A novel approach for distributed detection of infected machines in the cloud

Position paper

Mohammad Reza Memarian
University of Padua
Padua, Italy
memarian.m.reza@gmail.com

Mauro Conti
University of Padua
Padua, Italy
conti@math.unipd.it

Ville Leppanen
University of Turku
Turku, Finland
ville.leppanen@utu.fi

ABSTRACT

Abusing cloud services for cyber-crime intentions is a prevalent threat to cloud computing technology. Some of the cloud's essential characteristics are main origin of the threats for the cloud. One example of these kind of malicious usages is infected virtual machines forming botnets to conduct cyber-attacks such as Distributed Denial of Service (DDoS). DDoS can happen either against a system inside the cloud or a system outside of the cloud using cloud resources. While various works have been done to develop Intrusion Detection Systems (IDS) for cloud to mitigate threats like DDoS, unfortunately still the solutions for detection of botnets in the cloud are not mature as DDoS attacks using botnets are on rise. In fact most solutions are designed with limited view over a single host while having no communication with detection modules in other hosts. In this paper we present an approach that has a broad view over all virtual instances in the cloud to address the above concern. In our design we place the detection module in a higher level than host level in the cloud. Detection module coordinates and analyses information gathered in an agent-less manner from Virtual Machines (VMs) using Virtual Machine Introspection (VMI). The system gathers wide VM's system state information. Such information enables detection module to have a broad view over the entire cloud live images and detect malicious collaborative entities. As VMs in cloud are distributed over various clusters, cloud monitoring systems need to be distributed too.

1. INTRODUCTION

As users and companies are moving their data to clouds, attackers are redirecting their attack focus to cloud too. Cloud computing provides suitable infrastructure for both acceptable and malicious usages. Botmasters specifically benefit from the provided infrastructure as cloud computing offers them various possibilities to conduct their desired attacks [14]. Meanwhile cyber criminals are moving toward creating more sophisticated botnet intended malware that can scape from being detected. For instance malware authors integrate multiple backup forms of command and control. As described in [9], botnets are becoming more resilient and responding faster to countermeasures. Based on [11], DDoS attacks which are one of the consequences of botnet formation, tend to change their attack vector during a single attack. This matter can help attack sources to remain hidden while increasing attack impact. Indeed it is very vital for Cloud Service Providers (CSP) to detect botnets in their cloud, prevent and traceback to the botmaster as botmasters may use from cloud services to host their command and control servers [31]. On the other hand, design of network protocols forms a portion of the problem as some of them were not designed by security in mind manner. From time to time, vulnerabilities in network protocols get exploited which may result in massive attacks as well as system damages.

As mentioned earlier, conducting DDoS attack is one of the intentions of bot masters. Various recent DDoS attacks which are on rise and threaten cloud security are Network Time Protocol (NTP), Domain Name System (DNS) and Simple Service Discovery Protocol (SSDP) amplification attacks that are result of vulnerabilities in network protocols [11], [12]. In these attacks, vulnerable servers are used as reflector to direct huge amount of traffic to a victim in result of legitimate looking request from infected machines. So detection of these kind of attacks is not simple based on the network traffic analysis. As amplification attacks are on rise and have relatively high impact on the victim, research such as [18] are demanding to confront the attacks.

Based on [6], five essential cloud characteristics are on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service. From the mentioned characteristics, two which their side effects empower malicious activities are rapid elasticity and on-demand self-service. On demand self-service refers to ability for users to provision cloud services without human interaction. Rapid elasticity refers to ability of users to shape the cloud resources utilization based on their need. Nowadays attackers have the possibility to employ very low cost online botnet entities to conduct DDoS attack (i.e., about $5 [10]). As a result, even small companies or individuals can have access to vast amount of computing resources in a short period of

time with low financial requirements.

Cloud Security Alliance (CSA) has addressed top nine cloud security threats in year 2013 [8]. Abuse of cloud services is one of the threats placed in the mentioned list as well as DDoS attacks. Cloud resources can be misused to conduct attacks toward targets either inside or outside of the cloud. Fraudulent Resource Consumption (FRC) occurs by consuming the bandwidth of Web-based services. This Attack that results in increasing the financial burden of cloud consumer is an example of attacks by botnets toward a service hosted in the cloud [22], [16], [21].

