

VMWORLD 2006



PERFORMANCE TROUBLESHOOTING

November 7-9, 2006



CONTENTS

Contents 2

Introduction 4

 Formatting 4

 Lab Sections..... 5

 Timing Issue in Virtual Machines..... 5

 Resource Pool Conflicts 6

 CPU Performance 6

 Storage Performance 6

 Lab Architecture 6

 Login Information 7

 Pre-Lab Exercises..... 7

Section 1: Timing Issues in Virtual Machines..... 10

 Background Information..... 10

 Lab Practice 10

 Timing in a VM without Accounting for De-scheduled Time 11

 Timing in a VM - Accounting for De-scheduled Time with the *vmdesched* Driver 13

 Lessons Learned 15

 Reproducing This Lab at Home..... 15

Section 2: Resource Pool Conflicts..... 16

 Background Information..... 16

 Resource Pools, Shares, Reservations, and Limits 16

 Performance Charts 16

 Initial Setup 16

 Initial Configuration 17

 Lab Practice 17

 Step 1 17

 Step 2 18

 Step 3 19

 Step 4 21

 Step 5 21

 Step 6 23

 Step 7 24

 Step 8 25

 Step 9 26

 Step 10..... 26

 Step 11..... 27

 Step 12..... 28

 Lessons Learned 28

 Reproducing This Lab at Home..... 28

Section 3: CPU Performance..... 29

 Background..... 29

 Lab Practice 29

 Lessons Learned 35

 Reproducing This Lab at Home..... 35

Section 4: Storage Performance..... 36

 Background information 36

 Lab Practice 36

 Step 1 36

 Step 2 38

 Step 3 40

 Step 4 41

 Step 5 44

 Split Data Stores..... 45

 Hitting the Bounds of the HBA..... 46



Spanned VMFS Volumes and Performance.....	48
Lessons Learned	49
Reproducing This Lab at Home.....	49
After Lab Cleanup	50
Appendix A: Timing in a VM Lab Section Reproduction	51
Software Required.....	51
Hardware Required.....	51
Appendix B: Resource Pools Lab Section Reproduction	52
Software Required.....	52
Hardware Required.....	52
Creating Resource Pools and VMs	52
Adjustments for Hardware	52
Software in the VMs	52
Appendix C: CPU Performance Lab Section Reproduction	53
Software Required.....	53
Hardware Required.....	53
Virtual Machines	53
Appendix D: Disk Performance Lab Section Reproduction.....	54
Software Required.....	54
Hardware Required.....	54
Virtual Machines	54
Appendix E: The Pros and Cons of Using RDM Disks.....	55
Appendix F: CPUBusy Script Setup	58



INTRODUCTION

Welcome to the VMworld 2006 Performance Lab. Performance is a far reaching topic of discussion. It is the goal of this lab to lead you through some exercises that will help you become familiar with performance research so that you may use this knowledge in your own environment. This lab is not comprehensive of everything you will need to successfully diagnose a performance related issue in your environment. Instead this lab tries to highlight the most common performance misconceptions or trouble spots in a Virtual Infrastructure 3 environment. We hope that you find useful pieces of information throughout this lab that you can use to better understand performance related issues in your own environment.

A majority of the support requests opened with VMware are performance related. Out of the performance support requests a vast majority of those are dealing with **perceived** performance. Often the root cause of these perceived performance issues are mis-tuning of the environment or simply a mis-understanding of how Virtual Infrastructure works. It is the aim of this lab to help you distinguish between a perceived performance issue and a real one.

Formatting

The exercises in this lab are designed to prove a point. These points are highlighted by messages in rectangles.

Point:

- This is an example of a point being made.

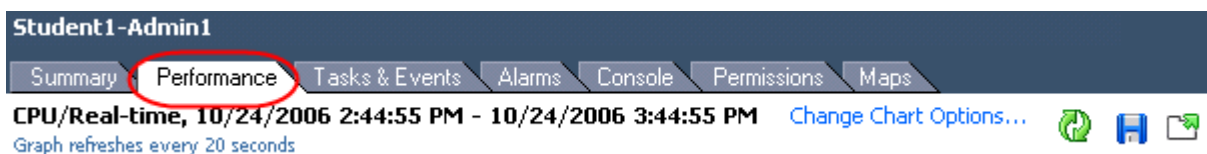
Please pay close attention to the points. Points are the key take-away from the lab.

As you progress through the lab you will also see symbols such as →. The → symbol represents a series of navigations. For example if you need to open the "Edit Settings" GUI and then open the Resources tab inside of the Edit Settings GUI you would see instructions like this: Go to Edit Settings → Resource tab. You may see a longer series of navigations such as: Go to Edit Settings → Resource → CPU → Check unlimited. After each → symbol look for the key word to navigate to the next instruction.

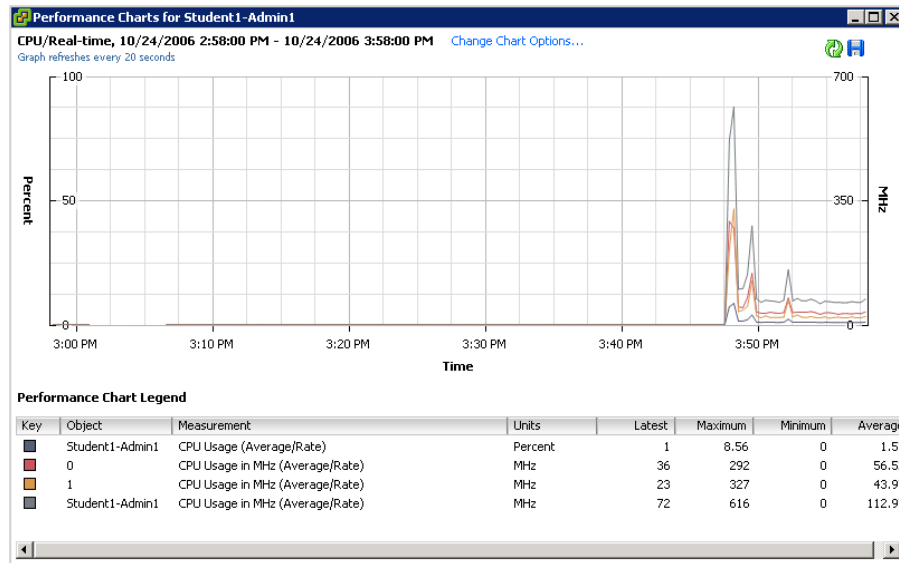
Point:

- The → symbol is used for navigation steps.


Throughout the lab you will be using the Performance Graphs in the Virtual Infrastructure Client. These graphs can be found by clicking on the name of a virtual machine and then navigating to the performance tab.



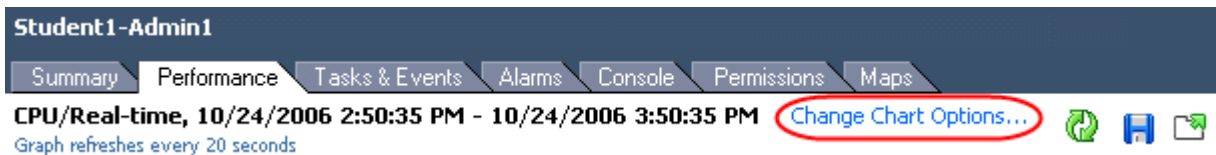
For usability you may want to use the popup chart icon in the graph in order to have the performance graph appear in a separate window.



Use this  to get this...

- Point:
- Use the popup chart icon  for better usability.

In different parts of this lab you will be asked to change the chart options for the performance graphs. This option is hidden in the top part of the performance graph itself. Below is a picture of where this option is located.



Each virtual machine that you will use has a unique name. Each student using this lab also has a unique name such as student1. Throughout this lab the name of a virtual machine will be referred to as Student#-virtualmachinename. For example, the "Admin1" virtual machine for student 1 will be displayed as Student1-Admin1 in the Virtual Infrastructure Client but referred to as Student#-Admin1 in the manual. Just remember to replace your student number for the # sign.

- Point:
- Each student has a number such as Student1, Student2, etc.
 - Virtual Machines will be referred to as Student#-Admin1, Student#-Admin2, etc.
 - Replace your student number with the # sign. For example student 1 would use Student1-Admin1 and Student1-Admin2, etc.

Lab Sections

There are four (4) sections to this lab. The sections do not need to be completed in order, however it is recommended that you start and the first section and work your way through to the end. The four sections are broken down as follows.

Timing Issue in Virtual Machines

This section covers one of the most frequent reports of performance problems – benchmarking results. Often customers or analysts will run an industry benchmark inside of a virtual machine and report horrible performance. In fact the results show perceived performance because virtual machines do not track time when they are not scheduled on physical CPUs.



Resource Pool Conflicts

Resource Pools are a new feature in Virtual Infrastructure 3. They take the concept of Shares, Limits and Reservations that existed in Virtual Infrastructure 2 at the virtual machine level and bring those concepts to collections of virtual machines and hosts. This section explores the various performance impacts that improper tuning of Resource settings can have on virtual machines and hosts.

CPU Performance

CPU scheduling is one of the more common performance bottlenecks on a Virtual Infrastructure 3 host. In Virtual Infrastructure 3, Virtual SMP was expanded to support 4 CPU Guest Operating Systems. This expansion has only compounded the need for proper planning and use of Virtual SMP with virtual machines. This section will demonstrate how to tell when Virtual SMP is being used properly or improperly.

Storage Performance

The last section centers its attention on storage performance. While the lab setup is very limited and by no means should reflect on the storage being used we do push the storage **architecture** to its limits. The limited equipment of the lab actually demonstrates why having good storage architecture is the best thing you can do to improve storage performance.

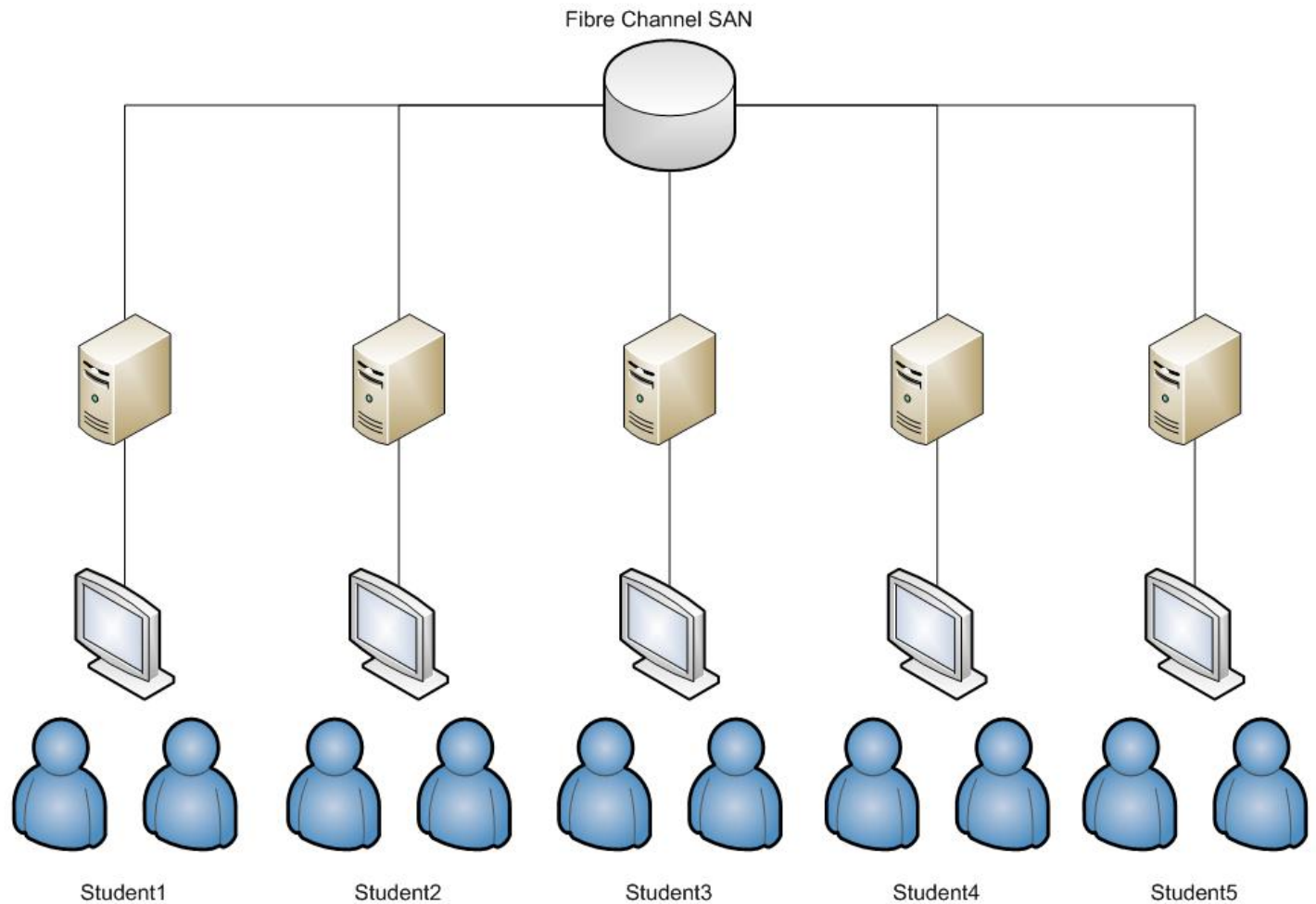
Lab Architecture

The diagram below shows the layout of the lab and its equipment. There will be a total of 40 people per lab session and therefore the layout below will be repeated 3 more times for a total of 4 quadrants or PODs.

Unfortunately we do not have an unlimited equipment supply. If you take this lab and repeat the sections in your own environment you may obtain different results. The concepts will remain the same.



This is representative of 1 quadrant. The entire lab setup will consist of 4 identical quadrants. Each station will house two users and one workstation. Each workstation will control one physical server. Each physical server will be connected to the fibre channel SAN.



Login Information

All virtual machines are set for automatic login as administrator. If you are prompted for a password at any time while logging into the virtual machines use "vmworld".

Your student login information for the Virtual Infrastructure Client is the label student and your student number. For example: Student1, Student2, etc. The password for all students is "vmware".

Point:

- Login to the Virtual Infrastructure Client with an ID of Student# and a password of "vmware". For example, Student1 with a password of "vmware".
- Virtual machines are set for autologon.
- If prompted for a password for the virtual machine use "vmworld".

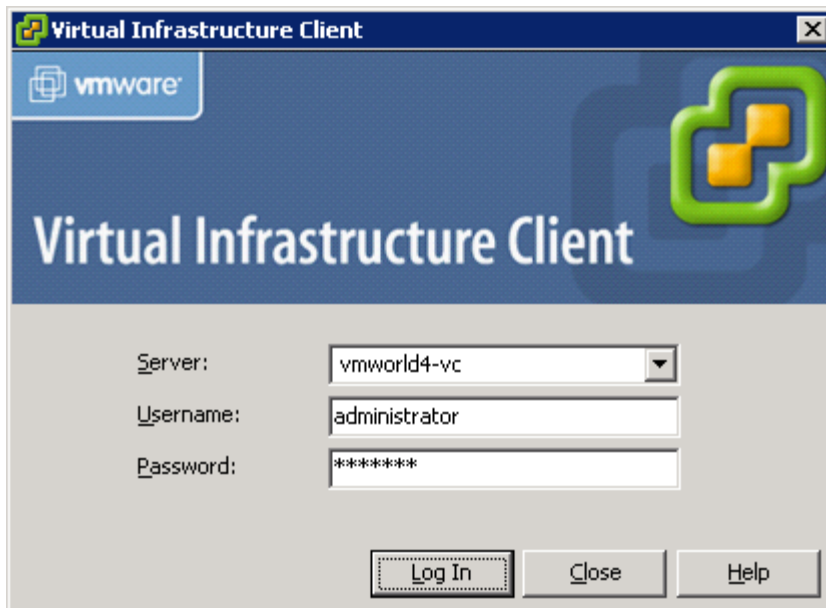
Pre-Lab Exercises

To become familiar with the lab components, please follow through this brief exercise. Let your lab instructors or proctors know if you have any issues with any of these steps by raising your hand.

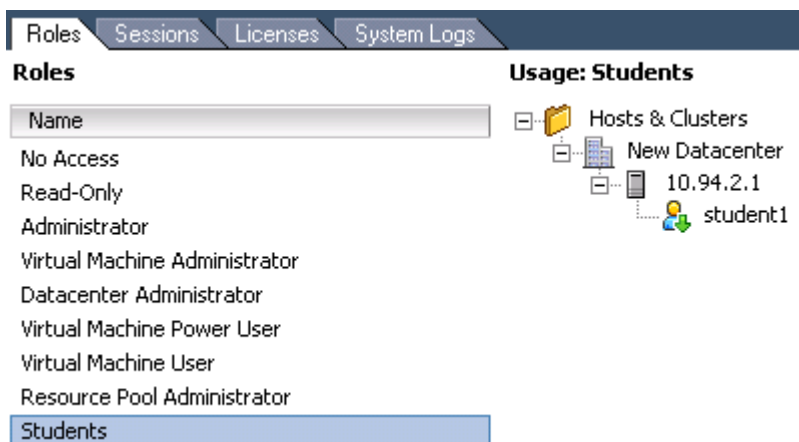
1. Login to the Virtual Infrastructure Client (VC) by double-clicking the icon on your desktop.



2. You will be connecting to *vmworld4-vc* as the Virtual Infrastructure Server. Login with the username *administrator* and the password *vmworld*.

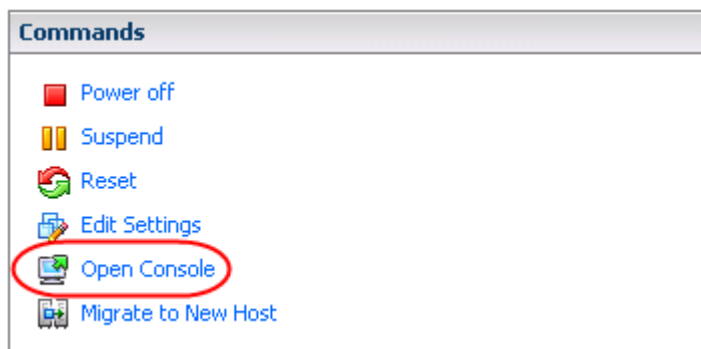


3. When you log on you'll notice you have limited permissions. We have set up particular roles in the environment to ensure the smooth running of the labs. Granular permission and roles are a new feature in Virtual Infrastructure 3.

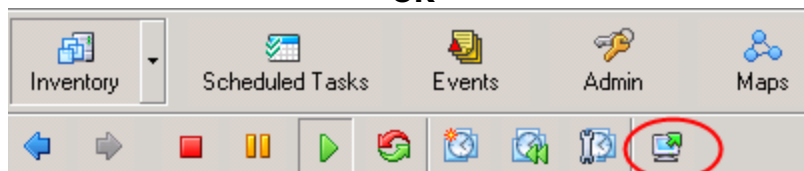


4. Power on virtual machines Student#-UPVM1 and Student#-UPVM2.
5. Open a remote console to Student#-UPVM1 and Student#-UPVM2.





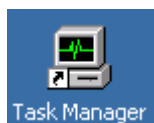
OR



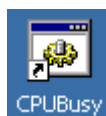
OR



6. Start Windows Task Manager on both Student#-UPVM1 and Student#-UPVM2.



7. On each of the above virtual machines (Student#-UPVM1 and Student#-UPVM2), Start CPUBusy. You do this by double-clicking on the CPUBusy shortcut on the desktop. This is a script which calculates the sine of a given function repeatedly which uses all of the CPU resources of a single VCPU. This script is run by a process called *cscript*. To learn how to build this script, see [Appendix F](#).



8. Another shortcut on the desktop is to IOMeter which is a disk I/O benchmark which we'll use later.



9. Watch for the appropriate instructions to power down your virtual machines throughout the lab. This needs to happen in the correct manner in order to get the correct results.

Now take one last stretch. This is a two hour lab. You're in for the long haul.

At any time during the lab if you have questions please raise your hand and a lab instructor or proctor will be more than happy to help you out. Have fun!!

Stop here!!! Wait for the group.



SECTION 1: TIMING ISSUES IN VIRTUAL MACHINES

Background Information

The VMware *De-scheduled Time Accounting driver (vmdesched)* is a new component to VMware Tools. It improves the accuracy of the percentage of CPU resources used in the guest operating system when physical CPU resources are over committed. The *vmdesched* thread represents explicitly the time that a virtual machine is not scheduled and allows performance monitoring tools and workload management software to account for de-scheduled time accurately. The driver also helps a virtual machine to catch up to real time quickly after it resumes execution.



