Hypervisor Security Assessment

For

Matt Hale

Assistant Professor of Interdisciplinary Informatics

IA capstone

University of Nebraska at Omaha

By

Afnan Albokhari

Jesse Hembree

May 4, 2017

Contents:

Hypervisor

What is a hypervisor?

In 1960s, IBM introduced a virtualization to separate the application from a specific hardware that used to make a Virtual environment. a virtual environment upgrades the resources usage. A virilization separates the resources logically by adding a hypervisor “is a thin software layer that provides abstraction of hardware to the operating system by allowing multiple operating system or multiple instances of the same operating system, turned as a guest, to run on a host computer” (Desai, Oza, Sharma, and Patel, 2013, p. 222). A hypervisor is a layer between an operating system and hardware and it known as Victual Machine Monitor(VMM). Each virtual machine represents the guest machine, and the computer that run a hypervisor represents the host machine. For example, the user runs multiple operating system in one host computer without mixing between the operating system.

A Hypervisor reduces the time of work because of the features that hypervisor provides, the user can be done multiple work at the same time. Moreover, it is a substitute of the actual resources by providing virtual resources and reduces the hardware cost.

Types of hypervisor

A VMM is grouped into two different types, type-1 hypervisor and type-2 hypervisor.

i. Type-1 hypervisor, is a Bare Metal hypervisor that runs directly on the top of the physical host server’s hardware. such as VM ware, Xen, Oracle VM server, and Hyper-V.

ii. Type-2 hypervisor, is a Hosted hypervisor that runs as an application inside the operating system such as Virtual Box, VMware player, and VMware workstation.

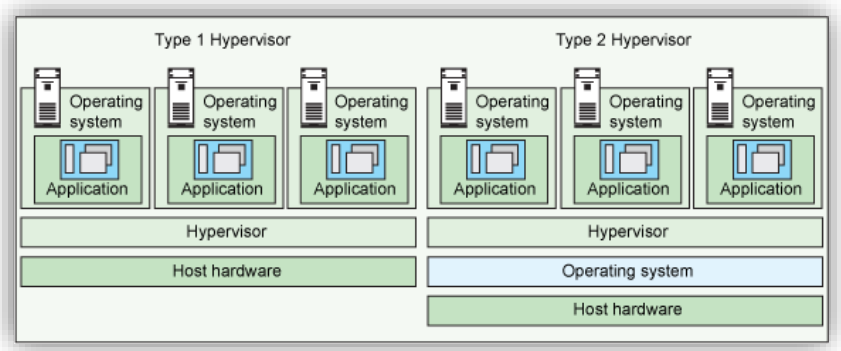


Figure 1 Describes two types of Hypervisor

Type-1 hypervisor -- Xen hypervisor

- Xen hypervisor is a “software system that allows the execution of multiple virtual guest operating systems simultaneously on a single physical machine” (Xen project.org). Xen hypervisor has two important domains.
    1. Dom 0 is a special domain that controls the hypervisor and start other guest virtual machines. it is one of the domain that have a permission to access physical resources. It used to create all quest domains which is Dom U. A Dom 0 “contains backend for network and local disk access request from Dom U that contain front end for accessing underlying hardware” (Ankita et al, 2013, p. 224).
    2. Dom U runs is the other guest machines that Dom 0 started and runs with a special operating system or special virtualization hardware because Dom U does not allow to access the physical resources. The “U” reoffered to unprivileged word which means another guest virtual machine cannot control the hypervisor.

MAC- based Security Architecture

In one of the research, the authors developed an architecture on Xen hypervisor which is a sHype. It is a “security architecture for virtualization environments that controls the sharing of resources among VMs according to formal security policies” (Sailer, Jaeger, Valdez, Caceres, Perez, Berger, Griffin, and Doom, 2005). Also, it supports most of the security function, and the relation with the secure service. A sHype goals based on the requirement that commercial environment provided, and it consists of:

1. Near-zero overhead on the performance-critical path
2. Non-intrusiveness with regard to exiting VMM code
3. Scalability of system management to many machines via simple policies
4. Support for VMM migration via machine- independent policies

These goals are used to achieve “medium assurance” (Sailer et al) by adding a number of lines to the code to the Xen hypervisor.