Security problems are one of the largest obstacles in adopting cloud computing by companies. There are some cloud-specific vulnerabilities that are sources of the security problems is cloud. But the question is, what are the characteristics of cloud-specific vulnerabilities that create cloud-specific threats? A vulnerability is applicable to cloud computing if that vulnerability exists in one of the core technologies of cloud computing [15]. As cloud computing is designed on the idea of reducing cost and increasing efficiency, it is constructed on virtualization technology. So virtualization technology is one of the core technologies in cloud computing architecture. Vulnerabilities that are threats to virtualization are indeed threats to cloud computing too. Tackling vulnerabilities in virtualization leads to increasing security of the cloud as virtualization security has direct relationship with cloud security.

In cloud, security of images running on virtual machine's structure while holding applications, are critical to the entire cloud security as they are the most inner entity in cloud virtualization design circle. Securing VM images and monitoring activities of these small entities, boots up overall security of the cloud as images construct base system of the cloud. VM images must have high integrity as they determine initial state of the VMs running in the cloud. Usage of these images may require some cautions as users in the cloud can use shared third party images. An example of these kind of shared image usage model can be a company advertising its application. The software firms configure the application on a VM image and publish that image for use and test of the application by cloud users. In the other case, users can publish their specific configured image and share it on the cloud either for free or to sell to others. Shared images can be published either to a specific group of users or to a public group of users with the aim of users using a homogeneous or pre-configured image. In this regard malicious users can configure and publish an infected image which may have malware embedded in it such as backdoor or a rootkit. Adoption of these infected images multiple times by users across the cloud can lead to cloud-wide infected image usage.

In [30], researchers explained risks that administrators, image publishers and image retrievers face in cloud image repositories. An image management system is proposed to control access to images, track source of images, and provide users and administrators with efficient image filters and scanners that detect and repair security violations. The mentioned research depicts the importance of risk reduction of image repositories and risks that can be involved in using shared cloud images.

While usage of shared images can have a high risk, CSPs do not have strong controls over image sharing as risk of using the shared images should be handled by image users. As an example Amazon a CSP, stated some security guides to help users to reduce risks of adopting shared images. It depicts that users must handle the risk in this case [1].

VMs running cloud core services are desired to have homogeneous images. Infection of one of these machines can reveal the vulnerability that exists in other homogeneous VMs too. It can end up in vast infection of homogeneous VMs across the cloud. Other scenario regarding this matter can be that one or multiple malicious users rent multiple VMs on the cloud. These virtual machines which are controlled from the beginning by specific users can do malicious activities in collaboration with each other.

According to matters discussed above, creating network of collaborative machines or in the other word botnet formation is easier in the cloud compared to conventional environments. When a bot enters a system, it should look for some distinctive vulnerabilities. But in the cloud (specifically in the case of image sharing), bots depend much less on victim's system software stack for exploitation. Ease of image sharing and interest for employing homogeneous images by user groups and CSPs are factors that accelerate malware propagation. Furthermore Infection methods may differ in a cloud botnet to mislead detection systems. However, recent works in the literature mainly focus on one way infection methods.

As it is possible to determine location of cloud instances, attackers can truly distribute and choose infected machines involved in a DDoS attack to conduct a strong distributed attack [19]. Refer to the mentioned concern and with refer to the fact that DDoS attacks and in general botnet related attacks are on rise in the cloud, presence of a system that has a wide look over the cloud and is able to correlate and coordinate all the information cloud-wide is vital. In this paper we review previous works done on cloud-botnet detection. Then we present a system design that adopts a decentralized approach for botnet detection in the cloud. Our approach takes advantage from virtualization as one of the cloud core technologies. Unlike previous works, goal of this research is to present a system to broadly detect malicious collaborative images in cloud-wide manner using VMI. Our design looks for symptoms leading to botnet instead of botnet behaviour itself in the VMs to detect and mitigate the threat from the source. It results in increase of detection rate while signature database size decreases.