When a virtual machine is waiting to be scheduled on a physical CPU then the guest operating system will not account for this lost time. Time inside the virtual machine is frozen until it is scheduled on a physical CPU again. The ESX Server scheduler ensures that time is given to all virtual machines regularly and that the time that a virtual machine has to wait to be scheduled on a physical CPU is small. Nevertheless, virtual machines to spend some amount of time "de-scheduled". Normally this does not affect any application inside the Guest OS. However, benchmarking applications rely heavily on timer interrupts to calculate performance over a given period of time. Current benchmarking applications are not virtualization aware and so they do not account correctly for the time a virtual machine is de-scheduled. Furthermore, performance monitoring applications from 3rd parties sometimes use timer interrupts and also do not account for the true amount of time a virtual machine is spending on a CPU.

As you will see throughout this section, the *vmdesched* driver was designed to allow the guest operating system to account for the time it is de-scheduled. At its most basic form the *vmdesched* driver simply uses CPU cycles in accordance with the time not spent on the physical CPU. For example a virtual machine may report that it is running at 100% CPU usage but it may only be getting 80% of the physical CPU time. The *vmdesched* driver will actually consume 20% of the resources inside of a Guest OS so that the performance inside of a VM shows 80% utilization.

This section will walk you through a demonstration of the effectiveness of the *vmdesched* driver and how to install it. This is important when troubleshooting performance since you may actually have false performance ceilings being reported by your Guest OS. It is now possible to see the real performance of a virtual machine in relation to how much CPU time it receives.

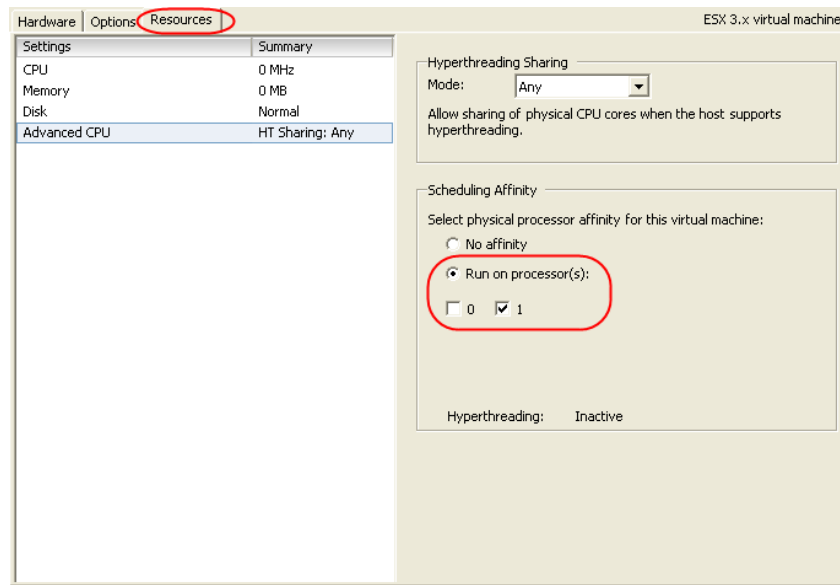
Lab Practice

You will find two Windows 2003 Enterprise Edition virtual machines underneath your ESX Server host. These are called Student#-UPVM1 and Student#-UPVM2. For this test we will pin both virtual machines to the same physical processor in order to show what happens with contention and time accounting in a virtual machine. We will use the second CPU in our 2 CPU server to avoid conflicts with the Service Console process. Let's start the lab by checking the settings of our virtual machines. They should be as follows:

	 Student1-UPVM1	 Student1-UPVM2
Processor Affinity	Processor 2	Processor 2
CPU Shares	Normal	Normal

Using the Virtual Infrastructure Client, select Student#-UPVM1 → Edit Settings → Resources → Advanced CPU. You can adjust the processor affinity while a virtual machine is running.





Timing in a VM without Accounting for De-scheduled Time

Each virtual machine has a script installed called CPUBusy. This script calculates the sine of a given function repeatedly which uses all of the CPU resources of a single VCPU. This script is run by a process called *cscript*. There is a shortcut to run this script located on the desktop of each virtual machine. To learn how to build this script, see [Appendix F](#). This script has already been built for these lab machines.



Do the following in both Student#-UPVM1 and Student#-UPVM2:

- 1) Start Task Manager, select the processes tab and double click on the CPU column header to sort the processes in order of activity.
- 2) Double-Click on the CPUBusy shortcut on the desktop. There may be other processes that take a few percent, but *cscript* running CPUBusy will take all the CPU capacity of the virtual machine.
- 3) Monitor the performance of the individual virtual machines for two minutes by viewing the Performance Tab for each in Virtual Center as well as the task manager output in each virtual machine.

Performance Chart Legend

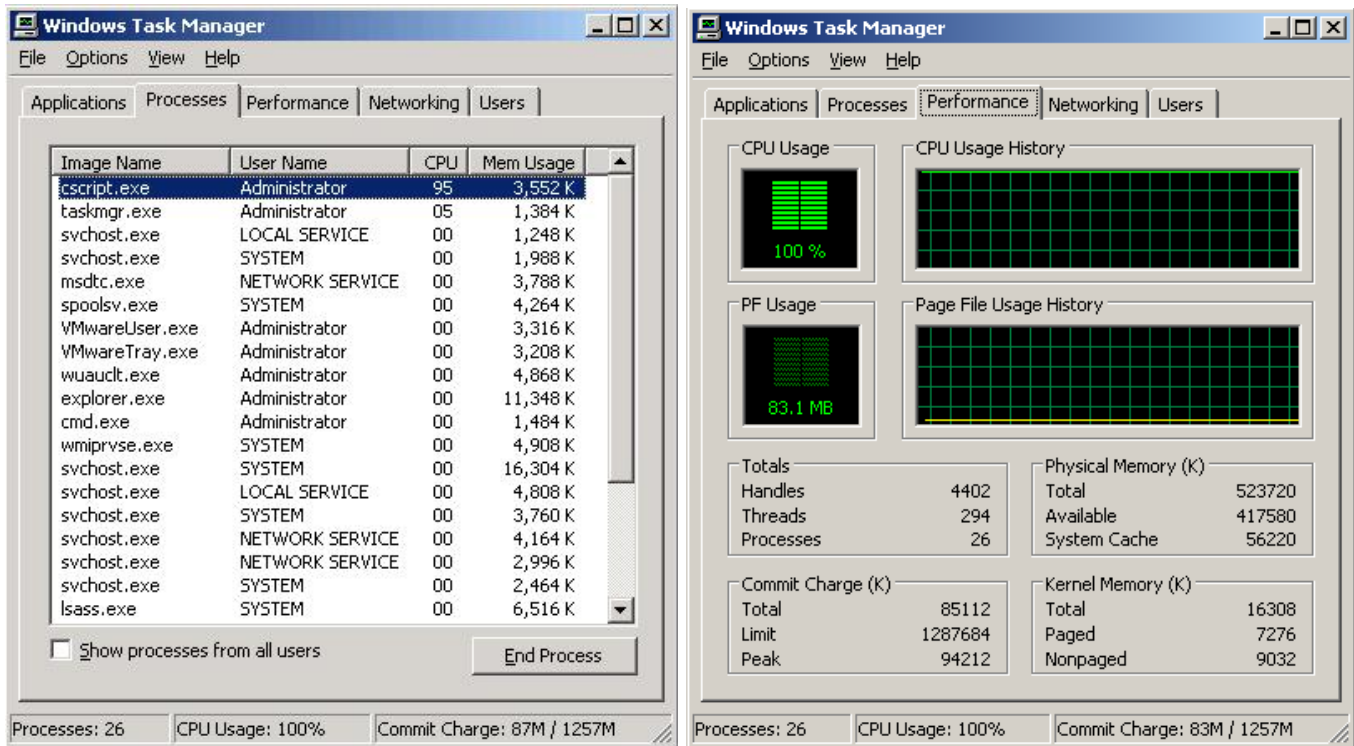
Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-UPVM1	CPU Usage (Average/Rate)	Percent	50.1	83.53	0	22.54

Performance Chart Legend

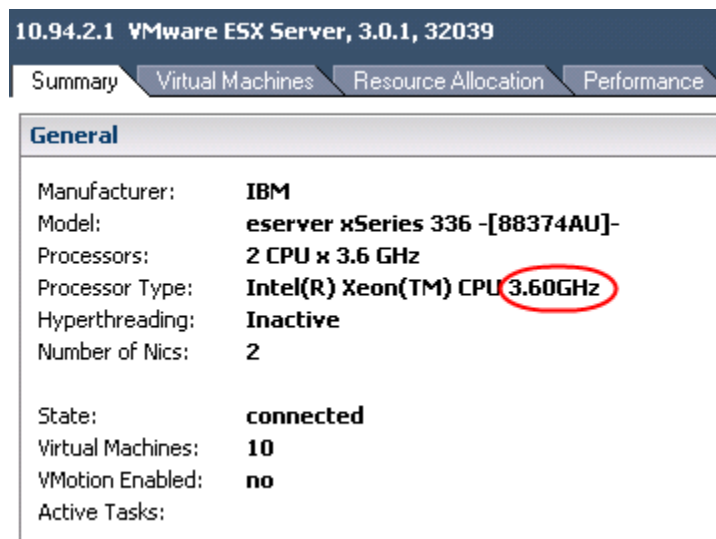
Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-UPVM2	CPU Usage (Average/Rate)	Percent	50.05	81.26	0	25.67

As you can see the VMs are each using 50% of the physical CPU. This is because they are currently both pinned to the same physical CPU and are competing for resources. Since neither VM is bound by implicit resource limits so they will split the CPU time. However, as you can see in the task manager screen shots below, the Guest OS still believes that it is getting 100% of the CPU time. The virtual machines are not be "de-scheduled" at this point since there is no upper bound on them. This is simply time slicing of resources.



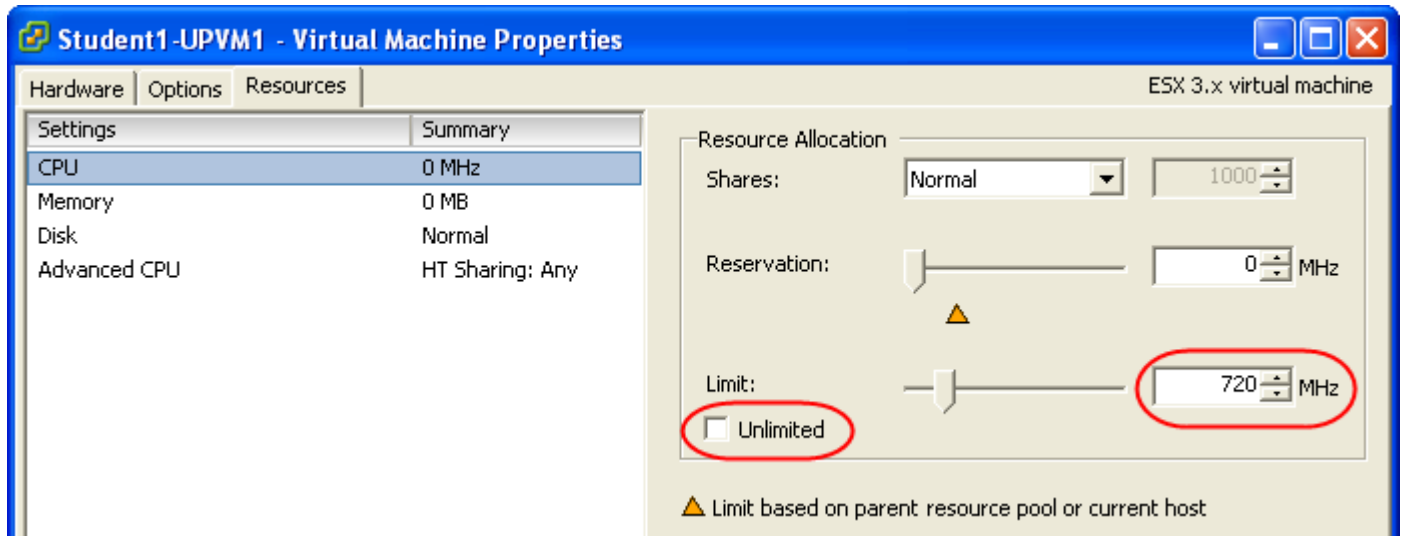


Now let's limit each Virtual Machine to use only 20% of a CPU. Using the Virtual Infrastructure Client, click on the Summary tab for the ESX Server you are using (ex. 10.94.2.1) and look for the processor speed. For example, the processors in the IBM x336 server are 3.6 GHz processors. Twenty percent (20%) of a single processor for this server is therefore 720 MHz (3600 x 0.20).



Using Virtual Infrastructure Client, select Student#-UPVM1 and click on the "Edit Settings" link. Then click on the Resources tab. Uncheck the Unlimited check box and enter 720 to the Limit box. Repeat these steps for Student#-UPVM2.





Now the virtual machine is still trying to use 100% of the CPU but is explicitly limited to only 20% of the CPU. The VM is still de-scheduled to only allow it to use 20% of the physical CPU resources. Look at the Task Manager and Processes tab. The *cscript* process will still be taking over 95% CPU. The Performance tab of the Virtual Infrastructure Client for this virtual machine will show that it is only getting 20% or 720 MHz of a CPU resource. This severe difference in performance reporting is what the *vmtoolsd* driver will solve. Now we will show you how great the *vmtoolsd* driver is at accounting for this de-scheduled time.

Timing in a VM - Accounting for De-scheduled Time with the *vmtoolsd* Driver

We are now going to repeat the same procedure as the previous section except we will be using the *vmtoolsd* driver as well. This will show how the *vmtoolsd* driver more accurately shows how much CPU resource the Virtual Machine is using.

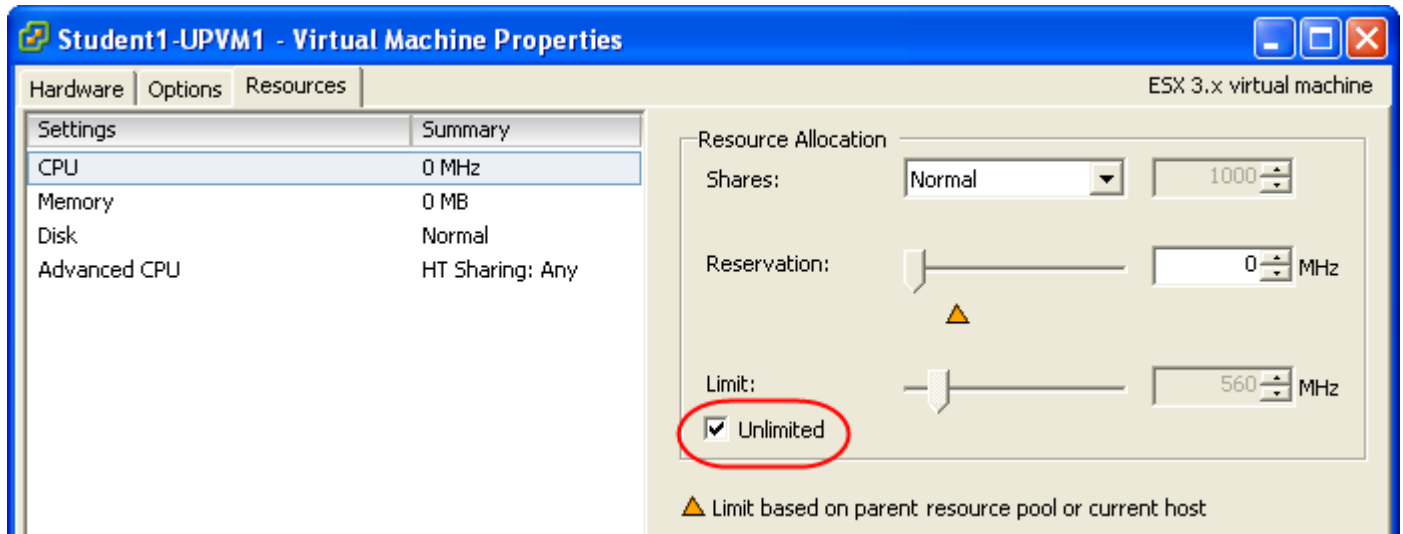
The *vmtoolsd* driver is part of the VMware Tools. The standard install of VMware Tools does not include *vmtoolsd*. For these labs, *vmtoolsd* has been installed in each virtual machine. For detailed instructions on a manual install of the *vmtoolsd* driver refer to the whitepaper at <http://www.vmware.com/vmtn/resources/526>. Note that the *vmtoolsd* driver is only included with version 3.x of VMTools.

Since *vmtoolsd* is already installed you can start *vmtoolsd* manually by issuing the following command from the Windows command prompt:

```
net start vmtoolsd
```

Alternatively you can start *vmtoolsd* by selecting Windows Start menu, choose Settings → Control Panel → Administrative Tools → Services. Right-click the VMware De-scheduled Time Accounting service entry and click on Start Service.

Please start the *vmtoolsd* driver in each of your virtual machines now. Also check that each virtual machine has unlimited CPU again. Using the Virtual Infrastructure Client, click on the Summary tab → Edit Settings → Resources. Check the Unlimited check box in the CPU section.

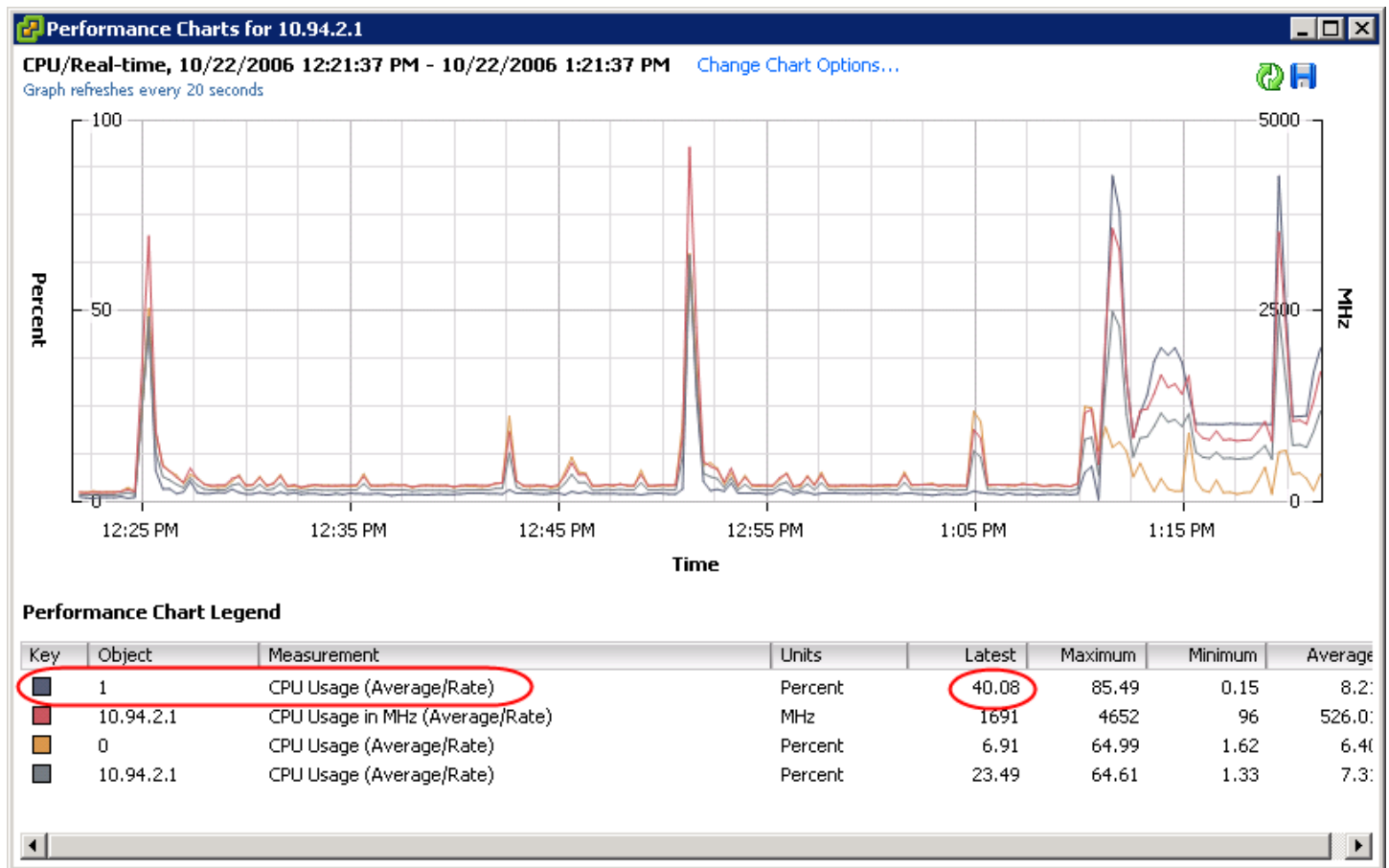


Select the Windows Task Manager of the Student#-UPVM1 and record the amount of CPU resource that the *cscrip.exe* and *vmtoolsd.exe* processes are using. You will see that the amount being used by the *cscrip* process is 95% or lower. You will also see that the amount of CPU being used by *vmtoolsd.exe* makes the difference of 100% (minus a little for task manager). For example, when *cscrip.exe* is using 95% CPU *vmtoolsd.exe* is using approximately 5%.

