# Advocating Isolation of resources among multi-tenants by containerization in IaaS cloud model

Javed A. Samo, [1] Dr. Zeeshan Ahmed, [2] and Aasia Shaikh[3]

1 Department of Computing and Techology, Indus University Karachi, Pakistan

2 Department of Computing and Techology, Indus University Karachi, Pakistan

3 Department of Computer Science, Sindh University Larkana Campus, Pakistan

## Abstract

Cloud Computing refers to the dynamic provisioning of resources to multiple tenants around the globe to process big volumes of data at high speed, remotely from any instance. It caters the multiple needs of users like storing huge variety of data, providing interfaces and environments to work collaboratively on projects using Platform as service model, failover server and disaster recovery of datacenter servers and many more. The technology also offers a major advantage by providing a multi-tenancy of physical infrastructures, platforms, applications or combination by the help of virtualization techniques like virtual machine managers/hypervisors and container. Both these techniques have few grey areas. The most vulnerable threat is “co-residence of multi-tenants/ virtual machines on same physical resources”. In this paper we performed experiments to assess security of both containers and hypervisors. We used Hyper-V as VMM and Docker as container. The experiments were performed to measure the vulnerabilities parameters such as side channel attack. The results suggest that Hypervisor lacks sufficient logical isolation configuration that a suspicious can attacks or peeks other VMs by side channel attack and causes security flaws among multi-tenants to compromise the confidentiality and integrity of data. On the other hand container is much more secure and provides much stronger isolation at micro level in terms of resources and overcome the issues raised in VMM among multiple tenants.

Keywords: Docker Container, Hyper-V Hypervisor, Isolation and Side Channel Attack

978-1-5090-3310-2/17/$31.00 ©2017 IEEE

# Introduction

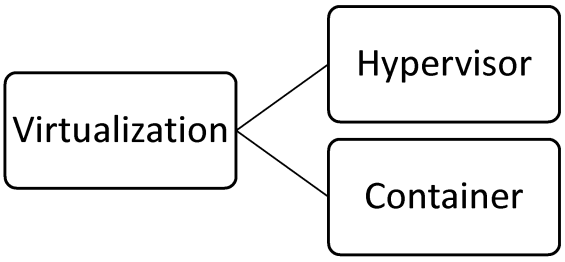
Cloud computing brings revolutionary change in the technology by giving the facility under and over provisioning of resources and pay as they go pricing model. Cloud computing provides three flavors of services mainly named as cloud computing models SaaS, PaaS and Iaas. In software as a service model thousands of applications are deployed over the Internet or Intranet and the customer is permissible to access it via web browsers like MS Office 365 online. SaaS is growing rapidly due to dynamic needs of customers, as application is not being installed at customer end so several customers can access single application with different features and pay as they go. Another cloud computing model is Platform as a service (PaaS) which facilitates the renter with different on demand platform like operating systems, development tools (programming IDEs, Database Management Systems and many more) to work as per their needs. Third and the last model of Cloud computing is IaaS (infrastructure as service) model which offers vigorous scalable resources to customers to like Storage devices, Processing devices Networking devices, Virtual machines or even Servers. IaaS gives the nearest similarity to the in-house datacenter contrasted by volatility. Infrastructure can be offered via two different ways either using bare metal approach or virtualization approach. In bare metal approach, physical resources are directly assigned to multiple tenants which may cause starvation among scalable tenants due to unavailability of on demand resources like servers, storage space, firewalls or physical cores. But virtualization approach overcomes the issues raised in bare metal approach by providing an intermediate layer among physical resources and multiple tenants. Furthermore, the paper is organized as follows section II contains virtualization techniques in detail. Section III describes motivation, Section IV contains Thread Model and V contains Experiments, results and conclusion.

**Multi-tenant and Isolation**: Individuals or corporate level organizations can acquire cloud computing services only on rental basis not permanent. Any customer who avails any of service is known as tenant. The term multi-tenancy refers that the identical service(s) availed by more than one tenant at the same time.

# SECTION II Virtualization

An essential element of Cloud computing is Virtualization which addresses to the get large scale elasticity of hardware resources and multi-tenancy isolation and configurability among

users, applications and different types of operating systems. There are two types of virtualization techniques used in cloud computing: ***Hypervisor* and *Containers***. Hypervisor is a system program which is used to create and execute several independent virtual machines simultaneously whereas Container is known as light weight virtualization technique means it is a simple program which runs with multiple instances as a single library program in kernel.




