Computational Modeling of Virtualization-Based Privacy Preservation Against Malware in Unstructured Network

Oviebor, E. O., Ogheneovo, E. E. & Egbono, F.
Department of Computer Science
University of Port Harcourt
Port Harcourt, Nigeria.
oviebor.emmanuel.o@gmail.com, edward_ogheneovo@yahoo.com, fubaraegbono@gmail.com

Abstract
This work creates a model to detect and prevent malicious code from accessing users' data privacy at runtime and the evaluation of its effectiveness. A thorough background study is conducted to gain knowledge about malicious files and their mode of carrying out attack on legitimate program. In this paper, we developed the Virtual Machine Hypervisor Introspection and Mandatory Access Control (VMHIMAC)system to detect and prevent intrusion by a large number of suspected files. A prototype system is developed and experiment perfumed on malicious and benign files to evaluate the accuracy of the newly developed system and generated reports are compared with Hypervisor Introspection (HI), Mandatory Access Control (MAC), and virtual Machine Introspection (VMI). Based on thorough background study the intrusion detection and preventions system we developed and after experiment performed using implemented prototype system on selected malware and benign samples. The experiment results show that VMHIMAC detected and prevented more malware and benign samples than the other existing systems. The results obtained shows that there is efficient preservation of virtual machine resources, including, intended changes to file system, registry, processes and other systems peripheral.
Keywords: Unstructured network,Virtualization, Malware,Computational Modeling

1.0 Introduction

Unstructured networks, like those used in cloud computing, allow for on-demand network access so that computational resources like storage space, network bandwidth, and applications can all be shared. It promotes quick scaling while requiring little service provider management (Kuyoro et al., 2011). In the case of cloud architecture, an unstructured network offers many advantages, including quick deployment, pay-for-use, lower costs, scalability, rapid provisioning, rapid elasticity, widespread network access, low-cost disaster recovery, data storage options, on-demand security controls, real-time system tamper detection, and prompt service reconstitution. While the network offers many benefits, many of the major players may be tempted to stay back until some of the drawbacks are better recognized (Subashini andKavitha, 2011).

Although cloud computing and other unstructured networks are intended to offer a healthier utilization of resources deploying virtualization techniques and to reduce most of the work load from the client, it is fraught with security threats (Keiko et al., 2013). Thus, we need to protect the data in the midst of untrusted programs (Kevin et al., 2010).While reducing cost is a major motivation for moving towards unstructured network managed by a cloud provider, reducing responsibility for security or privacy should never be. Ultimately, the organization is responsible for the overall state of the outsourced services. Monitoring, addressing security, as

well as privacy issues remain within the purview of the organization, just like other essential issues, like performance, accessibility, and recovery (Wayne, 2011).Nevertheless, the benefits and rapid development of unstructured networking in cloud, is accompanied with numerous security as well as privacy problems that must be worked upon (Rong et al., 2013; Bernsmed et al., 2012).For instance, it is crucial that the level of security provided by the cloud service provider is equivalent to or superior to the security offered by the users' traditional Information Technology (IT) environment if they are outsourcing their applications and data in order to use the unstructured network (Li et al., 2013; Rahulamathavan et al., 2013).

Unstructured computing is more popular, which makes unstructured-based networks more appealing to malware assault (Sgandurra and Lupu 2016; Banafar and Sharma, 2018).A particular approach to reduce the risk of privacy violation in the course of data generation is either by controlling the access to data or by falsifying data (Xu et al., 2014). The conventional security approach to protect data can be listed into four types which are file level data security schemes, database level data security schemes, media level security schemes, as well as application level encryption schemes (Hongbing et al., 2015). The conventional mechanism in protecting data security (Cao et al., 2014) as well as privacy (Soundararajan et al., 2014; Singla and Singh, 2013) for existing storage, storage architectures (i.e., direct attached storage, network attached storage, as well as storage area network) is a very dynamic research area that must not be ignored, nevertheless, that may not be explicitly relevant to this research to an extent. In some circumstances, it is not possible to prevent access of sensitive data.

Different intrusion detection techniques have been deployed to address the security challenges highlighted above in the unstructured network environment. They include signature-based detection or misuse detection, anomaly detection, heuristic detection, etc. Misuse detection techniques maintain rules for known attack signatures or bit sequence. These rules can be derived either by using the knowledge based systems which contain database of known attacks signatures or by using machine learning algorithms to determination the behavioral profiles of the users based on known suspicious activities (Preeti et al., 2017). Anomaly detection systems detect anomalies from the expected behavior of the system. Any deviation from the expected behavior is signaled as anomalous (Monowaret al., 2014). These existing techniques have certain drawbacks such as failure to detect and prevention of zero-day malwares, rootkits, high overhead computation as a result of periodic capturing and update of the knowledge-base and many more.