Wait about two minutes. Look at the virtual machine CPU performance in the Virtual Infrastructure Client Performance Tab and you will see that as you expected, the virtual machine is using half of a CPU just as before. There are two virtual machines both using as much CPU resource as they can.

Limit each Virtual Machine to use only 20% of a CPU just like before. Using the Virtual Infrastructure Client, click on the Summary tab for the physical ESX Server you are using (ex. 10.94.2.1) to calculate the appropriate value for the interface. For example, the processors in the IBM x336 server are 3.6 GHz processors. Twenty percent (20%) of a single processor for this server is therefore 720 MHz (3600 x 0.20).

You will see in the Processes tab in Windows Task Manager of the virtual machine the amount of CPU being used by *cscrip* is close to 40%. The amount used by *vmtoolsd.exe* is close to 60%. Looking at the Performance tab of the virtual machine you will see that the virtual machine is using only the 20% of a CPU. Why doesn't the guest display the same result as the Virtual Infrastructure Client performance graph? Let's take a look at the ESX server performance graph to get the answer.



As you can see the ESX Server performance graph reports 40% CPU utilization on CPU1. That means there's 60% de-scheduled time which is being accounted for by the *vmdesched* driver. The *vmdesched* driver is still being tuned and at this time and cannot differentiate between the de-scheduled time of the two virtual machines separately.

Lessons Learned

The *vmdesched* driver allows a guest operating system to account for time when the virtual machine is not scheduled. This allows applications such as benchmark tools to report more accurately. You also learned that the *vmdesched* driver is still a work in progress. It's up to you on how you would like to use the *vmdesched* driver. We mention it in this lab only so you can see the relationship between scheduled and de-scheduled virtual machine time and the impact it has on performance reporting.

Reproducing This Lab at Home

See [Appendix A](#) for instructions on how to reproduce this section of the lab in your own environment.



SECTION 2: RESOURCE POOL CONFLICTS

Background Information

Most of the support requests opened with VMware are concerning performance. Sometimes this is a perceived performance issue and sometimes it is real. VMware has found that a majority of the performance problems are self-inflicted as you've seen throughout this lab. One of the great features in Virtual Infrastructure 3 is the ability to split business functions into Resource Pools. Resource Pools are silos of guaranteed performance. This extends the concepts of Shares, Limits and Reservations from an individual virtual machine to a group of virtual machines. This section will demonstrate effects of Shares, Limits and Reservations as applied to Resource Pools and to individual virtual machines. We hope to explain how these tools can be useful and take some of the myth out troubleshooting a perceived shortage of resources.

Resource Pools, Shares, Reservations, and Limits

Resource Pools group VMs together. Shares, Reservations, or Limits for CPU or Memory accessed by a Resource Pool can be set for all its VMs collectively. Shares, Reservations and Limits can also be set on individual VMs. In this lab we will limit ourselves to discussing CPU access. Similar principles hold true for memory access.


Shares determine priority of VMs for access to resources only when there are not enough resources to meet demand. To guarantee or limit resources for a VM or group of VMs when there is no resource contention you must use Reservations and Limits.

Reservations are amounts of a resource guaranteed to a VM.

Limits specify the most that a VM can use, regardless of what is available.

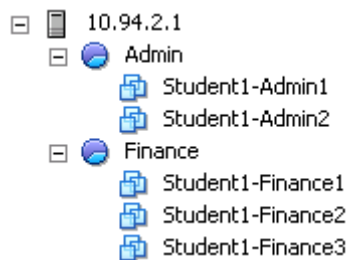
Performance Charts

Performance metrics are available on individual VMs or hosts, or Resource Pools. Select the item and click its Performance tab.

In this lab, it will be convenient to make the Performance chart open in its own window by clicking the popup chart icon at the top right of the display. The icon looks like this: 

Initial Setup

In the Virtual Infrastructure client, find the resource pools "Admin" and "Finance". Find the VMs inside them. There should be Student#-Admin1 and Student#-Admin2 inside the Admin Resource Pool and Student#-Finance1, Student#-Finance2, and Student#-Finance3 inside the Finance Resource Pool. The concept is that there are separate groups in your organization that have VMs, and you as a virtual infrastructure administrator bear responsibility for making sure that their VMs run properly.

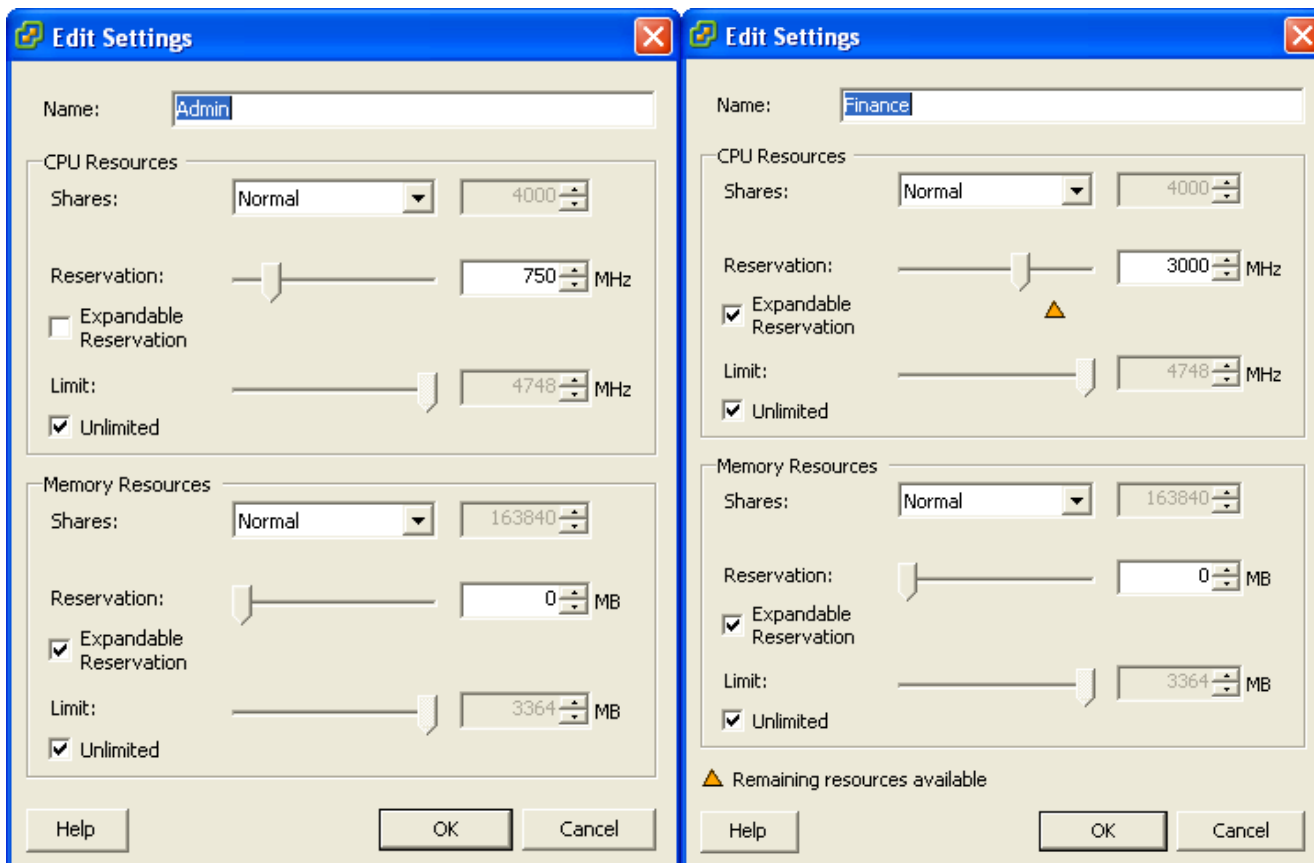


Initial Configuration

The initial configuration is listed below. All VMs will be powered off. This configuration has already been setup for this lab so you may skip to the [Lab Practice](#) section if desired. You may also feel free to check the settings if you desire.

Admin Resource Pool: CPU Reservation: 750 MHz, "Expandable Reservation" not checked.

Finance Resource Pool: CPU Reservation: 3000 MHz, "Expandable Reservation" checked.



Virtual Machines

- Student#-Admin1: CPU: Reservation 1500 MHz, Shares: Normal (2,000).
- Student#-Admin2: CPU: Reservation 2000 MHz, Shares: Normal (2,000).
- Student#-Finance1: CPU: Reservation 0 MHz, Shares: Custom (80,000).
- Student#-Finance2: CPU: Reservation 0 MHz, Shares: High (2,000).
- Student#-Finance3: CPU: Reservation 0 MHz, Shares: Normal (1,000).

Lab Practice

Step 1

Try to Power On VM Student#-Admin1.
What happens? Why?

Look at Student#-Admin1 and find the host running it. Look at CPU usage on the host (*hostname* → Performance). Are there resources lacking? Why do you think the VM will not start?



Look at Admin Resource Pool → Resource Allocation tab.

Admin

Summary Virtual Machines Resource Allocation Performance Tasks & Events Alarms Permissions Maps

CPU Reservation: **750 MHz**
 CPU Reservation Used: **0 MHz**
 CPU Unreserved: **750 MHz**
 CPU Reservation Type: **Fixed**

Memory Reservation: **0 MB**
 Memory Reservation Used: **0 MB**
 Memory Unreserved: **3372 MB**
 Memory Reservation Type: **Expandable**

View: CPU Memory

Name	Reservation - MHz	Limit - MHz	Shares	Shares Value	% Shares	Type
Student1-Admin1	1500	Unlimited	Normal	2000	50	N/A
Student1-Admin2	2000	Unlimited	Normal	2000	50	N/A

The Admin resource pool has a fixed reservation of 750 MHz. This means it can only get 750 MHz of CPU time. The Student#-Admin1 VM requires 1500 MHz of time but the resource pool it is in doesn't have that much to give to the VM.

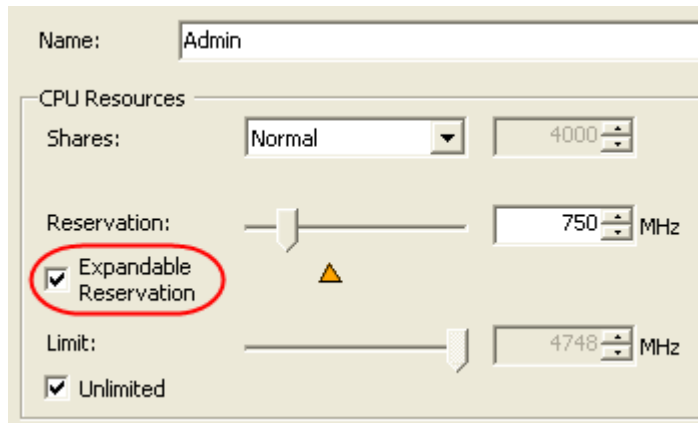
Points:

- Reservations that can't be met prevent a VM starting.
- Non-expandable reservations on a Resource Pool limit VMs' Reservations from being met even when resources abound.

Step 2

Now let's change the resource settings of the Admin Resource Pool to let our VMs power on.

1. Admin Resource Pool → Edit Settings → CPU Resources → check "Expandable Reservation".



2. Start Student#-Admin1.
3. Check Admin Resource Pool → Resource Allocation.
4. Check Finance Resource Pool → Resource Allocation.



Admin
 Summary Virtual Machines Resource Allocation Performance Tasks & Events Alarms Permissions Maps

CPU Reservation: **750 MHz** Memory Reservation: **0 MB**
 CPU Reservation Used: **1500 MHz** Memory Reservation Used: **88.21 MB**
 CPU Unreserved: **1620 MHz** Memory Unreserved: **3280.79 MB**
 CPU Reservation Type: **Expandable** Memory Reservation Type: **Expandable**

View: CPU Memory

Name	Reservation - MHz	Limit - MHz	Shares	Shares Value	% Shares	Type
Student1-Admin1	1500	Unlimited	Normal	2000	50	N/A
Student1-Admin2	2000	Unlimited	Normal	2000	50	N/A

Finance
 Summary Virtual Machines Resource Allocation Performance Tasks & Events Alarms Permissions Maps

CPU Reservation: **3000 MHz** Memory Reservation: **0 MB**
 CPU Reservation Used: **0 MHz** Memory Reservation Used: **0 MB**
 CPU Unreserved: **4620 MHz** Memory Unreserved: **3281.79 MB**
 CPU Reservation Type: **Expandable** Memory Reservation Type: **Expandable**

View: CPU Memory

Name	Reservation - MHz	Limit - MHz	Shares	Shares Value	% Shares	Type
Student1-Finance1	0	Unlimited	Custom	80000	96	N/A
Student1-Finance2	0	Unlimited	High	2000	2	N/A
Student1-Finance3	0	Unlimited	Normal	1000	1	N/A

Notice that Student#-Admin1 is not doing anything significant (nothing has been started running inside it), but it still has a reservation. This means that the VM is reserving all of its 1500 MHz as reflected in the Admin resource pool. So far the VM can get these resources because no other resource pools have reserved them.

Points:

- If and only if the resource pool has an Expandable Reservation, the sum of its running VMs' reservations can be more than the reservation defined for the Resource Pool itself.
- A running VM can have a Reservation even if it is not actively using the resources.
- There is a difference between reserving a resource and using it. You can reserve resources even if you are not using them. This will prevent other virtual machines from using those resources.

Step 3

Now that we have an expandable resource pool we can start all of the VMs in that resource pool.

1. Start Student#-Admin2.
 Why are there "Insufficient CPU resources" to start it?



2. Check the ESX host → Resource Allocation.

10.94.2.1 VMware ESX Server, 3.0.1, 32039

Summary Virtual Machines **Resource Allocation** Performance Configuration Tasks & Events Alarms Permissions Maps

CPU Reservation: **6120 MHz** Memory Reservation: **3369 MB**
 CPU Reservation Used: **4500 MHz** Memory Reservation Used: **154.98 MB**
 CPU Unreserved: **1620 MHz** Memory Unreserved: **3214.02 MB**

View: CPU Memory

Name	Reservation - MHz	Limit - MHz	Shares	Shares Value	% Shares	Type
Finance	3000	Unlimited	Normal	4000	33	Expandable
storage1	0	Unlimited	Normal	1000	8	N/A
Student1-UPVM3	0	Unlimited	Normal	1000	8	N/A
Admin	750	Unlimited	Normal	4000	33	Expandable
Student1-UPVM1	0	720	Normal	1000	8	N/A
Student1-UPVM2	0	720	Normal	1000	8	N/A

The CPU Reservation for the host is the total MHz it has to offer. In our example this is 6,120 MHz.

3. Check Admin Resource Pool → Resource Allocation. Why is the Admin Resource Pool CPU Unreserved only 1,620 MHz? Why is it not the host’s power (6,120 MHz) - Reservation Used by Admin (1,500 MHz) = 4,620 MHz?

4. Check Finance Resource Pool → Resource Allocation. Notice that it is set to 3,000 MHz.

Finance

Summary Virtual Machines **Resource Allocation** Performance Tasks & Events Alarms Permissions Maps

CPU Reservation: **3000 MHz** Memory Reservation: **0 MB**
 CPU Reservation Used: **0 MHz** Memory Reservation Used: **0 MB**
 CPU Unreserved: **4620 MHz** Memory Unreserved: **3213.02 MB**
 CPU Reservation Type: **Expandable** Memory Reservation Type: **Expandable**

View: CPU Memory

Name	Reservation - MHz	Limit - MHz	Shares	Shares Value	% Shares	Type
Student1-Finance1	0	Unlimited	Custom	80000	96	N/A
Student1-Finance2	0	Unlimited	High	2000	2	N/A
Student1-Finance3	0	Unlimited	Normal	1000	1	N/A

Notice that (Host Power) – (Admin Reservation Used) – (Finance Reservation) = 6,120 – 1,500 – 3,000 = 1,620 MHz = Admin CPU Unreserved.

Even though the Finance resource pool has no VMs running, the Finance pool’s reservation has reduced the reservation available to the Admin resource pool.

Points:

- There’s a difference between reserving a resource and using it.
- Reservations granted to a Resource Pool are reserved **even if the pool has no running VMs.**



Step 4

Now that we see where the resources are, let’s adjust the virtual machine resources to allow our virtual machine to start.

1. Student#-Admin2 → Edit Settings → Resources → CPU. Change CPU Reservation to be 0 MHz (zero).
2. Start Student#-Admin2. Does it now start?

3. Check Admin → Resource Allocation. Notice that the reservation used and unreserved are still the same.

Admin
Summary Virtual Machines Resource Allocation Performance Tasks & Events Alarms Permissions Maps

CPU Reservation: 750 MHz	Memory Reservation: 0 MB
CPU Reservation Used: 1500 MHz	Memory Reservation Used: 176.43 MB
CPU Unreserved: 1620 MHz	Memory Unreserved: 3123.81 MB
CPU Reservation Type: Expandable	Memory Reservation Type: Expandable

View: CPU Memory

Name	Reservation - MHz	Limit - MHz	Shares	Shares Value	% Shares	Type
Student1-Admin1	1500	Unlimited	Normal	2000	50	N/A
Student1-Admin2	0	Unlimited	Normal	2000	50	N/A

Points:

- Used virtual machine resources are not reflected in Reservations in the Resource Pool.

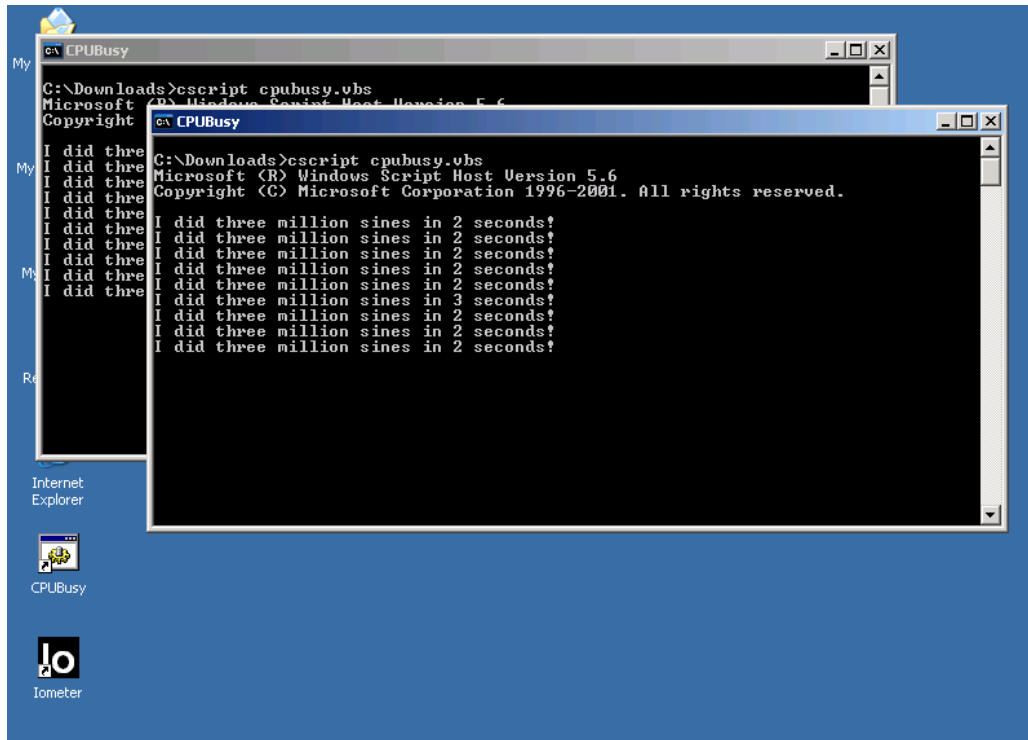
Step 5

Now that we understand a little about resource reservations when there is no load let’s start to get some work done. In this step we’ll run a load generation utility to simulate load inside of a virtual machine.