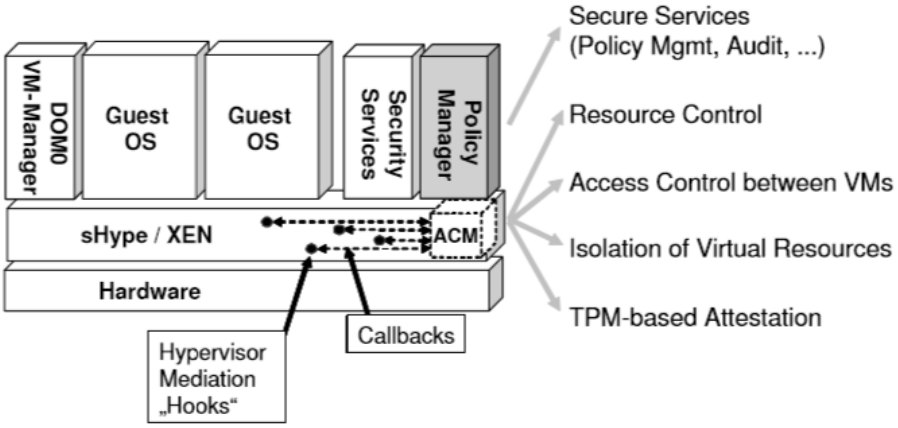


Figure 3. sHype architecture

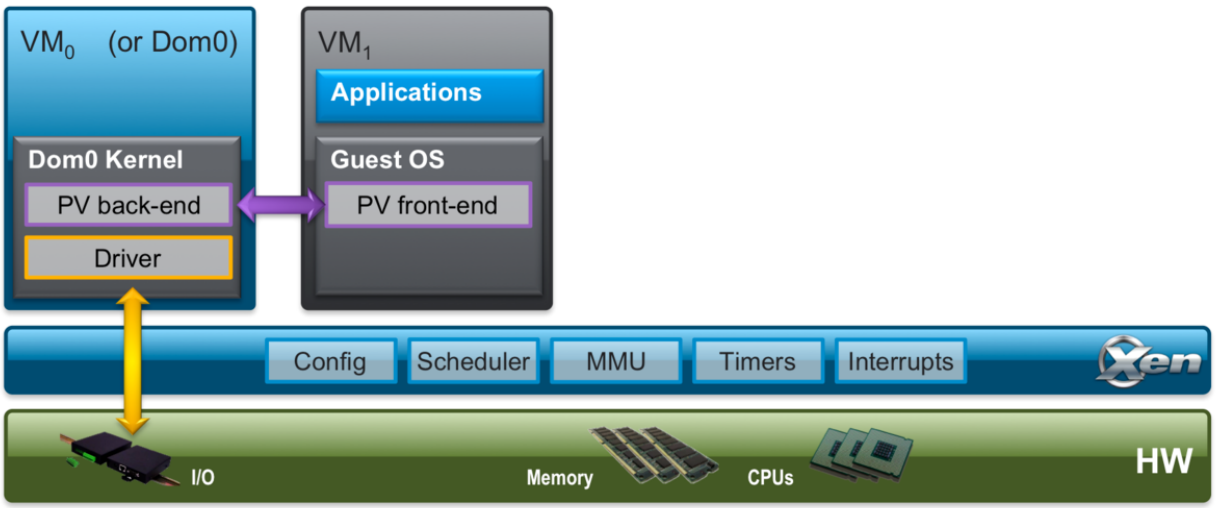
A sHype is creates an isolation between Mandatory Access Control (MAC) and virtual resources. The three major points that sHype designed for are, building on existing isolation properties of virtual resources, using bind-time authorization and controlling access, and enforcing formal security policies.
SHype implements a chinses wall policy to mitigate covert channel and simple type enforcement policies to enforce sharing virtual resources in the xen hypervisor.

Xen Project

Xen project creates a type-1 hypervisor (bara-metal hypervisor). It is supporting two types of virtualization:

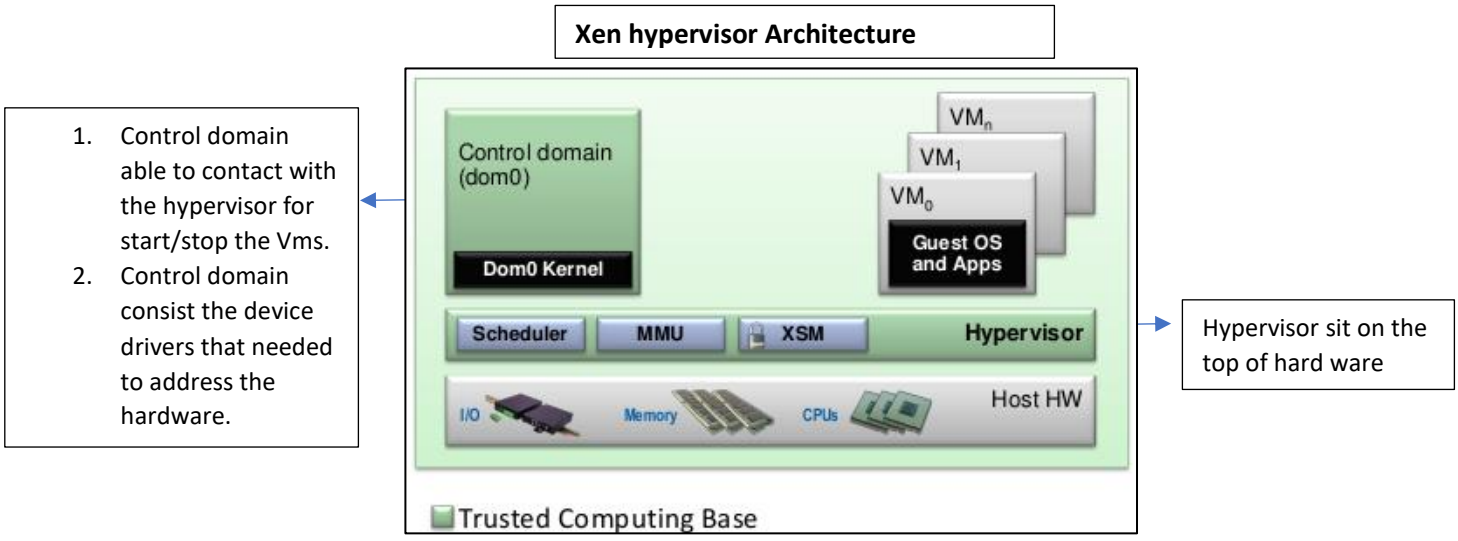
1. Paravirtualization developed by Xen project team; and it is a dynamic virtualization technique. Paravirtulization functions are, disk and network drivers, interrupts and timers, emulated motherboard and legacy boot, privilege instruction, and page table. Moreover, paravirtulization creates a channel to connect the hypervisor with the guest operating system using backend and frontend:
    1. A Backend placed in Dom 0 providing the virtual devices.
    2. A Frontend placed in the guest domain which allow the guest domain accessing the virtual devices.

