Seminar Report

on

Study of Techniques and Applications of VM based Event Logging & Co-location Detection

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology

by

Ankit Bhardwaj
Roll No: 133050024

under the guidance of

Prof. Purushottam Kulkarni

Figure 1

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai

April 27, 2014

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my guide,**Prof. Purushottam Kulkarni** for the consistent directions he has fed into my work. Also course CS 695 [Spring 2014] offered by my guide helped me a lot to know more about virtualization -basic concepts, problems and solutions to those problems.

Abstract

Cloud Computing is one of the buzzwords in computer science now a days.With the change in the speed of the Internet,service providers are providing various type of services like IaaS,PaaS,SaaS in which clients can get the service by giving some nominal amount of money to the provider or may be free also.To provide service to lot of users providers keep a pool of resources.Which they give to the clients when needed.The goal of the providers is to provide service to users and at the same time they don't want to waste their resources.They multiplex resources among users over time.Virtualization is the basic building block for the cloud environment to enhance flexibility.Virtualization abstracts computing environment while in a broad sense cloud determines how those virtualized resources are allocated and delivered to the clients. However virtualization can be used in other cases also and even this is not necessary to use virtualization in clouds.But in that case it will be very hard to manage and scale the clouds.Virtualization abstracts as well as isolates underlying hardware as VMs .We can use multiple VMs on a single physical Host.Aim of this seminar is to learn about virtualization and study about various problems in this field.I have read about " Execution Replay of Virtual Machines " and " Co-location Detection Virtual Machines " for my seminar work.

1 Introduction

Cloud computing[4] means computing over the network and broadly over the Internet.Instead of doing computing on the local machine we are doing this computation on a remote machine over the Internet.This computing can be of different types.Providers are giving various types of services like SaaS,PaaS,IaaS,*aaS. SaaS[2] means some other party is providing software deployed of some remote machine as a service to the user. User need not to install and manage this software on his local machine.Gmail can be as example of this type of service. In PaaS[2] service provider is giving computing platform as a service and user need not to take care about the various softwares needed,OS,storage,networking etc.But application and data is a responsibility of the user. In IaaS[2] service provider gives you an infrastructure on which the user can install any required platform,software etc. To provide above written services they keep a pool of resources which they give on rent or in some cases free to the users for different amount of time.The same pool of resources is multiplexed among various users.Virtualization is the technology which made it so easy to manage and scale resources for billions of users.

1.1 Need of Virtualization

Virtualization allows us to abstract hardware and also isolates it form the upper layer of computing.Which can be used extensively in cloud environment.First of all we can put more than one OS on a single physical machine and we can put different types of application on different OS. Packing of more than one VM on a single PM will increase the resource utilization.Beside that execution in VMs is more controlled and we can capture various information related to the execution which can be used to get about workloads ,failure situations etc.Server consolidation can be one of benefit which cloud providers can get by using virtualization.With the help of virtualization,we can do dynamic resource provisioning for different VMs according to the need.Like this there are many other benefits which we can get by using virtualization. Meaning of virtualization in computing environment is to create virtual version of something for example virtual memory.Here we are interested in Virtual Machines and in VMs we are looking at Sys-