**Hypervisor and Container**

**Hypervisor**

Hypervisors are also known as or virtual machine managers which enable system to create and manage virtual machines on demand at run time to overcome the problem of under provision and overprovision of virtual servers. Hypervisors are favorite technique for allocation of physical infrastructure (Hardware resources) among multiple tenants. Normally hypervisors or VMMs are being installed on host operating system and host virtual machines as many as needed to virtualize the hardware. VMWare, Virtual Boxes, Hyper-V and KVM are the good examples of hypervisor.

As Hypervisor providing many benefits but there are still so many grey areas which need to be addressed like: Security, Performance, Communication and speed.

A. **Security**: Security is the major concern of cloud computing due to virtualization of resources. Attacks can be of two types Internal Attack or External Attack.

***Internal Attack****:* For example once you are using same physical resources via virtual machine than it would be quite easy to obtain the control of other nearby or neighbor VM's by side channel attack. Or the intruder enters into the system as a legitimate user and can sniff or temper the data or code.

***External Attack**:* For example a network attacker which sniff data from network or launch DoS to disturb the legitimate user.

Furthermore, Virtualization advances offer new conservative and specialized conceivable outcomes. Despite, the expansion of another layer of programming presents new security concerns. A rundown of the difficulties raised by the virtualization that is talked about from now on.

i. **Scaling:** Virtualization empowers fast and simple production of new virtual machines. Hence, security arrangements of a system (setup, upgrades. . .) must be sufficiently adaptable to handle a quick increment in the number of machines.

ii. **Brevity:** With virtualization, machines are regularly added to or expelled from a system. This can obstruct the endeavors to balance out it. For instance, if a system gets tainted by a worm, it will be harder to discover correctly which machines were tainted and tidy them up when these machines exist just amid brief timeframes on the system. Correspondingly, contaminated machines or still defenseless ones can return after the disease was thought to be wiped out.

iii. **Programming lifecycle:** The capacity to reestablish a virtual machine into a past state raises numerous security concerns. In fact, beforehand fixed vulnerabilities (programs imperfections, deactivated administrations, more seasoned passwords. . .) may return. Besides, reestablishing a virtual machine into a past state can permit an assailant to replay some arrangements, which renders old any security convention in light of the condition of the machine at a given time.

iv. **Differing qualities:** In an association where security arrangements depend on the homogeneity of the machines, Virtualization expands the danger of having numerous forms of the same framework in the meantime on the system.

v. **Portability:** A virtual machine is viewed as like some other document on a hard drive. It can be duplicated and moved to another plate or another host. This element, referred to as an advantage of virtualization, additionally includes security requirements on the grounds that ensuring the security of a virtual machine gets to be identical to ensuring the security of each host it has been on.

vi. **Character:** Regular techniques used to recognize machines (like MAC locations) are not as a matter of course proficient with virtual machines. In addition, portability increments considerably more troubles to validate the proprietor of a virtual machine (as it can be duplicated or moved).

vii. **Information lifetime**: A hypervisor ready to spare the condition of its VMs can counter the activities made by a visitor to erase delicate information from its memory. For sure, there may dependably be a reinforcement variant of the VM containing the information.

viii. **Identifying the VMs:** Once the virtual machine of system is identified by the attacker than the attacker can peek into the VMs and so on

ix. **Breach in the isolation:** Rupture in the disconnection. One of the principle objectives of a hypervisor is to guarantee that the facilitated virtual machines are detached, which implies that one visitor framework is not ready to achieve a larger number of assets than it has been in truth, particularly the memory utilized by alternate visitors or by the host framework. Nonetheless, awful arrangement or outline imperfections inside the VMM can permit an assailant to break out of the detachment.

B. **Performance**: Performance is the second biggest challenge of cloud computing and virtualization using hypervisor techniques.

i. *Utilization of Executing Units (Cores)*: Performance of executing units is much slower due to granularity. Cores are very much busy by passing messages from guest to host operating systems.

ii. *Utilization of Memory Units*: There are several models for memory management in hypervisor virtualization. Likewise, ***Shared Memory Access***: Global address space is shared among all virtual machine and this may lead to security threads while having centralized memory access. Another big disadvantage of global address space is every time when any process or VM get/ detach memory address a signal is sent to all the connected devices so latency may be possible. ***Distributed Memory Access***: This memory model solves all problems arose in shared memory access model by allocating isolated memory units among virtual machines. But one big disadvantage of this model is it is not scalable.