Xen project Paravirtulization supports old operating system and rerun the operating system in new and efficient hardware. The connection between the operating system and hypervisor is very active which give the paravirtulization a high execution than hardware virtual machine.



Paravirtulization

2. Hardware virtual machine (HVM) called a full virtualization. The purpose of creating a full virilization is to “provide a virtual machine that was functionally nearly identical to a real machine” (Xenproject.org) and it functions are, disk and network devices, interrupts and timers, emulated platform: motherboard/device buses/ BIOS, legacy boot (16 bit – 64bit mode), privilege instructions, and page table.



Control domain creates an interface to the hypervisor which communicates with Xen software to change the hypervisors’ configuration. The device driver attached to the control domain using Linux drivers. Then, the control domain shares the resources with guest operating system using Paravirtualized devices such as disk, SCSI, USP, and PCI.

According to Xenproject.org, Xen requirements are:

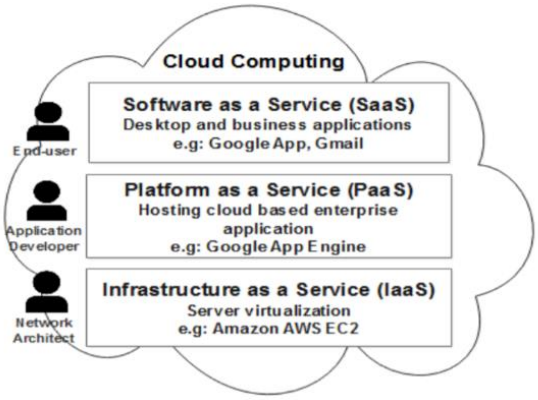
1. 64bit x86 computer with at least 1GB of RAM (this can be a server, desktop or laptop!)
2. (Optional) VT-d or AMD-V support
3. Sufficient storage space for your dom0 and whatever guests you want to install
4. A CD burner + blank CD (you can use a USB but this is not covered here)
5. Internet access for downloading and installing Debian
6. (Optional) Windows Server 2008R2 installation ISO, a trial copy is sufficient
7. (Optional) VNC client for installing HVM domain

In 2005, Linux developed Debian version 3.1 which is one of the operating system that support Xen project hypervisor; and it is simple and powerful. For more information about installing Debian visit here.

Hypervisor and Forensics in Cloud Computing

Hypervisor and Forensics in Cloud Computing

Cloud computing is “a model for enabling ubiquitous, convenient, on demand network access to a shared pool of configurable computing resources (e.g., network, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or services provider interaction” (Urias, Stout, and Young, 2016, p. 768); and it incorporates into different area such as government and private business. Recently, many applications support cloud computing such as Amazon, Google, Apple, and Microsoft which means most of the saved information is in the cloud. Cloud computing services divided into three models, Software, platform, and infrastructure.



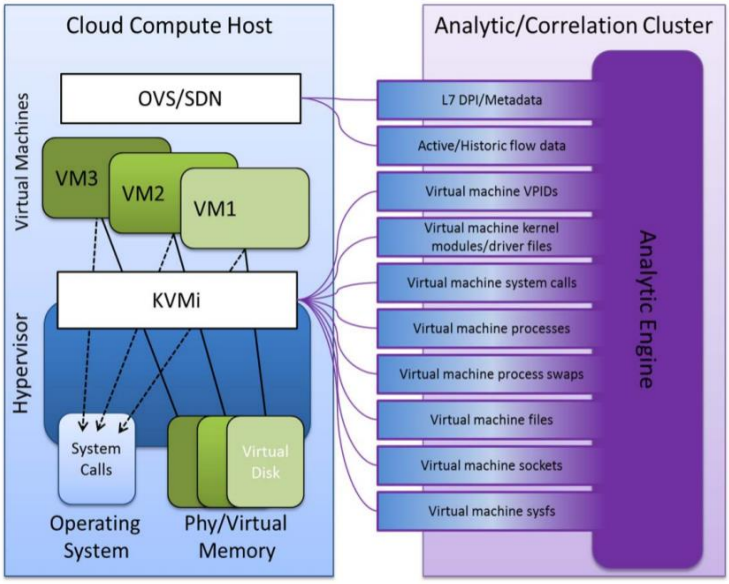
Cloud Computing services model

A software and platform controls the execution environments, and infrastructure gives the user permission to configure their virtual environment such as open stack and open shift platforms. Infrastructure has poor security and it invulnerable to malicious attack. The best practice of Cloud computing infrastructure is challenges the system in forensics. “the cloud infrastructure – with its distributed processing, storage, and resources – can be extremely complex because storage capacities can grow geometrically” (Urias, 2016, p. 769). To ensure the security of a data in cloud computing, the administrators must protect each users’ information and just let authorized user to access by using a virtualization technique for the environment. “When security incidents occur in violation of risk-reduction controls, the challenges involved in the cloud incident response and forensics begin to manifest” (Urias et al, 2016, 769)