Our proposed techniques tend to tackle these lapses. One of such techniques is the virtual machine introspection (VMI). The basic principle behind this technique is that it performs the introspection of programs running in a virtual machine (VM) to determine any malicious program change or execution of some abnormal or malicious code (Hebbal et al., 2015). Hypervisor Introspection (HVI) is another technique we will harness to the first one mentioned earlier. Its security and preventive approach depends mainly on the hardware assistance to perform introspection of hypervisor and the host OS kernel states and detect various attacks such as hardware attacks, rootkit attacks, and side channel attacks. Lastly, Policy Management in the Cloud is needed, that is why we intend to use the services of Mandatory Access Control to restrict unauthorized usage of sensitive information across the network. Existing techniques lack maintenance of reliable policies which in turn lead to deterioration of the efficiency of security

tools. Cloud administrator needs to set up policies for IDS where ever it is installed and also update it from time to time. The manual update of knowledge base architecture is time-consuming and as well can be prone to errors.

2.0Review of Unstructured Network

Unstructured network like Gnutella and others have no restriction on where data is stored and the network topology is arbitrary. In a loosely organized system like Freenet (Masood et al., 2018) other unstructured networks, the overlay organization and the statistics position are not precisely secure. In Freenet, mutually, the overlay topology and the statistics position are established on the basis of suggestion(s).Due to large costs and lengthy implementation durations, it is occasionally impossible to deploy a system in a real context. In this situation, building a model and studying the operation of the system would be more appropriate. A genuine system is to be replaced and sped up by modeling. The biggest problem in the current world is cyberattacks. It is crucial to comprehend harmful object behavior if we are to solve this issue. When it comes to computational modeling, this is crucial

Modeling can be used to comprehend and defend against malicious objects like viruses, worms, Trojan horses, spam, and specific technologies like instant messaging, bots, and phishing. It is possible to identify situations when certain essential presumptions, such as the data's lifetime, the time of collection, and the number of connections, can be applied. Calculations that need to be made and mathematical equations that indicate the current condition of the information should be used to estimate malicious items.

We are unable to predict the real-time of the subsequent attack since the attacks on the host VM are entirely random. Modeling the likelihood of the attack can only be done using probability principles. To express the attack time of the stochastic variable x_i (i = 1,2,3,…) the probability ofx_i is expressed by P(x_i). Where $n_i$is the number of attacks from a certain source and N is the total number of attacks.

$$\mathrm{P}(x_i) = \frac{n_i}{N} \tag{1}$$

Here, considering P(x_i) as a set of numbers, $\int_R \mathrm{P}(x_i)dx = 1$, that is, the area under the curve is 1

We can express $\sum_{i=1}^{\infty} \mathrm{P}(x_i) = 1$, which is the probability of density function. There can also be a probability distribution function that gives the probability of small or equal stochastic attacks to a given value.

$$\mathrm{f}(x_i) = \sum_{x_i \leq x} \mathrm{P}(x_i) \tag{2}$$

The stochastic system can be examined using a variety of metrics of the probability function, including mean, mode, median, and standard deviation. Both linear and non-linear characteristic equation models are possible. Partial differential equations can be used to represent the non-linear system.

Let's assume that the malicious object has P spread feature due to diverse other factors such as A, B, C,…. In this case, it can be body forth by P = (A,B,C,…). From here, taking the 1st and 2nd derivatives, respectively;

$$\text{Velocity } \frac{\partial p}{\partial t} = \frac{\partial f(A,B,C,\dots)}{\partial t} \quad (3)$$

$$\text{Acceleration } \frac{\partial^2 p}{\partial t^2} = \frac{\partial^2 f(A,B,C,\dots)}{\partial^2} \quad (4)$$

are calculated. The imitated conclusions acquired by using particular approximation techniques mentioned down can be used to complement the simulation-created data as well as to verify it.

The virtual CPU, memory, and disk contents are among the internal states and events that are observed and analyzed by the VMI tools. Our trampoline, acting in the capacity of an analyst, is granted access to that level of privilege by the hypervisor, which is in charge of managing interactions between hardware and the operating system (OS). It makes it possible for the hook to observe these actions from outside the virtual machines. Any attempt to modify a virtual machine after the hypervisor has established it will result in an alert being sent to the system administrator or tenant administrator at the beginning of execution.