The remaining of the paper is structured as follows. In Section 2 we provide a brief overview of the research literature on Botnets, cloud computing and VMI technique. Section 3 introduces and discusses the main existing approaches for cloud-botnet detection and Forensic Virtual Machine (FVM). In Section 4 we introduce our novel design for detection of a set of infected systems, while in Section 5 we discuss implementation details and show preliminary experimental results. Finally, in Section 6 we draw our conclusions, and discuss possible future research directions.

2. BACKGROUND

This section presents a study about botnet, cloud computing and virtual machine introspection technique.

Botnet. Botnet is a group of infected computer systems that enables a controller to have control over the group. The infected entities are called zombies and in most cases, users of infected systems are not aware that their system is compromised. There are various malicious usages of botnets for controllers. Examples of these usages are conducting DDoS attacks, spamming, search engine optimization poisoning, pay-per-click fraud, financial fraud, bitcoin mining and information stealing [7].

User's system gets infected by malware through various false user's behaviour such as opening an email with malicious content, visiting malicious website and installing pirated software. When systems are infected, malware installs a backdoor to enable bot master to have consistent communication with zombies. Infected systems may send primary victim system's information such as IP address, geographical location, operating system type, host name and other information which may help botmasters to classify zombies.

Infecting malware may download newer version of itself to remain updated. Infected machines and bot master may communicate with each other through various ways such as P2P protocol, HTTP commands and IRC channels. Nowadays, cost and skills of having a botnet is reduced. Some of the botnets source code can be found in underground black market websites with very low cost while they are designed to be easily worked.

Cloud Computing. Based on national institute of standards and technology (NIST), cloud computing is a model for enabling on-demand access to a shared pool of configurable computing resources. These resources are able to quickly be employed with minimal management effort and human interaction [6]. Indeed cloud computing is changing the way that computing services are offered.

Cloud services are offered through 3 service models which are: Software as a service (SaaS), Platform as a service (PaaS) and Infrastructure as a service (IaaS). Also cloud computing is deployed based on 4 deployment models. The choice of deployment model depends on the environment in which cloud is going to be deployed on. The 4 cloud deployment models are: private cloud, community cloud, public cloud and hybrid cloud. As mentioned earlier, security issues are one of the biggest obstacles in adoption of cloud through companies.

Virtual Machine Introspection (VMI). VMI is a technique, enabling monitoring VM by collecting system level data from memory of monitored VM. VMI is used for developing wide variety of security related applications. The prevalent use is in security monitoring applications that VMI enables VM's system state data collection from a secure place outside of the monitored VMs. Virtual machine introspection is done in an agent-less manner meaning that in order to collect monitoring data, there is no need for an agent to be installed in monitored system. Traditional monitoring methods required monitoring agent to be placed in the monitored VMs. One of the disadvantage of agent-oriented monitoring systems is that in case the monitored VMs are compromised, malware on the systems may deactivate the agent on the monitored systems. This results in lack of having accurate and on demand data. VMI tackles the problem of agent-oriented techniques in virtual environments. The monitoring can be done from a secure place like virtual machine monitor (VMM) or any other privileged virtual machine. Garfinkel and Rosenblum in [29], presented an intrusion detection system design based on VMI technique. As any entity in computing can be prone to attacks, so VMI is too. Researchers in [13] showed that VMI can be used to present inaccurate data to the monitoring system outside of monitored VMs.

3. RELATED WORK

In this section we review previous works done on botnet detection in the context of cloud and also discuss on FVM which is quite a new term.

Cloud-Botnet detection. Researchers in [20], designed a passive and an active malware detection module in VMM to actively look for information in VMs using VMI. The solution presented in their work mainly focuses on functions in one host and detects zombie machines, based on trained node about bot behaviours. The solution does not have a wide view over the cloud, and it makes the botnet profile based on just the API calls done by the applications.