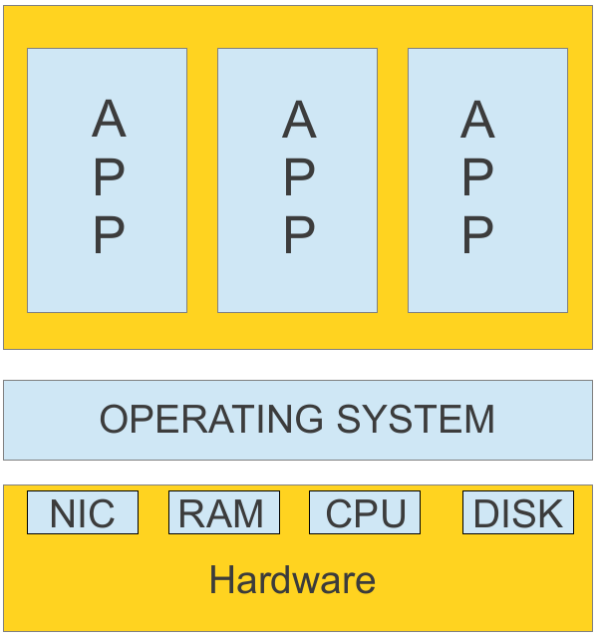

Figure 2: Without VMs: Single OS owns all hardware resources

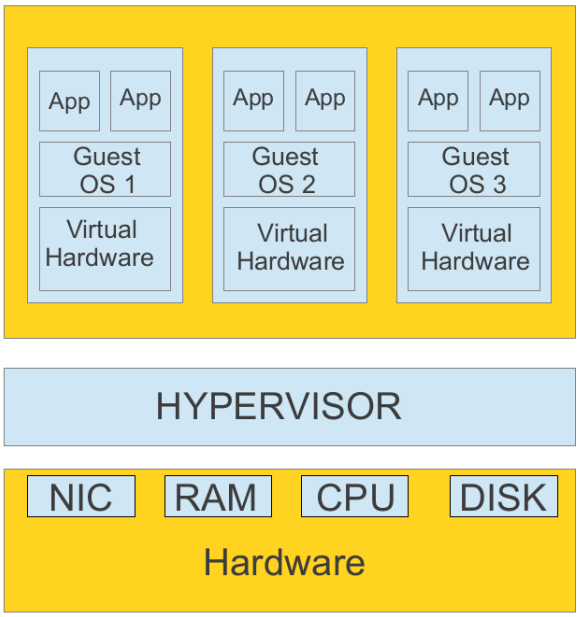

Figure 3: With VMs: Many OSes share hardware resources

tem VM.In the system VM environment host resources is shared among VMs with the help of ,a layer of software called, VMM or Hypervisor.In each VM there is an OS which see a virtual version of hardware resources.We can take an example of Hard-Disk,what we are seeing as a disk drive at VM level can be a file at hypervisor level.Hypervisor layer do the needed conversion to expose this file as a disk drive to the VM. This version of hardware resources may be similar to actual physical resources or can be different also.We can get more about virtualization and basic concepts in [13].There can be many more usage of VMs, which can be used for different scenarios.Here are some of the use cases :

1. Multiple Single Application VMs : Running each application in its own VM increases the robustness of the system.Say many applications are running same OS then single application can cause whole system to crash,while in case of VM environment most probably only one VM will crash.

2. Multiple secure environment : VMs provides isolation among VMs i.e one VM is isolated from the another VM.Many VM can run on a single PM and they are isolated from each other.We can use this benefit to give VM to different set of users.

3. Multiplatform application development : We can use single PM to run different VMs.Suppose we want to use Linux for development purpose and Window for administrative stuff.Instead of using two physical machine this can be done easily with VMs.

4. More controlled environment : In native system OS runs in highest priority mode and there is no way to get some sort of information which we can easily get in case of virtual environment.We can get resource needed to run an application and can dynamically allocate resources to the VM .Likewise there are hundreds of other such things which we can do in case of virtual environment.

5. Event Monitoring : In case of virtual machine we can trace the execution path and can log the events.Then we can replay the system execution to understand the behavior of the system.

6. System encapsulation : In case of VM ,it is encapsulated in a way that we can capture entire state of the system.This can be used for stopping the VM on one machine and resume the execution on some other VM.

   etc.

1.2 Scope of this seminar

OS is encapsulated in a VM and now we have more control over the execution.We can do different types of optimizations which is not possible in native case.For ex VM is encapsulated in such in a way that we can capture entire state of the VM and this gives us the opportunity to move this state across machine.We can migrate a VM from one PM to another PM and this can be very useful at various places.Likewise we can increase and decrease the resources for a VM and this can be used to solve various types of problems specially in data centers.So system virtualizations has solved many problems which was not possible in case of native environment.VM is running in a controlled environment and running in this manner give us the opportunity to Monitor various events happening in the VM.Scope of this seminar is to read about various approaches where we can exploit event monitoring capability and which problems can be solved using this.Beside that I am also looking at VM co-location detection approaches and use cases.There is a performance overhead on the machines if its running as a VM. If two or more VMs are running on the same PM then execution on one VM may affect the performance of the another VM.So we are interested in the question- is our machine running in a virtual environment or is it running on a PM where some other VM are also running ?

2 Background

2.1 Introduction to Virtualization

Virtualization, in terms of computers and computing environment can be explained as making of virtual copy of hardware or software resources instead of giving actual resource for use.Virtualization can be illustrated with example of Virtual Memory , suppose we don't have enough memory needed for a process,yet the OS can show this process that it has enough memory by using Virtual Memory concept.This is just an example and this philosophy can be extended further for various components.To address some problems virtualization can be used at process level or at system level and we are interested in system level virtualization.We can install many OS on a single machine and all of these OSs will think as it is using whole physical machine on its own.Each VM views similar or modified hardware.

2.1.1 Motivation for Virtualization

Virtualization is an old technology and was first introduced in 1972 by IBM[3].But in those days it was not as famous as it is now a days.However there are places where we can use virtualization and virtual machines but cloud computing is the main reason that we people are hearing virtual machines again and again.As described above that we have powerful computer hardware and most of machines are under-utilized.Cloud Service provider has a pool of resources and s/he tries to multiplex the resources among various users.So that service provider can do maximum utilization of resources.More than one Virtual Machine can be put on a single physical machine and these VMs can be shared among various users simultaneously.This will improve resource utilization and there are other benefits also,like we are packing more VMs on a single physical machine instead of one VM on one PM so PM running at a time will be low and also power consumption to run those PMs will be saved. Besides this we have more control over these machines we can move these VMs on different PMs according to resource need.We can save power by moving VM on few PMs if resource requirement is less and we can give more resources to VM if it has less resources.

2.1.2 Types of Virtualization

Here we are taking hardware virtualization into consideration in which computer hardware is virtualized to create a VM .There is three types in which this hardware virtualization can be exposed to the OS.To get more knowledge about hardware virtualization types refer [1].

1. Full Virtualization
2. Hardware Assisted Virtualization
3. Para Virtualization

To explain these concepts I am using Figure 4, 5,and 6 which are taken from [1]. Full Virtualization is achieved with a combination of binary translation and direct execution.User instructions are executed directly on the hardware and privileged instructions are executed with the help of binary translation.In Full virtualization we don't need to change the OS.In fact Full virtualization is the only option that requires no hardware assist or operating system assist to virtualize sensitive and privileged instructions.

