存储服务质量保障专题

施展 2019

课程信息

- 专题内容
  - 关于虚拟化环境存储系统的研究
    - 一些基础
      - https://github.com/cs-course/vagrant-tutorial
      - https://github.com/cs-course/cluster-monitoring-tutorial
      - https://github.com/cs-course/vagrant-presentation
    - 需要研究的问题
      - 存储服务质量保障

授课目标

- 研究对象
  - 虚拟化环境存储系统
    - 理解虚拟化环境下存储系统的I/O问题
    - 如何在虚拟化环境下保障存储服务质量

做个调查

- 单纯只是安装使用过
  - VirtualBox、VMware、Parallels Desktop
  - XEN、KVM、Qemu
- 学习过虚拟化、云计算课程
  - 云计算与虚拟化 24/1.5（4年级）
- 熟悉Linux且买过云主机
  - VPS（Virtual Private Server 虚拟专用服务器）

第一部分

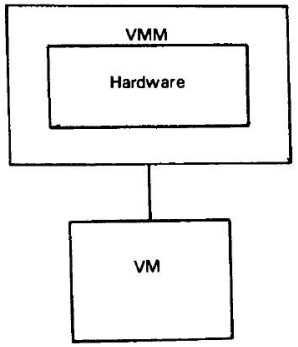
虚拟化环境

带上疑问

- 何谓虚拟化？
- 有什么用？
- 有什么不足？
- 有什么方法克服那些不足？

起源

Fig. 1. The virtual machine monitor.



"an *efficient, isolated duplicate* of the real machine"

- **Efficiency**
  - **Innocuous instructions should execute directly on the hardware**
- **Resource control**
  - **Executed programs may not affect the system resources**
- **Equivalence**
  - **The behavior of a program executing under the VMM should be the same as if the program were executed directly on the hardware (except possibly for timing and resource availability)**

Formal Requirements for Virtualizable Third Generation Architectures

Gerald J. Popek
University of California, Los Angeles
and
Robert P. Goldberg
Honeywell Information Systems and
Harvard University

Virtual machine systems have been implemented on a limited number of third generation computer systems, e.g. CP-67 on the IBM 360/67. From previous empirical studies, it is known that certain third generation computer systems, e.g. the DEC PDP-10, cannot support a virtual machine system. In this paper, model of a third-generation-like computer system is developed. Formal techniques are used to derive precise sufficient conditions to test whether such an architecture can support virtual machines.

Communications of the ACM, vol 17, no 7, 1974, pp.412-421

历史

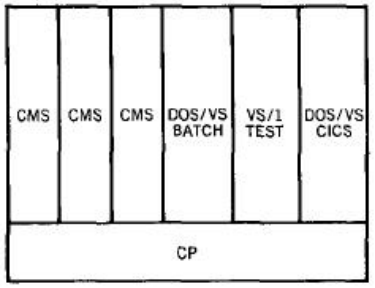
VM/370—a study of multiplicity and usefulness

by L. H. Seawright and R. A. MacKinnon

The productivity of data processing professionals and other professionals can be enhanced through the use of interactive and time-sharing systems. Similarly, system programmers can benefit from the use of system testing tools. A systems solution to both areas can be the virtual machine concept, which provides multiple software replicas of real computing systems on one real processor. Each virtual machine has a full complement of input/output devices and provides functions similar to those of a real machine. One system that implements virtual machines is IBM's Virtual Machine Facility/370 (VM/370).[1]

IBM Systems Journal, vol. 18, no. 1, 1979, pp. 4-17.

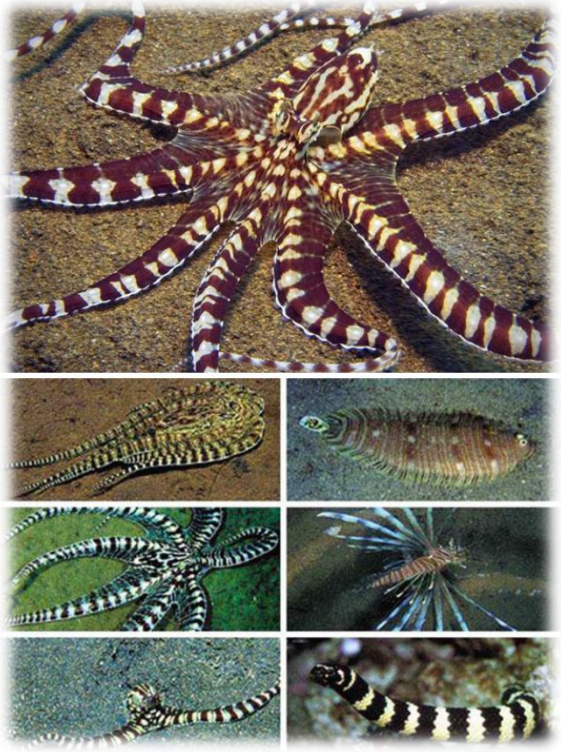
Figure 1 A VM/370 environment



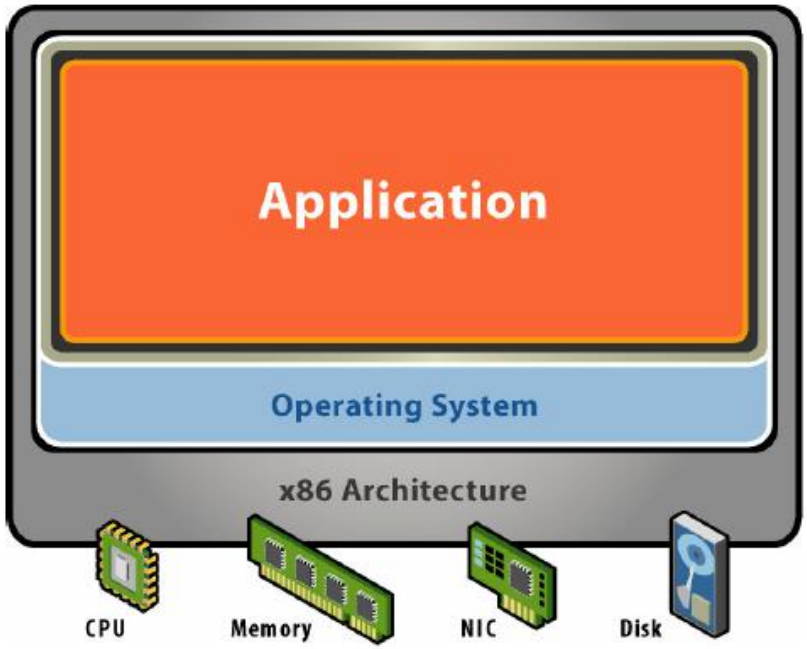
- Concurrent execution of multiple production operating systems
- Testing and development of experimental systems
- Adoption of new systems with continued use of legacy systems
- Ability to accommodate applications requiring special-purpose OS
- Introduced notions of "handshake" and "virtual-equals-real mode" to allow sharing of resource control information with CP
- Leveraged ability to co-design hardware, VMM, and guestOS