A research paper by Kebande and Venter [23], proposed botnet detection methods in the cloud environment using Artificial Immune System (AIS). This mechanism uses negative selection algorithm to match whether the botnet belongs to self or non-self pattern. It gets done by training some detectors on identifying malicious activities pattern. In the time of attack (when bots are attacking victims), AIS is trained to detect a malicious activity pattern by observing the behaviour based on the network traffic movement.

Beidermann and Katzenbeisser in [27] presented a research work to detect computer worms in the cloud based on the spreading behaviour. The solution looks for inconsistency in behaviour of machines. Presented system is limited to only looking for two kinds of information that are start of a new process or a loaded module which is not listed in the predetermined white list or its presence is not normal. In this case, start of a process which is not is the white list even on several machines can not necessarily indicate malicious activity. In this solution also monitoring stages are mentioned that have a random look into VMs which is not suitable for continues monitoring. The information obtained from each VM using VMI is sent to a central spreading monitor in the VMM layer that compares the lists. Although it is claimed that the system is able to detect various unknown malware but trojans today do not necessarily start a new process with an undefined name.

Watson in [24] presented a distributed detection system that combats malware in multi-server cloud. The research proposed having agents in each VMM of servers that com-

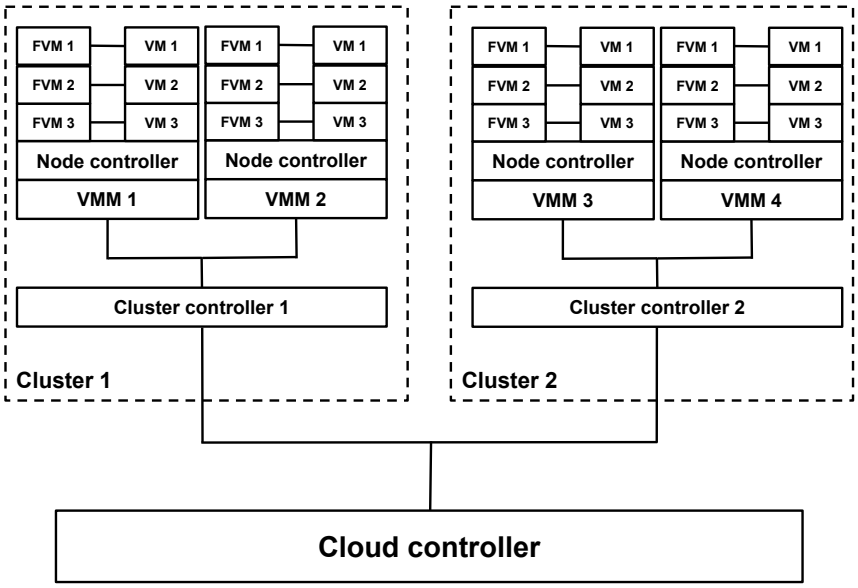

Figure 1: Typical cloud platform design with FVM incorporated inside

municate with each other and pass obtained information to each other. Without a central decisioning and information processing point, solutions will not be effective.

Forensic Virtual Machine (FVM). Due to modular design trend of malware, researchers in [17], used method of detecting malware by identifying the symptoms of malicious behaviour as opposed to looking for malware itself. To accomplish task of symptom detection, they presented small introspecting virtual machines. These machines that take advantage of VMI technique to look for existence of desired symptom are called FVM. FVMs are small independent privileged VMs which inspect memory page of other introspected virtual machines. While introspecting VMs, each FVM looks for existence of only one symptom. Findings of FVMs are reported to a central command and control module. FVMs choose the introspected machine randomly using mobility algorithm embedded in them. When an FVM finds existence of a symptoms, other FVMs will be commanded to check existence of other symptoms too on the same system. So one VM may not be introspected by any FVM at a given time while other VMs are under introspection by several FVMs.