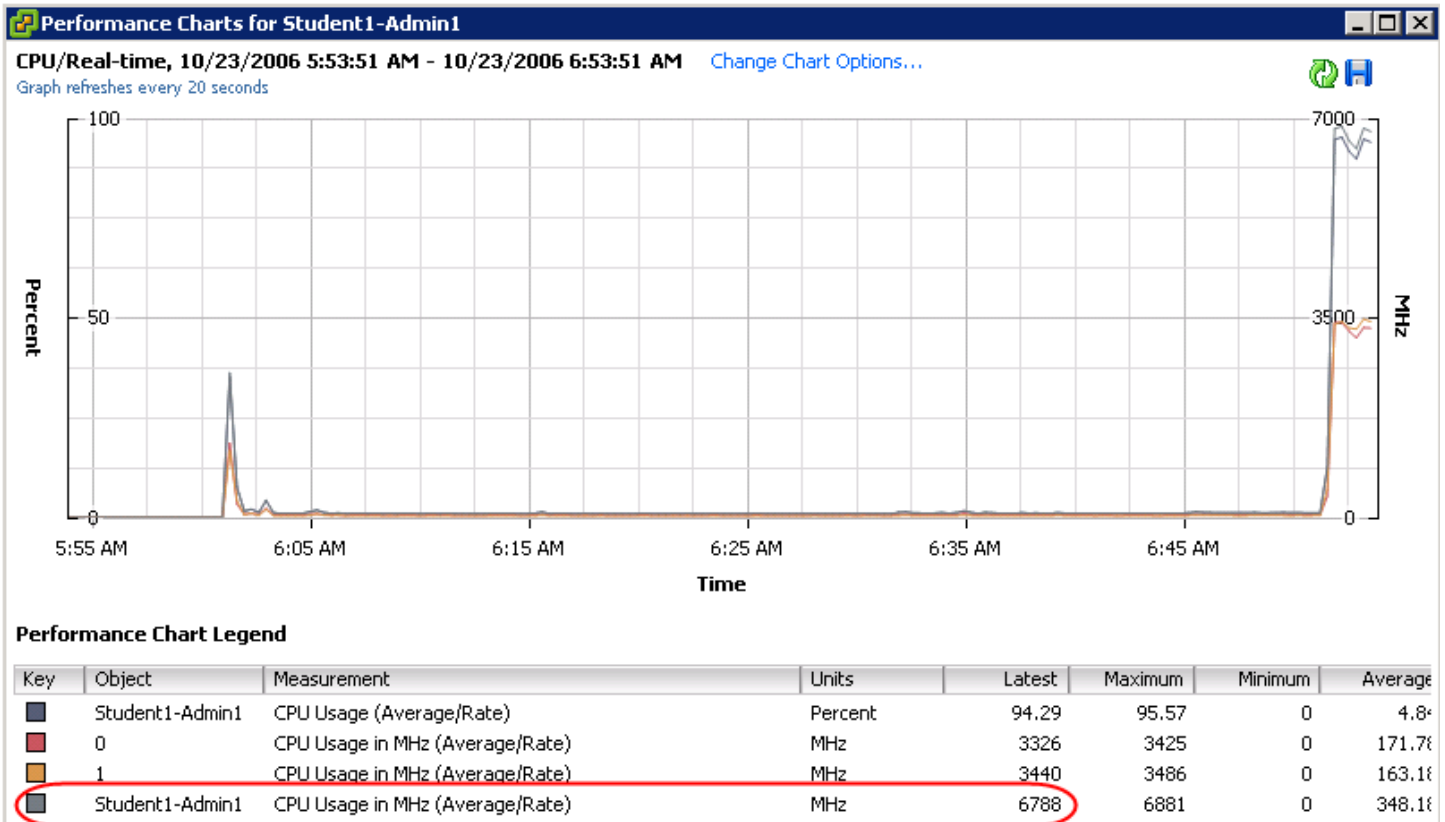
1. Connect to the console on Student#-Admin1.
2. Start 2 instances of CPUBUSY (click the CPUBUSY icon on the desktop once, see a new window open, then repeat and see a second window open).



You should now have two windows with text saying they are performing mathematical computations.



3. Check the Performance tab on Student#-Admin1 and look at CPU Usage and CPU Usage in MHz. What do you notice?



Observe that the CPU MHz used by Student#-Admin1 is greater than Student#-Admin1’s reservation (1,500 MHz). In fact it’s so big that it’s using CPU cycles that are reserved by the Finance resource pool. It can do so because Finance has reserved those cycles, but it has no VMs that are actively using those cycles.

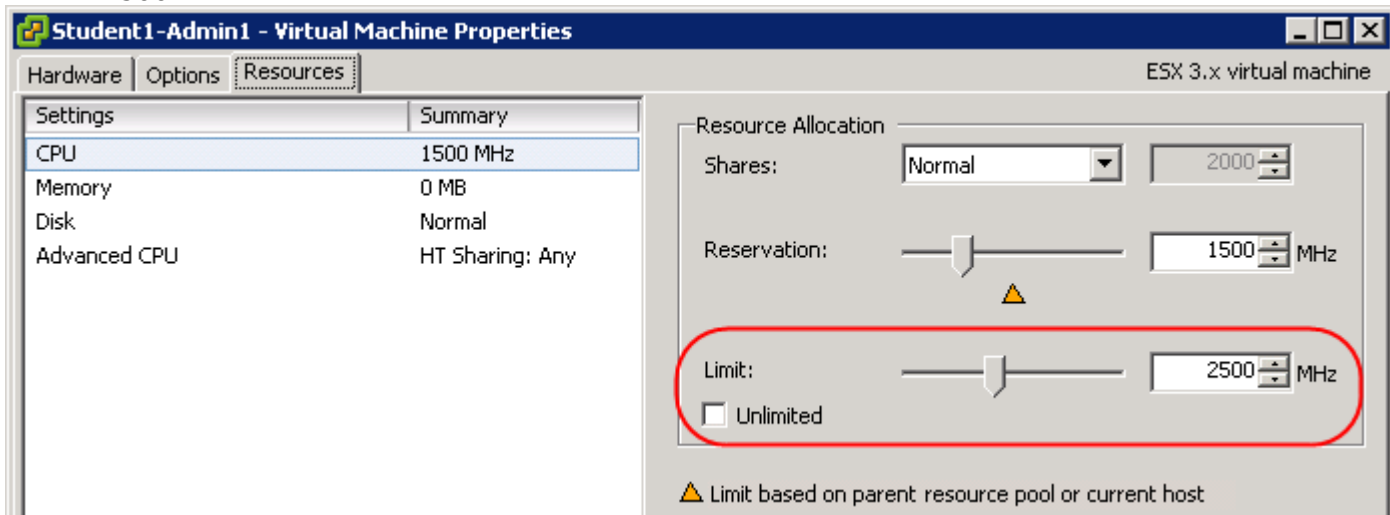
Points:

- There’s a difference between reserving a resource and using it.
- Reservations granted but not used are not wasted. If a reservation holder does not want to use its reserved resource, that resource can be used (though not reserved) by another VM.

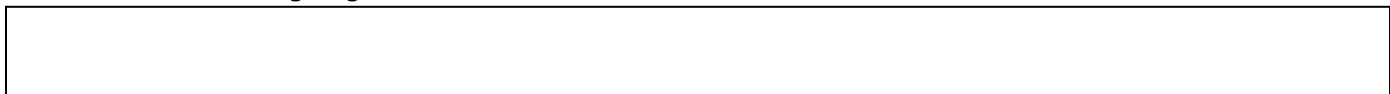
Step 6

Now that we understand reservations between Resource Pools, let’s look at the impact that limits impose.

1. Student#-Admin1 → Edit Settings → Resources → CPU, uncheck “Unlimited” and set a Limit of 2500 MHz.



Look at Student#-Admin1 → Performance. What’s happened? Look at the host's Performance tab. Are there CPU resources going unused?



Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Admin1	CPU Usage (Average/Rate)	Percent	34.79	97.14	0.79	15.0%
■	0	CPU Usage in MHz (Average/Rate)	MHz	1238	3497	28	534.4
■	1	CPU Usage in MHz (Average/Rate)	MHz	1246	3500	16	535.0
■	Student1-Admin1	CPU Usage in MHz (Average/Rate)	MHz	2505	6994	56	1083.6

Points:

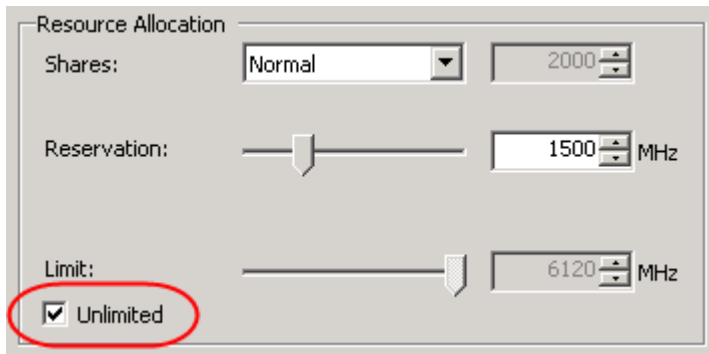
- A limit will limit what a VM can do, even if resources abound.



Step 7

Now that you understand the concept of limits, let's go and see what happens when there's resource contention.

1. Student#-Admin1 → Edit Settings → Resources → CPU, check "Unlimited".

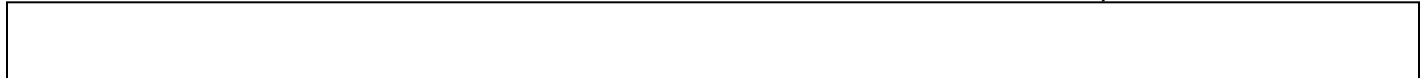


2. Start Student#-Admin2.
3. Login to Student#-Admin2.
4. Start two (2) instances of CPUBUSY.

Current state:

- Student#-Admin1: Running 2xCPUBUSY. CPU Reservation: 1500 MHz; CPU Shares: Normal (2,000); CPU Limit: none.
- Student#-Admin2: Running 2xCPUBUSY. CPU Reservation: 0 MHz; CPU Shares: Normal (2,000); CPU Limit: none.

Look at Performance tabs for Student#-Admin1 and Student#-Admin2. What do you notice?



Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Admin1	CPU Usage (Average/Rate)	Percent	49.14	97.14	0.79	23.90
■	0	CPU Usage in MHz (Average/Rate)	MHz	1735	3497	28	848.00
■	1	CPU Usage in MHz (Average/Rate)	MHz	1784	3500	16	858.50
■	Student1-Admin1	CPU Usage in MHz (Average/Rate)	MHz	3538	6994	56	1722.60

Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Admin2	CPU Usage (Average/Rate)	Percent	48.94	48.94	0	4.60
■	0	CPU Usage in MHz (Average/Rate)	MHz	1766	2043	0	175.30
■	1	CPU Usage in MHz (Average/Rate)	MHz	1741	1741	0	144.90
■	Student1-Admin2	CPU Usage in MHz (Average/Rate)	MHz	3523	3523	0	336.00

Student#-Admin1 and Student#-Admin2 are each trying to use 2 PCPUs at 100%, i.e. 4 PCPUs of demand. The host has only 2 PCPUs (hyper-threading is disabled). Even though Student#-Admin1 has a CPU reservation and Student#-Admin1 does not, the two VMs have equal Shares of CPU. Student#-Admin1 and Student#-Admin2 use equal amounts of CPU because their **shares** are equal, and because Student#-Admin1's **reservation is small** enough that the host can treat Student#-Admin1 and Student#-Admin2 equally. Student#-Admin1 is still meeting its reservation however there is contention for resources and so shares start to take effect.



Point:

- Equal shares means equal overall access to the resource - if that's possible after reservations are met.

Step 8

Now that you understand that shares only kick in when resources are in competition, let's look at the impact of adjusting shares.

1. Student#-Admin1 → Edit Settings → Resources → CPU, change its CPU Reservation to 0 MHz (zero MHz).
2. Student#-Admin2 → Edit Settings → Resources → CPU, change its CPU Shares to High (4,000).

Student#-Admin1

Student#-Admin2

Current state:

- Student#-Admin1: Running 2xCPUBUSY. CPU Reservation: 0 MHz; CPU Shares: Normal (2,000); CPU Limit: none.
- Student#-Admin2: Running 2xCPUBUSY. CPU Reservation: 0 MHz; CPU Shares: High (4,000); CPU Limit: none.

Look at the Performance charts for Student#-Admin1 and Student#-Admin2.

Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Admin1	CPU Usage (Average/Rate)	Percent	32.7	97.14	0.91	39.7%
■	0	CPU Usage in MHz (Average/Rate)	MHz	1137	3497	31	1410.9
■	1	CPU Usage in MHz (Average/Rate)	MHz	1202	3500	21	1434.3
■	Student1-Admin1	CPU Usage in MHz (Average/Rate)	MHz	2354	6994	65	2863.7

Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Admin2	CPU Usage (Average/Rate)	Percent	65.49	65.49	0	20.1%
■	0	CPU Usage in MHz (Average/Rate)	MHz	2356	2356	0	729.0
■	1	CPU Usage in MHz (Average/Rate)	MHz	2342	2342	0	706.3
■	Student1-Admin2	CPU Usage in MHz (Average/Rate)	MHz	4715	4715	0	1452.0

Notice that Student#-Admin2 is now using about twice as many CPU MHz as Student#-Admin1. This is because there are not enough MHz for both VMs. And Student#-Admin2 has twice the CPU Shares as Student#-Admin1. When there are conflicts for CPU, Student#-Admin2 wins about twice as often as Student#-Admin1.

Points:

- Twice the shares, means twice the access to the resource--when the resource is scarce.



Step 9

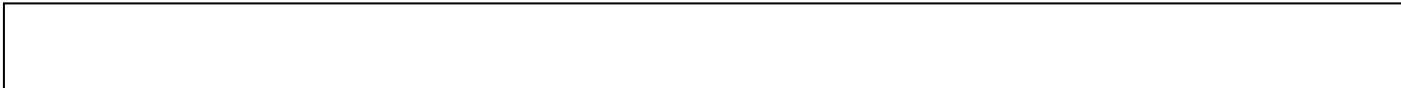
Now let's look at what happens to CPU shares between Resource Pools.

1. Student#-Admin2 → Shut Down Guest.
2. Admin Resource Pool → Edit Settings → CPU Resources; change Reservation to 0 MHz (zero MHz).
3. Finance Resource Pool → Edit Settings → CPU Resources; change Reservation to 0 MHz (zero MHz).
4. Start Student#-Finance1.
5. Login to Student#-Finance1 and start running two instances of CPUBUSY (2 x CPUBUSY).

Current state:

- Admin Resource Pool: no CPU Reservation. CPU Shares Normal (4,000).
- Finance Resource Pool: no CPU Reservation. CPU Shares Normal (4,000).
- Student#-Admin1: 2 VCPU, 2xCPUBUSY. Reservation 0 MHz, CPU Shares normal (2,000).
- Student#-Admin2: shut down.
- Student#-Finance1: 2 VCPU, 2xCPUBUSY. Reservation 0 MHz, CPU Shares custom (80,000).

Look at Performance charts for Student#-Admin1 and Student#-Finance1. There is CPU contention because there is 4 PCPU-worth of demand but only 2 PCPUs in the host. Why are they using CPU equally when Student#-Finance1 has 40x the CPU Shares?



Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Admin1	CPU Usage (Average/Rate)	Percent	46.62	97.44	1	47.61
■	0	CPU Usage in MHz (Average/Rate)	MHz	1665	3497	33	1689.5
■	1	CPU Usage in MHz (Average/Rate)	MHz	1674	3524	23	1719.3
■	Student1-Admin1	CPU Usage in MHz (Average/Rate)	MHz	3357	7016	72	3428.0

Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Finance1	CPU Usage (Average/Rate)	Percent	49.09	49.18	0	49.13
■	0	CPU Usage in MHz (Average/Rate)	MHz	1754	2188	0	15
■	1	CPU Usage in MHz (Average/Rate)	MHz	1763	1790	0	14
■	Student1-Finance1	CPU Usage in MHz (Average/Rate)	MHz	3534	3541	0	3

Point:

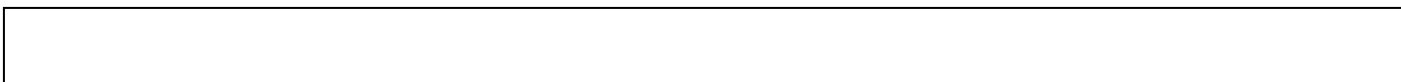
- VM Shares do NOT matter across Resource Pools, only within Resource Pools.

Step 10

What about shares for resource pools? How does that affect a virtual machine?

1. Admin Resource Pool → Edit Settings → CPU Resources, change CPU Shares to High (8,000).

Compare Student#-Admin1 and Student#-Finance1 performance graphs. Why is Student#-Admin1 now using twice the MHz as Student#-Finance1 is using?



Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Admin1	CPU Usage (Average/Rate)	Percent	65.4	97.44	30.99	51.16
■	0	CPU Usage in MHz (Average/Rate)	MHz	2320	3497	1107	1815.5
■	1	CPU Usage in MHz (Average/Rate)	MHz	2370	3524	1101	1848.5
■	Student1-Admin1	CPU Usage in MHz (Average/Rate)	MHz	4709	7016	2231	3683.5

Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Finance1	CPU Usage (Average/Rate)	Percent	32.81	49.21	0	15
■	0	CPU Usage in MHz (Average/Rate)	MHz	1165	2188	0	15
■	1	CPU Usage in MHz (Average/Rate)	MHz	1181	1796	0	15
■	Student1-Finance1	CPU Usage in MHz (Average/Rate)	MHz	2362	3543	0	31

Point:

- Resource Pool shares matter across Resource Pools.

Step 11

This last step will confirm the fact that shares only kick in when resources are scarce.

1. Shut down Student#-Admin1 and Student#-Finance1.
2. Start Student#-Finance2 and Student#-Finance3.
3. On Student#-Finance2 and Student#-Finance3, start 1xCPU_BUSY in each VM. These two VMs are single processor VMs so 1xCPU_BUSY script will run them to 100% utilization.

Current state:

- Student#-Admin1, Student#-Admin2, and Student#-Finance1: shut down.
- Student#-Finance2: 1 VCPU, running 1xCPU_BUSY. CPU Shares 2,000.
- Student#-Finance3: 1 VCPU, running 1xCPU_BUSY. CPU Shares 1,000.

Look at the performance charts for Student#-Finance2 and Student#-Finance3. Student#-Finance2 has twice as many shares as Student#-Finance3; is Student#-Finance2 running twice as much?

Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Finance2	CPU Usage (Average/Rate)	Percent	98.42	99.95	0	8
■	0	CPU Usage in MHz (Average/Rate)	MHz	3528	3582	0	28
■	1	CPU Usage in MHz (Average/Rate)	MHz				
■	Student1-Finance2	CPU Usage in MHz (Average/Rate)	MHz	3543	3598	0	28

Performance Chart Legend

Key	Object	Measurement	Units	Latest	Maximum	Minimum	Average
■	Student1-Finance3	CPU Usage (Average/Rate)	Percent	94.33	94.83	0	70
■	0	CPU Usage in MHz (Average/Rate)	MHz	3381	3400	0	253
■	1	CPU Usage in MHz (Average/Rate)	MHz				
■	Student1-Finance3	CPU Usage in MHz (Average/Rate)	MHz	3396	3413	0	253

Points:

- Shares matter only when resources are scarce. When the host has enough resources to meet all demand, shares are irrelevant.



Step 12

You are done with this section. Please shut down your virtual machines.

1. Shut down Student#-Finance2 and Student#-Finance3.

Lessons Learned

In this section you learned how to apply Limits, Reservations, and Shares. You learned that shares defined for virtual machines are only valid inside of a Resource Pool. Shares defined for Resource Pools are valid across Resource Pools. You also learned that applying reservations in your environment may impact the ability to start virtual machines in other areas of your environment.

Over tuning a virtual environment can create a performance issue. There are still resources left to fix those performance issues but the tuning you have done is actually preventing you from using those other resources.

When tuning your environment use broad strokes. Leverage the power of DRS to move resources around inside of a cluster. DRS and ESX do a great job of managing access to physical resources in a fair manner. Shares, Reservations, and Limits should be used carefully as they restrict options for managing resources by ESX and the global scheduler of the DRS cluster.

Reproducing This Lab at Home

See [Appendix B](#) for instructions on how to reproduce this section of the lab in your own environment.



SECTION 3: CPU PERFORMANCE

CPU tuning is perhaps the most obscure concept to grasp firmly and yet the easiest to tune and receive worse performance. This section will concentrate on the various CPU metrics that can be tuned and their consequences on VM performance.