Digital forensics is a combination of tools and information used in a computer and the location of the devices to find evidence for criminal investigation. Digital forensics investigators are in high level of experts that achieved the collection of the digital data; and they are able to determine the five forensics protocols, identification of an incident from its source, acquisition of evidence, preservation of the state of the evidential data, analysis of evidential data, and reporting the result. Also, it is very challenging to the analyst to have a clear image to help in forensics because each cloud has different strategies. According to Urias et al, there are eight major area that could challenges the analyst which are:

1. Architecture such as diversity, complexity, provenance, multitenancy, and data segregation.
2. Data collection such as data integrity, data recovery, data location, and imaging.
3. Analysis such as correlation, reconstruction, time sync, logs, metadata, and timeline.
4. Anti- forensics such as obfuscation, data hiding, and malware.
5. Incident first responders such as trustworthiness of cloud providers, response time, and reconstruction.
6. Role management such as data owners, identity management, users, and access control.
7. Legal such as jurisdiction, laws, SLA, contracts, subpoenas, international cooperation, privacy, and ethics.
8. Standards such as operating procedures, interoperability, testing, and validation.
9. Training such as forensics investigators, cloud providers, qualification, and certification.

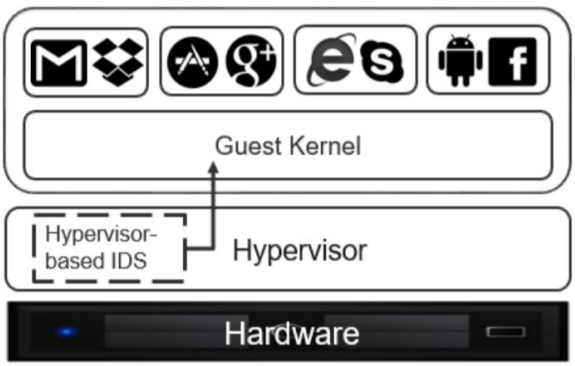
Most of the data are stored in the cloud and the hypervisor is where the data collects' critically. A hypervisor is a service in the VM which helps in the forensics; It can work with guest securely. A Kernal- based Virtual Machine introspection (KVMi) developed a (KVM) which is a hypervisor on Intel; it "implemented as a single loadable kernel module for Linux" (Urias et al, 2016, 771). A KVMi discusses ways to secure a hypervisor, personal and system securities. for more information visit here



KVMi forensics architecture

Cybercriminal could happen if vulnerably found in the VM, but there is different way to protect VM.

1. Virtual Fire wall is a virtualization environment that includes network firewall.
2. Intrusion detection system
    i. Intrusion detection and prevention system
    ii. Network based intrusion detection system
    iii. Intrusion detection system – host based
    iv. Hypervisor- based intrusion detection system – cloud computing using Hypervisor- based intrusion detection system to prevent any attack; is located between the hypervisor and guest kernel. A "Hypervisor-based IDS observes the system metrics through cloud requests from the hypervisor and detect any possible misuse trends" (Dildar, Khan, Abdullah, and Shahid Khan, 2017)



Hypervisor -based intrusion detection system

Open Source Hypervisor Security Assessment Project

Executive Summary

Many desirable features can be achieved through the use of virtualization including reducing costs by being able to run multiple systems on the same hardware, gaining an additional level of security for sensitive assets, and cloud service providers being able to host multiple clients on the same hardware due to the clients being logically isolated. Due to these many advantages, virtualization has become a rapidly growing domain in computing. Primary to the function of virtualization is the hypervisor, a hardware or software device that creates, runs, and manages the multiple virtual guests machines being hosted on the physical hardware.

As the usage of virtualization continues to grow, security of this primary computing piece, the hypervisor, grows in importance as well. If a user is able to escape from a virtual machine to the hypervisor, they may be able to move between virtual machines which would otherwise be isolated

User Stories Realization

At the current stage, we have successfully configured the Xen hypervisor and installed two paravirtualized hosts. In the next week, an HVM will be added to the configuration and full testing for vulnerabilities will begin. The purpose of this phase was to ensure the hypervisor was appropriately configured and to ensure the implementation would support multiple guest systems being installed.

The initial attempt to configure the Xen hypervisor consisted of installing Debian Linux as a virtual machine inside VMware due to hardware limitations. Multiple issues arose when attempting to load the Xen hypervisor on the Debian install. Our assumption is that there were incompatibility issues with trying to run a virtualization manager within a virtualized environment. After the initial attempt failed, standalone hardware was procured and setup was able to be conducted successfully.

The initial setup was performed following the steps provided in the Xen project's beginner guide (link). This consisted of installing Debian, partitioning the disk to allow for storage of the LVM (logical volume manager), and bridging the ethernet interface to allow the virtualized hosts to use the host OS ethernet port for networking. Once Debian was installed, we installed the Xen Project packages to allow PVM and HVM guests to be installed on the system. Next, two PVM hosts were configured using the xen-tools package and were both confirmed to be up and running. These hosts were set up initially with SSH and HTTP services running.