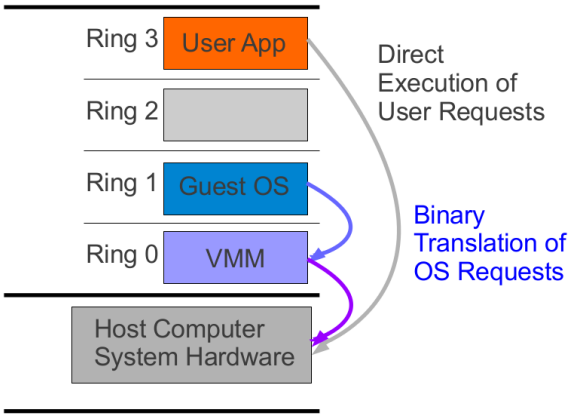

Figure 4: Full-Virtualization

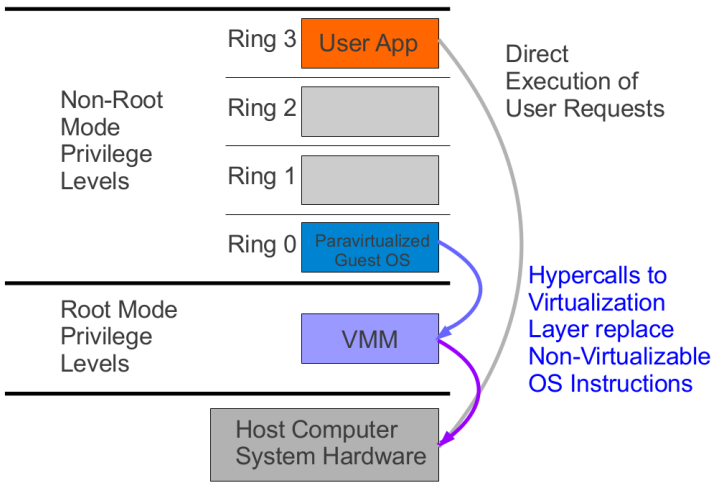

Figure 5: Para-Virtualization

In Para Virtualization guest OS is modified to replace the non-virtualizable instructions with the hypercalls that talks to the hypervisor and hypervisor takes care about the execution of such instructions.Para virtualization is not useful where we can't make change in the OS.Figure 6 shows about the paravirtualization.

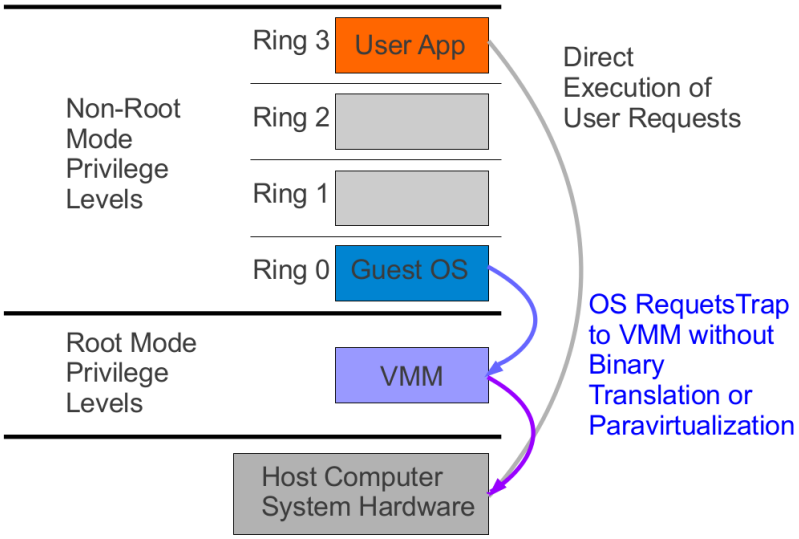

Figure 6: Hardware Assisted Virtualization

In Hardware Assisted Virtualization execution is done normally but for non-virtualizable instructions a trap is sent to the hypervisor. As show in the figure 5 privileged and sensitive calls are set to automatically trap to the hypervisor, removing the need for either binary translation or paravirtualization.Table **??** is showing comparision of x86 processor virtualization Techniques which is inherited from [1].

2.1.3 Essential Components of Virtualization

We have hardware and OS in case of native environment as well as virtual environment.What else we need to pack more than one OS on a single PM.There should be something in between

	Full-Virtualization	Hardware-Assisted Virtualization	Para-Virtualization
Technique	Binary Translation and Direct Execution	Exit to Root Mode on Privileged Instructions	Hypercalls
Guest OS compatibility	Unmodified Guest OS Excellent compatibility	Unmodified Guest OS Excellent compatibility	Guest OS codified to issue Hypercalls.Poor compatibility; Not available on Windows OSes
Performance	Good	Current performance lags Binary Translation virtualization on various workloads but will improve over time	Better in certain cases
Used By	VMware,KVM	VMware,Xen	VMware,Xen
Guest OS Hypervisor Independent	Yes	Yes	XenLinux runs only on Xen Hypervisor

Table 1: Comparison of x86 processor virtualization techniques

hardware and VM which can show similar or some modified version of hardware to the VM so that a VM can execute on this hardware.This middle layer which resides in between hardware and VM is called Hypervisor.More about Hypervisors is given in the next section.