3.0 Algorithm for the VMI

Pseudocode

Function: detect inconsistent behavior over the life of the OS, not just between training and classification like other machine learning algorithms do
Pass in : guest OS information
Direct the VMI applications to inject hooks (In-VM agent) into the host of guest OS
Pass Out: display all information's that meet the set pre-defined access policies the proposed system is built upon
End Function

Algorithm

Input: (Host List, VMI List).
Output: (Map_to_VMI).
Step 1: Start
Step 2: Input: Host List, VMI List
Step 3: Output: Map_to_VMI
Step 4: Sort VMI List in Order to Increase CPU Utilization
Step 5: For VMI in VM list do
Step 6: For host in Host list do
Step 7: If host has enough resources to VMI Then
Step 8: Map_host ←Host
Step 9: If Map_host! = Null Then
Step 10: And (Map_host, VMI) to Map_to_VM
Step 11: Return Map_to_VMI
Step 12: Stop

Fig. 1: virtual machine introspection (VMI)

The VMI detects inconsistent behavior within the OS, not just between training and classification like other machine learning algorithms do. VMI tools attempt toreconstruct a range of information, including the set of running processes, sensitive file contents, network sockets, disk contents, network traffic within its boundaries through memory introspection, where the hypervisor draws inferences about guest behavior from the contents of memory and CPU registers.

Hypervisor Introspection (HVI)

One of the essential elements of Virtual Machine Introspection (VMI)-based intrusion detection systems is the hypervisor, which offers a privileged and secure location for the proposed system's VMI tools to function. Due to the VMI-based presumption, that the hypervisor is a more reliable and secure area within the cloud environment. Our solution uses the Hypervisor Introspection (HVI) technique, which examines data structures, memory areas, hypercalls, control flow data, non-control flow data, etc. connected to the hypervisor and also works to avoid and identify hypervisor attacks.

Algorithm for the HVI

Pseudocode	**Algorithm**
Function : secure and prevent a compromised hypervisor to launch further attacks on VMs running over it	Input: (X_1, X_2, …X_n,). Output: (X_2, is greater than B>C). Step 1: Start Step 2: Read X_1, X_2, …X_n, X_1=eliminate A X_2 = Eliminate B X_3 = Eliminate C
Pass in : HVI to eliminate possible attracts from reaching the VMS	Step 3: Input: A, B, C
Pass Out: display all eliminated attacks End Function	Step 4: A=hardware attacks Step 5: B= rootkit attacks Step 6: C= Side channel attacks Step 7: If A> B> C Step 8: Print X_1 Step 9: Else Step 10: Print X_2, is greater than B>C Step 11: Stop

Fig. 2: The hypervisor introspection (HVI)

The hypervisor introspection make sure the hypervisor is secure and stop any compromised hypervisor from attacking the VMs using it. Using hardware-assisted virtualization-enabled technologies, the hypervisor can be observed. To perform introspection of the hypervisor/host OS kernel states and identify various threats, including hardware assaults, rootkit attacks, and side channel attacks, the Hypervisor Introspection (HVI) based security technique primarily relies on hardware assistance.

MAC-based hypervisor protection

While the guest VMs will be introspected through protectedin-VM monitoring, the security of the hypervisor will be enforced through mandatory access control. When a VM is created, it will

be assigned with the access policies which reflect its intended purpose. Requests made by the guest VM will be granted by looking up to the system admin security policies defined in the hypervisor protection module running in the hypervisor. Any attempts by the guest VM to violate the assigned access policies will be notified to the control monitor running within the controlled VM.

Detailed working of the MAC technique

The MAC technique deployed in the proposed system typically consists of three components, namely, reference monitor, enforcement hooks, and access control policies.

Reference Monitor

It is responsible for monitoring all resource access requests in the host domain. It uses the access enforcement hooks to intercept any resource access request and deliberates about the requestaccording to a set of pre-defined access policies set out by the system admin across the unstructured network.

Algorithm for the Reference Monitor

Pseudocode	**Algorithm**
Function : Monitoring all resource access requests in the host domain Pass in : Resource access request Direct the hooks to intercept any resource access request and deliberates about the request according to a set of pre-defined access policies set out by the system admin across the unstructured network Pass Out: All resource access requests that meet the set pre-defined access policies. End Function	Input: (A, B). Output: (C, B is greater than A). Step 1: Start Step 2: Read A, B Step 3: Initialize monitoring in A, B, C Step 4: A=hook 1 Step 5: B=Resource access request 1 Step 7: C= Resource access request 2 Step 8: If A>B>C Step 9: Print B Step 10: Else Step 11: Print C, B is greater than A, Step 12: Stop

Access Enforcement Hooks

They are invoked when a guest VM makesresourceaccessrequest.Theyinterceptanyresource access from the VMs and pass the intercepted information to the reference monitor.