Scans were run against the test environment using Nmap to ensure the ports were showing as accessible from an external environment. This uncovered two objects of note. First, OS fingerprinting quickly determined that the virtualized hosts were indeed running in a virtual environment due to the way their MAC addresses are configured. Additionally, scans showed that on the host system, no services were running by

default. Due to this, our initial hypothesis was that any attacks against the hypervisor are going to be a result of weak security practices on the guest OS allowing us to use these guests as a pivoting point against the host itself.

For further testing the VASTO (Vulnerability assessment toolkit) module was added in to Metasploit. This toolkit has multiple modules for reconnaissance and exploitation of virtualized hosts. However, a major shortcoming exists in this tool for the purposes of our test requirements. Of the multiple modules included in VASTO, the majority are targeted towards VMWare exploits and recon. Similarly, nearly all built in modules in Metasploit dealing with virtual environments are specific to VMWare and none are specific to the Xen hypervisor. Due to these limitations, automated testing was not a viable option.

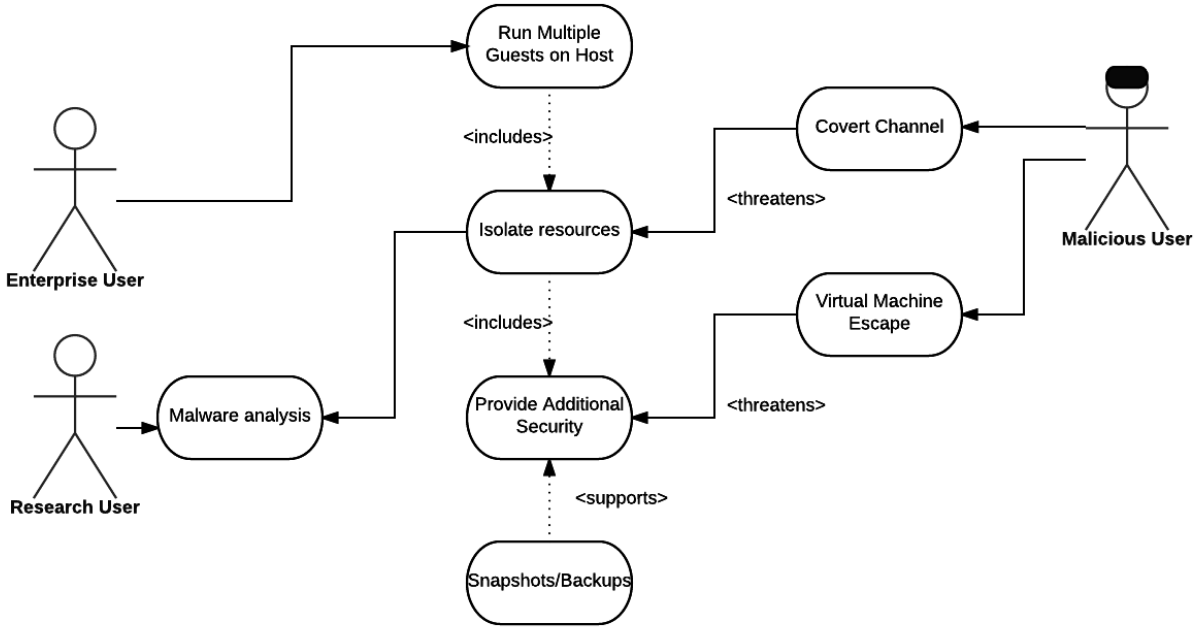
Turning our focus to the previously released and current vulnerabilities for the Xen project Xen Vulnerability Database revealed an interesting commonality. Of the 215 vulnerabilities that have been disclosed a disproportionate number of vulnerabilities have been published relating to weaknesses in how Xen manages the memory being allocated to the guest OS. As depicted previously in the memory allocation segment, the Xen system uses guest physical frame number (gpfn) pages for virtual memory and then links these to machine frame numbers (mfns) to utilize the memory of the host machine. When memory is not in use in the guest, a balloon driver is used to fill up the unused memory in the guest OS and allow the host to allocate that memory to other resources. If this freed memory is not properly deallocated or cleared before being allocated to a guest, it could allow for the guest to access memory contents intended for another host resulting in data leakage or with instructions that run at a higher privilege level than expected resulting in a possible escalation of privileges.
The Xen platform is constantly being tested and vulnerabilities submitted for review and resolution. Due to this and technical limitations we were unable to find a unique vulnerability. Our next step was to then attempt to replicate a vulnerability submitted by another researcher. For this we selected the vulnerability classified as CVE-2017-7228 - broken check in memory_exchange(). This vulnerability exploits a weakness in the function call memory exchange() where only the start address is checked to see if it is within a valid address range for the guest. Using a crafted argument for the function call an attacker could write to a protected memory area. We were unable to get the exploit to run in our test environment, possibly due to the fact that the exploit was tested in an environment running Xen on Ubuntu while we were running Xen on Debian Linux.