虚拟化

- Virtualization deals with
  - "extending or replacing an existing interface so as to mimic the behavior of another system"
- Virtual system examples
  - Virtual Private Network
  - Virtual Memory
  - Virtual Machine



从物理机开始



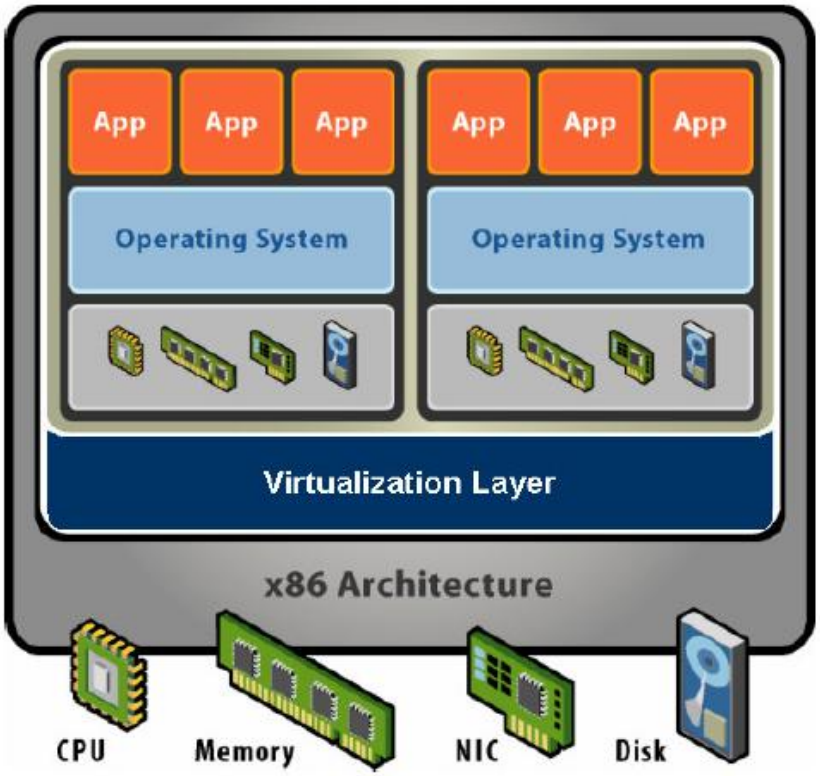
Physical Hardware

- Processors, memory, chipset, I/O bus and devices, etc.
- Physical resources often underutilized

Software

- Tightly coupled to hardware
- Single active OS image
- OS controls hardware

何谓虚拟机？



Hardware-Level Abstraction

- Virtual hardware: processors, memory, chipset, I/O devices, etc.
- Encapsulates all OS and application state

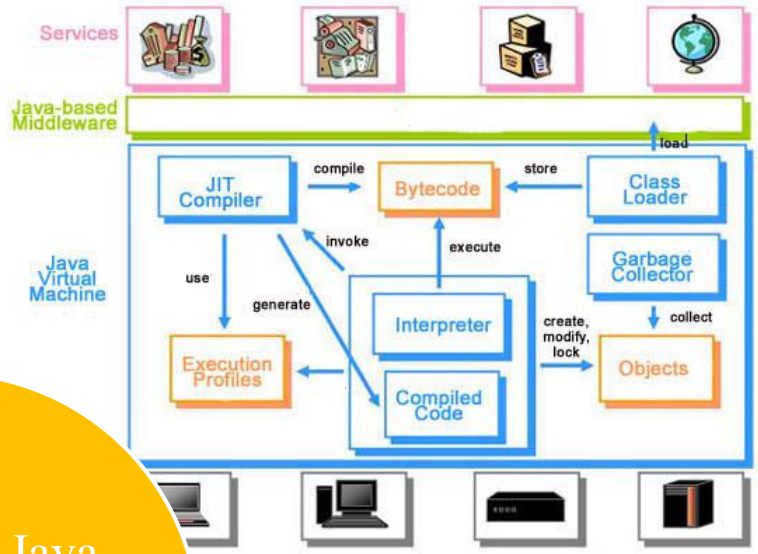
Virtualization Software

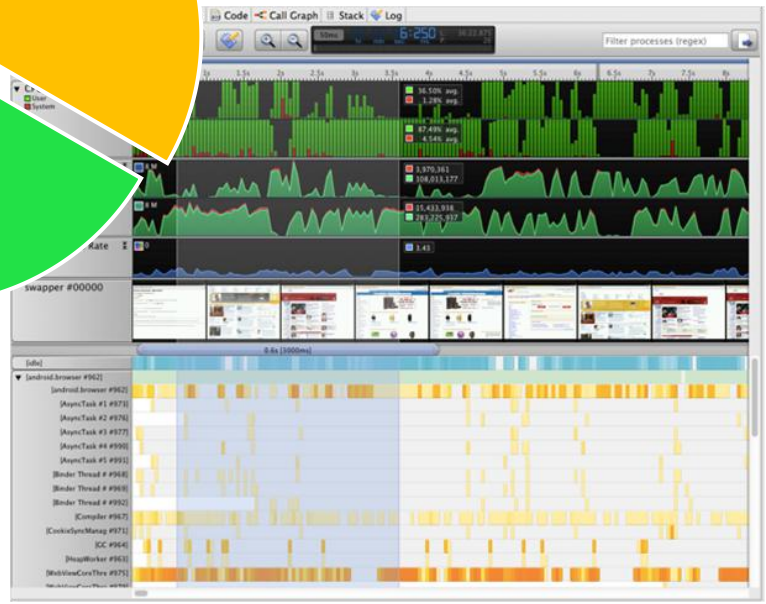
- Extra level of indirection decouples hardware and OS
- Multiplexes physical hardware across multiple “guest” VMs
- Strong isolation between VMs
- Manages physical resources, improves utilization

