Virtual Machine Introspection based Intrusion Detection System using Software Defined Networks

Sujay Vaishampayan
Arizona State University
ssvaisha@asu.edu

Tejas Khairnar
Arizona State University
tkhairna@asu.edu

Gaurav Patil
Arizona State University
gauravpatil@asu.edu

Project Mentor
Ankur Chowdhary
Arizona State University

***Abstract*— Virtual Machine Introspection (VMI) involves monitoring the state and contents of a Virtual Machine in real-time in order to analyze it for any discrepancies from it's regular working. This takes place without the guest system having any knowledge of these tasks being performed in the background. This technique can serve as an Intrusion Detection System (IDS) which runs on the host machine. Virtual Machines (VM's) are created using the native virtualization method embedded, QEMU-KVM, within the Linux kernel. QEMU directly uses the virtualization feature of the respective Intel or AMD processor without going through any third party software. The main aim of our project is to use Virtual Machine Introspection as an IDS on the host machine and analyze the processes that are running on the guest system. The tasks involved in this project would be:**

i. **Installation of QEMU/KVM, libVMI and their dependencies**
ii. **Installation of virtual machines and interfacing them with libVMI**
iii. **Deploying scripts that will grab the process list, analyze the memory, etc. of the guest system and report all the information on the host system.**
iv. **Successfully deploying host machines in a SDN Environment using OpenFlow Protocol and OpenDaylight Controller.**
v. **Deploying scripts to analyze the information received for abnormalities and classifying them as intrusion attempts or not**
vi. **Deploying a live attack and performing introspection using our VMI based IDS**

The first phase of the project will be to set up VM's using the KVM Hypervisor and successfully interface it with the libVMI library for VMI. The second phase will be to write our own script in order to grab data from the guest system. The third phase will involve analyzing the information received for meaningful patterns and classifying them as malicious or not. The final phase of our project will be to employ multiple host machines on a Software Defined Network (SDN) using OpenFlow Protocol and OpenDaylight Controller. Thus, we will be able to perform Intrusion Detection on multiple host systems with their respective guest systems using a single controller.

Keywords — Virtual Machine Introspection, Hypervisor, Intrusion Detection System, Software Defined Networking

I. INTRODUCTION

In this project we are trying to implement Virtual Machine Introspection based Intrusion Detection System using Software Defined Networking approach. Our project can act as a solution security over cloud computing. In order to implement virtual machine introspection we are going to use the popular libVMI library[8]. This library has in-built functions in order to access the virtual machine's memory and system maps. Intrusion detection system can be implemented by introspecting these system and memory maps of the virtual machines which can signal the host machine about malicious activities. The outcome of this project is building a Unix based package which can be installed on

OpenDaylight[9] or floodlight controller to perform VMI based intrusion detection.

II. System Models

A. System Model

The system model is specififed in the file attached with this proposal.

B. Software

Software's required in order to implement the project:

1. libVMI[8]
2. QEMU/KVM Hypervisor[9]
3. openvSwitch[10]
4. OpenDaylight Controller
5. Drakvuf[11]
6. Volatility[11]
7. Rekall[11]

Virtual Machines used to introspect:

1. Windows 7
2. Ubuntu 14.04
3. Kali v2.0

C. Security Model

The virtual machines which we will be introspecting are/maybe susceptible to various cyber security threats. These machines have a bridged access to the internet via the host machine hence there is a high possibility that these machines can be a victim to a cyber hack. Hence, we will be deploying an ethical hack which we will try to detect using our Intrusion detection software on the host machine. The attack source will be a generic rootkit which when triggered might start a remote server for the attacker to perform remote access. These rootkits are basically a shell script or executable which starts running in background once triggered. Our aim through virtual machine introspection is to detect this malicious running process or a wider scope will be detecting the download of such kind of rootkits.

III. Project Description

The Project description can be summarized as follows:

A. Project Overview:
   In order to complete this project we have divided the work done into 13 tasks which are to be performed in the listed fashion.

B. Task 1: Performing required installation:
   To start working with on the project we have to perform some basic installation on our host machine on which introspection is to be performed. This includes installation of the mentioned software's, various dependencies, which includes linux packages which supports QEMU/KVM hypervisor and deploying virtual machines to introspect.

C. Task 2: Learning KVM/QEMU Hypervisor:
   In order to introspect it is important to learn how virtual machines work on virtual technology embedded in Intel processors. Hence learning and understanding of hypervisor functionality is an important part of this project.

D. Task 3: Understanding libVMI:
   The introspection which we are talking about can be performed using a library written specially for enthusiasts who are eager to learn about virtual machine introspection and using it for developing new software's. Hence learning to use libVMI API is one of the most important part of this project.

E. Task 4: Learning virtual machine introspection using libVMI:
   libVMI package contains various other packages which helps the user to perform introspection. It also has a python wrapper API which might be handy to write the IDS system.