Algorithm for Access Enforcement Hook

Pseudocode	**Algorithm**
Function : intercept any resource access from the VMs and pass the intercepted information to the reference monitor	Input: (VM makes resourcesaccess). Output: (a, is lesser). Step 1: Start Step 2: Initialize the access

Pass in : Resource access request Direct the guest VM to make resource access request and deliberates about the request according to a set of pre-defined access policies set out by the access enforcement hooks Pass Out: All resource access requests that meet the set pre-defined access policies. End Function	enforcement hooks counter 0 to 10 Step 3: loop Step 4: Read VM makes resources access a=Resource access request 1 b= Resource access request 2 accumulate input (a,b) Step 5: While counter 0 to 10 Step 6: Print b Step 7: If a>10>b Step 8: Else Step 9: Print a, is lesser, Step 10: Stop

Methodology of the Proposed System

In order to analysis and design the proposed system we employed the services of object oriented programming (OOP), with some unified modeling language (UML) behavioural diagram to show the objects and methods used in the design.In order to analyze the proposed design, usecase diagrams and sequence diagram are used to illustrate the design in order show the behavour and components of the proposed system. The unified modeling language used shows the various parts that made up some of the internal and external critical design of the system. This programming paradigm is deployed because the components of the system are made up of objects which involve methods and classes.

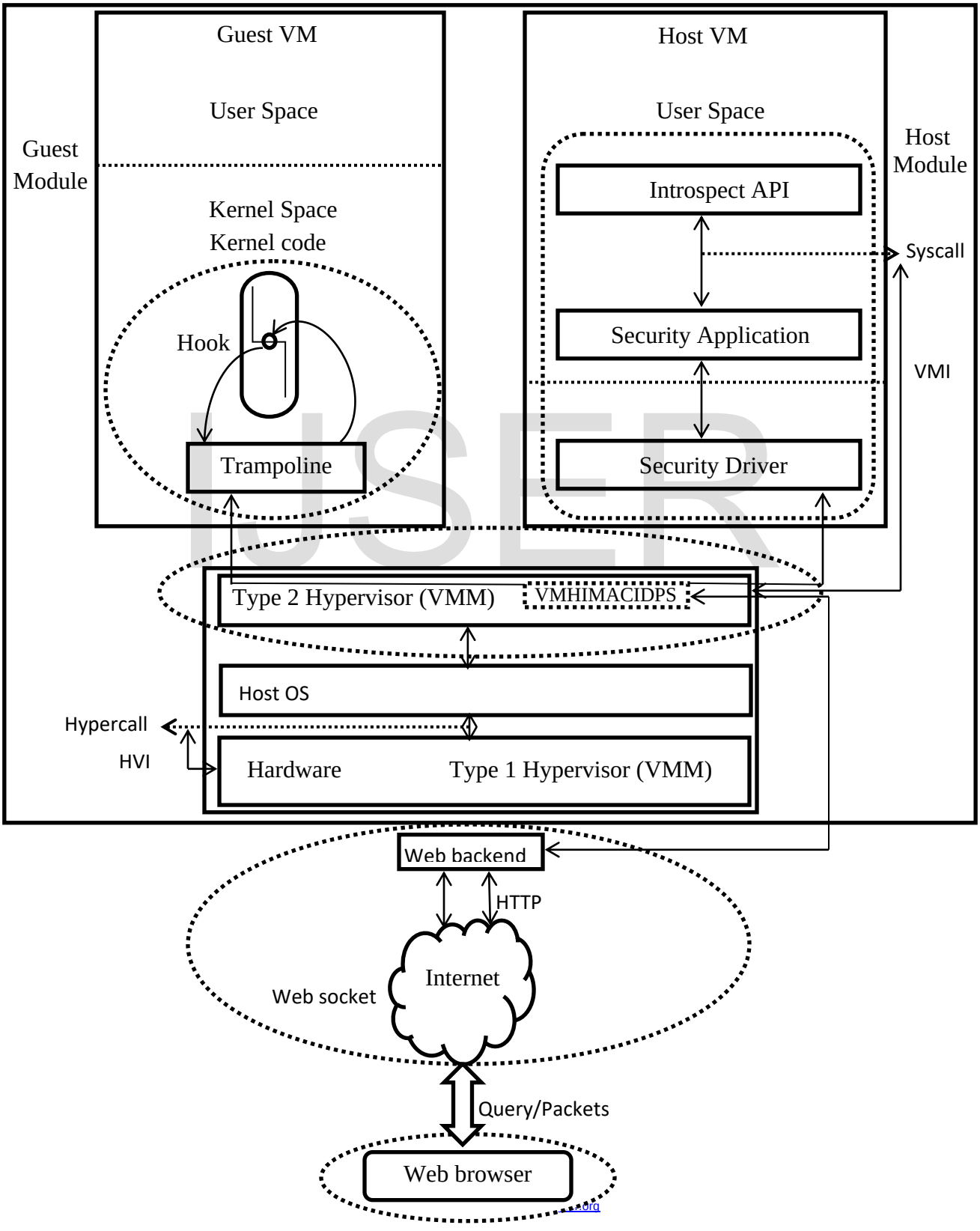
Detailed description of some ofthe components in the proposed model are shown in figure 3

User: sends request through the web browser to any host machine or server and the server or host machine responds by providing the requested service if the request is legitimate. The users' programs run on top of the VMOS, and communicate directly with the secure server via the trusted path.