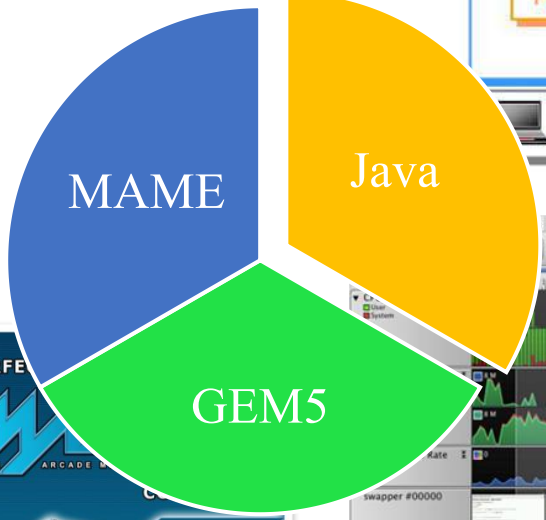
何谓虚拟机？


REVENGE OF
ALLADDIN'S
CASTLE
CYCLONE

THE PERFECT
MAME
COLLECTION
Vol. 1 Contents:
ALPHA
MISSION II
BUBBLE BOBBLE
TAITO
THE KING OF FIGHTERS
'94
SNK
THE PERFECT MAME COLLECTION
1
DVD


Services
Java-based Middleware
load
JIT Compiler
compile
Bytecode
store
Class Loader
Java Virtual Machine
invoke
execute
Garbage Collector
use
generate
Interpreter
create, modify, lock
collect
Execution Profiles
Compiled Code
Objects


Code
Call Graph
Stack
Log
swapper #00000


MAME
Java
GEM5

隔离

Secure Multiplexing

- Run multiple VMs on single physical host
- Processor hardware isolates VMs, *e.g.* MMU

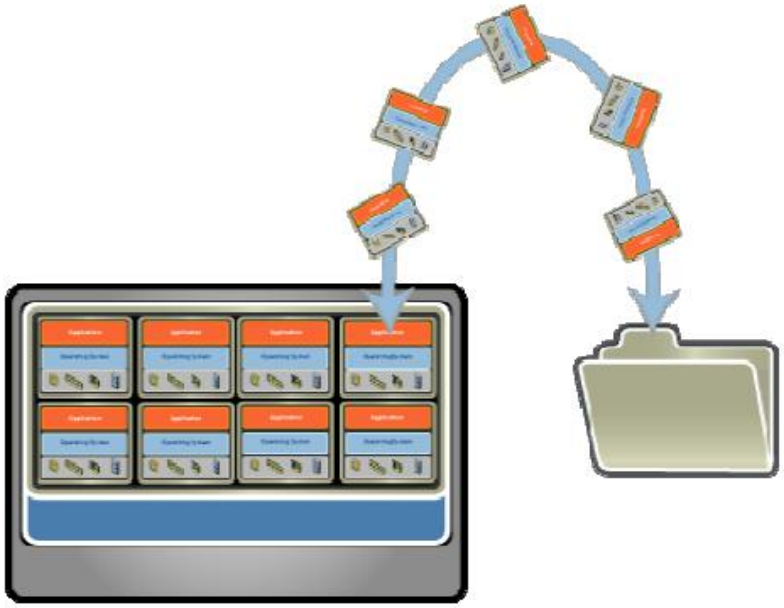
Strong Guarantees

- Software bugs, crashes, viruses within one VM cannot affect other VMs

Performance Isolation

- Partition system resources
- Example: VMware controls for reservation, limit, shares

封装



Entire VM is a File

- OS, applications, data
- Memory and device state

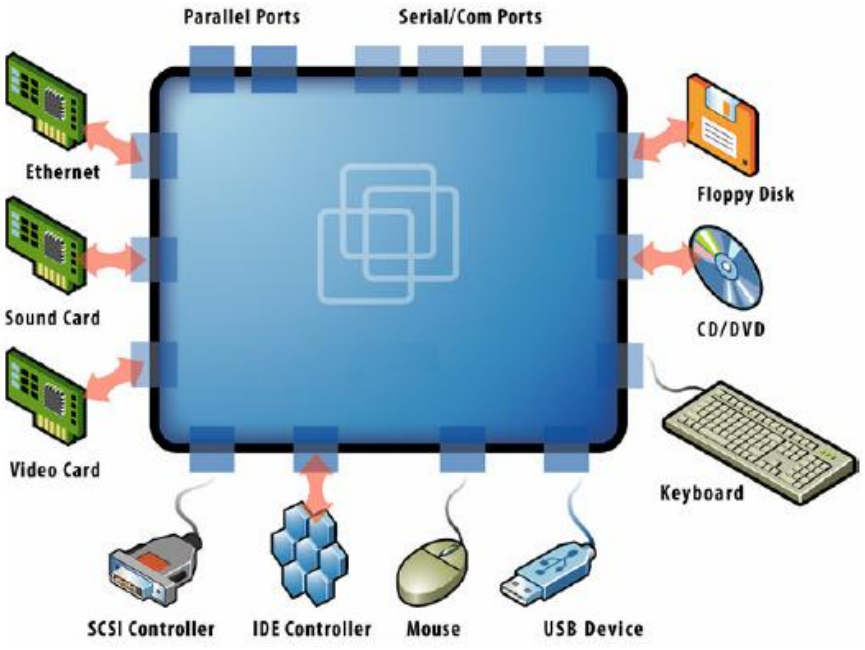
Snapshots and Clones

- Capture VM state on the fly and restore to point-in-time
- Rapid system provisioning, backup, remote mirroring

Easy Content Distribution

- Pre-configured apps, demos
- Virtual appliances

兼容



Hardware-Independent

- Physical hardware hidden by virtualization layer
- Standard virtual hardware exposed to VM

Create Once, Run Anywhere

- No configuration issues
- Migrate VMs between hosts

Legacy VMs

- Run ancient OS on new platform
- *E.g.* DOS VM drives virtual IDE and vLance devices, mapped to modern SAN and GigE hardware

用途

Test and Development – Rapidly provision test and development servers; store libraries of pre-configured test machines

Business Continuity – Reduce cost and complexity by encapsulating entire systems into single files that can be replicated and restored onto any target server

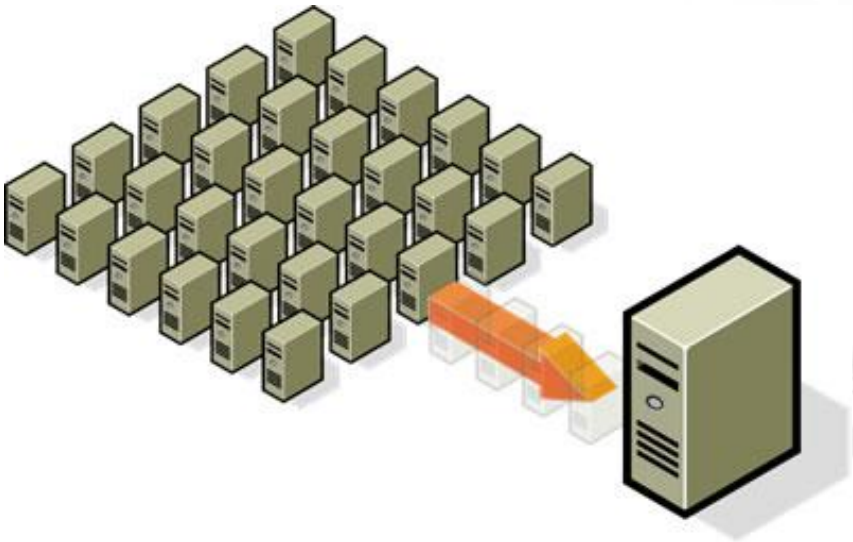
Enterprise Desktop – Secure unmanaged PCs without compromising end-user autonomy by layering a security policy in software around desktop virtual machines