So as monitoring of symptoms are done randomly, a symptom may appear and disappear before the FVM arrives on the system or it may start after FVM introspection is done. As number of FVMs has a direct relationship with number of symptoms the system looks for, FVMs should be small size VMs. Researchers in [28], implemented FVMs using MiniOS. MiniOS is a small operating systems distributed by Xen source code and is intended to be used for dom0 disaggregation. Although MiniOS lacks variety of a normal OS functionalities, but they have advantage of being very small in size. As an instance basic networking functionalities can be added to it by using a mini TCP-IP stack module.

4. OUR PROPOSED APPROACH

In this section, we present our approach toward information gathering, analysis and processing of obtained data.

4.1 Data gathering method

As mentioned earlier, VMI can help security applications by enabling access to system level information of guest VMs. So we get advantage of VMI technique to gather needed information from memory of target VMs. Introspection technique can give a set of system-level information at any given point in time about each live virtual machine. Examples of these informations can be list of running processes, loaded modules, opened files and running services. In our design, we use concept of FVM for data gathering entity. As depicted in Figure 1, we assign one FVM per VM to retrieve system level information from each VM's memory page at any given time. Each FVM in our design is dedicated to one VM and looks for existence of all symptoms in the VM assigned to it.

Assigning one FVM per VM as opposed to [28], has various benefits. Introspected VMs trust only to one FVM which results in monitored VM's Trusted Computing Base (TCB) reduction. In addition, all symptoms are checked at once and frequently instead of random check which is not feasible in terms of symptom detection. In time of FVM infection, other monitored machines can remain immune from consequences of one of the FVM's infection as FVMs do not randomly check other VMs. In Figure 1, a high level architecture design of cloud platform with FVMs added is shown. Cloud controller is outermost entity is conceptual cloud platform design.

4.2 On gathered data

List of checked symptoms need to be defined by determining inconsistencies in VMs as our approach focuses on system level inconsistencies. Based on malware analysis information available on [2], we constructed a table depicting common modifications that 9 botnet intended malware do to victim systems after infecting them. Under each malware's name the year that was discovered has been mentioned. For example, all malware listed in Table 1 create registry entry after infecting victim. So it can be resulted that creating registry entry is a prevalent action among malware. Creation of registry value under a specific key would be a symptom as opposed to entry value. Ofcourse creation of registry value happens by legitimate processes too. For this reason, system looks for set of diverse symptoms. Occurrence of number of them among several machines would trigger security alert.

Based on information depicted in Table 1 and [25], a sample set of inconsistencies that can be considered prevalent among botnet intended malicious codes are: File/Folder creation, registry key creation, existence of hidden (unlinked) processes, existence of hidden (unlinked) DLL and existence of abnormal loaded service in memory.

4.3 Categorization of obtained data

As depicted in Figure 3, information gathered at each host gets in relevant data structure. These information will be sent to cloud controller through node controller and VMM. Digging down in categorization of obtained data, as shown in Figure 2, after data gathering module in FVM gathers the data, it passes them to a basic symptom search module. This module only indicates existence or non existence of symptoms in VMs. The obtained data structure is transferred to data set matching module in cloud controller. This module may use machine learning techniques such as similarity learning to identify data sets that even do not exactly have same entries in their symptom detection list. In case of accurate matching, one notification will be sent to logging module to log the events. Another command can be sent to privileged domain of each host to take necessary action to stop the threat such as quarantining the suspicious VMs.

As analytic module is placed in cloud controller level, VMM does not have any role in analysing data which results in offloading this task from VMM. Furthermore as in virtualization, VMM (and in case of Xen, Dom0) takes important roles such as sharing the physical resources, it may be target of attacks. There are various vulnerabilities that can be mapped to each important functionality of the VMM [26].

5. IMPLEMENTATION AND EXPERIMENTS

In this section, we describe an overview of configured system.