2.2 Introduction to Hypervisor

Hypervisor is a software layer which provide support for various OSes to execute on a single PM irrespective of other VM on that PM.Every VM execute in such a manner he is the only one which has all the resources of that PM.Hypervisor allows many VM exist on same PM..Hypervisor is also called as VMM i.e Virtual Machine Monitor.There are two places where a hypervisor can reside.Based on that we can give types of hypervisors :-

2.2.1 Types of Hypervisors

There is two types of Hypervisors/VMM i.e

1. Type-1 or Bare-Metal Hypervisor
2. Type-2 or Hosted Hypervisor

Type 1 hypervisor also called bare-metal hypervisor, directly runs on the top of the hardware.Guest OS is hosted on the hypervisor.In this approach hypervisor has direct control over the hardware.
Type 2 hypervisor also called Hosted hypervisor is hosted on an OS and guest OS is hosted on the layer above the hypervisor.In Type 1 hypervisor has more control over hardware as compared

to Type 2.

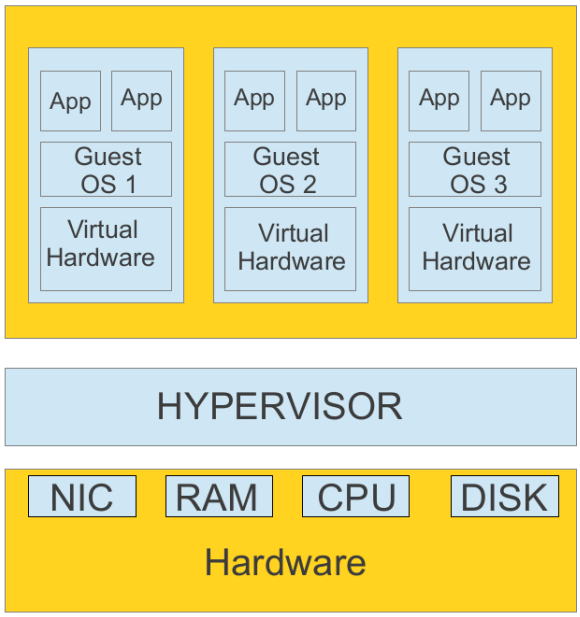

Figure 7: Type 1 Hypervisor

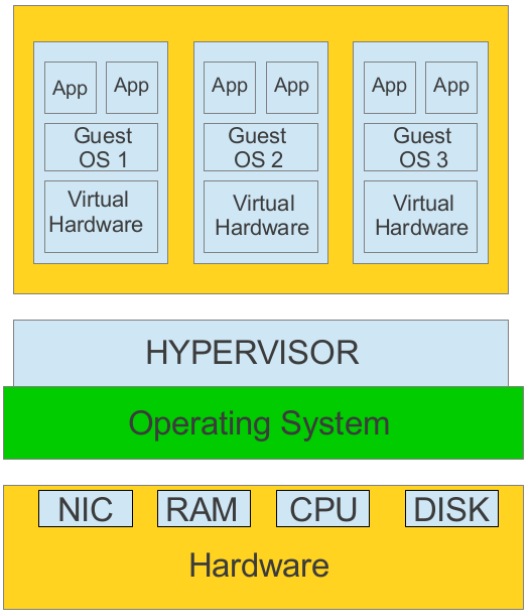

Figure 8: Type 2 Hypervisor

2.3 Scope of this Seminar

I am looking at basically two types of research papers in this seminar.Firstly I am looking at papers which are exploiting event monitoring capabilities on a VM.We can to use this monitoring to replay the execution.We are recording the events occurring in the VM and then we want to replay the execution for that VM.Record and replay of VMs has lots of use cases.If we are using some approach where we are doing recording and replaying efficiently then R&R can be used in application diagnosis[11], Application debugging,VM based intrusion detection[7] etc. Performance in native environment and virtual environment is not same.Virtualization imposes some extra overhead on the machine. Normally for a machine there is no way to get if it running in native environment or in virtual environment.So one of the problem to solve is to get information about the machine if its running in virtual environment or in native environment.Beside that there are some security problems if two VMs are on same machine.One VM can exploit this co-residency to get information about the another VM.So anyone can be interested in knowing about the co-residency of VM.If there are other VMs which are running on same PM. In the second half of the report I will be talking about the various proposed approaches to get about the VM co-location detection.

3 VM based Excution Recording and Replaying

3.1 Motivation

Physical machines consists of different kind of hardware resources and in the native system OS executes directly on the hardware and its very difficult to log the events required to do the execution replay.But execution in virtual environment is more controlled and most of external devices are accessed through hypervisor and its comparatively easy to record events in case of virtual environment.But one the main question to ask is Why we are interested in Execution Replay of VMs.There are different types of problems which can be solved with the help of record and replay of VM.Record-replay has many useful application, ranging from fault tolerance and forensics to reproducing and diagnosing bugs or to get the information about the intrusion in the system. By replaying the execution we can find the bug in the application.Suppose we want to know about the abnormal behavior of an application then we can replay the execution again and again to get the idea about the bug in the application.Likewise we can get when or due to which part of the code our VM's security has been compromised.So we can find about the intrusion in the system using record and replay.

3.2 Record and Replay of VMs

In Record and replay of execution we log events which are happening during the execution and then using that information we replay same set of events.In a machines there are lots of events which happens to properly execute some process or application.In the record phase we have to record events in such a way that we can replay the execution deterministically.This is almost impossible in case of native system but in case of VM, execution is more controlled and we can record events generated by various type of devices.We need not to record the deterministic events as these events can be generated normally executing the process or application.In the next few section we will look at what to record and how to do replay to get the desired information to solve out various types of problems.Replay stage depends on record stage,if we are not doing recording properly then there may be problems in the replay phase.While recording any tool should keep these things in mind.