用途

- Run legacy software on non-legacy hardware
- Run multiple operating systems on the same hardware
- Create a manageable upgrade path
- Manage outages (expected and unexpected) dynamically

用途

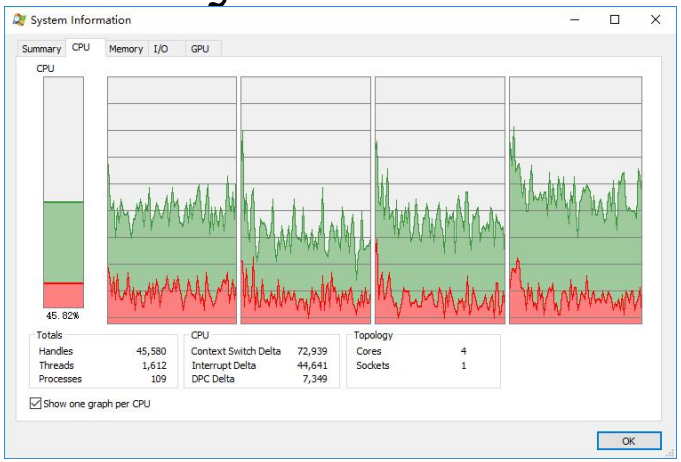
- Reduce costs by **consolidating 整合** services onto the fewest number of physical machines

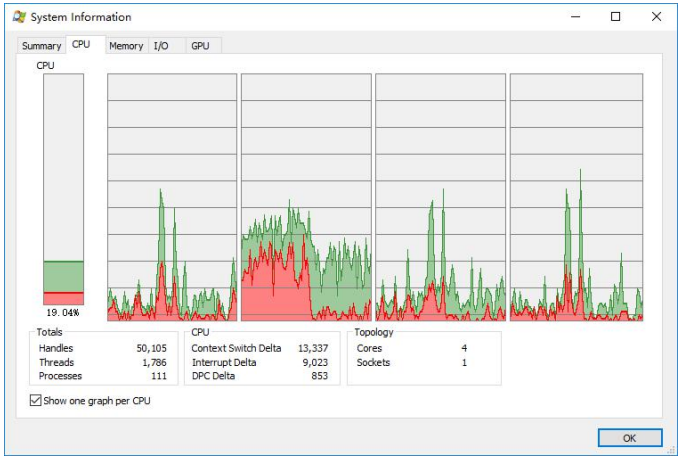


http://www.vmware.com/img/serverconsolidation.jpg

What IF Non-Virtualized

- Too many servers/cores for too little work



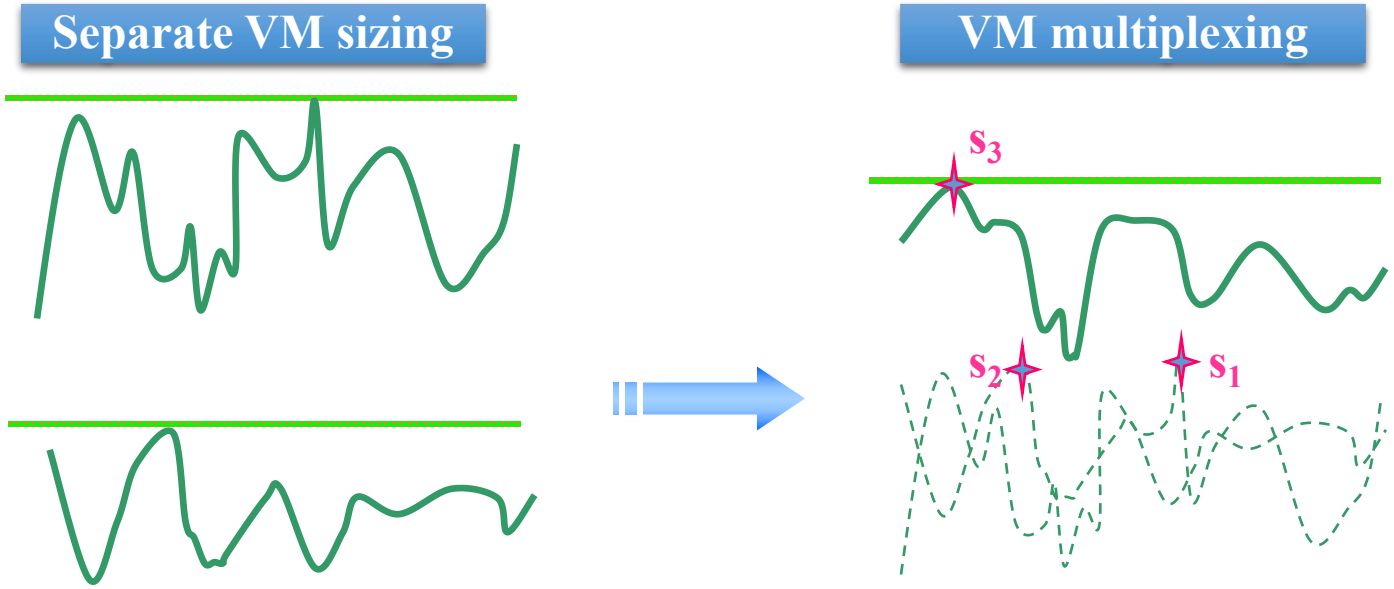


- High costs and infrastructure needs
  - 维护 Maintenance
  - 网络 Networking
  - 空间 Floor space
  - 冷却 Cooling
  - 能耗 Power
  - 容灾 Disaster Recovery

Dynamic

- Virtualization helps us break the "one service per server" model
- Consolidate many services into a fewer number of machines when workload is low, reducing costs
- Conversely, as demand for a particular service increases, we can shift more virtual machines to run that service
- We can build a data center with fewer total resources, since resources are used as needed instead of being dedicated to single services

VM workload multiplexing



We expect $s_3 < s_1 + s_2$. Benefit of multiplexing !