Starting with version 2.1 of ESX Server VMware introduced VSMP or Virtual SMP. In version 3 of ESX Server SMP support was extended to allow 4 CPU implementations within a virtual machine. VSMP was designed to broaden the reach of the VM to include more production applications. Some applications such as SQL, Oracle, and Exchange really like multiple processors more from a threading perspective than actual raw throughput. However, in the physical world most applications do not scale linearly across multiple processors. In fact most applications are not even multi threaded and will not gain any performance from adding additional processors.

Background


VSMP was designed to allow multiple threads (one per virtual CPU) to be scheduled across one or more physical CPUs. VSMP requires that two VM worlds get scheduled onto two separate physical CPUs (PCPUs) or in the case of hyper-threading on the server, logical CPUs (LCPU). Since both processors have to be scheduled at the same time and kept in state with each other having one processor run idle because it's not being used wastes valuable time in the OS and in the underlying vmkernel. Processors that are scheduled but run idled are accounted for in CPU Wait%.

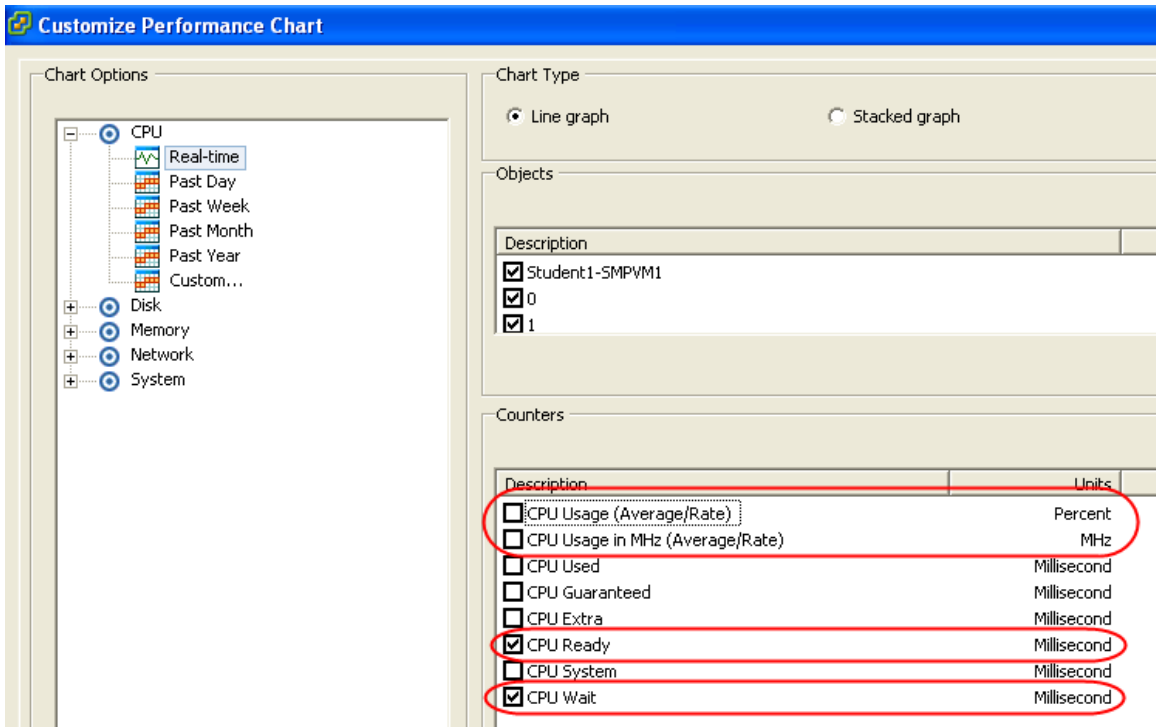
Lab Practice

During this lab you will be using the following virtual machines: Student#-UPVM3 and Student#-SMPVM1. All other virtual machines should be powered off. If they are not powered-down already, then please power off all virtual machines.

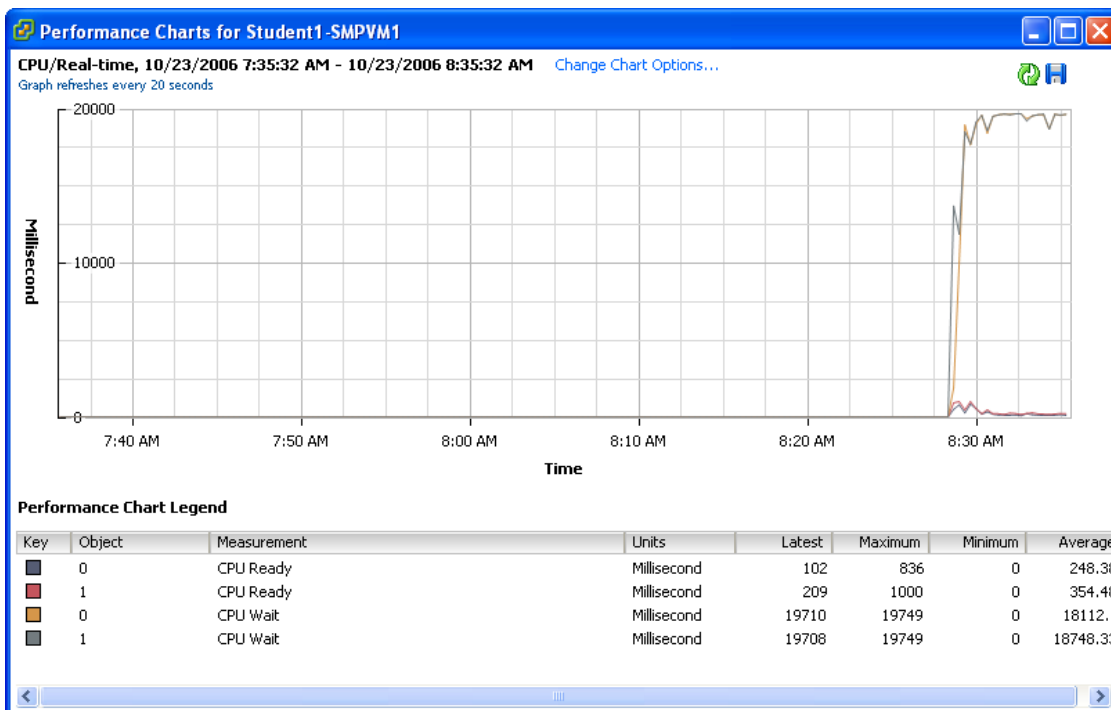
First, we will look at CPU usage in our SMP virtual machine. For this lab we will only look at 2-way SMP because we only have 2 physical processors available to us and hyper-threading is currently turned off. The same principals learned in this lab can be applied to 4-way SMP as well.

Start by powering on Student#-SMPVM1. The virtual machine will automatically login as administrator.

With the virtual machine powered on open the performance graph  for Student#-SMPVM1. Change the chart type to show the %READY and %WAIT time for the virtual machine. You will have to deselect CPU Usage and CPU Usage in MHz since the graph can only handle two CPU metrics at a time.



Monitor the performance graph. After the virtual machine has powered on and settled down you will notice a low %READY time and a high %WAIT time.

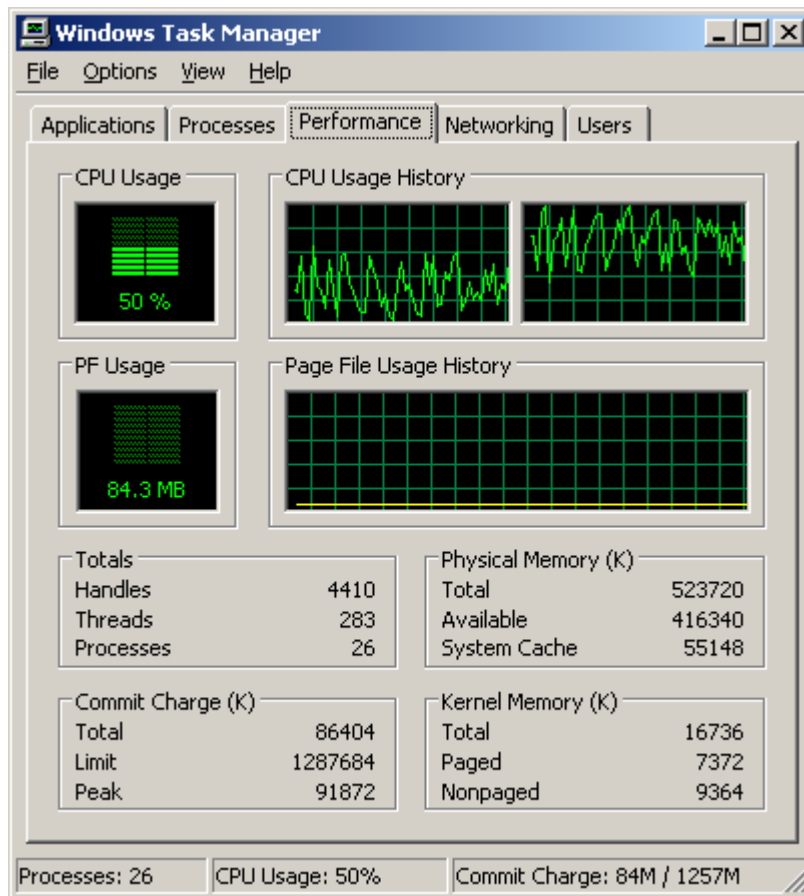


%READY is the amount of time a virtual machine is ready to run but can't get scheduled due to other virtual machines using the physical resources. Does one CPU have more %READY time than another one or are they about the same? There are always helper processes and the Service Console running on one of the physical CPUs so even with just one virtual machine running there could be some %READY time.

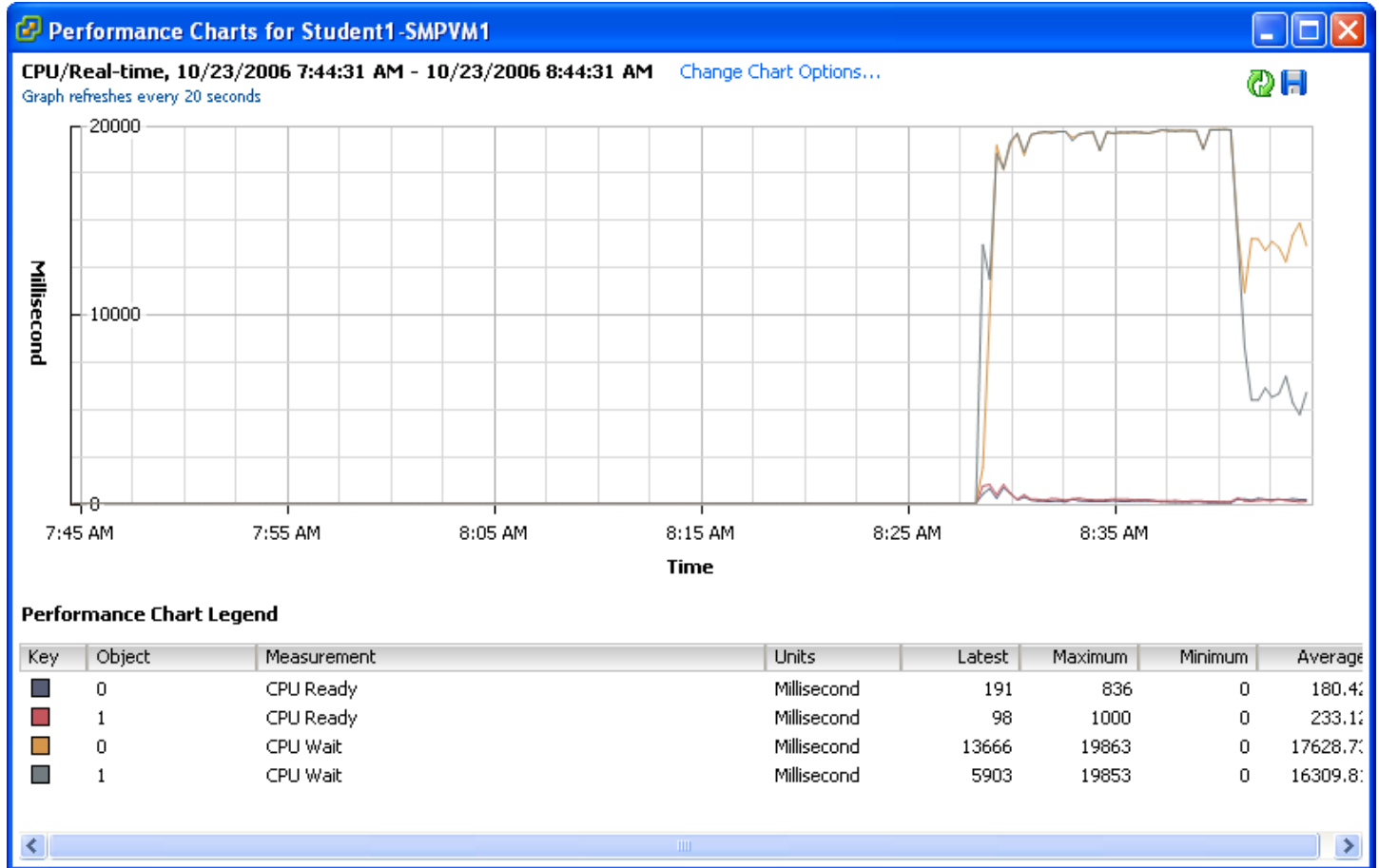


%WAIT is the amount of time a virtual machine did get scheduled but the processors have nothing to process and so the CPU simply waits while the scheduled time for the virtual machine clicks by. With just one virtual machine running and no activity in the virtual machine we can see both processors are just sitting idle.

Now that the VM is started let's launch one instance of the CPUBusy script located on the desktop. Also open Task Manager and change to the Performance tab. What do you see in the virtual machine? You should see two virtual CPUs in the performance graph and the total performance of the system at 50% utilization. The CPUBusy script is a single-threaded application so it will only keep one CPUBusy at a time.



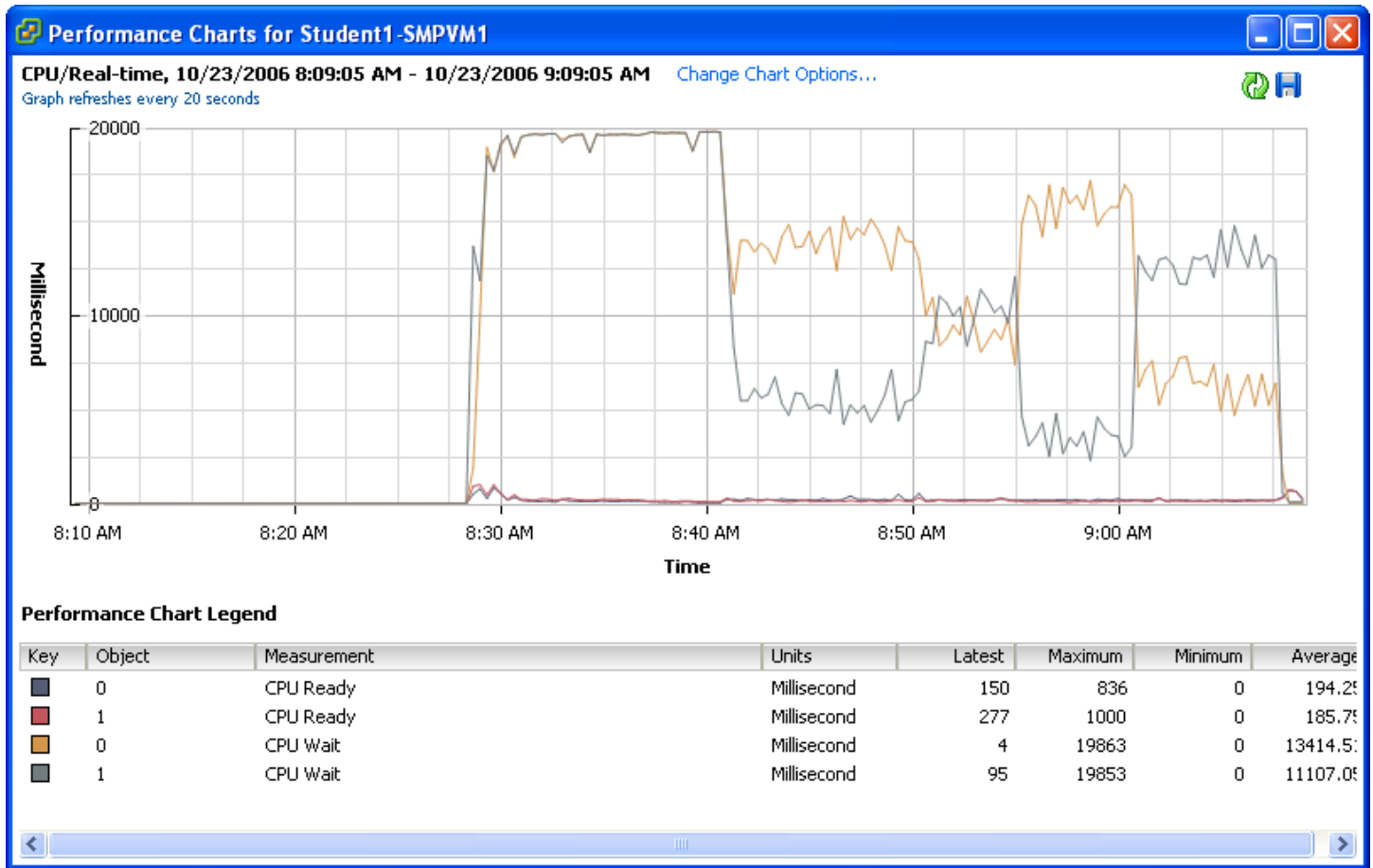
Now change back to the Virtual Infrastructure Client performance graph. What do you notice in this graph? Has the %READY or %WAIT time changed at all? Are the percentages balanced across all CPUs?



As you'll see the %READY time really didn't change that much. This is because the virtual machine really doesn't have any competition for the CPUs. The %WAIT time has gone down considerably but only on one of the CPUs. This is because the application inside the virtual machine is only a single threaded application and it does not use the extra CPU.

If you see unbalanced %WAIT times in your normal application workloads then you should adjust your virtual machine so that it only uses a single virtual CPU.

Next, let's go back to Student#-SMPVM1 and start a second CPUBusy script. What happens to the performance displayed in task manager? Now the virtual machine is running at 100% utilization because there are two single threaded applications. The operating system is spreading each thread across a different virtual CPU. Go back to the performance graph for Student#-SMPVM1. Has anything changed?



The %READY time is still low because there are still no resource contentions. The %WAIT time for both CPUs is also low now since both virtual CPUs are busy. The two virtual CPUs get scheduled on separate physical CPUs.

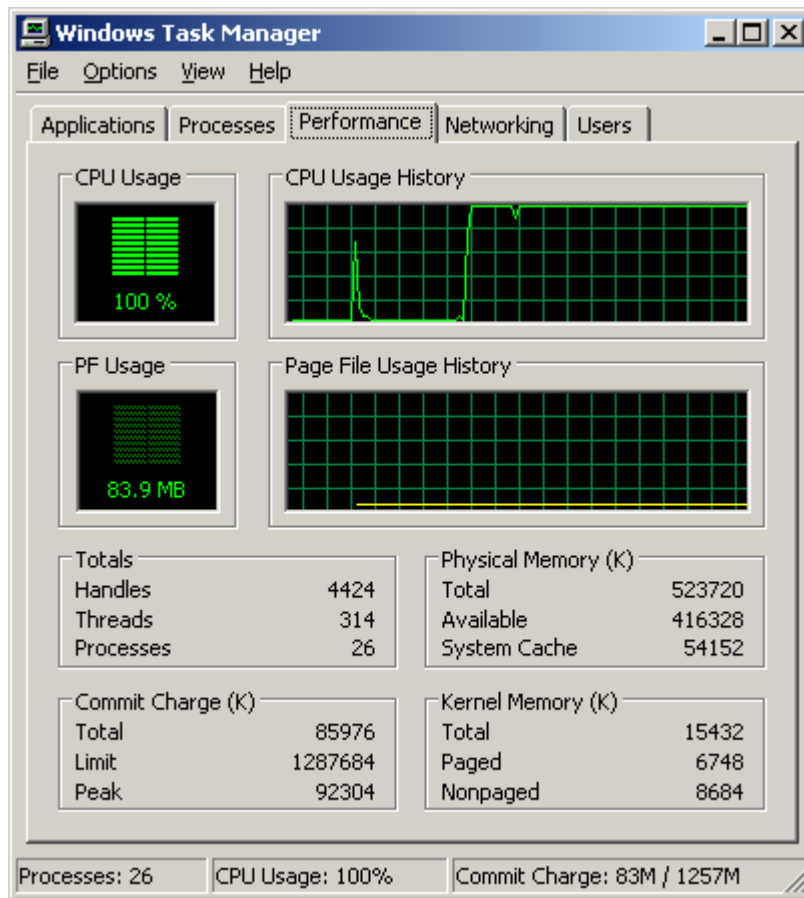
If you see balanced %WAIT times and they are low then you have used VSMP appropriately in your environment.