*iii.* Utilization of Storage Devices: Server virtualization joins couple of servers on storage devices, this frequently make bottlenecks as virtual machines (VMs) vie for storage resource. Desktop virtualization can likewise make bottlenecks since such a large number of desktops keep running on a solitary host. Along these lines, effectively overseeing virtual servers regularly requires a joint effort from storage, and also desktop and server administrators. To help information stockpiling experts capitalize on these testing situations, we've gathered our main five tips on overseeing stockpiling in a virtual server environment.

iv. *Distributed and Network File systems*: An issue brought about by shared file system is the absence of arrangement separation. Different applications can have clashing necessities for framework wide setup settings. Shared library conditions can be particularly dangerous since cutting edge applications use numerous libraries and frequently distinctive applications require diverse variants of the same library. At the point when introducing different applications on one working framework the expense of framework organization can surpass the expense of the product itself.

**C. Inter process Communication (IPC)**:

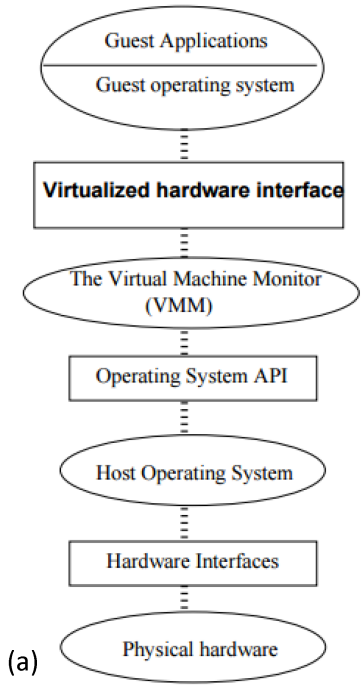
i. Hypercalls: Hypercalls in hypervisor behaves like system calls in operating system. It is an interface in between guest operating system (Virtual Machine) and host

operating system. There are certain issues with hypercalls inter process communication mechanism like: Granularity, Security and speed.

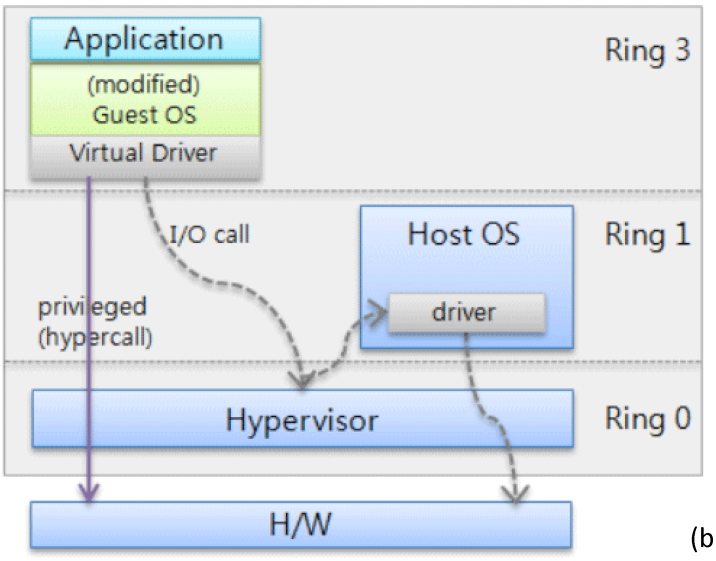
**Granularity**: Hypercalls increase the problem of granularity by increasing communication ratio than processing ratio as multiple guest operating system requests for resources to host operating system installed over hardware layer.

**Security**: hypercalls are exceedingly privileged; correctness should be in the bag. It may be problem to sniff data using hypercalls from one guest operating system to other guests or host operating system by IPC. So it may be necessary that the call must be well equipped with cryptographic algorithm.

**Speed**: Normally system calls are very fast in communication but Hypercalls are little bit slower as an application is executing on virtual machine and needs a resource to complete the process so respective VM will request for the particular resource to host OS or Virtual machine manager and it will take long time to process.