Web Browser: it enables the passage of packets and the access to resources in participating virtual machines in the network. When a user requests for resources, the web browser retrieves the necessary content from the appropriate host machine or server if the request fulfil the restriction imposed by the VMHIMACIDPS mechanism and then displays the resulting fileson the user's device.

Web Socket: it is used for real-time communication between theweb backend and the web browser where packets are sent across the network having various virtual machines.

Web Backend: is design upon VMHIMACIDPS mechanism for analyzing virtual machine state, hypervisor inspection, access control to system resources, and gathering data about system events.


Guest VM
User Space
Guest Module
Kernel Space
Kernel code
Hook
Trampoline
Host VM
User Space
Host Module
Introspect API
Syscall
Security Application
VMI
Security Driver
Type 2 Hypervisor (VMM)
VMHIMACIDPS
Host OS
Hypercall
HVI
Hardware
Type 1 Hypervisor (VMM)
Web backend
HTTP
Internet
Web socket
Query/Packets
Web browser

User

Fig. 3: Proposed Model

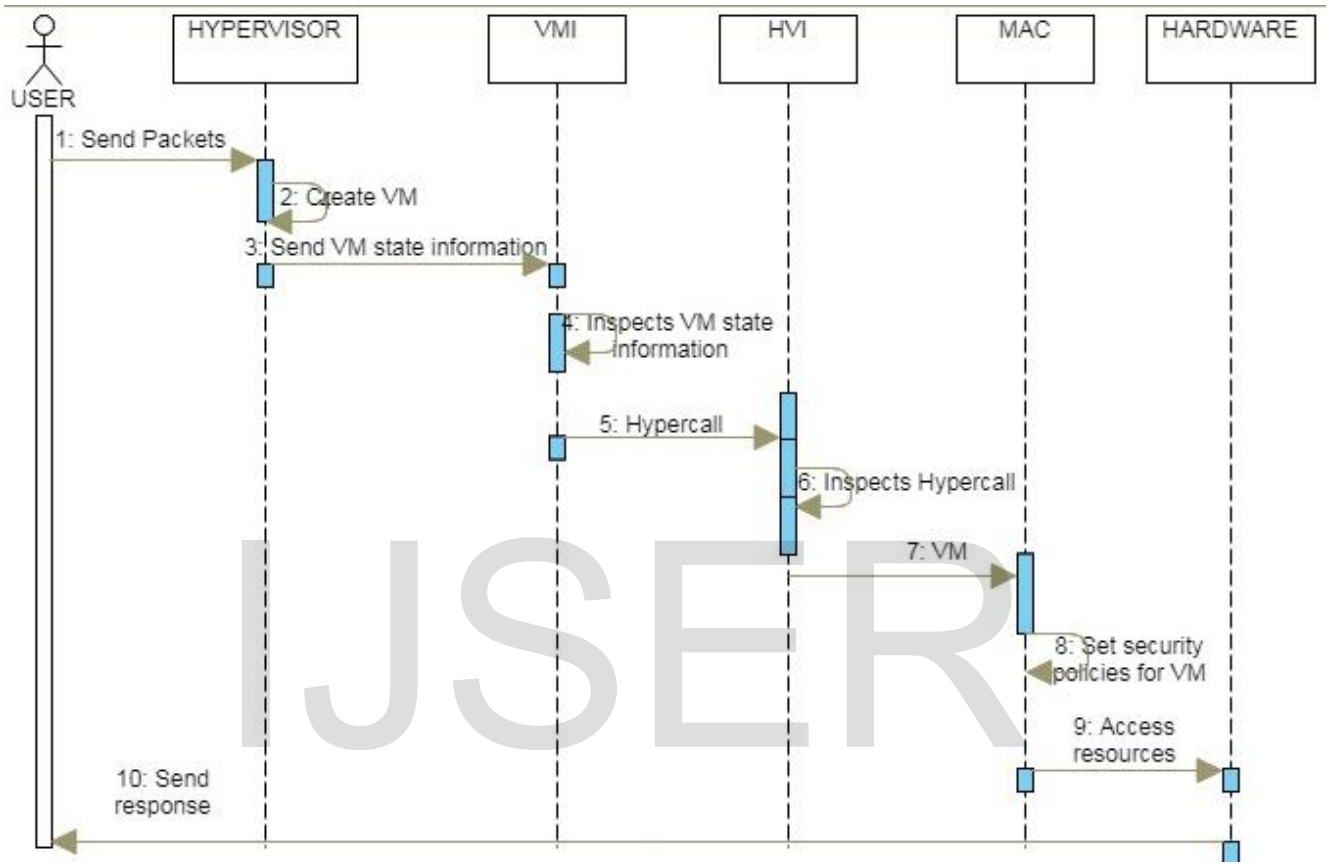


Fig. 4: Sequential diagram of the proposed system

Sequential Diagram

Sequence diagrams are sometimes called event diagrams or event scenarios. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. The sequential diagram in fig. 2 shows the events that take place when packet is sent across the unstructured network. The hypervisor create a virtual machine for the packets and then, move the VM across the appropriate host VM. The HVI introspects the hypercalls from the hypervisor to the hardware if there is abnormally when a Guest VM intend to access the underlying host hardware. The hypervisor also sends the created VM state information to the VMI tools for recording and inspection when the Guest VM commences execution, if there is any deviation of its initial execution request. the MAC components set certain security policies for the Guest VM, if the condition is met, then, the unprivileged VM can been given access to the resources within the hardware or elsewhere in the network, otherwise access will be deny.

