches and their controllers as to the Software Defined Network (SDN). The deployed security mechanism focuses on providing a non-intrusive approach to prevent attackers from exploring vulnerable VMs and use them as a stepping stone for further attacks.

4.3. System Components

In this section, we explain each component of NICE

4.3.1 Nice-A The NICE-A is a Network-based Intrusion Detection System (NIDS) agent installed in each cloud server. It inspects the traffic going through the bridges that control all the traffic among VMs and in/out from the physical cloud servers. It will sniff a mirroring port on each virtual bridge in the Open v Switch. Each bridge creates an isolated subnet in the virtual network and connects to all related VMs. The traffic generated from the Virtual Machines on the mirrored Software Bridge will be mirrored to a specific port on a specific bridge using SPAN ERSPAN or RSPAN methods. It's more efficient to scan the traffic in cloud server since all traffic in the cloud server needs go through it; however our design is independent to the installed VM. The false alarm rate could be decreased through our architecture design.

4.3.2 VM Profiling Virtual machines in the cloud can be profiled to get precise information about their state, open ports, services running etc. One major factor that counts towards a VM profile is its connectivity with other Virtual Machines. Also required is the knowledge of services running on a VM so as to verify the authenticity of alerts relating to that VM. An attacker can use port scanning program to perform an intense examination of the network to look for open ports on any Virtual Machine. So information about any open ports on a VM and the history of opened ports plays a significant role in

determining how vulnerable the Virtual Machines is. All these factors will form the Virtual Machine profile. VM profiles are maintained in a database and contain comprehensive information about vulnerabilities, traffic and alert.

4.3.3 Attack AnalyserThe major functions of NICE system are performed by attack analyzer, which consist of procedures such as attack graph creation and update, alert correlation and countermeasure selection. The process of constructing and using the scenario Attack (*SA*) consists of three phases: information gathering, attack graph construction, and potential exploit path analysis. With this information, attack paths can be modeled using SAG. Each node in the attack graph represents an exploit by the attacker. Each path from an initial node to a goal node represents a successful attack.

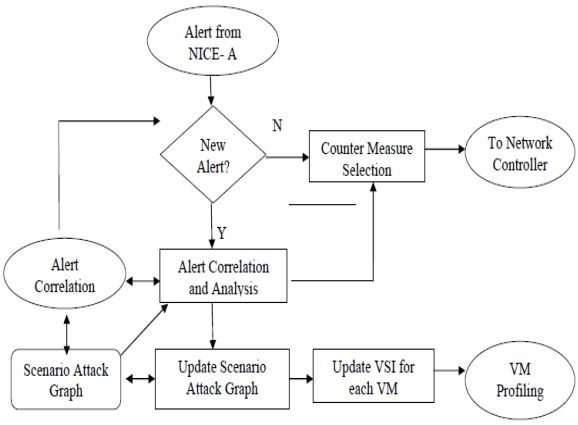

Fig:2 Flow Chart for Attack Analyzer

4.4.4 Network Controller The network controller is a key component to support the programmable networking capability to realize the virtual network reconfiguration. In NICE, we combined the control functions for both Open V Switch and Open Flow Switch into the network controller that allows the cloud system to set security/filtering rules in an integrated and comprehensive manner. The network Controller is responsible for collecting network information of current Open Flow network and provides input to the attack analyzer to construct attack graphs. In NICE, the network control also consults with the attack analyzer for the flow access control by setting up the filtering rules on the corresponding Open V Switch and Open Flow Switch. Network controller is also responsible for applying the countermeasure from attack analyzer. Based on Virtual Machines Security Index and severity of an alert, countermeasures are selected by NICE and executed by the network controller.

5. PERFORMANCE ANALYSIS

The amount of CPU utilized by MDIDS is less when compared to other intrusion detection system. The graph below highlights this fact.

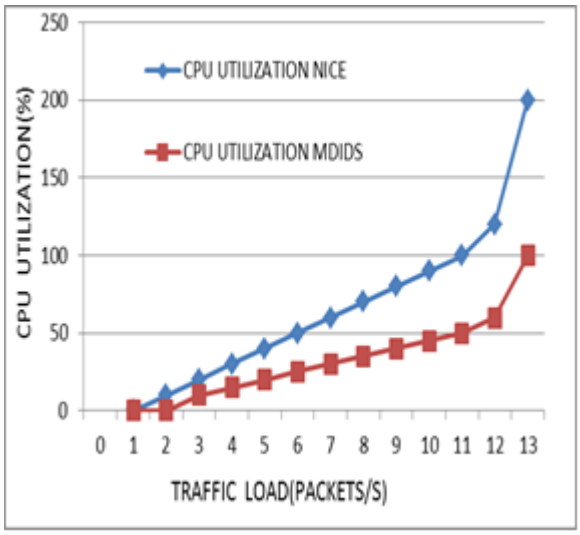

Fig. 3 CPU Utilization in MDIDS

The CPU utilization in mirror based MDIDS [11], proxy based MDIDS and MDIDS are compared. The CPU utilization is very low in MDIDS than in other two intrusion detection system. The CPU utilization is given in percentage against traffic load measured in packets per second. Proxy based MDIDS utilizes twice the amount of CPU than MDIDS for a traffic load of 15 packets per second. Mirror based MDID utilizes 35% more CPU than MDIDS for an equal traffic load. The IRT is analyzed in two phases. The IRT is calculated with the number of applications plotted against its response time measured in milliseconds. In the first phase IRT is calculated without the introduction of MDIDS in cloud.