1. Size of the logged data should not be large.
2. It should record events which are necessary depending on the application where we are using record and replay.
3. Recording phase should not interfere with the other application execution.
4. Overhead in terms of CPU cycles should be less.

In the next few section we will learn about Record and Replay.

3.3 VM Recording

Recording or logging is done on events and while replaying,this log is used in the forward direction of events.The type of application where we are using record and replay determines the type of information that needs to be recorded.In a process execution most of events are of deterministic type and we need not to record these events as the process can re-execute these events in the

same way during replay.That is why we are concerned only about the non-deterministic events.In these kinds of events we can put external device interrupts.Just getting the information about the non-deterministic events is not sufficient if we don't know about the exact timing information that when this event occurred.Timing information is needed to insert the same event at the same time in replaying.In figure 9 we can see that we are recording timestamp and the log information related to the event.

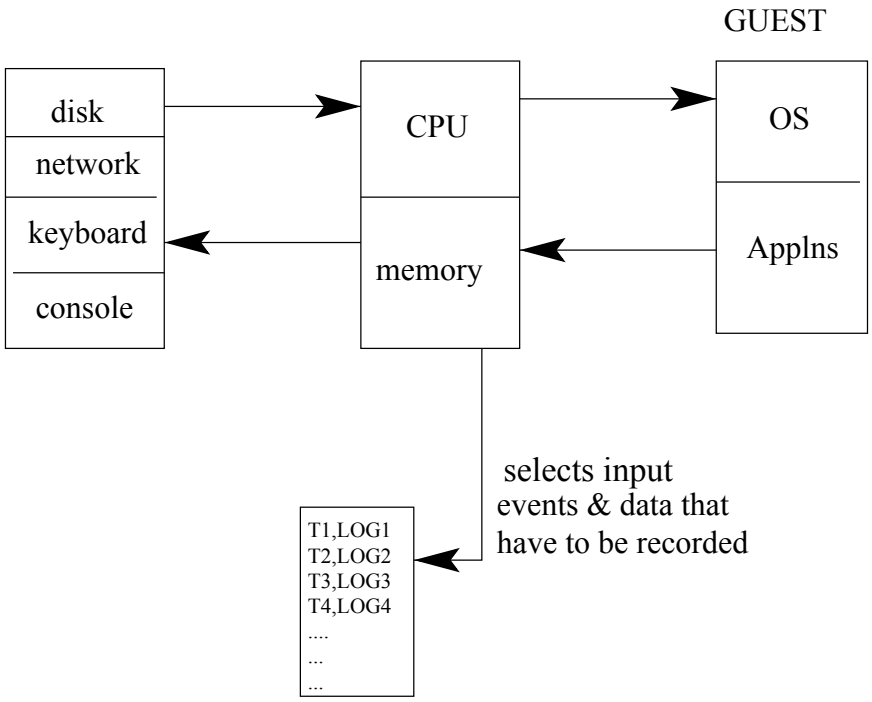

Figure 9: Basic Recording

Broadly non-deterministic events fall into two categories: Time and external input.External output to devices does not affect the replaying process so we don't save such events in our record.

3.3.1 Recording Nondeterministic Events

When an interrupt occurs hypervisor get to know about the interrupt and we can easily record the event with the help of hypervisor but what about the timing information.We can't use clock time to record the events because it depends on many different factors for example what if in replaying process scheduling of processes is not in the same way as in case of recording.Moral of the story is we can't use clock time for timing information.To get the exact time when the event occurs we store Timestamp as "branch counter,instruction pointer,ECX value ".And with this we use the same information in case of replaying process.With the help of these three value we can uniquely identify the point where the interrupt occurred at recording time. Recording phase must log only those non-deterministic events which can affect the execution of the Virtual Machine process. Some of the non-deterministic events at host need not to logged unless those interrupt signals are not sent to the process in the VM.For example scheduling order of other processes at the host doesn't affect the process execution in the VM. Before delivering the interrupt to the virtual machine process enough information is logged to re-deliver the interrupt at the same point during replay.In addition with the external interrupt information,input from the external device is also logged.However we try to minimize the amount of logged information

for example if we are keeping the image consistent then we need not to log the input from the hard disk because that we can get from the disk while replaying.

3.3.2 About Timestamp

We have already looked at the need of the timestamp in the replay phase and it has already been stated that why we can't use normal clock time as a timestamp.To identify any point in execution we use three tuple timestamp of (Branch count, Instruction pointer, Value of ECX).Which statement in the program is executing can be identified with the help of instruction pointer but in case of a branch statement instruction pointer is loaded with a new value every time.So we can't use only instruction pointer to uniquely identify the point where interrupt occurred in the record phase.To identify a point at which an interrupt has occurred, we use number of branches taken so far and instruction pointer at the instance of interrupt. In Intel we can repeat an instruction 'x' number of time by specifying the value of 'x' in ECX register.Say we want to execute an instruction 50 times then the value of ECX register is set to 50 and every time the instruction is executed ,value of ECX register is decremented by 1 and the instruction will be repeated till the value of ECX register becomes 0.What is the need of value of ECX register in timestamp ? To capture how many iteration were pending when event occurred,we record the value of ECX . Using the ECX value we can wait till the point of execution to inject events.