In reviewing the vulnerabilities that were discovered two things became evident. First, the community involved in the Xen project is very active in testing the platform for vulnerabilities. In the year 2017 alone, 11 vulnerabilities have been discovered and reported for the Xen project. The second is that the community not only works fervently towards discovery but also to determine mitigations and resolution steps. Even the vulnerabilities that were reported most recently, May 2nd at the time of this writing, have a fully fleshed out discussion on mitigation techniques but more importantly a patch release is included that resolves the reported issue entirely. With this in mind, it appears that tried and true security methods are once again king. First, implementers of Xen should ensure that the attack surface of the guests and host OS. By limiting the points

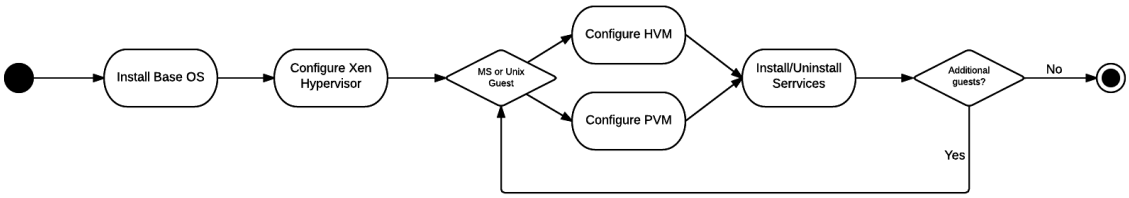
of access for an attacker we reduce the risk of exploitation of vulnerabilities resulting in VM escape. Secondly, regularly apply patches for Xen along with patching other hosts. Thanks to the quick disclosure of vulnerabilities with a resolution action, patching can be a major tool in minimizing the likelihood of successful exploitation.