5.1 XEN

We used Xen as VMM for our system configuration. Ubuntu 14.04 (Linux kernel 3.13.0-24-generic) is our choice for Dom0 Implementation. The reason for using Xen is that while Xen is a widely used VMM, it is an open source virtualization solution. Large CSPs like Amazon use Xen as the VMM for virtualization infrastructure [5]. VMs running on Xen are referred as domains. There are two domain types in Xen, privileged domain and unprivileged domain. A supervisor

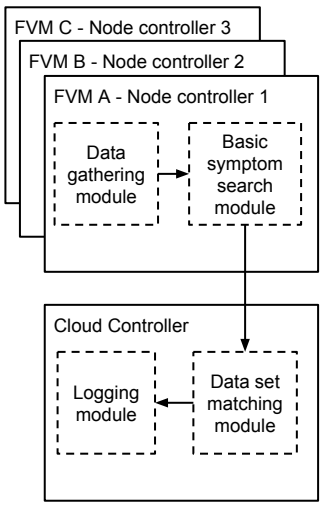

Figure 2: Architecture of detection system

like VM, called dom0 runs as privileged domain. It is the first VM that boots after VMM starts and holds hardware drivers and control software stack. Other guest VMs run as unprivileged domains. Dom0 is allowed to have access to memory pages of other VMs so VMI is possible to be done from Dom0 level. By modifying Xen access control, other VMs will be able to introspect memory of other VMs too.

VMs running on Xen should run on one of the two available modes. The two modes are Hardware Virtualization (HVM) and Para Virtualization (PV). In HVM, VMs are not aware that they are running in a virtualized environment. Whereas in PV mode, VMs experience some modification and are aware that they are running on virtual platform. VMs running in PV mode experience faster system performance. Closed source operating systems like Microsoft windows, must run is HVM mode as modification of them is not possible. We run a guest operating system running in HVM mode while having Microsoft windows 7 as operating system.

We used LibVMI version 4.4 to conduct VMI for obtaining information from VM's memory. Libvmi is an application programming interface (API) which enables introspecting VM to read from and write to memory of introspected VM. Libvmi is written in C while offering a python wrapper for users to be able to write the introspection applications in python as well [3]. Libvmi supports Virtual machines running on Xen and KVM. For introspection to take place using Libvmi, introspecting VM should have access to kernel symbols of introspected VM. So when an application requests to view some data through introspection, kernel symbols of the target system are accessed through accessing to system.map file which should be placed in /boot directory of introspecting machine to receive virtual address (VA) of the symbol. Then through several mappings, correct data page is found and returned to LibVMI and LibVMI returns the requested data to the VMI application.

5.2 Introspection tools

System modification	Rustock.B (2006)	Virut (2007)	Koobface (2008)	Tidserv (2008)	Qakbot (2009)	Pilleuz (2009)	Zbot (2010)	Carberp.B (2014)
File/Folder created	✓		✓	✓	✓	✓	✓	✓
Files/Folder deleted				✓		✓		
Files/Folder modified		✓		✓	✓	✓		
Registry entry created	✓	✓	✓	✓	✓	✓	✓	✓
Registry entry deleted								
Registry entry modified								
Kernel Modification	✓							
Code injection into process/DLL/App	✓	✓	✓	✓	✓	✓		

Table 1: Botnet intended malware functionalities based on [2]

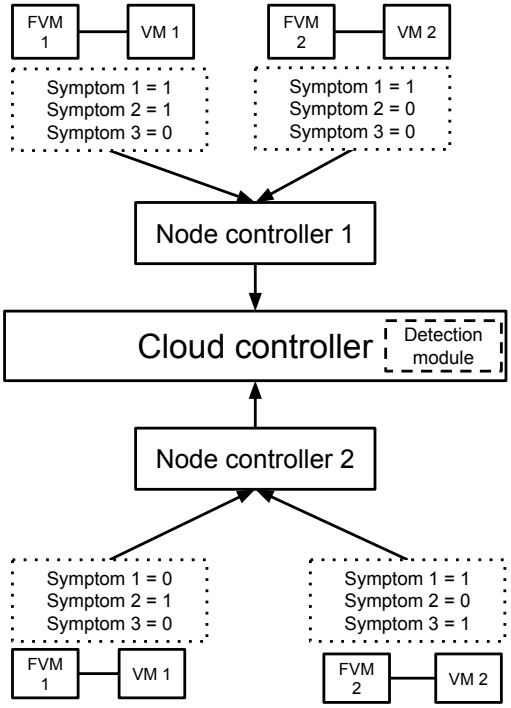

Figure 3: Transfer of gathered data to cloud controller

LibVMI is well integrated with volatility [4]. Volatility is an advanced open source memory forensic framework which offers variety of functions to users for extraction of data from memory snapshots and dumps. As Libvmi has integrity with volatility, it is possible to use volatility functions on live VMs while using Libvmi by adding Libvmi address space to address space list of volatility.