Therefore, security monitoring working mechanism of the proposed system can be viewed as

M(S, P) → {True, False}(5)
Where M = denotes the security enforcing mechanism
S = denotes the current systemstate, and
P = denotes the predefined policy.

If the current state S satisfies the securitypolicy P, then it is in a secure state (True), and if M is an online mechanism, it can allow continued execution. Otherwise, it is insecure (False); an attack is detected, andM can halt the execution (for prevention) or report that there is an attack instance.For some of the existing systems that use classification mechanism to detect attack, P is the signatures of viruses; if *M* identifies that there is any running process orsuspicious file having one of the signatures defined in *P*, the system will raise an alarm. The proposed system (VMHIMAC) which is a system call–based intrusion detection system, S,can denote the currentsystem call and P can denote the correct state machines for S; if M identifies thatS deviates from P, then it can raise an intrusion alert. The system also has a security tools (e.g., security reference monitors) for its security monitoring efficiency.In the experiment we process the system call dataset and prepare the feature vectors using stand and modified vector space representation in the simulation model. These feature vectors are used as detection prototype. In the detection level, we collect the system call traces from the introspection and prevention tools running on each node.

4.0 Results and Discussion

Algorithm	FP Rate 1	FP Rate 2	FP Rate 3	FP Rate 4	FP Rate 5	FP Rate 6
HI	0.998	0.986	0.985	0.989	0.902	0.902
MAC	0.176	0.451	0.452	0.452	0.452	0.452
VMI	0.176	0.451	0.452	0.452	0.451	0.451
VMHIMAC	0.073	0.005	0.065	0.052	0.052	0.052
Term-Size	1	2	3	4	5	6

Table 1. values for each algorithms showing their respetive FP Rate as the term-size increases in