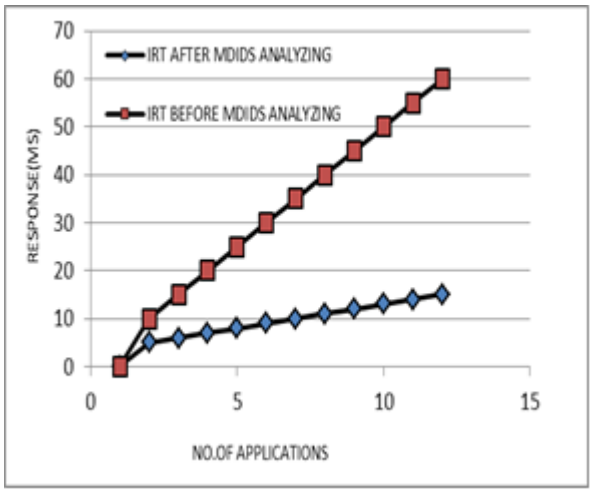

Fig. 4 Infra-structure Response Time Calculation

The IRT is very high before introducing MDIDS. The IRT becomes very low after the introduction of MDIDS. This clearly indicates that there is growth in the performance of cloud in MDIDS. The service to the clients is implemented

rapidly in MDIDS. The time taken to create a required VM before detecting DDOS using MDIDS is more.

The time in milliseconds and the number of VM created are noted and plotted. Before analyzing the DDOS , the VM creation taking 100milliseconds after detecting the DDOS using MDIDS the VM creation taking only 10milli-seconds The time taken to create one VM before detection is ten times the time taken to create it after detection.

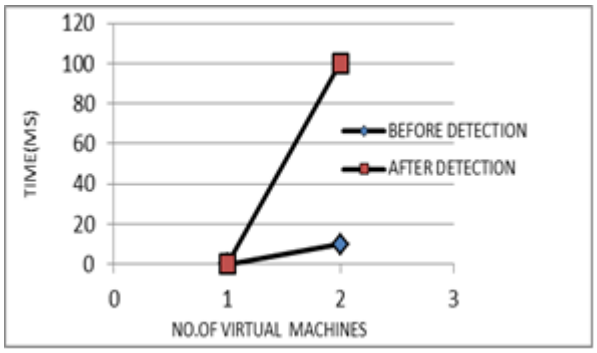

Fig. 5 Time Taken to Create Virtual Machine

6. CONCLUSION

This paper provides a Review on Network Intrusion detection and Countermeasure selection and the related security concepts. NICE, which is proposed to detect and mitigate collaborative attacks in the cloud virtual networking environment. NICE uses the attack graph model to conduct attack detection and prediction. The proposed solution investigates how to use the programmability of software switches based solutions to improve the detection accuracy and defeat victim exploitation phases of collaborative attacks. The system performance evaluation indicates the feasibility of NICE and shows that the proposed solution can significantly reduce the risk of the cloud system from being exploited and misused by internal and external attackers

REFERENCES

[1] Cloud Security Alliance, "Top threats to cloud computing1.0,"https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf, March 2010.
[2] A. Vijayan and B. Joshi, "Securing Cloud Computing Environment against DDoS Attacks," Proceedings IEEE International Conference Computer Comm. and Informatics, Jan. 2012.
[3] J.B. Joshi, H. Takabi and G. Ahn, "Security and Privacy Challenges in Cloud Computing Environments," IEEE Security and Privacy, vol. 8, no. 6, pp. Dec. 2010.
[4] Z. Duan, P. Chen, F. Sanchez, Y. Dong, M. Stephenson and J. Barker, "Detecting Spam Zombies by Monitoring Outgoing Messages," IEEE Trans. Dependable and Secure Computing, Apr. 2012.
[5] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "BotHunter: Detecting Malware Infection through IDS-driven Dialog Correlation," Proc. 16th USENIX Security Symp. (SS '07), pp. 12:1-12:16, Aug. 2007.
[6] X. Ou, S. Govindavajhala, and A.W. Appel, "MulVAL: A Logic-Based Network Security Analyzer," Proc. 14th USENIX Security Symp., pp. 113-128, 2005.
[7] "Citrix XenServer." [Online]. Available: http://www.citrix.com/xenserver.
[8] "Kernel based Virtual Machine (KVM)." [Online]. Available:http://www.linuxkvm.org/
[9] "Openflow."http://www.openflow.org/wp/learnmore/, 2012.
[10] "Open vSwitch project," http://openvswitch.org, May 2012
[11] S.Usha, Dr.A.Tamilarasi, R.Kalaivanan "MDIDS: Multiphase Distributed Intrusion Detection in Virtual Network Systems" IJSET(ISSN : 2277-1581) Volume No.3 Issue No.4, pp : 355-358