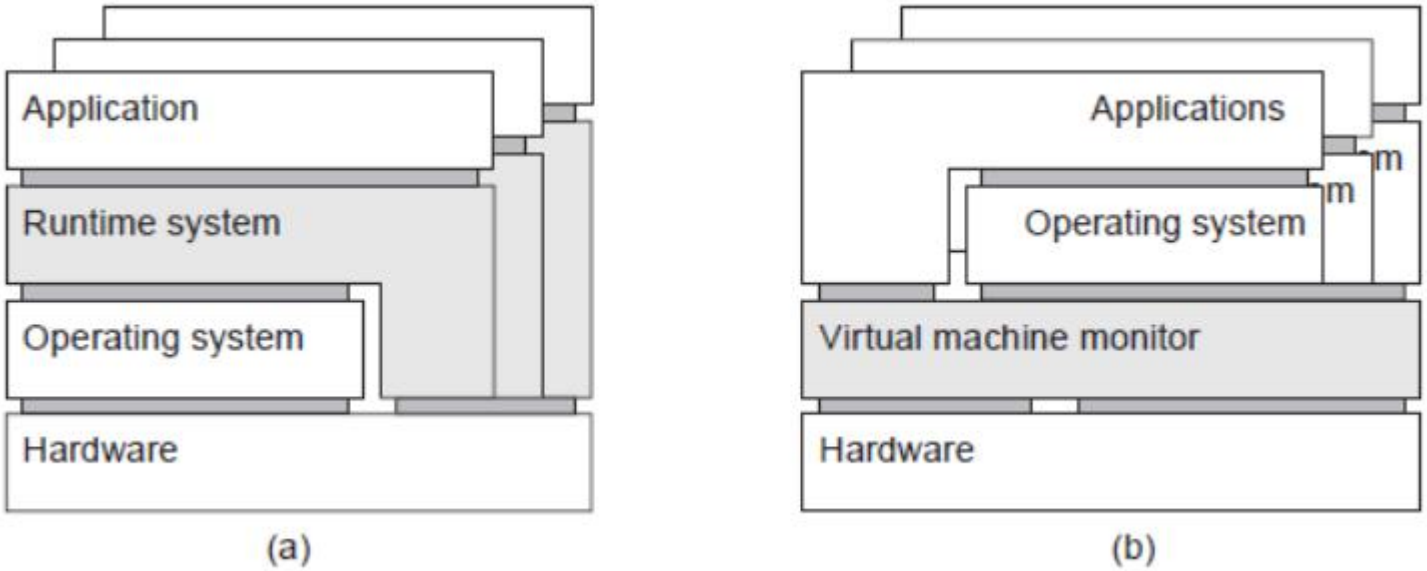
- 工作负载复用
  - Multiplex VMs' workload on same physical server
  - Aggregate multiple workload. Estimate total capacity need based on aggregated workload
  - Performance level of each VM be preserved

虚拟机管理器

A virtual machine is taken to be an *efficient, isolated duplicate* of the real machine. We explain these notions through the idea of a *virtual machine monitor* (VMM). See Figure 1. As a piece of software a VMM has three essential characteristics. First, the VMM provides an environment for programs which is essentially identical with the original machine; second, programs run in this environment show at worst only minor decreases in speed; and last, the VMM is in complete control of system resources.

- Java 虚拟机？
- 显然不是，Java 虚拟机以及各类高级语言虚拟机目标不在于模拟各类机器部件设备
- 虚拟机管理器需要从 CPU 指令层面准确模拟各种硬件

两种虚拟机



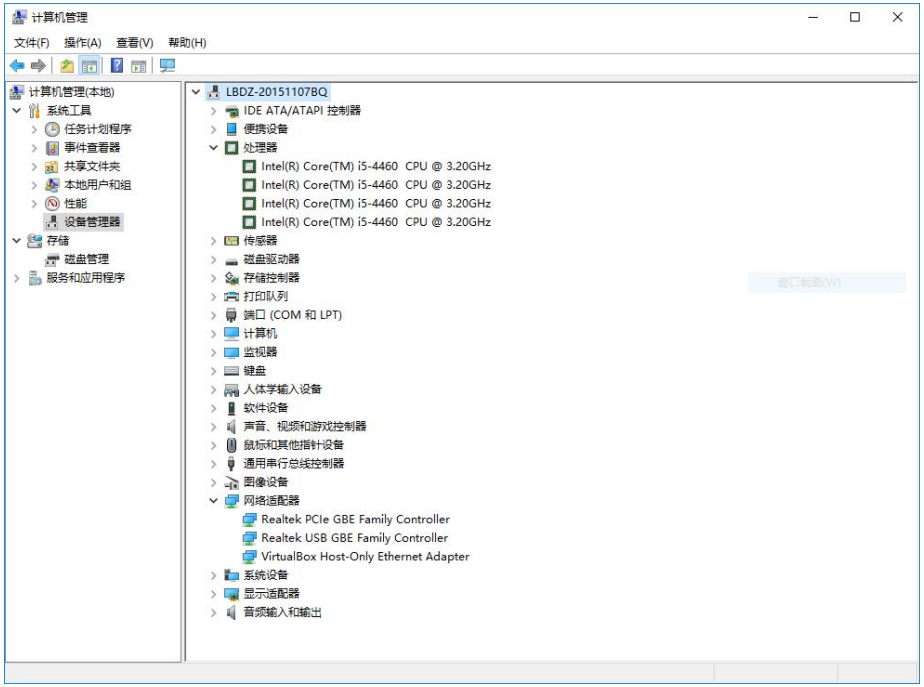
- Process VM 进程虚拟机
  - 程序编译为 Intermediate Code 中间代码交由“Runtime运行时”执行
  - 为编程语言提供“跨平台/平台无关”特性
    - Java VM, Python VM
- VM Monitor 虚拟机管理器
  - 用于模拟另外一套硬件指令集的特殊软件层
  - 足以支持一整套操作系统及其应用软件运行
    - VMWare, VirtualBox, Hyper-V