(a)




(b)

Fig. 3 IPC

## Containerization

Another virtualization technique is known as Containerization. Container is a light weight process that runs inside an operating system root (kernel) as it is host. Containerization does extremely well in executing all kind of micro services-based architecture that is appropriate more and more for cloud-native web apps. Containers are of two types:

**System Container:** OS containers are virtual atmosphere that distribute the kernel of the host operating system but offer user space isolation. For all useful reasons, one can believe of OS containers as Virtual Machines. Individual can install, configure and run diverse apps, libraries just as it would be on any Operating system. But in this scenario anything executing within a container can only perceive resources that are assigned to it. There are two flavor of system containers; identical and multiple. In identical OS Container, there is an array of homogeneous containers as in figure a, but in multi-OS container a choice is for user to host different operating systems containing different containers of their nature,

**Application Container**: OS containers are designed to run multiple programs and services, whereas application containers are planned to box up and run a particular service. Simple single service as a container is known as application container.

The container host all the required libraries and supporting files with that particular application/service to execute in that particular domain. This feature help to process fast and avoid granularity that not all the time application communicate with the host operating system

***Characteristics of Container***

1. As a customer, you condition containers, not hardware virtual machines or even bare metal servers.
2. Those containers run on native systems, not in a VM.
3. Each container is an identical examine on the network to which it is attached, by its hold IP stack sovereign of its meticulous host; containers must not be group in the host's system.
4. The container hypervisor has experienced and believed security isolation for containers; nasty or wrecked code in one container must not be able to break other containers as it was possible in VMs.
5. As a public cloud client, he/she has to pay by the container, not for the infrastructure and surely not for a bunch of VMs they force to run on.

***Benefits of Container***

1. **Fast:** Runtime performance near bare metal speed
2. **Flexible:** Have facility to Containerize a “system” and Containerize “application(s)”
3. **Inexpensive:** Most containers are freeware or open source
4. **Ecosystem:** Due to its benefit and characteristics it is growing rapidly and getting enough popularity
5. **Light Weight:** It is light weight process as other processes running concurrently on operating system. Just enough OS Minimal per container penalty

Section IV Background

# SECTION III MOTIVATION

*The main theme which enforces to work on the title cited above is:*

**The principle of least privilege” or the principle of least authority”,** it is mechanism is a model in security, endorsing negligible user profile **rights** on computers and based on users' job requirements. And sharing same instance among diverse programs is always risky. If any of the program is disturbed then it disturb few more programs or the entire programs in the same instance like covert storage channel. Suppose the security of one program is compromised than all other programs sharing same instance or state will be affected. This concept brings “***least common mechanism principle”***. Hypervisor virtualization techniques do not satisfy the problems discussed in this section. So need another virtualization technique to address the problem and in this report I advocate containerization

Literature Review

# SECTION IV THREAD MODEL

In this area, we talk about dangers and vulnerabilities in CC, some normal dangers and vulnerabilities are highlighted in detail, and record and administration seizing.

Powerlessness alludes to a shortcoming or blemish in a framework that can be abused by an occasion called an assault (Hashizume et al. 2013; Tianfield 2012). A danger (see Figure 1) is an abuse ofany known powerlessness that can bring about genuine loss of information and data (Modi et al. 2013). These dangers and vulnerabilities are seen as a hindrance for associations wishing to embrace cloud administrations (Hashizume et al.

2013). The Cloud Security Alliance (CSA) has highlighted beat dangers to CC (Alva et al. 2013).

Information Breaches and Loss: These are exceptionally extreme dangers in view of cloud administrations' shared design. Information misfortune can happen if information are being erased without having reinforcement or if there is any catastrophe and no appropriate components are embraced to recuperate it (Khorshed et al 2012).

Privacy, uprightness and accessibility of client's information are put at hazard (Alva et al. 2013)when an assailant has an opportunity to capture a client's record points of interest by taking client qualifications furthermore, data. An aggressor may utilize different techniques, for example, phishing, DoS and Man-in the- middle assault to take client accreditations (Khorshed et al. 2012) and afterward utilize them to spy on a client's exercises and exchanges. More refined assaults include mixes of these techniques for monetary pick up. A few strategies for which fruitful assaults might be accomplished include:

Unreliable Application Programing Interfaces (APIs): A CSP offers framework, programming and stage administrations to clients and gives them access to administrations through Application Programing Interfaces (APIs) (Khorshed et al 2012; Chhabra and Dixit 2015; Ayala et al. 2013). Clients oversee administrations they devour by utilizing these APIs (Modi et al. 2013), for instance, PaaS APIs give get to and usefulness to cloud, SaaS APIs give availability to applications, IaaS APIs gives control and dispersion over particular cloud assets.