Currently we have one virtual machine running on our server with no resource contention. Now let's power on Student#-UPVM3. This will start to introduce contention for the physical CPUs. While Student#-UPVM3 powers on watch what happens to the performance of Student#-SMPVM1. You should see the %READY time continue to climb for Student#-SMPVM1. You may also see a very high %READY time for Student#-UPVM3. Starting an operating system is a very resource intensive task.

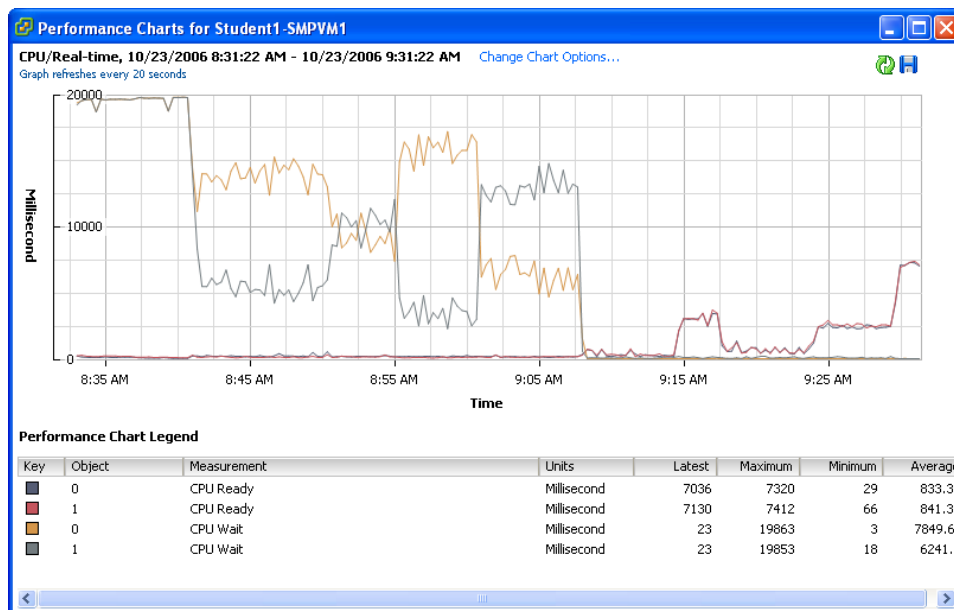
If you have several virtual machines running then you should not perform a lot of power operations on your virtual machines since this will drastically degrade performance.

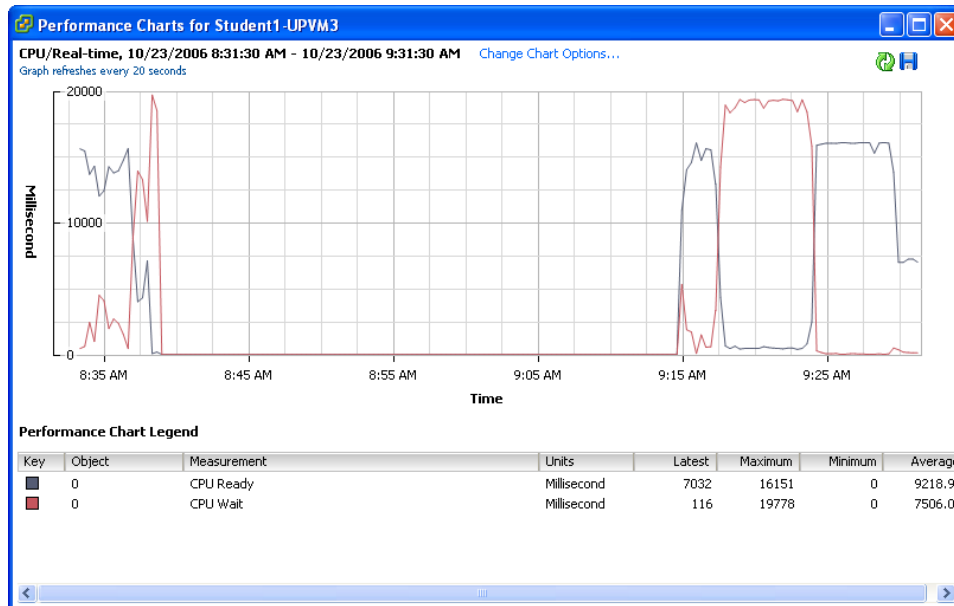
After Student#-UPVM3 has settled down you will notice that the %READY time comes back down. You will also notice a higher %WAIT time for Student#-UPVM3. Since there is nothing going on inside Student#-UPVM3 there will be a high %WAIT time and low %READY times.

Next, let's start a CPUBusy script in Student#-UPVM3. Also open the Performance tab in Task Manager so you can see what happens to the resources inside the virtual machine.



As you can see the single threaded application is taking up 100% of a single virtual CPU as expected. Now look at the performance graphs for Student#-UPVM3 and Student#-SMPVM1. What happened to the %READY time? What about the %WAIT time?





As you can see there is now contention for resources. All of the virtual CPUs for all of the VMs have a high %READY time. Since all of the virtual machines has the same number of shares and are not limited by CPU settings they should split the time on the CPU equally and as such all of the %READY times should be approximately equal. The %WAIT time is still low because the virtual CPUs are being used every time they are scheduled on the physical CPUs.

For additional workload analysis and to see the affect of VSMP on normal workloads repeat these lab concepts with SMP workloads in your environment. SQL and Exchange are simple and good tests for SMP workloads in Windows. File Servers are simple tests for apps that scale poorly in Windows SMP environments (SMP should be avoided). Kernel compiles or any product builds are examples of single-threaded applications in a Linux environment (SMP should be avoided).

Lessons Learned

In this section we learned that SMP is not the answer to every performance issue and could very well be the cause of some of your performance issues. At the same time if you have a SMP aware application like databases then VSMP could help your performance. Tests should definitely be performed with different apps in a lab configuration to select the best number of processors for that given application.

We also learned the impact that poor VSMP selection can have on a virtual machine. We learned how to monitor %READY and %WAIT to determine if we have resource contention or extra resources when using VSMP.

Reproducing This Lab at Home

See [Appendix C](#) for instructions on how to reproduce this section of the lab in your own environment.



SECTION 4: STORAGE PERFORMANCE

Background information

Disk related performance issues usually surface when using disk intensive applications. Disk activities are the most time consuming part of virtualization processor-wise. The running virtual machine must be swapped with the virtualization layer to process the I/O interrupt. The I/O is serviced and placed into primary memory (RAM or cache). The memory access also takes CPU cycles away from the virtual machine. Finally, the virtual machine is scheduled on the physical processor again and allowed to run. For I/O intensive applications this can become a potential bottleneck.

VMware has two different disk access mechanisms – VMFS and RAW Disk Maps (RDMs). For more information about the differences please consult the Virtual Infrastructure 3 Server Configuration Guide (http://www.vmware.com/pdf/vi3_server_config.pdf). In some cases Raw disk access may yield better performance. Depending on the workload and your architecture you may want to consider the use of RAW disks sparingly. For a good look at the pros and cons of RAW disk usage see [Appendix E](#).

The aim of this section is to show how the performance of different storage access methods differs. We will also look at the limitations of storage configurations and lend guidance on where a particular architecture would yield better performance. In this section you will run an industry standard disk benchmarking tool to evaluate the performance of fibre channel attached disks in both VMFS and RDM access modes. You will also discover where bottlenecks may reside in our lab architecture and how to improve upon this architecture in your own environment. Additionally, you will look at different storage mediums such as iSCSI and NAS. For the interest of time the lab instructors have already run the tests for iSCSI and NAS datastores.


Please note that this test is not intended to be a competition between vendors or protocols. This section was designed to help you understand the different trade-offs when selecting and architecting storage. Performance can be a complicated function of many factors. Unfortunately we do not have an unlimited amount of equipment for the VMworld 2006 Performance Lab. At times you will experience mixed results and new performance bottlenecks. These situations will be highlighted as areas where a different architecture could yield better results. Performance bottlenecks are not always a limitation of the ESX Server architecture or the type of array you are using. We encourage you to take this knowledge back to your own environment to find the limits of your current setup so you can plan your storage and virtual machine layout accordingly.

Lab Practice

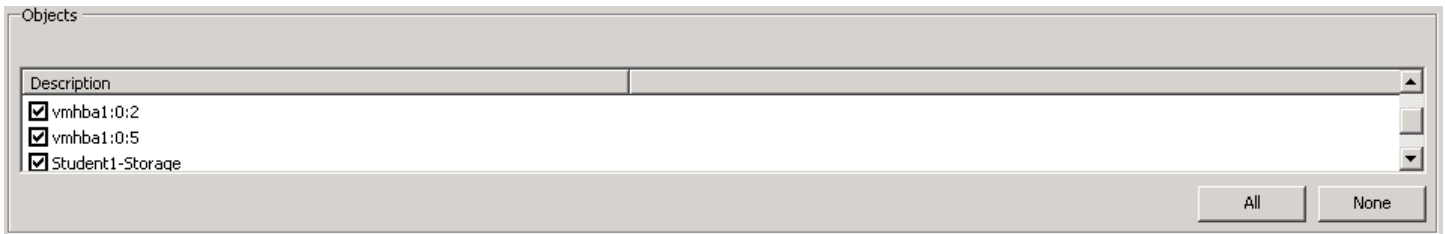
During the following tests you may notice out of disk space warnings. These are normal during an IOMeter test since IOMeter fills the disk with test data. It is safe to ignore these errors.

Step 1

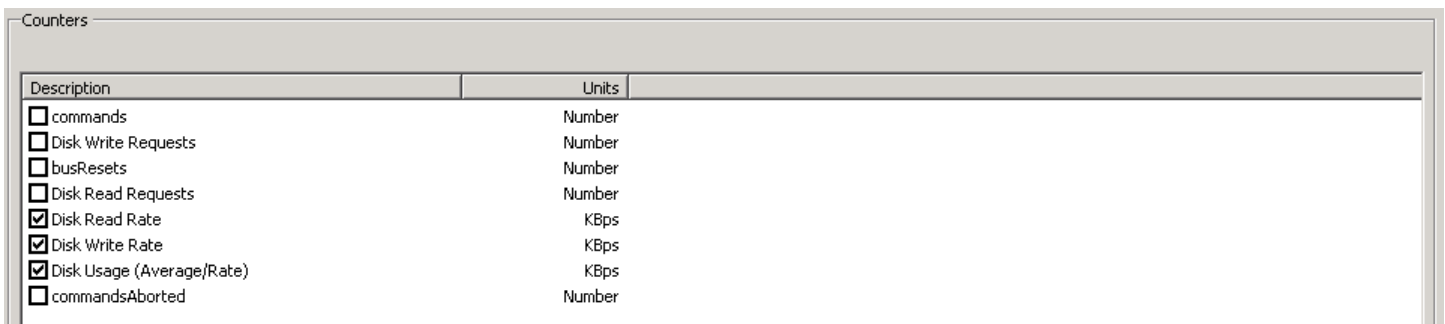
You will be working with a virtual machine called Student#-Storage.

1. Power on Student#-Storage virtual machine.
2. Open a remote console to the virtual machine. The virtual machine should automatically logon.
3. Select the Performance tab for Student#-Storage and then select the popup chart icon  to display the performance chart in its own window.
4. In the performance chart window, click on Change Chart Options → Disk → Real Time.

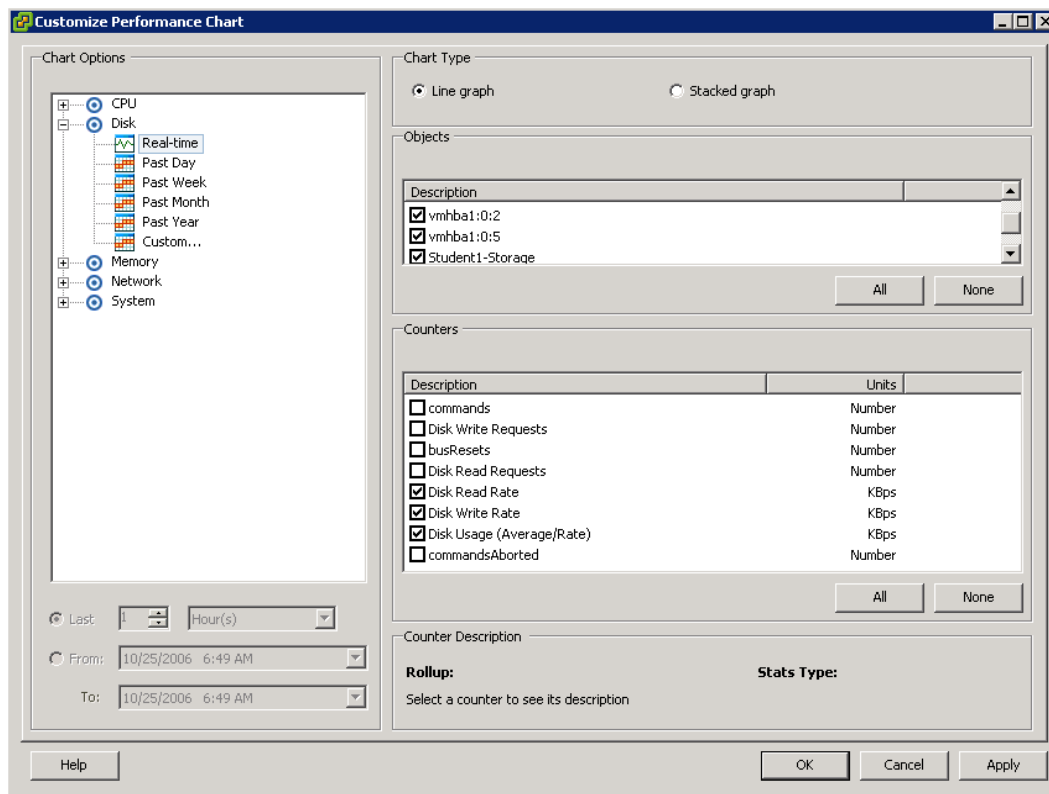
5. Select the **ONLY** following checkboxes in the Objects area:
 - a. Vmhba1:0:2
 - b. Vmhba1:0:5
 - c. Student#-Storage



6. Select **ONLY** the following checkboxes in the Counters area:
 - a. Disk Read Rate (in KBps)
 - b. Disk Write Rate (in KBps)
 - c. Disk Usage Rate (in KBps)



7. Your final setup should look similar to the picture below.



Points:

- These settings will be used to measure disk performance at the ESX Server level.



Step 2

In this step we will run the first of our storage tests and record the results next to our pre-run tests for later comparison. We are going to use a disk benchmarking tool called IOMeter to create I/O load within the virtual machine. This utility is freely available on the Internet and has many different settings. The tool has already been setup for the tests that we will run. For more information about setup for this lab see [Appendix D](#).

The virtual disk for this test is already setup and connected to the Student#-Storage virtual machine and appears as drive letter E inside the Guest OS. For this test we will be testing a standard 10 GB virtual disk (vmdk) placed on a single VMFS partition. The VMFS partition is on an array shared by 5 other ESX Servers. The VMFS volume is only viewable by your ESX Server.

The conditions that IOMeter uses for this test are saved in My Documents. (VMworld-FC-VMFS.icf)

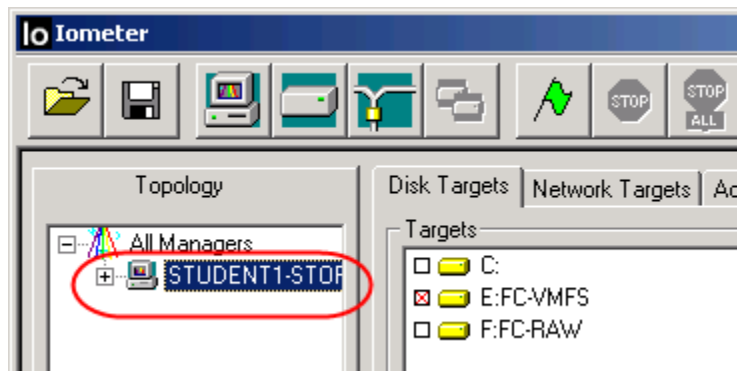
1. Launch IOMeter by double-clicking on the icon on the desktop.



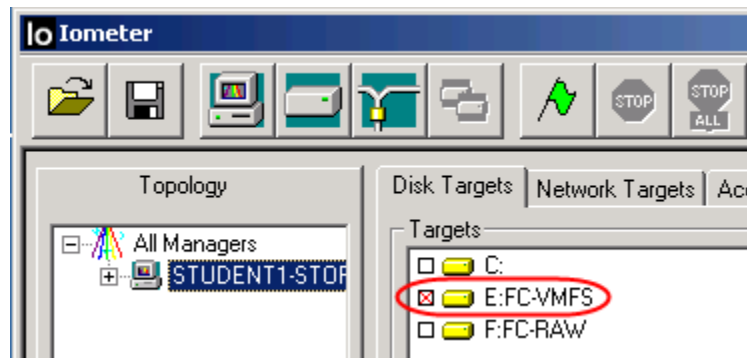
2. Open the test settings for the VMFS test.
 - a. Use the open icon to browse for the test settings (VMworld-FC-VMFS.icf).



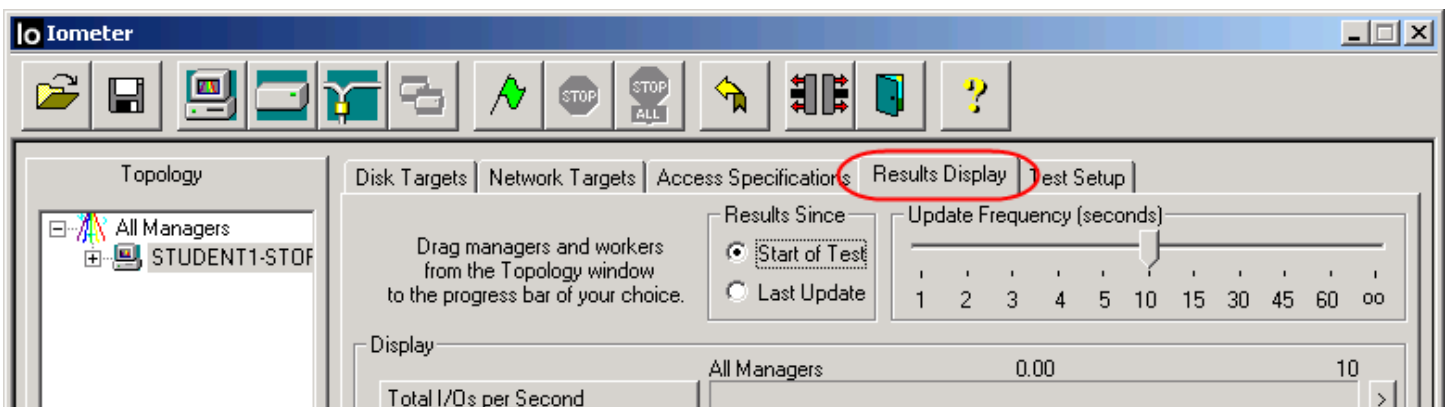
- b. Open the test settings.
3. Once the test settings are open highlight the only manager in the list called Student#-Storage.



- Confirm the disk is set to the FC-VMFS disk with the little red X in the checkbox next to drive E.



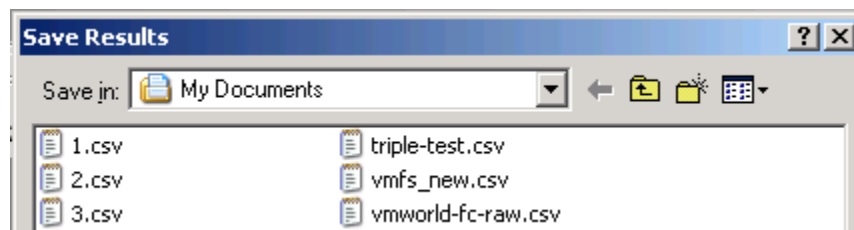
- Change to the Results Display Tab to monitor the results.



- Start the test by clicking the start flag in IOMeter.