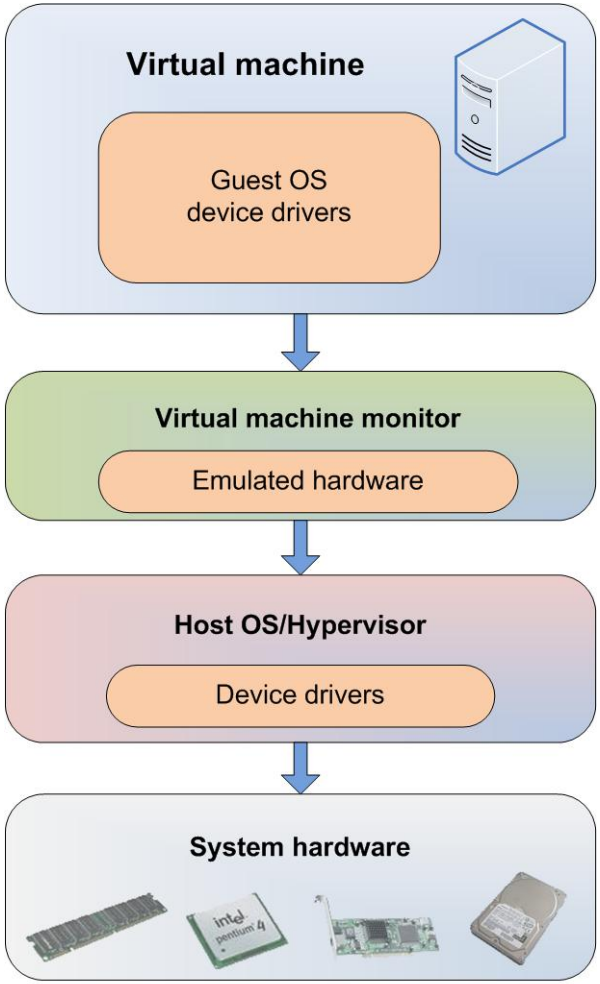
三种虚拟化方法

- 全虚拟化 Full Virtualization
- 半虚拟化 Paravirtualization
- 硬件辅助虚拟化 Hardware-Assisted Virtualization

Full Virtualization

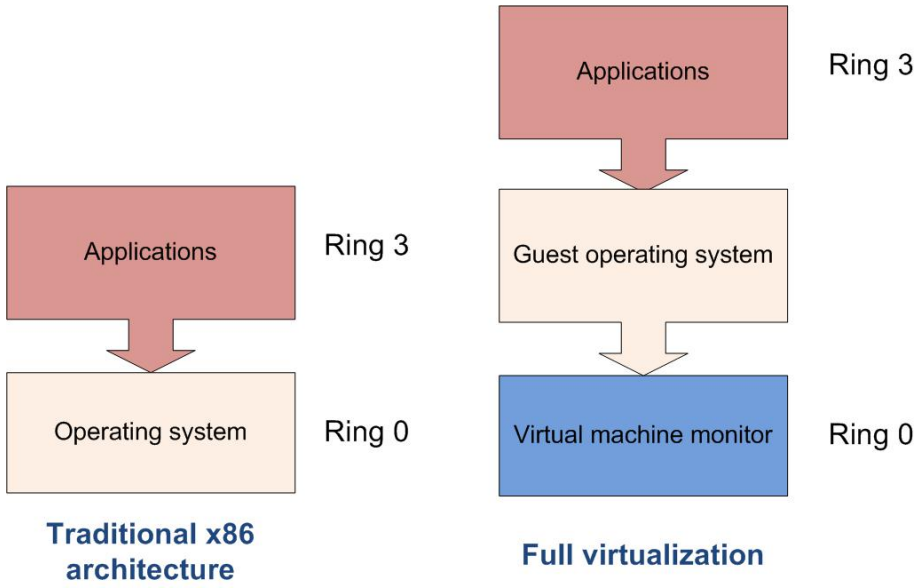
- Everything is virtualized
- Full hardware emulation
- Emulation = latency





Privileged Instructions

- Privileged instructions:
  - OS kernel and device driver access to system hardware
- Trapped and emulated by VMM

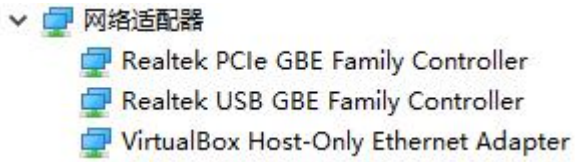


Full Virtualization *Pros and Cons*

- **Pros**
  - Disaster recovery, failover
  - Virtual appliance deployment
  - Legacy code on non-legacy hardware
- **Cons** – LATENCY of core four resources
  - RAM performance reduced 25% to 75%
  - Disk I/O degraded from 5% to 20%
  - Network performance decreased up to 10%
  - CPU privileged instruction dings nearing 1% to 7%

Paravirtualization

- OS or system devices are virtualization aware
- **Requirements**:
  - OS level -- recompiled kernel
  - Device level – paravirtualized or “enlightened” device drivers

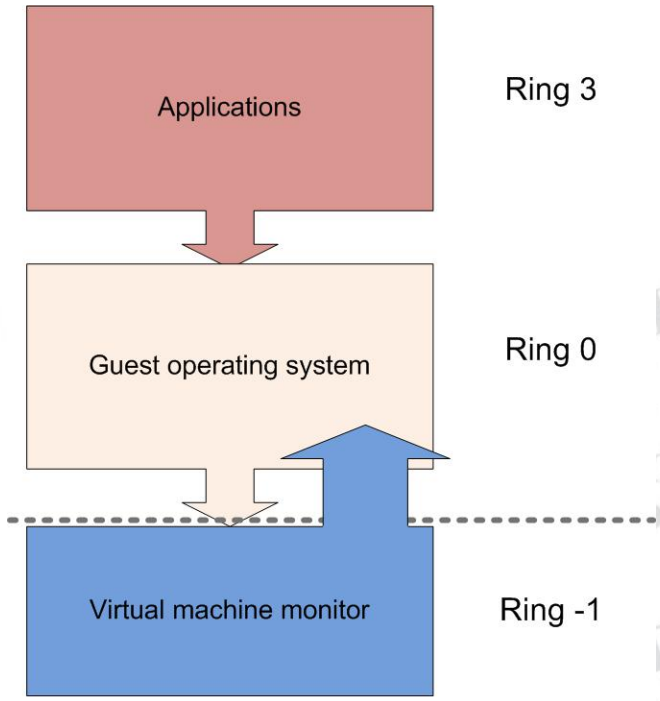


Paravirtualization *Pros and Cons*

- **Pros**: fast!
- **Cons**: requires ***a specially modified guest OS***, thus precludes the ability to run off-the-shelf and legacy OS in para-virtual environments.