Activity diagrams

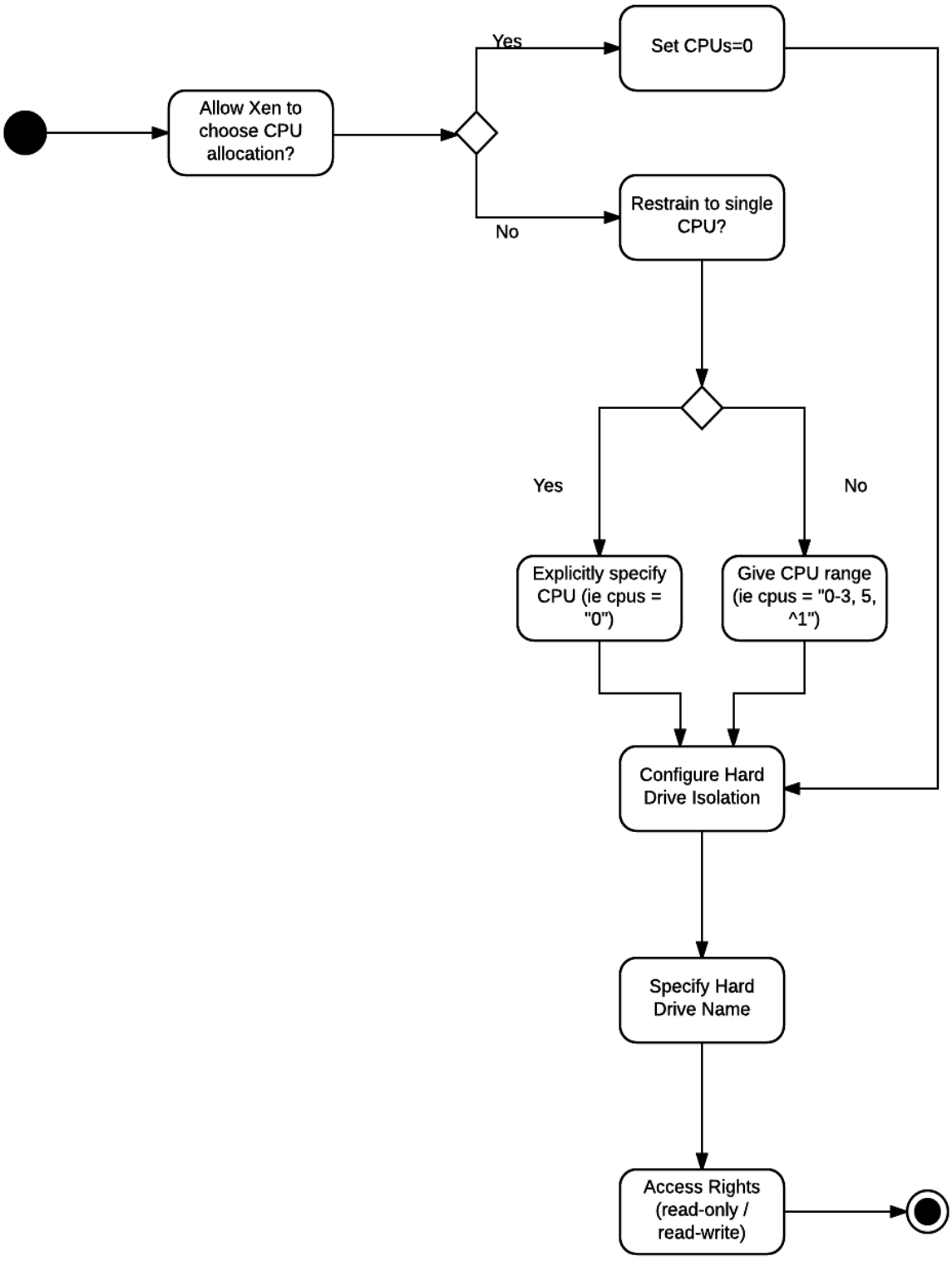
Hypervisor assessment use case



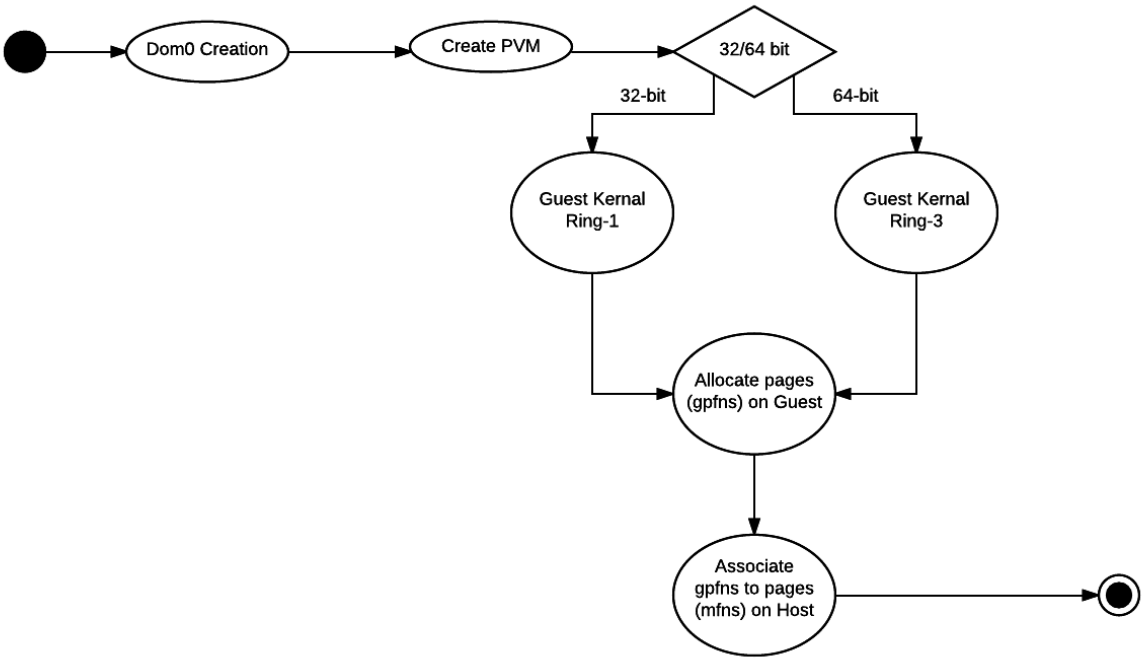
Running multiple host


Install Base OS
Configure Xen Hypervisor
MS or Unix Guest
Configure HVM
Configure PVM
Install/Uninstall Serrvices
Additional guests?
No
Yes