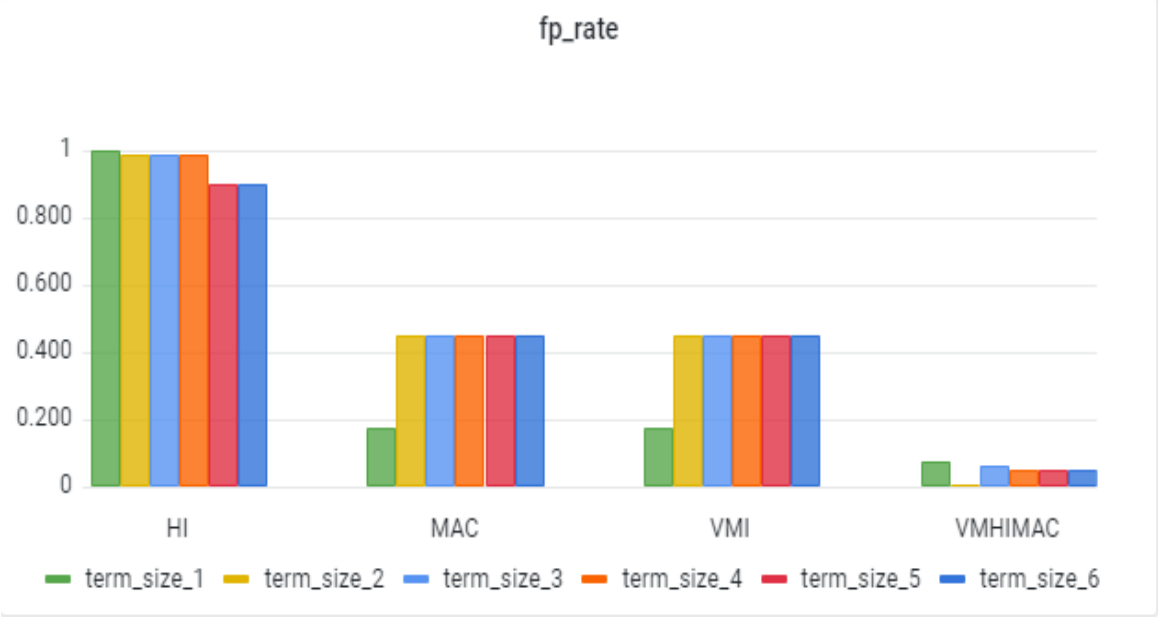


Fig. 5: Performance Metric for each System with respect to FP Rate

F P Rate: The specificity of the proposed system (VMHIMAC) to its closest revival (MAC and VMI) as regards their term-sizes ranging from 1, 2, 3, 4, 5, and 6 are 7%, 5%, 6%, 5%, 5%, and 6% respectively for our system and 45%, 45%, 45%, 45%, 45%, and 45% for MAC and VMI, since they have same FP Rate values. Our system outperforms other existing systems with the considerable lower false positive rate recorded.

Table 4.13 values for each algorithms showing their respetivePrecision as the term-size increases

in Fig. 4.14

Algorithm	Precision1	Precision2	Precision3	Precision4	Precision5	Precision6
HI	0.925	0.911	0.913	0.91	0.92	0.92
MAC	0.984	0.96	0.96	0.96	0.96	0.96
VMI	0.984	0.96	0.96	0.96	0.96	0.96
VMHIMAC	0.994	0.996	0.998	0.998	0.998	0.998
Term-Size	1	2	3	4	5	6

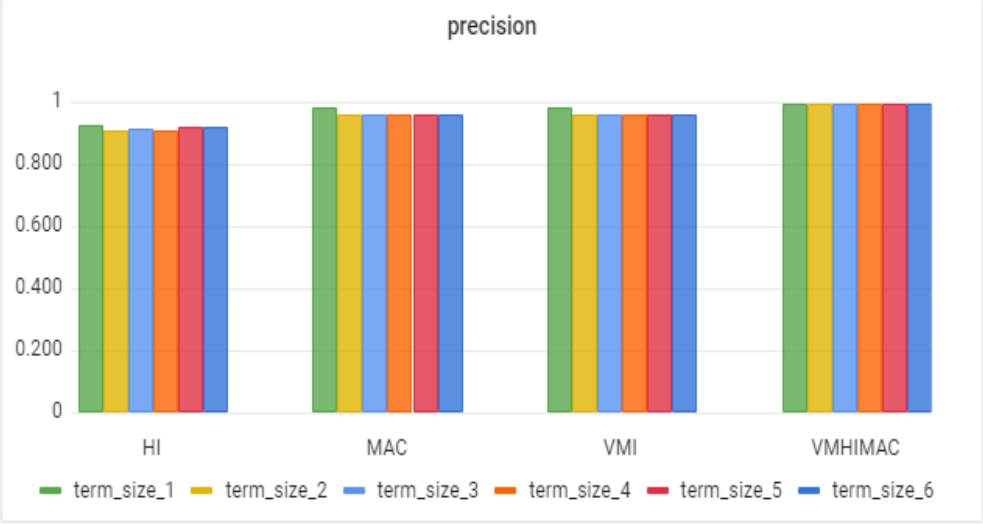


Fig. 6: Performance Metric for each System with respect to Precision

Precision: As regards precision the proposed system (VMHIMAC) have a better rating than its closest revivals (MAC and VMI) as regards their individual term-sizes ranging from term-size 1, 2, 3, 4, 5, and 6,which are 99%, 99%, 99%, 99%, 99%, and 99% respectively for our system and for MAC and VMI they have same precision values of 98%, 96%, 96%, 96%, 96%, and 96%, signifying that our system outshinesthe existing systems with a very reasonable precision advantage.

5.0 Conclusions

In the course of this study, we found out that the existing systems find it difficult to track down attackers because of low cognizance in their technique and reactive way of resolving poisonous program in a network. But, in ours (VMHIMAC), we developed an appropriate policy and awareness to limit such malicious activities while malicious nodes are identified proactively.The basic principle behind one of our techniques the HVI is that it performs the introspection of programs running in a VM to determine any malicious program change or execution of some abnormal or malicious code. The different approaches to the VM introspection such as guest-OS hook based, VM state access based, kernel debugging based, interrupt based and hypercall authentication based tend to bridge the semantic gap in interpreting the low-level information available at a VM to high level semantic state of a VM. Why the Hypervisor Introspection (HVI) based security approach mainly depends on the hardware assistance to perform introspection of hypervisor/host OS kernel states and detect various attacks such as hardware attacks, rootkit attacks, and side channel attacks. The mandatory access control (MAC), access to resources is controlled through access policies to avoid unauthorized usage of sensitive information. The VMI helps in identifying high level view modification in VM by suspicious code during execution.