Hardware-assisted Virtualization

- Server hardware is virtualization aware
- Hypervisor and VMM load at privilege Ring-1 (firmware)
- Removes CPU emulation bottleneck
- Memory virtualization coming in quad core AMD and Intel CPUs



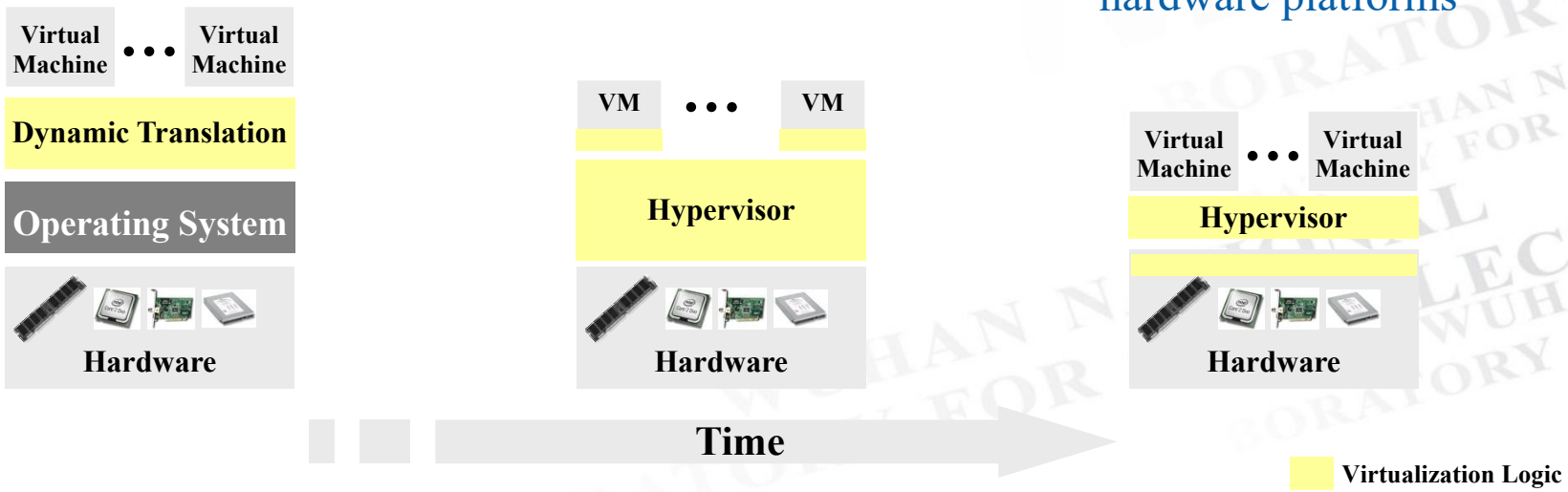
Hardware-assisted virtualization

Evolution of Software solutions*

1st Generation: Full virtualization (Binary rewriting)
- Software Based
- VMware and Microsoft

2nd Generation: Para virtualization
- Cooperative virtualization
- Modified guest
- VMware, Xen

3rd Generation: Silicon-based (Hardware-assisted) virtualization
- Unmodified guest
- VMware and Xen on virtualization-aware hardware platforms



*This slide is from Intel® Corporation

带上疑问

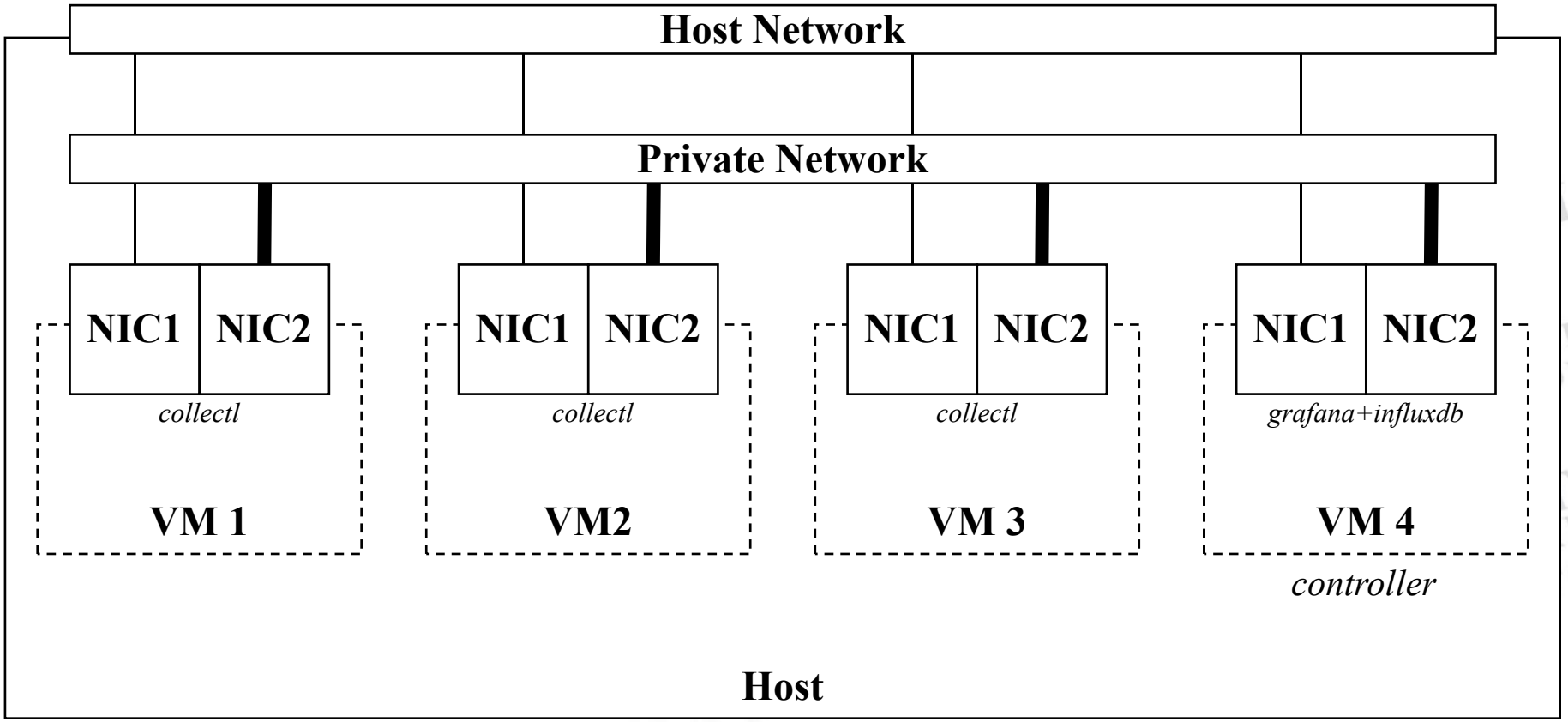
- 如何虚拟化？
- 该怎么用？
- 怎样观察分析？
- 具体怎么处理？

初步实践

- 所需材料
  - VirtualBox(All)/HyperV(Windows10)/Parallels(MacOS)
  - Vagrant
  - https://github.com/cs-course/vagrant-tutorial

小型系统

四机操演


Host Network
Private Network
NIC1
NIC2
collectl
VM 1
NIC1
NIC2
collectl
VM2
NIC1
NIC2
collectl
VM 3
NIC1
NIC2
grafana+influxdb
VM 4
controller
Host