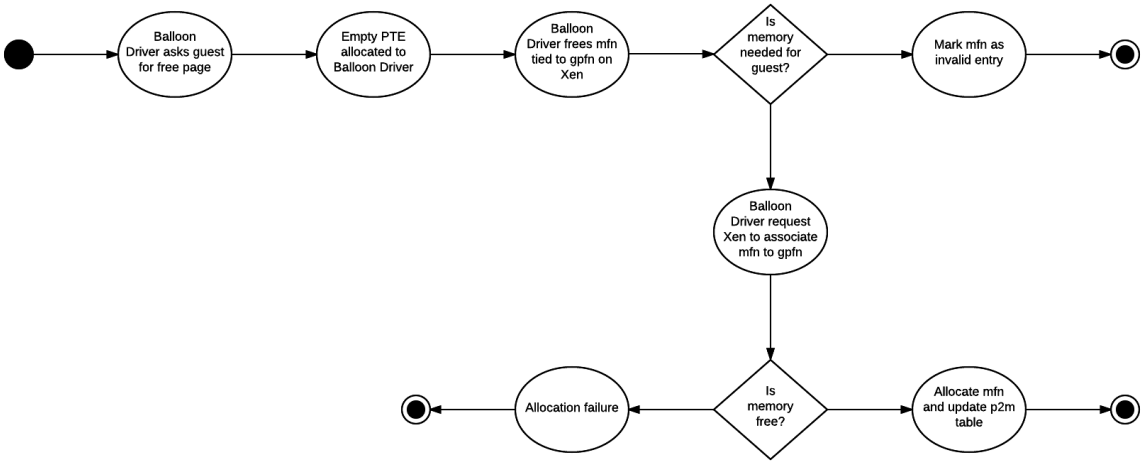
Resource isolation



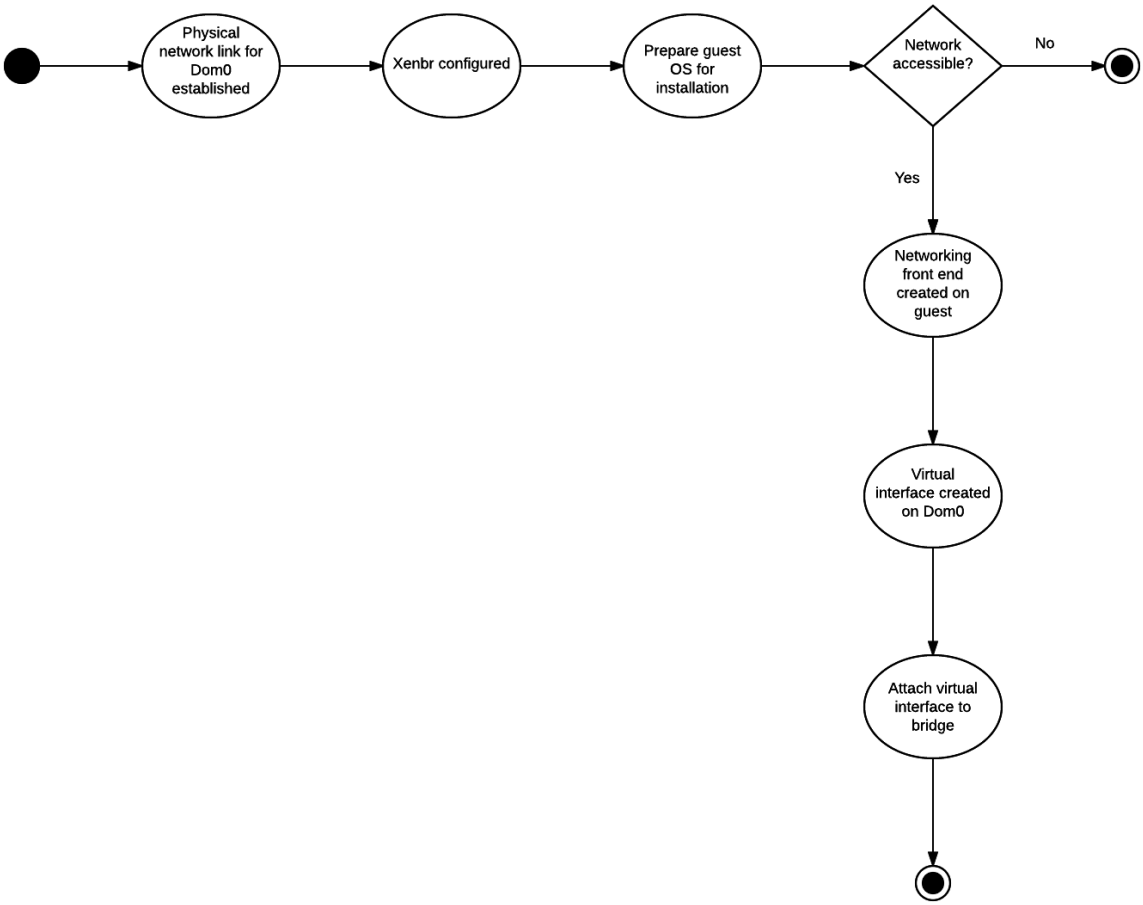
Memory allocation



Memory management



Network bridging



Assessment results

- Successfully installed Xen Hypervisor running on Debian Linux with two PVM hosts and one HVM host installed
- VASTO, while a good tool, is not very useful for testing Xen.
    - Of the 18 modules available in VASTO only 1 was for Xen
- The Xen platform is widely tested and supported by it's community
    - Most recently published vulnerabilities from 5/2
    - Largely related to weaknesses in the memory allocation and memory management processes
- Unsuccessful in replicating results from previously released vulnerabilities
    - Source code not very accessible
    - Differences in environments

References:

Desai, A., Oza, R., Sharma, P., and Petel, B. (Hypervisor: A Survey on Concepts and Texonomy) international journal of innovative technology and exploring engeneering, 2013, volume 2 issue 3.

Dildar, M., Khan, N., Abdullah, J., and Khan, A. (Effective Way to Defend the Hypervisor Attack in Cloud Computing) Anti-Cyber Crimes (ICACC), 2017 2nd International Conference, 2017. http://ieeexplore.ieee.org.leo.lib.unomaha.edu/stamp/stamp.jsp?arnumber=7905282

Sailer, R., Jaeger, T., Valdez, E., Caceres, R., Perez, R., Berger, S., Griffin, J., and Doorn, L. (Building a MAC -based Security Architecture for the Xen Open Sources Hyperviaor). 21st annual computer security applications conference, 2005, 1063-9527. http://ieeexplore.ieee.org.leo.lib.unomaha.edu/stamp/stamp.jsp?arnumber=1565255

Thonhthua, A. and Ngamsuriyaroj, S. (assessment of Hypervisor Vulnerabiliy). 2016 International Conference on Cloud Computing Research and Innovations, 2016. http://ieeexplore.ieee.org.leo.lib.unomaha.edu/stamp/stamp.jsp?arnumber=7600180

Urias, V., Stout, W., and Young, J. (Hypervisor Assisted Forensics and Incident Response in the Cloud) 2016 IEEE International Conference on Computer and Information Technology, 2017. http://ieeexplore.ieee.org.leo.lib.unomaha.edu/stamp/stamp.jsp?arnumber=7876418

https://wiki.xenproject.org/wiki/Xen_Project_Beginners_Guide