Classification, trustworthiness and validation of a cloud administration are in particular subject to how APIs are planned and created (Hashizume et al. 2013; Vaquero et al. 2011; Ayala et al. 2013). Accordingly, it is the obligation of the architect and designer to make it troublesome for malevolent clients to endeavor potential vulnerabilities. For instance, plan defects of the APIs to the fundamental SSL libraries (Georgiev et al.

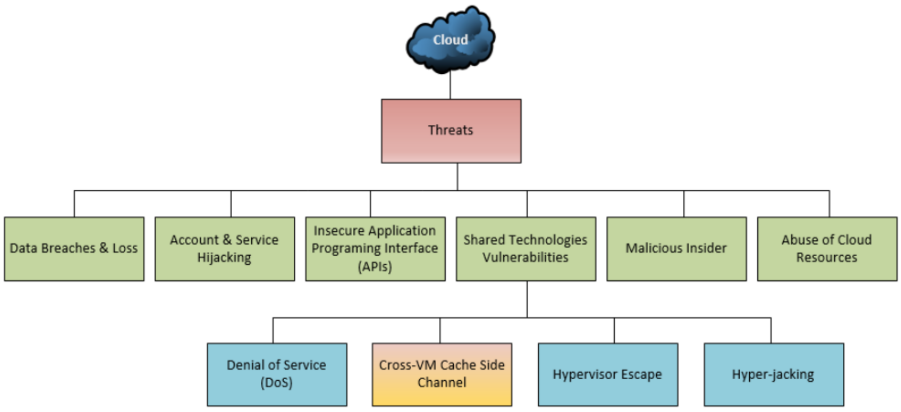
2012) give influence to a client with noxious purpose to execute unsafe activities (Srinivasan et al. 2012).

Noxious Insider: Confidentiality, uprightness and accessibility might be damaged if client information mishandle happens from somebody with approved get to yet with vindictive purpose (Alva et al. 2013). The vindictive insider danger may get to be distinctly clear in light of the fact that of an absence of straightforwardness in CSP procedures and

methodology and the level of get to each worker needs to client information (Modi et al. 2013; Srinivasan et al. 2012).

Manhandle of Cloud Resources: CSPs give free trials of IaaS to clients with a specific end goal to persuade them to buy this administration later on. Be that as it may, because of an absence of control practiced by the CSP, clients may accept the open door to perform malignant exercises utilizing the capable equipment stage gave by the CSP (Khorshed et al. 2012; Modi et al. 2013). Besides, such equipment gives them opportunity to split encryption keys (Modi et al. 2013; Zhang et al. 2013).

Shared Technology Vulnerabilities: Virtualization gives multi-tenure through the sharing of equipment assets like CPU centers, abnormal state reserve, stockpiling gadgets what's more, and system interface cards among various occupants (Ayala et al. 2013). The hypervisor gives a stage to virtualization and is in charge of separation between various VMs running on a mutual server. By abusing vulnerabilities or design blemishes, an assailant can increase unapproved access to a hypervisor to control a cloud stage and adventure other clients' VMs (Khorshed et al. 2012; Modi et al. 2013). For instance, foundation sharing is one center cloud benefit but since of equipment sharing impediments, needs essential assurance systems to ensure client arrange movement, information and different applications. This gives an aggressor an opportunity to seize client accreditations, listen in on data also, and control other clients' VMs (Khorshed et al. 2012).