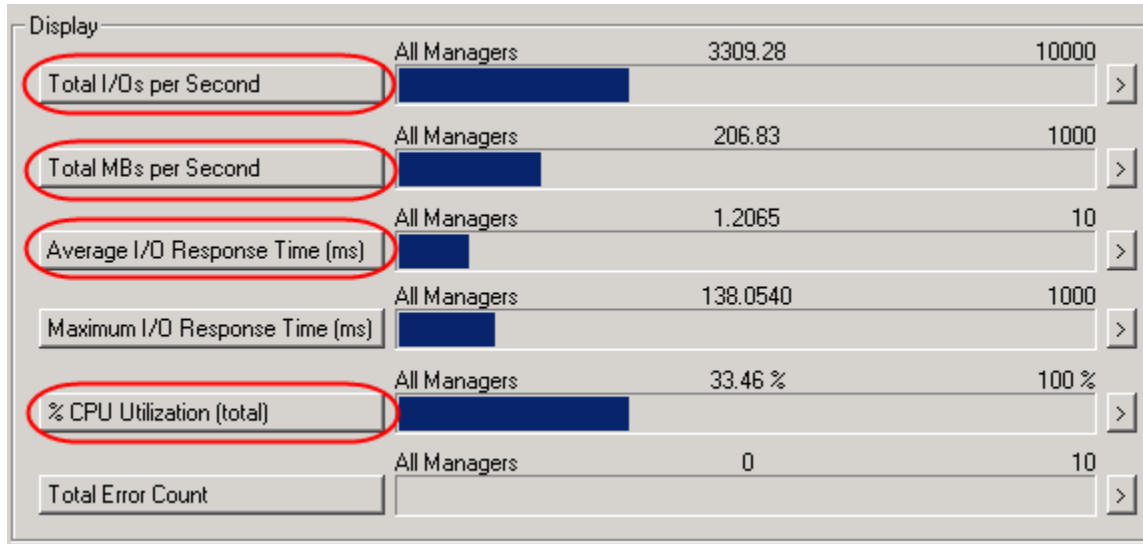
- The test will prompt to save the results as a csv file. We will not need these results so you can use any name you want for the file and click save.



- The test will run for a 30 second warm up period where no results will be displayed followed by a 5 minute run to test I/O.

- Once the test is completed the red stop sign  will disappear and the green flag  will return.

10. Record the results of this test in the chart in [Step 4](#). Note that not all results will be used. Only record the circled results.



Did your results match up to the pre-run fibre channel VMFS test? Why or why not?

More analysis will be done as part of [Step 4](#).

Step 3

One of the features of Virtual Infrastructure 3 is the ability to connect directly to a storage volume using something called Raw Device Mappings. Raw Device Mappings were created to allow direct SCSI command access to the underlying storage. This is important when using SAN management tools for managing SAN snapshots for example.

Customers often ask if going directly to the disk is faster than going through the VMFS file system. The answer is it depends on the workload. In this step you will run the same performance tests as Step 2 only you will use a Raw Device Mapping this time.

For more information about the pros and cons of using Raw Disk Mappings see [Appendix E](#).



The Raw Device Mapping for this test is already setup and connected to the Student#-Storage virtual machine and appears as drive letter F inside the Guest OS. We will be testing a standard 10 GB SAN volume connected as a Raw Device Mapping. The SAN volume is on an array shared by 5 other ESX Servers. The SAN volume is only viewable by your ESX Server.

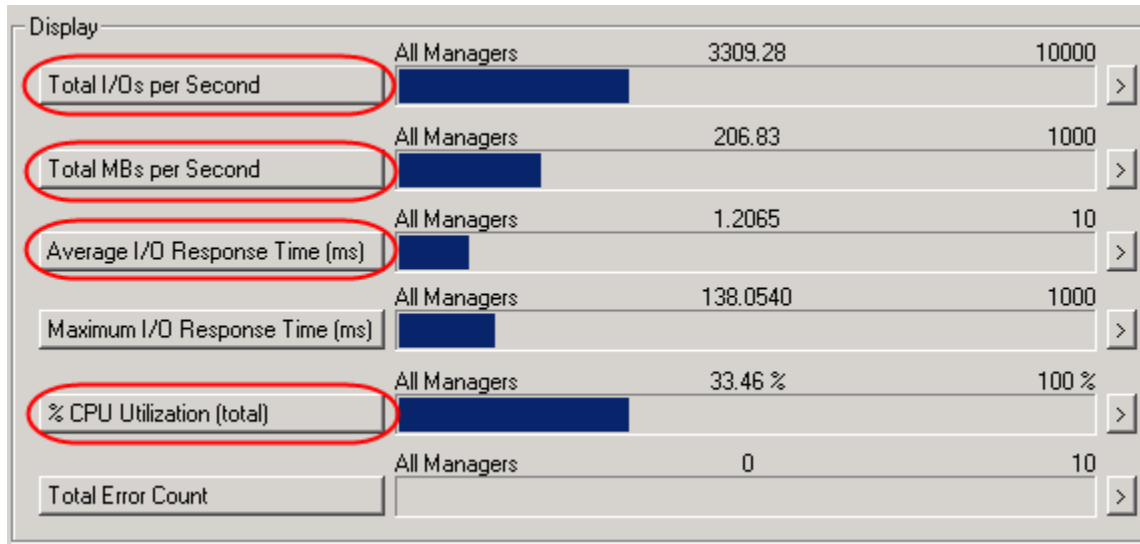
The conditions that IOMeter uses for this test are saved in My Documents. (VMworld-FC-RDM.icf)

The steps for this test are nearly identical to the previous step. Refer to the screen shots in [Step 2](#) for assistance.

1. Launch IOMeter by double-clicking on the icon on the desktop.
2. Open the test settings for the VMFS test.
 - a. Use the open icon to browse for the test settings (VMworld-FC-RDM.icf).
 - b. Open the test settings.
3. Once the test settings are open highlight the only manager in the list called Student#-Storage.
4. Confirm the disk is set to the FC-RDM disk with the little red X in the checkbox next to drive F.
5. Change to the Results Display Tab to monitor the results.



- 6. Start the test by clicking the start flag in IOMeter.
- 7. The test will prompt to save the results as a .csv file. We will not need these results so you can use any name you want for the file and click save.
- 8. The test will run for a 30 second warm up period where no results will be displayed followed by a 5 minute run to test I/O.
- 9. Once the test is completed the red stop sign  will disappear and the green flag  will return.
- 10. Record the results of this test in the chart in [Step 4](#). Note that not all results will be used. Only record the circled results.



Did your results match up to the pre-run fibre channel RDM test? Why or why not?

Were your test results for Raw Disk Mappings better or worse than the VMFS tests? Why or why not?

More analysis will be done as part of [Step 4](#).

Step 4

In this step we will take a look at our results and try to figure out what these performance numbers mean. The conclusions drawn in this section are partly related to the limited equipment we have to test on and are by no means completely conclusive as to what will happen in your environment. You should take these notes and these tests and see where the performance bottlenecks reside in your environment.



	Fibre Channel Student Results		Fibre Channel Pre-Run		iSCSI Pre-Run		NAS Pre-Run
	VMFS	RDM	VMFS	RDM	VMFS	RDM	VMDK
Total I/Os per Second (IOPS)			3294	3353	1813	1865	1691
Total MBs per Second (Throughput)			206	209	113	116	105
Average I/O Response Time (ms)			1.21	1.19	2.20	2.14	2.36
% CPU Utilization (total)			33.87%	27.26%	24.00%	19.40%	23.00%

Question 1: Now you have run two tests. One of your tests benchmarked a VMFS partition and the other benchmarked a Raw Disk Mapping. Did you notice any difference between the two? Use the pre-run numbers for comparisons as well.

In general there are very few workloads that take any advantage of RDMs. RDMs should be used sparingly in your environment and not as general crutches for performance gains. More analysis of RDMs will be done in [Step 5](#). For more information on the benefits of RDMs see [Appendix E](#).

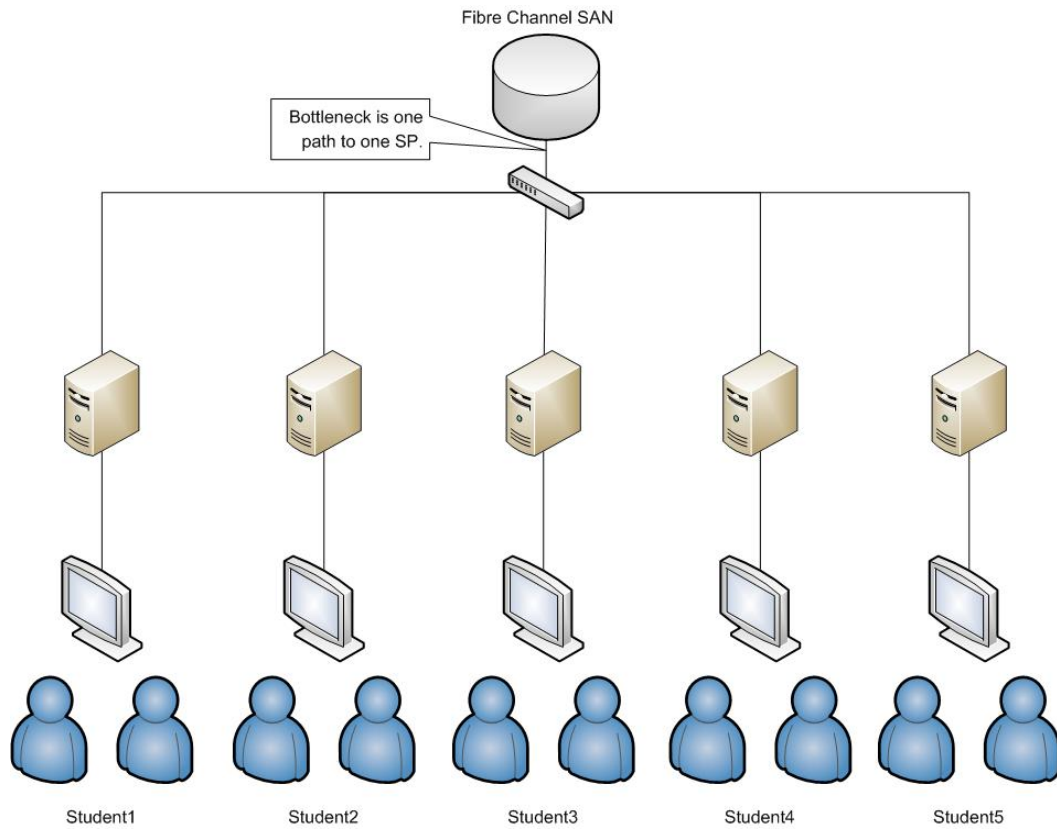
Point:

- RDMs do not always give better performance than VMFS partitions.

Question 2: Were your numbers very different from the pre-run numbers? Why do you think that is?

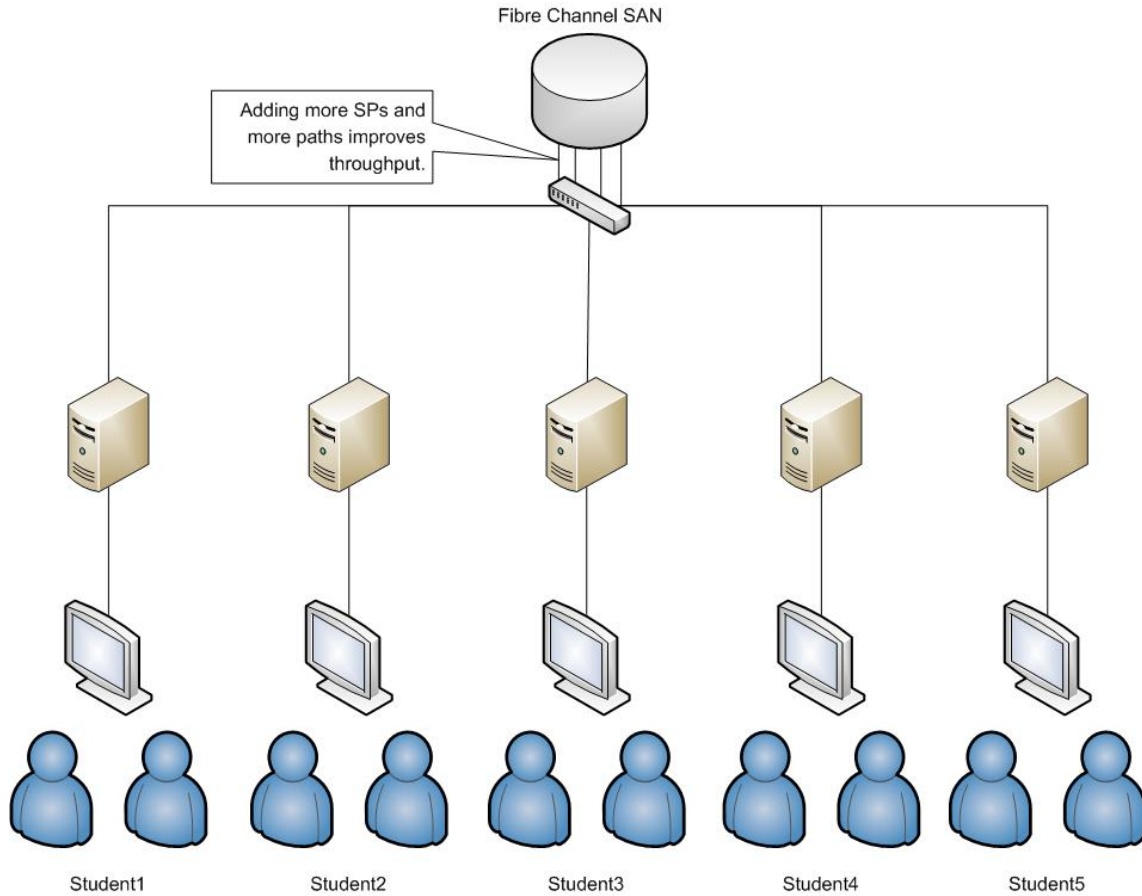
The pre-run numbers were gathered in a clean environment when there was only one student using the system. Each POD in our architecture has a single HBA going through a single fibre channel switch into a single storage processor (SP) on the array. There is only one path for all traffic between the five ESX Servers in the POD and the one SP in the array. Because of our poor SAN fabric architecture the storage processor becomes the bottleneck for performance. The array actually has plenty of processing power left. The more disk tests that are run the more the available storage processor I/O is split between the different requesters. Since you are not the only student using the array currently there is competition from other students at the storage processor level.





Question 3: How do you think you could fix the storage processor bottleneck?

The best way to fix the storage processor bottleneck is to add more storage processors and paths between the SAN fabric and the storage processors.



Point:

- Use more storage processors and paths to the storage processors to increase the available throughput in your SAN.
- The best way to improve storage performance is to have a good architecture for your SAN fabric.

Question 4: How did the numbers compare between NAS, iSCSI, and Fibre Channel? Were you surprised by any of the results? Keep in mind that we are still storage processor bound for these results.

In general iSCSI offers good performance for non-disk intensive apps. Some iSCSI vendors offer better performance than other so these numbers are not indicative of everything you might experience with iSCSI. Some vendors offer better performance while some offer worse performance. It is always a good idea to evaluate a vendor in your own environment to test performance results.

Step 5

The last section of this lab has further analysis included. There are no lab exercises in this section however there are some good data points. It is highly suggested that you keep reading. If you need to end your lab now please pay close attention to the instructions at the [end of this section](#).



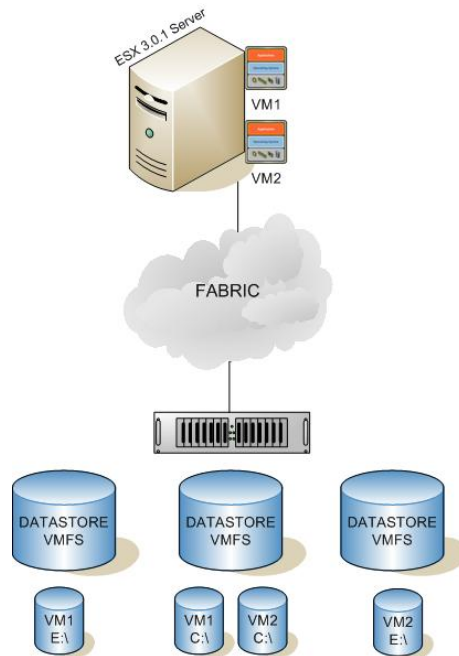
Due to the lack of equipment in our lab setting the lab instructors performed some follow-up tests before VMworld on this same equipment. The goal of the tests was to see what else we could tune in our environment to get the best possible storage performance.

In all of the test results below we will show the results in a chart. The following labels will be used in the chart:

- IOPS – This shows the total number of I/O operations per second. For example, total number of reads and writes.
- MB/s – This shows the total throughput per second expressed in Megabytes (MB).
- Latency – This shows the average amount of time an I/O request took to process. Time is reflected in milliseconds.
- %CPU – This is the percentage of CPU time the performance test consumed. In general this does not have an impact on our test results but we still like to track it to make sure we are not CPU bound.

Split Data Stores

The first test we ran was from a single ESX Server to two different VMFS datastores on the same array. Each virtual machine had a second virtual disk that appeared in the GuestOS as drive letter E.



	IOPS	MB/s	Latency	%CPU
VM1	1961	123	2.04	22.27%
VM2	1983	123	2.01	22.37%
Total	3944	246		

As expected, our performance bottleneck is still at the single path to the one storage processor. The I/O is still be divided between all requesters. However, if we add up the total IOPS and MB/s we see that we far exceeded the results of our Pre-Run single VMFS tests. Why do you think this is?

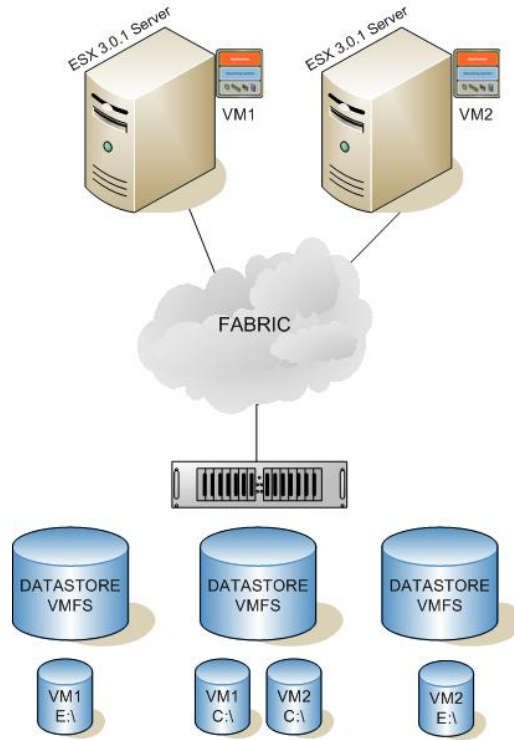
In Virtual Infrastructure 3 each volume gets its own disk queue to work from. It appears that with a single VMFS we were actually flooding the disk queue on the local ESX Server. By adding datastores to our environment we actually increased performance. Please note that at this point we could have adjusted the queue depth in the Windows Guest, on the HBA driver, in the HBA BIOS and even on our storage array as per best practice guidelines. The settings that would give us the best performance are different for every configuration and this is where your storage specialist or vendors can help you find the best settings for your environment.

Point:

- Splitting heavily accessed datastores up may lend better performance.
- This is a time when RAW Disk Mappings for each heavily hit volume may make sense in order to increase storage performance.

Hitting the Bounds of the HBA

To make sure we were not bound by the HBA we decided to run a test from two separate ESX Servers to the same two VMFS partitions that we used in the previous test. The two VMFS partitions are still on the same array with the single path, single storage processor bottleneck.



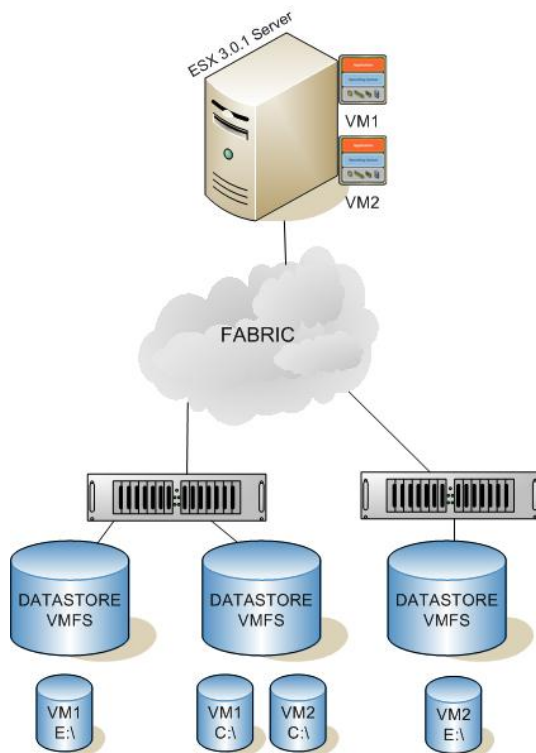
	IOPS	MB/s	Latency	%CPU
VM-Host1	1980	124	2.02	20.30%
VM-Host2	1989	124	2.01	20.70%
Total	3969	248		

The IOPS and MB/s really didn't change significantly. This means that we're still bound at the same performance point – the single path and the single storage processor. To investigate the potential for where our HBA bound might be we decided to run one more test.