References

1. Banafar, H. and Sharma S. (2018).Secure Cloud Environment Using Hidden Markov model and rule based generation. International Journal of Computer Science and Information Technologies, 5(3), 4808–4841.

2. Preeti, M., Emmanuel, S. P., Vijay, V. and Udaya T. (2017). Intrusion detection techniques in cloud environment: A survey. Journal of Network and Computer Applications. (77) 28-47.

3. Bernsmed, K., Jaatun, M. G., Meland, P. H. and Undheim A. (2012). Thunder in the clouds: Security challenges and solutions for federated clouds. In IEEEFourth Intl Conf. Cloud Computing Technology and Science (CloudCom), 113–120.

4. Cao, N., Wang, C., Li, M., Ren, K. and Lou W. (2014).Privacy-preserving multi-key words ranked search over encrypted cloud data.IEEE Trans. Parallel Distrib. Syst., 25(1), 222-233.

5. Hebbal, Y., Laniepceand, S. and Menaud, J.-M. (2015). Virtual machine introspection: Techniques and applications. In: 10th International Conference on Availability, Reliability and Security (ARES). IEEE, 676-685.

6. Hongbing, C., Chunming, R., Kai, H., Weihong, W. and Yanyan, L. (2015). Securebig data storage and sharing scheme for cloud tenants.ChinaCommun., 12(6): 106-115.

7. Kuyoro, S. O., Ibikunle, F. and Awodele, O. (2011).Cloud Computing Security Issues and Challenges International Journal of Computer Networks (IJCN). 3(5), 247-255.

8. Li, F., Rahulamathavan, Y., Rajarajan, M. and Phan, R. C.-W. (2013). Low complexity multi-authority attribute based encryption scheme for mobile cloud computing. In Proc. IEEE 7th Intl Symp. Service Oriented System Engineering (SOSE), 573–577.

9. Masood, S., Shahid, M. A., Sharif, M. and Yasmin, M. (2018). Comparative Analysis of Peer to Peer Network. International Journal of Advanced Networking and Applications. Vol 9. (4). 3477-3491.

10. Monowar, H. B., Bhattacharyya, D. K. and Kalita, J. K. (2014). Network Anomaly Detection: Methods, Systems and Tools. IEEE Communications, Surveys and Tutorial, Vol. 16, No. 1.

11. Rahulamathavan, Y., Phan, R. C.-W., Veluru, S., Cumanan, K. and Rajarajan M. (2013). Privacy-preserving multi-class support vector machine for outsourcing the data classification in cloud. IEEE Trans. Dependable and Secure Computing, (99): 1–1.

12. Rong, C., Nguyenand, S. T. and Jaatun, M. G. (2013). Beyond lightning: A survey on security challenges in cloud computing. Computers & Electrical Engineering.39(1): 47 – 54.
13. Saroiu, S., Gummadi, P. K. and Gribble, S. D. (2001). A measurement study of peer-to peer file sharing systems. In Electronic Imaging.International Society for Optics and Photonics.156-170.
14. Keiko, H., David, G. R., Eduardo, F-M. and Eduardo B. F.(2013). An analysis of security issues for cloud computing. Journal of Internet Services and Application. 5.
15. Sgandurra, D. and Lupu, E. (2016).Evolution of attacks, threat models, and solutions for virtualized systems. ACM Computing Surveys, 48(3), 46.
16. Singla, S. and Singh, J. (2013).Cloud data security using authentication and encryption technique.Global J. Comput. Sci. Technol., 13(3). 2232-2235.
17. Soundararajan, O. M., Jenifer, Y. Dhivya, S. and Rajagopal, T. K. P. (2014).Data security and privacy in cloud using RC6 and SHA algorithms.Netw.Commun. Eng., 6 (5). 202-205.
18. Subashini, S. and Kavitha V. (2011). A survey on security issues in service delivery models of cloud computing. Journal of Network and Computer Applications (34) 1-11.
19. Kevin, H., Murat, K., Latifur, K. and Bhavani, T. (2010). Security Issues for cloud computing, International Journal of Information Security and Privacy, 4(2). 39-5.
20. Wayne, A. J. (2011). Cloud Hooks: Security and Privacy Issues in Cloud Computing. Proceedings of the 44th Hawaii International Conference on System Sciences, 1-10.
21. Xu, L., Jiang, C., Wang, J., Yuanand J. and Ren, Y. (2014). Information security in big data: Privacy and data mining, inIEEE Access, vol. 2, 1149-1176.