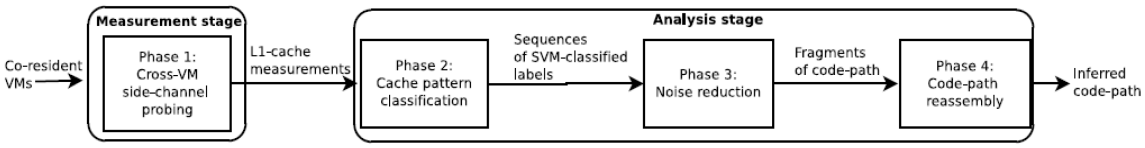



**Side Channel Attack**

The present day x64 microchip has various levels of reserve: Level 1, Level 2 and Level 3. In a processor with different centers and three levels of store, every processor center has its own isolate L1 and L2 reserve while L3 store is shared among every one of

the centers of the processor (Yarom and Falkner 2013; Kim et al. 2012; Zhang et al. 2011). In this way, the common design of L3 reserve among various centers makes an open door for a vindictive client to misuse it and dispatch a side channel assault.

In an average situation, when the processor needs information, it checks L1, L2 and L3 reserves separately. On the off chance that the asked for information are available at any level then the information are perused and a reserve hit happens. Be that as it may, if information are not accessible in any reserve level, then a store miss happens and the processor gets information from RAM and places it in the store prepared for the following call. In the occasion of a reserve miss, the processor may likewise exploit spatial region by getting information from close reserve areas, since it is normal that information in close store areas are going to be gotten to soon (Irazoqui et al., 2014a; 2014b). Additionally, information brought from RAM also, put in store are in settled measured pieces called reserve lines, and a reserve passage is made amid this procedure.




**Flash Reload**

The fundamental building block of a Flush- Reload assault is as per the following. A lump is a cacheline-sized, adjusted locale in the physical memory that is mapped into the foe's address space. For instance, if a cacheline is of size 64B, then every address that is a various of 64B characterizes the piece beginning at that address.

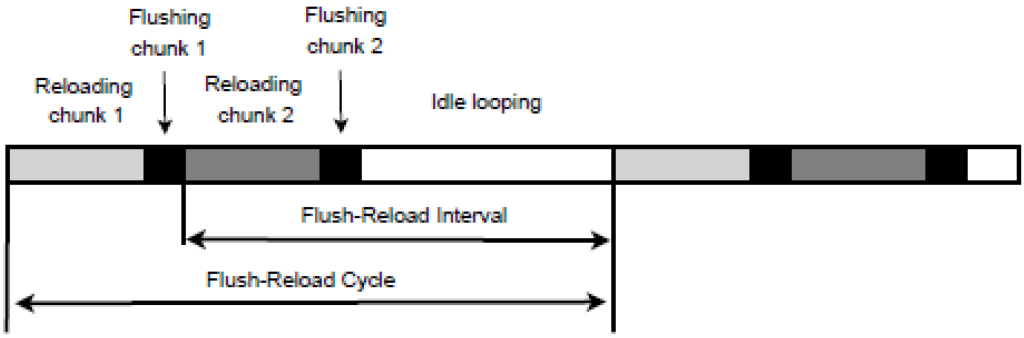
• Flush: The enemy flushes pieces containing particular directions situated in a memory page it offers with the casualty out of the whole reserve progression (counting the shared last-level reserve) utilizing the clflush direction.

• Flush-Reload interim: The enemy sits tight for a pre- determined interim while the last-level store is used by the casualty running on another CPU center.

• Reload: The enemy times the reload of the same pieces into the processor. A speedier reload proposes these lumps were in the last-level reserve as were executed by the casualty amid the Flush-Reload interim; a slower reload proposes something else.

We allude to the lumps being Flush-Reloaded by the advertisement versary as being observed, since Flush-Reload basically screens access to information in the lump. Flushing a piece by means of clflush, thus observing that piece, should be possible without knowing the physical address of the lump, since clflush takes the lump's virtual address (for this situation, in the adversary's address space) as its operand. We call a quicker reload amid the Reload stage a watched occasion or perception. We additionally embrace ideas from measurable order and utilize the term false negative to allude to missed perceptions of the casualty's entrance to the checked piece and false positives to allude to watched occasions that are brought about by reasons other than the casualty's entrance to the observed piece.

We characterize a Flush-Reload convention, in which the foe procedure screens a rundown of lumps at the same time and over and again until taught generally. It will first attempt to Reload the principal lump, record the reload time and Flush it promptly thereafter. At that point it will rehash these means on the second lump, the third, and so on, until the last lump in the rundown. At that point the foe will sit tight for a precisely computed day and age before beginning once again from the primary piece, so that the interim between the Flush and Reload of a similar lump is of an objective level 3 from the Reload of the main lump to the end of the holding up period is called one Flush-Reload cycle. An outline of the Flush-Reload convention is appeared in following Fig.