5.3 Experiments

We setup an environment to run some preliminary experiments and simulation. We run a guest VM (domU) to infect it with a malware sample. In this test we infect domU with an instance of Koobface. Libvmi and volatility are configured to gather information from Dom0 level. We used from psscan plug-in of volatility to gather all running processes including hidden (Unlinked) processes that are not shown in the task list of the operating system and terminated processes which with random introspection technique is not detectable. When domU became infected, a process with name bolivar30.exe started. Then immediately dllhost.exe and regedit.exe were started and terminate. They were shown by psscan plug-in as terminated process.

As a result of regedit.exe start and termination, a registry entry was created to start the malicious process each time the system starts. So in this case our framework looks into the registry key which is prevalent for creation of value to check whether any value is added to the key or not. It does not check what values are added. Of course creation of registry entry can be done for legitimate purposes too but existence of this symptoms in conjunction with others in a cloud-wide manner can be a brief indication of infection. For the detection system to be able to distinguish among different infected groups, diverse set of symptoms must be checked.

6. CONCLUSION AND FUTURE WORK

Abusing cloud services is a prevalent threat toward cloud computing security. The need for having distributed and efficient monitoring systems is vital to empower security of the cloud. By making sure about security of the most inner entities of the cloud computing which are VM images and running VMs, the entire security of the cloud can be increased. Having distributed monitoring systems that are capable of widely and actively monitor, log and report malicious activities and movements is essential. In this paper we presented a novel approach which is able to act as a super system looking into its subsystems and strongly resulting in relating events occurring in each cluster and detect collaborative running malicious images.

There are two future directions which we would like to follow as our future work. First we would like to move more toward depicting detection ratio of our approach by completely implementing the proposed approach and testing under various attack vectors and obtain related benchmark results. Second is enhancing privacy of data obtained from guest VMs using VMI as is a very important point. In the researches which VMI is involved, privacy of obtained user's data is generally overlooked in favour of secure monitoring.

VMI technique can reveal various private information from inside the guest VMs. These information can be target of misuse by malicious insiders. So our second future research direction will be enhancing privacy of user's data in cloud monitoring systems using VMI.

7. REFERENCES

[1] Building AMIs for AWS Marketplace. https://aws.amazon.com/marketplace/help.

[2] Symantec website. www.symantec.com.

[3] VMItools. https://code.google.com/p/vmitools/.

[4] Volatility foundation website. www.volatilityfoundation.org.

[5] XEN project website. www.xenproject.org.

[6] Defenition of cloud computing. Technical report, National Institute of Standards and Technology (NIST), 2011.

[7] Anatomy of a Botnet. Technical report, Fortinet, 2012.

[8] Nine threats to cloud. Technical report, Cloud Security Aliance (CSA), 2013.

[9] Security threat report 2014. Technical report, Sophos, 2014.

[10] The continued rise of DDoS attacks. Technical report, Symantec, 2014.

[11] Verisign distributed denial of service trends report, 2nd quarter 2014. Technical report, Versign, 2014.

[12] Verisign distributed denial of service trends report, 3rd quarter 2014. Technical report, Versign, 2014.

[13] Bahram, S. and Xuxian Jiang and Zhi Wang and Grace, M. and Jinku Li and Srinivasan, D. and Junghwan Rhee and Dongyan Xu. DKSM: Subverting Virtual Machine Introspection for Fun and Profit. In *Proceedings of 2010 29th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 82–91, 2010.