Point:

- We have confirmed that multiple storage processors and multiple paths (a good SAN architecture) are key to unlocking more storage performance.

In this test we ran from a single ESX Server with a single HBA to two different VMFS partitions on two different storage arrays. This test essentially gives us twice the number of storage processors and paths on the array side but still only one path on the ESX Server side.



	IOPS	MB/s	Latency	%CPU
VM-Array1	2048	131	1.90	20.88%
VM-Array2	2153	134	1.86	20.08%
Total	4201	265		

This test saw a good jump in IOPS and MB/s for individual virtual machines as well as for the combined results. The increase in performance was not equal to our single VMFS, single virtual machine, and single host tests because we still have a bottleneck. The bottleneck has now moved to the HBA in the physical ESX Server. Increasing the queue depth may help but only if you find the right combination between the Guest OS, the driver, the BIOS, and the array. A better way to increase performance would be to add another HBA to the system. Please note that ESX Server 3.0.1 and prior do not load balance dynamically across HBAs. You can however manually assign the preferred path for Array1 to HBA1 and the preferred path for Array2 to HBA2. This will balance the load for the two VMFS partitions between the two HBAs and increase your performance.

If you are using an active/passive array then you **MUST** use the MRU failover policy in ESX Server. This means that all ESX Servers must have the same path to a given LUN as their primary path. With an active/active array you can have different ESX Servers use different paths and storage processors to the same LUN at the same time. With active/active arrays you get the ability to do more granular manual load balancing.

Points:

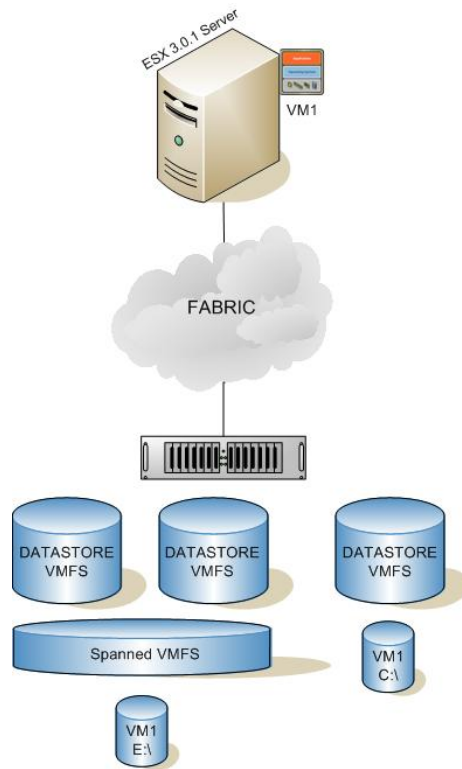
- Adding additional HBAs to your system will increase your performance.
- You can manually load balance LUNs down different HBAs.
- Using an active/active array with a fixed failover policy in ESX Server will enable you to balance load even more.



Spanned VMFS Volumes and Performance

If more LUNs help with performance then will spanning a single logical VMFS volume across two VMFS extents increase performance? What do you think?

The next test starts by creating a single logical 20 GB VMFS partition out of two 10 GB VMFS partitions. The smaller partitions become extents of the larger logical volume. To individuals creating virtual machines it appears as one 20 GB volume in the interface. We ran the test with a single virtual machine with its disk on the 20 GB logical volume. The ESX Server still has a single HBA and we are using a single array again. You should compare these results to the pre-run Fibre Channel VMFS test in [Step 4](#).



	IOPS	MB/s	Latency	%CPU
Student#-Storage	3328	208	1.20	32.74%

As you can see, the tests revealed no significant increase or decrease in performance using spanned VMFS partitions. Contrary to popular belief, spanned VMFS volumes are simply concatenated. The volume begins to fill from the beginning of the first extent to the end of the first extent and then from the beginning of the second extent to the end of the second extent. There is no stripping in a spanned VMFS volume.

Points:

- Spanned VMFS volumes should only be used to increase storage space and should never be used to try and increase performance.
- It is better to add a second, separate VMFS volume to increase disk space rather than span into another extent.
- Spanned VMFS volumes have 1 disk queue. Separate VMFS volumes have separate disk queues. More disk queues increase storage performance.



Lessons Learned

In this section you learned how to tell where a storage bottleneck may exist and ways to correct the issue. This lab section is not a complete inclusion of every type of storage performance issue that you may face. If you encounter storage related performance issue you are encouraged to open a VMware Support Request or contact your local storage vendor.

The tests in this lab can and should be run in your own environment to find out if you have any hidden storage performance bottlenecks. Please make sure you read the note below before running these tests in your environment.

NOTE: Every environment is different. If you decide to run this test in your environment your numbers may be different for a variety of reasons. Many things will change the results of your tests such as SAN fabric architecture, speed of disks, speed of HBAs, number of HBAs, etc. The numbers introduced in this lab are by no means meant to be an official benchmark of the lab equipment. The tests run were simply used to create a desired performance issue so that a point could be made. Please consult your storage vendor contacts for official benchmarking numbers on their arrays in a number of environments

Reproducing This Lab at Home

See [Appendix D](#) for instructions on how to reproduce this section of the lab in your own environment.



AFTER LAB CLEANUP

You are now done with your lab. Please shut down any remaining VMs so the next set of students can start fresh. Thank you and enjoy your time at VMworld.



APPENDIX A: TIMING IN A VM LAB SECTION REPRODUCTION**Software Required**

- Windows 2003 Server (Enterprise Edition was used in this lab)
- CPUBusy script. Please read the notes below.
- VMTools
- Virtual Infrastructure 3 or later (ESX Server 3, Virtual Infrastructure Server, etc)

Hardware Required

- At least 2 physical CPUs
- Hyper-threading is preferred to be disabled for this test.
- At least 2 GB of memory
- At least a 15 GB VMFS3 partition

To repeat the experiments in this section two virtual machines are needed. The guest operating systems used were Windows 2003 Enterprise Edition. Each virtual machine was defined with a single vCPU, 512 MB of RAM and a 5 GB disk.

The virtual machines need to have VMTools installed. The *vmtoolsd* driver is included in the VMTools installation media but is not installed by default. The [Lab Practice](#) includes a link to a white paper on the VMware website showing how to install the driver.

Important! Both of the virtual machines need to be pinned to the same logical CPU.

The utility that puts a load onto the vCPU is a script called CPUBusy. For information on creating this script, please refer to [Appendix F](#).

You can create a shortcut on the virtual machine desktop to the CPUBusy script if you wish for easy access. Ensure that both virtual machines have the script file and the batch file.

The *vmtoolsd* driver can run on other guest operating systems. The current version currently supports uniprocessor versions of Windows 2000, Windows XP and Windows 2003. It also supports all uniprocessor releases of 2.4x and 2.6x kernels.

The *vmtoolsd* driver is still a work in progress. Please report any inconsistencies to VMware support.

APPENDIX B: RESOURCE POOLS LAB SECTION REPRODUCTION

Software Required

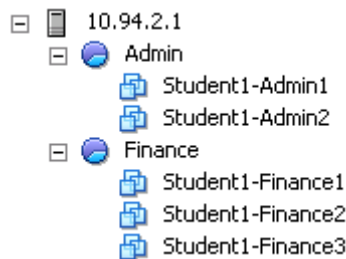
- Windows 2003 Server (Enterprise Edition was used in this lab)
- CPU Busy script. Please read the notes below.
- VMTools
- ESX Server 3.0 or later

Hardware Required

- Virtual Infrastructure Server
- At least 2 physical CPUs
- Hyper-threading is preferred to be disabled for this test.
- At least 2 GB of memory
- At least a 15 GB VMFS3 partition

Creating Resource Pools and VMs

Create an "Admin" resource pool and a "Finance" resource pool. Populate the Resource Pools with five (5) VMs (Admin1 and Admin2 in the Admin Resource Pool and Finance1, Finance2, and Finance3 in the Finance Resource Pool).



Initially configure the resource settings for the Resource Pools and Virtual Machines as follows:

Admin Resource Pool: CPU Reservation: 750 MHz, "Expandable Reservation" **not** checked.

Finance Resource Pool: CPU Reservation: 3000 MHz, "Expandable Reservation" checked.

Admin1: 2 VCPU. CPU: Reservation 1500 MHz, Shares: Normal (2,000).

Admin2: 2 VCPU. CPU: Reservation 2000 MHz, Shares: Normal (2,000).

Finance1: 2 VCPU. CPU: Reservation 0 MHz, Shares: Custom (80,000).

Finance2: 1 VCPU. CPU: Reservation 0 MHz, Shares: High (4,000).

Finance3: 1 VCPU. CPU: Reservation 0 MHz, Shares: Normal (2,000).

Adjustments for Hardware

Do not worry about the numerical values of shares; just be sure the relative ratios are the same.

The lab was written assuming that there is a single ESX host that has 2 CPUs at 3.0 GHz. Adjust the Reservations up if the CPUs have a higher speed such as 3.6 GHz. Adjust the Reservations down if the CPUs have a lower speed such as 2.0 GHz. These CPUs are not dual-core, and hyper-threading if any has been disabled. If a host has dual-core CPUs, count each core as a CPU. If you have 4 CPUs, double the VCPUs in each VM. When running CPUBUSY in 4-way VSMP virtual machines, sections that say to run one instance of CPUBUSY stay the same; in sections that say to run 2xCPUBUSY, run 4xCPUBUSY instead.

Software in the VMs

The virtual machines should ideally have VMTools installed.

The utility that puts a load onto the vCPU is a script called CPUBUSY. For information on creating this script, please refer to [Appendix F](#). Please install CPUBusy on each virtual machine.

APPENDIX C: CPU PERFORMANCE LAB SECTION REPRODUCTION**Software Required**

- Windows 2003 Server (Enterprise Edition was used in this lab)
- CPU Busy script. Please see [Appendix F](#) for instructions on building this script.
- VMTools
- ESX Server 3.0 or later

Hardware Required

- Virtual Infrastructure Server
- At least 2 physical CPUs
- Hyper-threading is preferred to be disabled for this test.
- At least 2 GB of memory
- At least a 15 GB VMFS3 partition

Virtual Machines

- Two virtual machines
 - One virtual machine with 1 virtual CPU
 - One virtual machine with 2 virtual CPUs
- 512 MB RAM each
- 5 GB virtual disk for C drive



APPENDIX D: DISK PERFORMANCE LAB SECTION REPRODUCTION**Software Required**

- Windows 2003 Server (Enterprise Edition was used in this lab)
- IOMeter (<http://www.iometer.org/>)
- VMTools
- ESX Server 3.0 or later

Hardware Required

- Virtual Infrastructure Server
- At least 2 physical CPUs
- Hyper-threading is preferred to be disabled for this test.
- At least 2 GB of memory
- 1 – 30 GB VMFS partition for the C drive of all virtual machines
- 2 – 10 GB VMFS partitions for the E drives and VMFS based tests
- 2 – 10 GB empty SAN volumes for the F drive and RDM based tests

Virtual Machines

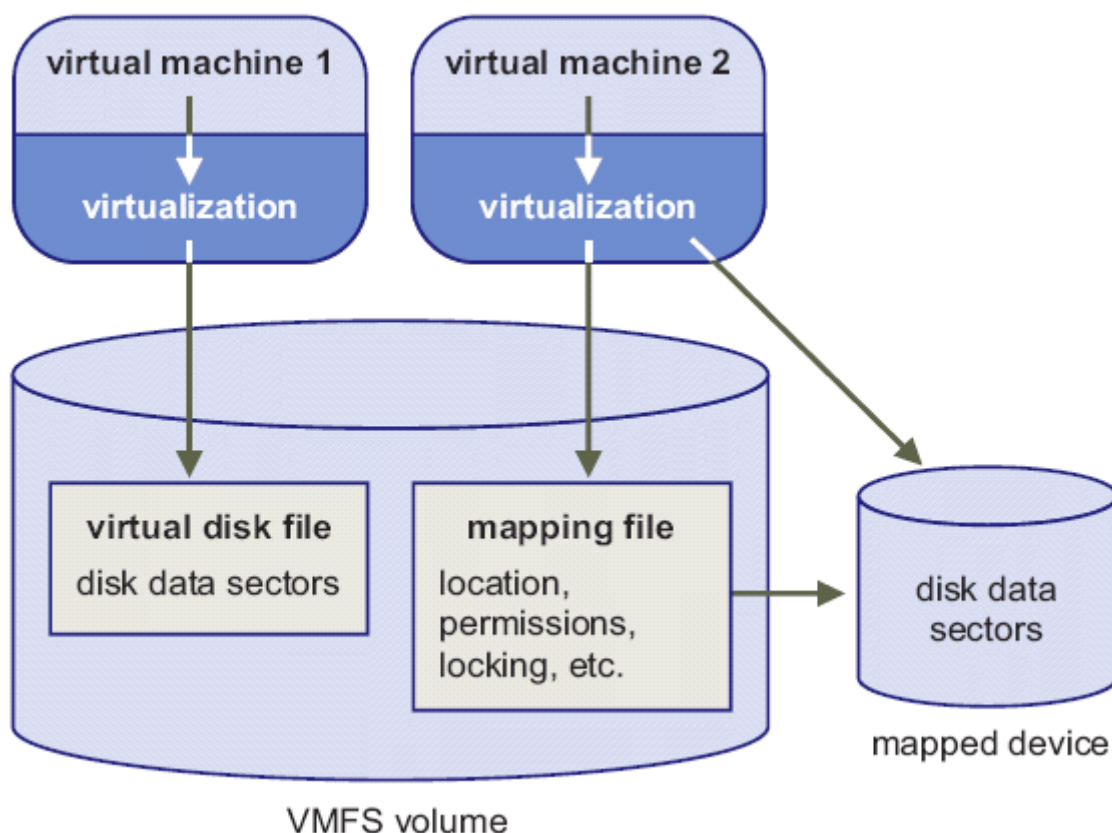
- 2 Windows 2003 Server virtual machines
 - 512 MB of RAM
 - 1 - 5 GB virtual disk
 - IOMeter Loaded
 - VMTools

NOTE: Every environment is different. If you decide to run this test in your environment your numbers may be different for a variety of reasons. Many things will change the results of your tests such as SAN fabric architecture, speed of disks, speed of HBAs, number of HBAs, etc. The numbers introduced in this lab are by no means meant to be an official benchmark of the lab equipment. The tests run were simply used to create a desired performance issue so that a point could be made. Please consult your storage vendor contacts for official benchmarking numbers on their arrays in a number of environments.

APPENDIX E: THE PROS AND CONS OF USING RDM DISKS

A virtual machine can use either a virtual disk file that resides on a VMFS datastore or a Raw Device Mapped (RDM) SAN LUN as a place to store system and production data. Given this basis, the question becomes which device is the best to use in a given situation? This appendix addresses this question and provides best practice recommendations.

When building a VMFS datastore, the underlying LUN should be geared towards the applications that will reside there. For example, write-intensive virtual machines should have their virtual disks stored on a VMFS datastore with a RAID 1+0 underlying LUN. Other low bandwidth virtual disks can reside on a VMFS datastore with a RAID 5 underlying LUN. These could include system OS virtual disks, test/development virtual disks and infrastructure virtual disks.



Production data should take advantage of Raw Device Mapped (RDM) SAN LUNs. These LUNs map directly to a LUN in the SAN, which can be created to provide a guaranteed number of I/Os and write requests. VMFS datastores cannot provide a guaranteed level of performance as they are usually servicing any given number of virtual disks. RDMs, on the other hand, map 1-to1, so the performance level is assured.

RDMs should be used with care. Several disadvantages come into play. First, you can wind up with a large number of RDMs, making SAN administration more cumbersome. The number of LUNs can increase further when provisioning new VMs, since you must clone the SAN storage as part of the provisioning process. Furthermore, template capabilities cannot be used in conjunction with RDMs. A good practice is to combine both technologies for the best output. The overall number of RDMs can be reduced by placing low-bandwidth virtual disks into a few VMFS datastores. These could include the types mentioned in paragraph 2. The remaining production volumes could be placed in limited VMFS datastores or mapped directly to SAN LUNs using RDMs.

PERFORMANCE TROUBLESHOOTING APPENDIX E: THE PROS AND CONS OF USING RDM DISKS

When using RDMs in an environment, performance can be improved in some arrays by aligning the RDM with the disk stripe. This can be done at the virtual machine level using *diskpar* for Windows VMs, and *fdisk* for Linux VMs. However, it should be noted that in most cases the benefits from adjusting the stripe are typically offset by the increased amount of administration necessary to make the modifications.

Another concern when using RDMs is System Heap Availability. ESX uses a system heap for certain operations, and the use of a large number of LUNs can diminish this heap, leading to problems. The vmkernel system heap is used to allocate memory that is used for a variety of operations. Some configurations will pre-allocate memory from the system at boot-time, leaving an inadequate amount for post-boot operations. The amount of memory pre-allocated depends on:

- The HBA driver installed (which depends on the HBA card installed)
- The number of HBA ports
- Number of LUNs
- Number of paths to the LUNs

Configurations that use Emulex memory require larger amounts of heap space. In any configuration, system heap space must remain above 6MB to avoid system errors.

To ensure that adequate heap memory is still available, poll */proc/vmware/mem*:

```
[root@admv010 root]# cat /proc/vmware/mem [Press enter key]
Unreserved machine memory: 44735 Mbytes/62081 Mbytes
Unreserved swap space: 23831 Mbytes/35095 Mbytes
Reclaimable reserved memory: 11264 Mbytes
Machine memory free: 51118 Mbytes/63306 Mbytes
Shared memory (shared/common): 13844816 Kbytes/700012 Kbytes
Maximum new 1-vcpu VM size: 3600 Mbytes
Maximum new 2-vcpu VM size: 3600 Mbytes
System heap size: 32768 Kbytes (33554432 bytes)
System heap free: 5678 Kbytes (5814728 bytes)
System map entries free: 1529
System code size: 4096 Kbytes
System memory usage: 528444 Kbytes
Node -Total-/MB    -FreeHi/MB    FreeLow/MB    Reserved/MB    Kernel/MB
0    8173312/31927    6354200/24821    198895/776    161867/632    21409/83
1    8033254/31379    6695145/26152    0/0    0/0    110702/432
TOTALS 13049345/50974 198895/776
```



PERFORMANCE TROUBLESHOOTING APPENDIX E: THE PROS AND CONS OF USING RDM DISKS

Below is a sample calculation of several LUNs connected to the same ESX Server. The effects on heap memory can clearly be seen.

System Heap Calculations

System Heap	Case A (Emulex LP10000)	Case B (Qlogic)		
Base Mem Availabe	21,000	21,000	KB Free	< Do not modify this
Driver memory	11,000	5,500	KB Used	< Do not modify this
Remaining	10,000	15,500	Remaining before load	< Do not modify this

After loading drivers but before turning on the fiber with LUNs and paths set-up

Paths	4	4		< Modify this
LUNs	24	24		< Modify this
Overhad per unit	31	15.5		< Do not modify this
Overhead used in config	2,976	1,488	kBytes Used	< Calculated

Remaining system heap free after fiber connected:

Free Amount	Free Amount
7,024	14,012

For more information on the use of Raw Disk Mappings (RDMs) in your environment consult chapter 8 of the Virtual Infrastructure 3 Server Configuration Guide (http://www.vmware.com/pdf/vi3_server_config.pdf).



APPENDIX F: CPUBUSY SCRIPT SETUP

The CPUBusy script is written in Visual Basic and very simply computes the sine of a function. There are two parts to setup and no programming skills or software are required. Below are the instructions for the two files that you need to create. Place both files in the same directory on your C: drive.

First you need to create the main Visual Basic script. It doesn't matter if you understand the code or not. Just copy and paste the information between the lines below into a standard text editor such as Notepad. Save and name the file *cpubusy.vbs*.

Cpubusy.vbs

```
Dim goal
Dim before
Dim x
Dim y
Dim i

goal = 2181818

Do While True
    before = Timer
    For i = 0 to goal
        x = 0.000001
        y = sin(x)
        y = y + 0.00001
    Next
    y = y + 0.01
    WScript.Echo "I did three million sines in " & Int(Timer - before + 0.5) & "
seconds!"
Loop
```

The next part you need is a simple batch file to launch the visual basic script with a service name. We used "cscript" to launch our script. Copy and paste the information between the lines below into a standard text editor such as Notepad. Save and name the file *cpubusy.bat*.

Cpubusy.bat

```
cscript cpubusy.vbs
```