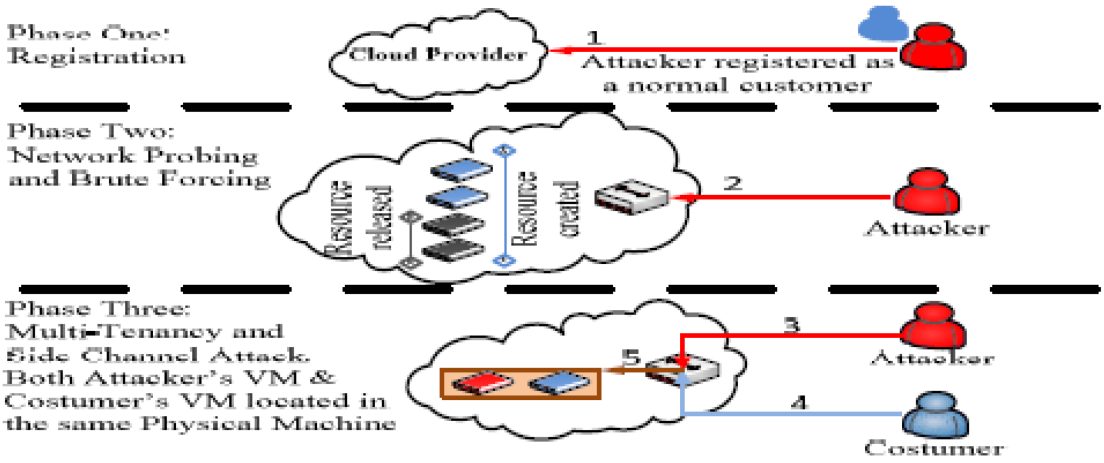



# SECTION V EXPERIMENTS

### Experiment Setup/ Proposed attack scenario

Here we execute experiments on two systems. One system is a laptop and the second system is a dedicated server facilitating different services. We have installed windows server 2012 on both machines. First of all we used NMap to find OS finger printing the

using the Nassus we found the vulnerabilities available in VMs. Then generated side channel attack on both machines using flush + reload attack to get cryptographic keys and peeks inside the VMs whereas docker provides much more isolation and it was quite difficult to get all keys and peeking mechanism in docker.




Algorithm Flush Reload

Hardware Specification

| | System 1 | System2 |
|---|---|---|
| Use | Laptop | Dedicate Server |
| OS | Windows Server 2012 | |
| CPU | Intel® Core™ i3-4010U CPU @ 1.70Ghz | Intel Xeon 2.8 Ghz |
| Cores | 02 Physical<br>04 Logical | 04 Physical<br>08 Logical |
| RAM | 4GB | 8GB |

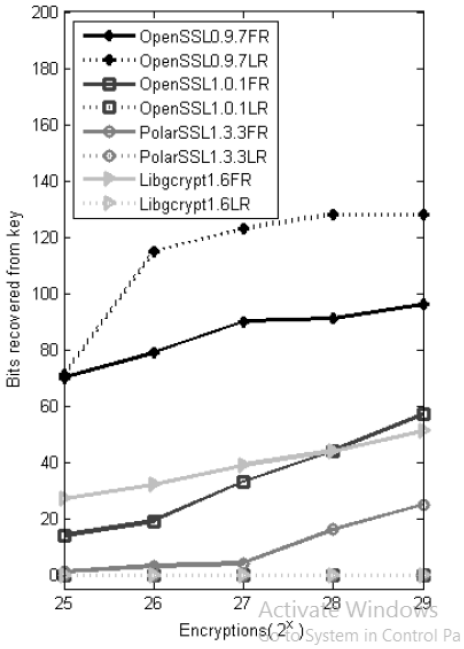
| | | |
|---|---|---|
| L1 | 128 KB | 256KB |
| L2 | 512 KB | 01 MB |
| L3 | 03 MB | 08 MB |