3.4 VM Replaying

With the help of recorded information we now execute the same process in a controlled manner.All the deterministic events happened in the normal execution.As we have timestamp information in our recorded data,we insert external interrupt exactly at the same time, it occurred in the record phase.Likewise VM replaying is done and VM replay can be used for different applications.As written earlier we record information according to the need ,which somehow depends on the usage of the VM record and replay.In the replay phase all the external devices are accessed in a controller manner .For example we are not allowed to give input from the keyboard as an input to the process etc.In figure 10 we can see that if in the recording phase some input is given to the application then in replaying phase we will give this input from the log file and not from the IO device.Only selected data is sent to the device,remaining is sent to the sink and logged information is used to replay the execution.

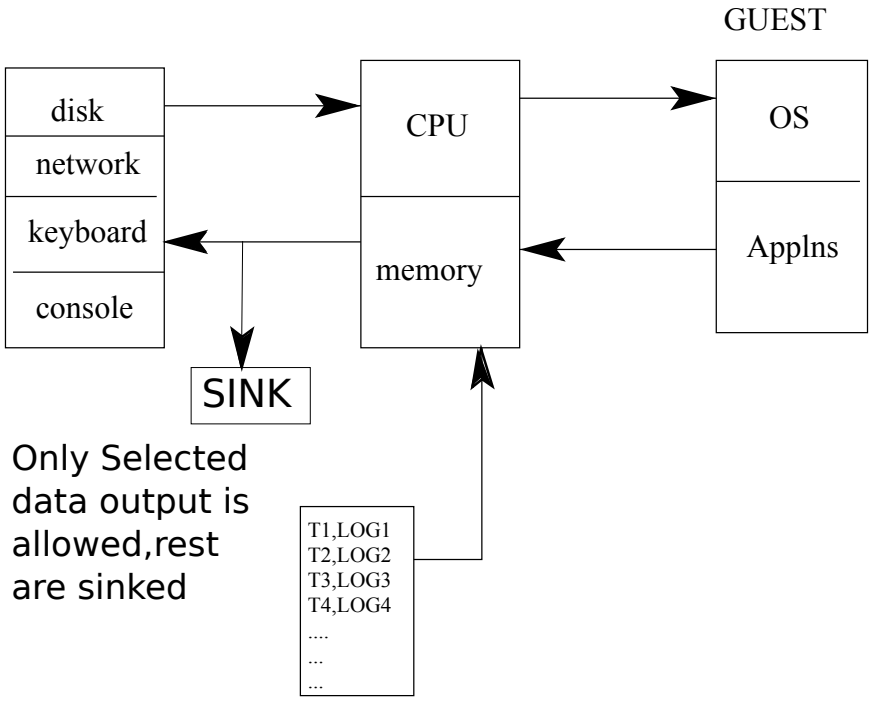

Figure 10: Basic Replaying

3.4.1 Replaying Events Deterministically

After recording of non-deterministic events,the execution is done in a controlled environment for replaying i.e external events to the CPU are given in a controlled manner.The process or application is executed normally on the CPU and with the help of recorded information and whenever time stamp reaches to the point of interrupt.This interrupt is fed to CPU in a controlled manner.Whenever CPU ask for some value as an input ,this value is fed from the recorded information instead of taking it from the input devices. The other challenge in the replaying phase may be delivery of the network packets.In the recording phase also packet consumption may be non-deterministic.In the next section we will see how to delivery non-deterministic arrived network packets to the process deterministically.Beside this we may talk about various other devices like disk drive,keyboard,registers etc. If the disk image is consistent during record as well as replay phases then we can perform reads and writes to the disk as we do in normal execution as the final result in both the cases will be the same.In replay phase input from the keyboard is not allowed If there is some point in process execution where it need input then instead of giving input from the keyboard ,input is given from the record file,which was recorded in the record phase.

3.5 Use Cases of Record and Replay of VM

There can be many use cases ,where we can use record and replay.We are talking about some of the use cases in this section. [7] explains that how we can use record and replay for intrusion analysis.While in [11] record and replay is used for application diagnosis.[6] explains how high-availability can be achieved in virtualization environment. Though this solution is not based on record and replay, the basis of the solution relies on state synchronization across two machines.

3.5.1 Intrusion Detection

Application level logging is not enough to get about the intrusion in the system or some application is affected by the intrusion.As the system can be compromised and information related to the intrusion can be modified by the intruder.But in case of virtualized environment we have full control over the execution and with the help of record and replay we can get about the intrusion in the system.First we will record the execution and with the help of recorded information execution is replayed instruction by instruction.If we see some suspicious activity we can replay the execution from that point.[7] explains how we can use record and replay for intrusion detection.The same setup can be used for debugging purpose also.

3.5.2 Application Diagnosis

[11] talks about how record and replay can be used for application diagnosis.As a use case they are using record and replay for lock contention detection and possible deadlock detection.First they talk about the events they are interested in and with the help of wrapper libraries.Wrappers are created by using the LD_PRELOAD environment variable.With the help of these wrapper libraries information about the events of our interest is given to the diagnosis system.First information is gathered about the application and then this information is used for the possible deadlock detection.In the recording phase only nondeterministic events information is logged and recording is done in the execution phase.In the replaying phase if some event occurs which is of our interest then with the help of event probe information about this event is sent to the diagnosis system.The diagnosis system collects and saves the information in the event database.Then diagnosis tools can process on this database.Till now [11] is using R&R for lock contention analysis and potential deadlock detection.In this they are interested in the lock related events i.e Lock(),Unlock(),TryLock(),TimedLock() etc and after getting every information in the event database,Diagnosis tools can use this information to make Wait-for graph get about the lock dependencies and the potential deadlock.