性能观测


vm dashboard
Zoom Out
Last 30 minutes
Refresh every 5s
disk-read
1.0
0.5
0
-0.5
-1.0
11:30
11:32
11:34
11:36
11:44
11:46
11:48
11:50
11:52
11:54
11:56
11:58
2019-09-24 11:36:52
node1.diskinfo.readkbs.sda.mean: 0
node2.diskinfo.readkbs.sda.mean: 0
node3.diskinfo.readkbs.sda.mean: 0
node1.diskinfo.readkbs.sda.mean
node2.diskinfo.readkbs.sda.mean
node3.diskinfo.readkbs.sda.mean
disk-write
40
30
20
10
0
11:30
11:32
11:34
11:36
11:38
11:40
11:42
11:44
11:46
11:48
11:50
11:52
11:54
11:56
11:58
node1.diskinfo.writekbs.sda.mean
node2.diskinfo.writekbs.sda.mean
node3.diskinfo.writekbs.sda.mean

找出问题

- 开监控

```
for i in 21 22 23
do ssh -i insecure-key vagrant@192.168.33.$i \
   'nohup \
      collectl \
         -i1 -scdmnstxyCDMNTXY --dskfilt sd -f /vagrant_data/collectl --rawtoo \
         --export graphite,192.168.33.127:2003,d=1,s=cdmnstxyCDMNTXY > /dev/null &'
done
```

- 上负载

```
# 分别在node{1..3}上，同时执行(tmux同步输入命令)，观察仪表盘，分析资源隔离情况
dd if=/dev/zero of=/dev/null bs=$(( 1024*1024*1024 )) count=2
```

```
# 分别在node{1..3}上，同时执行(tmux同步输入命令)，观察仪表盘，复现前述资源冲突
dd if=/dev/zero of=test_2GB bs=$(( 1024*1024*1024 )) count=2
```

avg disk write

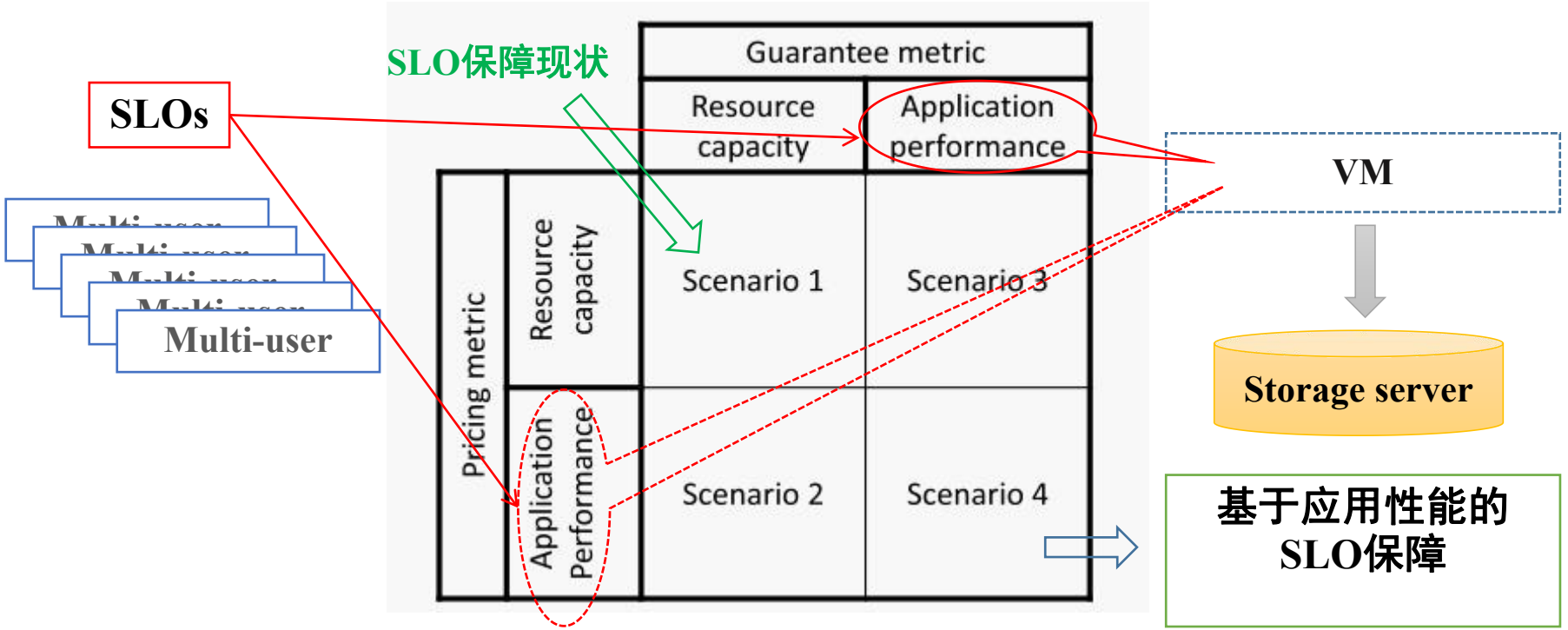
Time	node1.diskinfo.writekbs.sda.mean	node2.diskinfo.writekbs.sda.mean	node3.diskinfo.writekbs.sda.mean
2019-09-24 12:05:00	1.03	455.42	1.18
2019-09-24 12:00:00	1.01	1.14	1.14
2019-09-24 11:55:00	1.00	1.14	1.13
2019-09-24 11:50:00	0.99	1.11	1.14
2019-09-24 11:45:00	1.01	1.14	1.14
2019-09-24 11:40:00	1.01	1.14	1.15

意味着什么

- 直观认识
  - 物理机开一台就是一台
  - 虚拟机开一台 … 不见得有一台的资源
- 严格描述： 服务等级协议SLA *Service Level Agreement*
  - SLA含多个服务等级目标SLO *Service Level Objective*，每个SLO规定一项性能需求的量化指标。
  - 越来越多的企业级用户将应用部署在大规模的虚拟机集群中，虚拟机的SLO保障水平对用户应用的服务质量而言至关重要。

进一步剖析

现有的企业级数据中心或私有云的解决方案主要还是依据应用性能峰值（最坏情况下）的资源需求量作为参考来进行资源供给以保障应用的SLO



不足：

- 造成SLO保障中的资源过量供给
- 无法充分挖掘现有资源优化VM性能

因此

- 如何尽可能避免性能保障中的资源过量供给，用必需而不过量的资源保障虚拟机性能达到用户要求的SLO，即SLO精确保障的问题。
- 在SLO 精确保障的基础上怎么进一步利用可用的资源优化虚拟机性能，即SLO约束下的性能优化问题。

本讲小结

- 虚拟化环境存储系统基础知识
- 实验环境
- 服务质量保障问题