Supporting Software

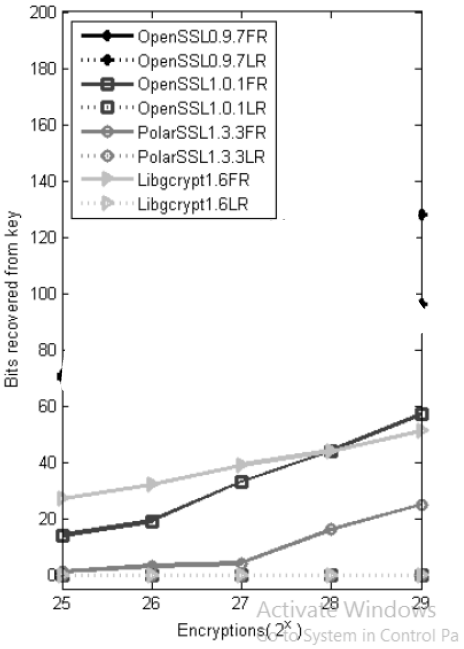
| Name | Function |
|---|---|
| Ubuntu and Windows Server 2008 | Ubuntu and Windows server 2008 are being used to host multiple servers on Virtual machines |
| Web Server | |
| Docker Container and Microsoft Hyper-V | Docker<br><br>Docker Container cape a bit of Software in an entire file system that contains everything expected to run: code, runtime, framework instruments, framework libraries – anything that can be introduced on a server. This ensures the product will dependably run the same, paying little respect to its surroundings.<br><br>MS Hyper-V<br><br>Microsoft Hyper-V, codenamed Viridian and some time ago known as Windows Server Virtualization, is a local hypervisor; it can make virtual machines on x86-64 frameworks running Windows |
| NMap | Nmap is a security app used to find has and benefits on a PC organize, hence making a "guide" of the system |
| Nessus | Nessus is a remote security checking device, which filters a PC and raises a caution in the event that it |

| | |
|---|---|
| | finds any vulnerabilities that noxious programmers could use to access any PC you have associated with a system |
| GnuPG and Open SSL | GnuPG is an entire and free execution of the OpenPGP standard as characterized by RFC4880 (otherwise called PGP). GnuPG permits to encode and sign your information and correspondence, highlights a flexible key administration framework and additionally get to modules for a wide range of open key indexes. |

# Results




System - A




System -B

## Conclusion

It has been observed from experiments that container provides much more isolation among multiple users/ multi-tenants in cloud virtualization as compared to virtual machines. Taking the example of Docker container which is light weight, more secure, and fast processing virtualization technique and getting much more familiarity due to its characteristics. Also Container provides isolation at every instance of virtualization like; at process level, at file system level, network level and at inter process communication (IPC) level.

# References


1. An Updated Performance Comparison of Virtual Machines and Linux Containers Wes Felter, Alexandre Ferreira, Ram Rajamony, Juan Rubio IBM Research, Austin, TX fwmf, apferrei, rajamony, rubiojg@us.ibm.com

2. A Unified Operating System for Clouds and Manycore: fos David Wentzlaff, Charles Gruenwald III, Nathan Beckmann, Kevin Modzelewski,, Adam Belay, Lamia Youseff, Jason Miller, and Anant Agarwal

3. Containers and Cloud: From LXC to Docker to Kubernetes, DAVID BERNSTEIN

4. Containers and Clusters for Edge Cloud Architectures – a Technology Review Claus Pahl Irish Centre for Cloud Computing and Commerce IC4 & Lero, the Irish Software Research Centre Dublin City UniversityDublin 9, Ireland

5. Containerisation and the PaaS Cloud, Claus Pahl

6. http://www.slideshare.net/BodenRussell/kvm-and-docker-lxc-benchmarking-with-openstack

7. http://stackoverflow.com/questions/16047306/how-is-docker-different-from-a-normal-virtual-machine.

8. https://www.toptal.com/linux/separation-anxiety-isolating-your-system-with-linux-namespaces

9. Isolation in Cloud Computing and Privacy-Enhancing Technologies Suitability of Privacy- Enhancing Technologies for Separating Data Usage in Business Processes, Prof. Dr. Noboru Sonehara
Prof. Dr. Isao Echizen, Dr. SvenWohlgemuth, National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo
sonehara@nii.ac.jp.

10. Performance Isolation and Fairness for Multi-Tenant Cloud Storage
David Shue, Michael J. Freedman, and Anees Shaikh, Princeton University, yIBM TJ Watson Research Center