3.5.3 High Availability

[6] explains how high-availability can be achieved in virtualization environment. Though this solution is not based on straight record and replay, the basis of the solution relies on state synchronization across two machines. Replication may be achieved by copying the state or by replaying input deterministically but second approach is not feasible in case of multi-processor systems.Remus runs paired servers in an active-passive configuration.At primary,for each epoch changes in the state of the system is buffered and as an optimization execution is started after buffering and in the next step this buffered state is transmitted and stored on the backup.Backup sends an acknowledgement to the primary and then primary releases the outgoing network packets.Instead of recording every non-deterministic events,we capture the state change for a period and instead of replaying the logged events, we apply the state change on the backup in such a manner that backup can take control if primary goes down.

3.6 Evaluation

In record and replay some question are generic enough that can be asked for every technique we use for record and replay.

1. How to show the correctness of record and replay system.

2. How much overhead record and replay system is imposing on the system.Since log file will use space so we are interested in space requirement of record and replay.

3.6.1 Correctness of the system

In InSight[11] They are using top command to show the correctness.They used the top command and recorded the events and then using that recorded information replaying is done and in the end results were identical.In the second experiment they used CPU intensive work load(Syn_cont) and compared various parameters like context switches,major page fault,execution time.All the parameters value were identical in record and replay phase. In ReVirt[7] They booted up the computer and built two applications.There were total 15,000,000 function calls and 55,000 virtual interrupts.And ReVirt replayed it same as the run in the record phase.

3.6.2 Record and Replay Overhead

For [7] both CPU overhead as well as memory overhead is not much.Because recording of events doesn't impose much CPU overhead and that is true for both InSight[11] & ReVirt[7].While space overhead for InSight[11] is much higher than ReVirt[7] because in ReVirt[7] they are not recording network packet data as well as they need not to consider DMA data which is a considerable overhead. In InSight[11] they are recording both network packet data as well as DMA data.What to store and what not to store in the log file also depends on the use case i.e for what we are doing record and replay.

4 VM Co-location Detection

4.1 Background

If two or more VMs are on same PM then we can say that one VM is co-located with the other VM and process of detecting about this co-location can be said as VM co-location detection.There may be different questions which can be answered by co-location detection but we are interested in :

1. Am I running in a VM ? [16]
2. Is there another VM on same PM ? [16]
3. Can I influence another VM on same PM ? [12]

First of all, why we are interested in giving answer to such question ?
There is a difference in performance while executing in a VM and native environment.So anyone can be interested in giving answer to the question whether OS is running in a VM or directly on the hardware.In case of Cloud services,some clients don't want to put his VM with some another VM or another VM from different organization,in that case VM co-location detection can be used to check whether his VM is placed according to his requirements or not.Some intruder may be interested in placement of his VM with some target VM and after placement ,he can get information from the other VM with the help of cache based side channel. (e.g.,[10, 9])

4.2 Approaches to Co-location Detection

Depending on the situation we can use different approaches to detect co-location.[12] is interested to answer about Can we get information from another VM co-located on same VM ? While [16] answer the question,Is my VM is co-located with VM which doesn't belong to me or with the VM I don't my VM to be co-located with ?In the first case authors map the IP address distribution of Amazon's EC2 and with the help of this mapping they are placing their VM on the same machine on which the target VM is already present.As a co-location confirmation i.e their VM and target VM are on same PM,they are doing network based co-residence check:

4.2.1 Network Based Co-residency Check

In [12] they are using this approach.

1. With the help of trace-route get the IP address of the Dom0 on which target VM is placed.
2. Check the IP address of the Dom0 on which our VM is places with the help of trace-route.
3. If IP addresses in both the cases are same then both the VMs are located on same PM.

if the Dom0 is not responding to trace-route then this technique can't be used for co-residence detection.

4.2.2 Side-Channel based Co-residency Check

[16] detect co-location in the cloud via side-channel analysis.Some of the other work shows that side channel in shared hardware can also be used to extract sensitive data across VMs.But in this paper side-channel(in L2 cache) is used to detect undesired co-residency.They are using Prime-Probe Protocol for this purpose :

1. PRIME : Entity U fills an entire cache set S by reading memory region M from its own memory space.
2. IDLE : Entity U waits for a pre specified PRIME-PROBE interval while the cache is utilized by monitored entity V.
3. PROBE : Entity U times the reading of the same memory region M to learn Vs cache activity on cache set S

Where U is the monitoring entity and V is the monitored entity.If V do some activity in the prime-probe period the it is very much possible that U's data will be evicted from the cache and will be replaced by the data accessed by the V which will result in higher access time in Probe phase. However some other factor may also affect Prime-Probe protocol,like:

1. TLB misses : If TLB miss occurs then it may add noise to the timing measurements in the Probe phase.
2. Hardware Prefetching : Several page lines from the memory page is prefetched before execution this can also add significant noise to the timing measurements.
3. Multi-core Architecture: If the system we are executing on is the mutli core then it may be possible that after a context switch entity U is scheduled on some other core,it give a false positive or in some cases L2 caches can be shared.