F. Task 5: Writing code to perform introspection of Virtual Machines
   This tasks is self-explanatory as its name suggests. Our code will be running on host machine on which other virtual machines are running. This task will take a while to be complete and hence consumes maximum amount of time.

G. Task 6: Deploying our code on host machine running KVM hypervisor:
   This task is pretty much similar to the above task.

H. Task 7: Understanding SDN Environment:
   The aim of this project is to use VMI as IDS on software defined networks. Hence in order to deploy this software which will perform this intrusion detection we need to learn on what network this is supposed to be deployed. This task will containing gaining knowledge about various newly released software's which replaces the generic networking devices such as routers, switches.

I. Task 8: Deploying SDN Environment with our project/host machine:

We need to integrate our project into a software defined network which would consists of a central SDN controller. Through this controller various other host machines can be controlled on which different virtual machines are running. This is more optimistic goal of this project.

J. Controlling different Host machines running virtual machines using OpenDaylight Controller:
ODL controller is a recently released software used to controller various machine on a software defined network. We will be using this software to control our whole software defined network.

K. Understanding Host & Network based Intrusion Detection System:
This is something basic which is to be known. Knowledge about how host based or network based IDS works will come in handy.

L. Using the introspection code introspect for malicious behavior similar to an IDS:
The introspection code output can be used to detect malicious behavior on any virtual machine. Ideally an IDS throws an alert based on the signatures/rules provided to it. Signature based attack detection is also an optimistic goal to achieve through this project.

M. Deploying a live attack and introspecting the same using our VMI based IDS:
We will try downloading a rootkit software on any virtual machine and using our deployed IDS try to detect about this malicious download.

N. Project documentation:
This is the final phase of this project where we will work on documenting our work in the form of Github repository or a word document. A project poster will also be made in order to explain our project in brief.

A. Project Task Allocation

The workload will be divided equally among all team members. This is because we believe that academic projects the most vital source of learning in a coursework. Therefore, all team members will try to perform each task mentioned, except the coding part which can be divided according to the functionality to be implemented.

B. Deliverables

To implement VMI based IDS application in the form of a graphical user interface and to deploy an attack on the network to test the functionality of the IDS.

C. Project Timeline

We have prepared a gantt chart for project timelines and deliverables. This gantt chart is attached at the end of this documents.

IV. RISK MANAGEMENT OF THE PROJECT

The major risk in implementing this project is performance of KVM hypervisor with libVMI API. According to the collected resources major work of virtual machine introspection is done using Xen hypervisor. A failover task will be to use Xen hypervisor to achieve the second phase of this project which we will be successfully performing virtual machine introspection.

V. CONCLUSION

Virtual Machine Introspection is a novel way of implementing an Intrusion Detection System on a host machine. It's more efficient than most IDS' as it can directly interact with the guest system without the knowledge of the user. Moreover, implmenting this system within an SDN could be one of the most efficient ways of performing intrusion detection on multiple systems without the installation of dedicated software on each machine. Using VMI in a IDS inside a SDN has many potential applications such as being deployed on a large scale in enterprises, in restricted testing environments, monitoring a guest system's behaviour to a particular malware without actually affecting the host system and many more.

ACKNOWLEDGMENT

We would like to thank our project mentor Mr. Ankur Chowdhary for his guidance on how to proceed and his assistance in obtaining the necessary documentation in order to understand VMI as well as the libVMI API. We would also like to thank our professor Dr. Dijiang Huang for educating us with the basic concepts related to the academic course CSE 548 – Advanced Computer Network Security.

REFERENCES

[1] *A Virtual Machine Introspection Based Architecture for Intrusion Detection, Tal Garfinkel & Mendel Rosenblum; NDSS, 2003*

[2] Lecture by Tamas K Lengyel & Thomas Kittel, 6297, 31th Chaos Communication Congress [31c3] of the Chaos Computer Club, Hamburg, Germany

[3] Understanding the Linux Kernel - Daniel P. Bovet & Marco Cesati
[4] GitHub – Repository: libVMI, Valerio Aimale
[5] https://publish.illinois.edu/assured-cloudcomputing/files/2015/05/041915-Virtual-Machine-Instrospection-Overview.pdf
[6] http://nob.cs.ucdavis.edu/bishop/papers/2009-ares/vmimpl.pdf
[7] https://www.sec.in.tum.de/assets/studentwork/finished/Kittel2010.pdf
[8] www.libvmi.com
[9] Libvmi Setup - Mctrain's Blog, http://ytliu.info/blog/2013/08/04/libvmi-setup/
[10] www.openvswitch.org
[11] www.drakvuf.com
[12] www.opendaylight.org
[13] Write Introspection Tools Using Libvmi – Mctrain's Blog
[14] Virtual Machine Introspection – WikiXenProject.org