[14] Clark, Kassidy P. and Warnier, Martijn and Brazier, Frances M. T. Botclouds - The Future of Cloud-based Botnets? In *Proceedings of the 1st International Conference on Cloud Computing and Services Science (CLOSER 2011)*, pages 597–603, 2011.

[15] Grobauer, B. and Walloschek, T. and Stocker, E. Understanding Cloud Computing Vulnerabilities. *IEEE Security Privacy*, 9(2):50–57, 2011.

[16] Gruschka, N. and Jensen, M. Attack Surfaces: A Taxonomy for Attacks on Cloud Services. In *Proceedings of 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, pages 276–279, 2010.

[17] K. Harrison, B. Bordbar, S. Ali, C. Dalton, and A. Norman. A Framework for Detecting Malware in Cloud by Identifying Symptoms. In *Proceedings of the 2012 IEEE 16th International Enterprise Distributed Object Computing Conference (EDOC)*, pages 164–172, 2012.

[18] Herzberg, Amir and Shulman, Haya. DNS Authentication As a Service: Preventing Amplification Attacks. In *Proceedings of the 30th Annual Computer Security Applications Conference (ACSAC)*, pages 356–365, 2014.

[19] Herzberg, Amir and Shulman, Haya and Ullrich, Johanna and Weippl, Edgar. Cloudoscopy: Services Discovery and Topology Mapping. In *Proceedings of the 2013 ACM Workshop on Cloud Computing Security Workshop (CCSW)*, pages 113–122, 2013.

[20] S.-W. Hsiao, Y.-N. Chen, Y. Sun, and M. C. Chen. A cooperative botnet profiling and detection in virtualized environment. In *Proceedings of 2013 IEEE Conference on Communications and Network Security (CNS)*, pages 154–162, 2013.

[21] Idziorek, Joseph and Tannian, Mark and Jacobson, Doug. Detecting Fraudulent Use of Cloud Resources. In *Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop (CCSW)*, pages 61–72, 2011.

[22] Idziorek, Joseph and Tannian, Mark F. and Jacobson, Doug. The Insecurity of Cloud Utility Models. *IT Professional*, pages 22–27, 2013.

[23] V. Kebande and H. Venter. A cognitive approach for botnet detection using Artificial Immune System in the cloud. In *Proceedings of 2014 Third International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec)*, pages 52–57, 2014.

[24] M. R. Watson. Malware detection in the context of cloud computing. In *Proceedings of 13th annual post graduate symposium on the convergence of telecommunication, networking and broadcasting*, 2012.

[25] A. H. Michael Sikorski. *Practical malware analysis*. 2012.

[26] Perez-Botero, Diego and Szefer, Jakub and Lee, Ruby B. Characterizing Hypervisor Vulnerabilities in Cloud Computing Servers. In *Proceedings of the 2013 International Workshop on Security in Cloud Computing (SCC)*, pages 3–10, 2013.

[27] Sebastian Biedermann, Stefan Katzenbeisser. Detecting Computer Worms in the Cloud. In *Proceedings of 2011 Open Problems in Network Security (iNetSec)*, 2011.

[28] Shaw, A.L. and Bordbar, B. and Saxon, J. and Harrison, K. and Dalton, C.I. Forensic Virtual Machines: Dynamic Defence in the Cloud via Introspection. In *Proceedings of 2014 IEEE International Conference on Cloud Engineering (IC2E)*, pages 303–310, 2014.

[29] Tal Garfinkel, Mendel Rosenblum. A virtual machine introspection based architecture for intrusion detection. In *Proceedings of 2003 Network and Distributed System Security (NDSS) Symposium*, 2003.

[30] Wei, Jinpeng and Zhang, Xiaolan and Ammons, Glenn and Bala, Vasanth and Ning, Peng. Managing Security of Virtual Machine Images in a Cloud Environment. In *Proceedings of the 2009 ACM Workshop on Cloud Computing Security (CCSW)*, pages 91–96, 2009.

[31] Wenjie Lin and Lee, D. Traceback Attacks in Cloud – Pebbletrace Botnet. In *Proceeding of 2012 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 417–426, 2012.