4.3 Possible Attacks due to Co-residency

There can be many types of attacks possible to get the information from physically co-located VM.Some of them are enlisted here:

1. Extracting cryptographic key using cache based side channel [9]
2. Keystroke Timing Attack [14]

4.4 Usage

If our VM is co-located with the target VM i.e physical hardware for both the VMs is same then we can exploit this residency. There is lot of work has been done in this field.At present our concern is VM based co-location detection and not how to get information from the another co-located VM.For example [17, 9, 14] explains the work where we can exploit this residency to get the information about the another VM.So one use case may be for attackers.After getting information about the physical co-location he can use some method to get the information from foe VM located on same PM. VM co-location detection can also be used by the clients to verify its VM's exclusive use of the physical machine.Since we know that physical co-location of VMs may cause problem or may extract information from our VM and he asks cloud provider to give

an exclusive machine for use.Now he can verify with co-location detection also whether he got exclusive machine or not.

5 Conclusion

Here in this seminar my aim was to get about the problems and solution in the areas I was looking at.However due to time constraint I read only few of them.In "Record and Replay" related papers, I have seen papers which provides efficient recording and replaying for uniprocessor system but in case of multi-processor system this approach doesn't seem to work.Beside that ECX register we is used in a timestamp tuple is specific to IA-32 architecture and I have no clue about the other counterpart for other architectures.I have came across the paper related to only few use cases.Most of my readings were about ideas and basic algorithms related to record and replay but low level implementation details were not given.[11] is the first paper which allow recording for both network packets as well as DMA.But it will increase the size of log file.On the other hand [7] is also using the same approach but they are not saving the content for a network packets and DMA events.InSight[11] is based on KVM[8] hypervisor and they are using Linux as an OS.While in ReVirt [7] they are using UMLinux hypervisor and Linux as a Guest OS.

In "VM Colocation Detection" related papers,most of them are related to security.Main concern was security i.e if two or more VM are co-located on a single PM then security breach may be a problems for such VM and I could not find any other possible use case.One more possible usecase can be verification of exclusive use of a PM.

References

[1] *Understanding fullvirtualization,paravirtualization, and hardwareassist*, `http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf`.

[2] *Understanding the cloud computing stack saas, paas, iaas*, `http://broadcast.rackspace.com/hosting_knowledge/whitepapers/Understanding-the-Cloud-Computing-Stack.pdf`.

[3] *Vm*, `http://en.wikipedia.org/wiki/VM_(operating_system)`.

[4] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, and Matei Zaharia, *Above the clouds: A berkeley view of cloud computing*, Tech. report, 2009.

[5] Paul Barham, Boris Dragovic, Keir Fraser, Steven H, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield, *Xen and the art of virtualization*, In SOSP (2003, 2003, pp. 164–177.

[6] Brendan Cully, Geoffrey Lefebvre, Dutch Meyer, Mike Feeley, Norm Hutchinson, and Andrew Warfield, *Remus: High availability via asynchronous virtual machine replication*, In Proc. NSDI, 2008.

[7] George W. Dunlap, Samuel T. King, Sukru Cinar, Murtaza A. Basrai, and Peter M. Chen, *Revirt: Enabling intrusion analysis through virtual-machine logging and replay*, In Proceedings of the 2002 Symposium on Operating Systems Design and Implementation (OSDI, 2002, pp. 211–224.

[8] Avi Kivity, *kvm: the Linux virtual machine monitor*, OLS '07: The 2007 Ottawa Linux Symposium, July 2007, pp. 225–230.

[9] Dag Arne Osvik, Adi Shamir, and Eran Tromer, *Cache attacks and countermeasures: The case of aes*, Proceedings of the 2006 The Cryptographers' Track at the RSA Conference on Topics in Cryptology (Berlin, Heidelberg), CT-RSA'06, Springer-Verlag, 2006, pp. 1–20.

[10] Colin Percival, *Cache missing for fun and profit*, Proc. of BSDCan 2005, 2005.

[11] Senthilkumaran R and Purushottam Kulkarni, *Insight: A framework for application diagnosis using virtual machine record and replay.*

[12] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage, *Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds*, Proceedings of the 16th ACM Conference on Computer and Communications Security (New York, NY, USA), CCS '09, ACM, 2009, pp. 199–212.

[13] Jim Smith and Ravi Nair, *Virtual machines: Versatile platforms for systems and processes (the morgan kaufmann series in computer architecture and design)*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[14] Dawn Xiaodong Song, David Wagner, and Xuqing Tian, *Timing analysis of keystrokes and timing attacks on ssh*, Proceedings of the 10th Conference on USENIX Security Symposium - Volume 10 (Berkeley, CA, USA), SSYM'01, USENIX Association, 2001, pp. 25–25.

[15] Carl A. Waldspurger, *Memory resource management in vmware esx server*, In OSDI, 2002.

[16] Yinqian Zhang, Ari Juels, Alina Oprea, and Michael K. Reiter, *Homealone: Co-residency detection in the cloud via side-channel analysis*, In Proceedings of the 2011 IEEE Symposium on Security and Privacy (2011), 2011, pp. 313–328.

[17] Yinqian Zhang, Michael K. Reiter, and Flow Controls, *Cross-vm side channels and their use to extract private keys*, Proceedings of CCS 2012, ACM Press, 2012.