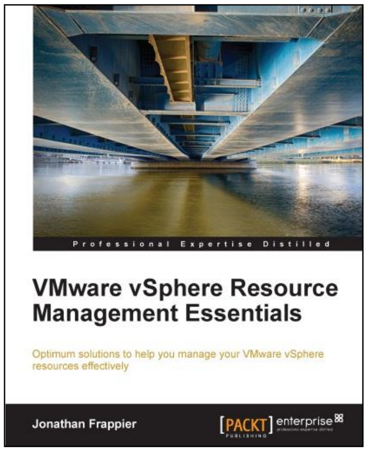
[PACKT] enterprise
PUBLISHING professional expertise distilled

VMware vSphere Resource Management Essentials

Jonathan Frappier


Professional Expertise Distilled
VMware vSphere Resource Management Essentials
Optimum solutions to help you manage your VMware vSphere resources effectively
Jonathan Frappier
[PACKT] enterprise

Chapter No. 2
"Assigning Resources to VMs"

In this package, you will find:

A Biography of the author of the book

A preview chapter from the book, Chapter NO.2 "Assigning Resources to VMs"

A synopsis of the book's content

Information on where to buy this book

About the Author

Jonathan Frappier is a hands-on technology professional with over 15 years of experience in VMware-virtualized environments, focusing on system interoperability. He has specialization at the intersection of system administration, virtualization, security, cloud computing, and social enterprise collaboration.

He had not touched a computer until high school but then quickly found his passion. Jonathan holds a Bachelor's degree in Computer Science from Newbury College and a Master's degree in Computer Information Systems from Boston University, which he completed while working full time. He holds VMware certifications, including VCAP5-DCD, VCP5-DCV, VCA-Cloud, DCV, and WM.

Jonathan has worked in enterprises and start-ups throughout his career and has become a self-defined jack of all trades, but he is most passionate about virtualization and its community, and was honored as a vExpert 2013 for his contributions.

You can find Jonathan on Twitter `@jfrappier`, and on his blog at `www.virtxpert.com`, as well as at almost every Virtualization Technology User Group (VTUG) meet. He also supports the #vBrownBag podcast at `professionalvmware.com`.

First and foremost, I'd like to thank my wife for putting up with me for all these years and my daughter who will have to put up with me for many more. I am thankful to my parents for supporting me even when I had no idea why I was going to college. I would also not be here if it were not for my grandparents, who I was fortunate to know well in my adult life, and in loving memory of my grandfather, Norman L'Heurex, and Pepere Lionel Frappier. Just as much love is due to my friends Jim, Jeremy, Manny, Bob, and Igor.

I'd also like to thank the following people from the virtualization community: Matt, Sean, Luigi, and others—the "Nerdherd". I'm not sure I'd be doing this if it weren't for that dinner at Ichigo Ichie. Also, a big thank you to the #vBrownBag crew for giving me the opportunity to give back through this incredible organization; to all the folks out there blogging, sharing, and helping others; and the folks at VMware who support the community. Your contribution inspires me.

VMware vSphere Resource Management Essentials

VMware vSphere Resource Management Essentials provides readers with a high-level understanding of the various components, methodologies, and best practices for maintaining and managing resources in a virtual environment.

Readers will begin the book by going through an explanation of the requirements for ESXi and the foundation for VMware vSphere. Also, this book will provide readers with an understanding of how resources are supplied and the features that enable resource and virtual machine availability.

With an understanding of the requirements to build and run your environment, you will then move into understanding how ESXi manages resources such as CPU, memory, disk, and networks for multiple virtual machines and ensures there is resource availability.

With VMs built and resources assigned, readers will get to know the advanced features as well as the monitoring and automation tools included to make your VMware vSphere environment more efficient.

What This Book Covers

Chapter 1, *Understanding vSphere System Requirements*, provides specific resource requirements for installing ESXi and vCenter, as well as providing links to online resources such as the VMware HCL.

Chapter 2, *Assigning Resources to VMs*, covers how virtual machines use physical resources provided by ESXi hosts, and provides various techniques that ESXi uses to manage the allocation of physical resources.

Chapter 3, *Advanced Resource Management Features*, provides an overview of the various tools and features licensed with VMware vSphere to increase resource utilization and availability.

Chapter 4, *Automation and Monitoring Options*, takes a look at the two main automation tools available with VMware vSphere, PowerCLI, and vCenter Orchestrator; it also covers monitoring solutions built into vCenter and vCenter Operations Manager.

2
Assigning Resources to VMs

Now that you understand the various resource requirements for installing ESXi and vCenter, we will look at how VMware vSphere manages resources assigned to an individual VM and how multiple VMs on a single physical host can affect resource availability. Throughout this chapter, we will compare and contrast common practices used while deploying operating systems to bare metal systems and then in deploying operating systems as VMs.

The topics we'll be covering in this chapter are as follows:

- The basics of overcommitment and virtualization
- CPU scheduling and the effect of multiple vCPU VMs
- Memory assignment and management
- Storage considerations and their effects on performance
- Networking, Virtual Switches, VM to VM, and VM to physical system communication

The basics of overcommitment and virtualization

As hardware evolved, processors and memory far outpaced the rate that applications could consume those resources. When multicore processors came out, many applications were still only able to utilize a single core, that left an entire CPU core idle unless some other application or process was multiprocessor capable. While an underutilized resource provides headroom for the applications, it reduces value to the business because the resource does not receive full returns on its hardware investment. When a system or application was idle, the investment made by the organization was further wasted.

Virtualization helps to solve this problem by being able to assign physical resources to virtual servers. VMware made this technology available to all organizations. It simplified deployment and built features that could be leveraged by a small organization with only a handful of servers, or by the world's largest organizations with thousands or even tens of thousands of servers.

Using the two example servers from *Chapter 1*, *Understanding vSphere System Requirements*, let's consider the benefits of virtualizing your servers compared to installing them on dedicated hardware.

We looked at two physical servers, each with a dual core CPU, so two processing cores. The Windows server, even at its peak utilization, never used more than 35 percent of the available processing power of the server, leaving 65 percent of the resources idle. Our Linux server (also with a dual core CPU) was almost entirely idle, and even when we added a workload, we still only averaged out 45 percent CPU utilization. This means your organization purchased two physical servers, while the application only required the resources available in one physical server.

Let's take this example even a step further. While the Windows server required a peak of 35 percent of the CPU, it did not require this for the entire duration in which we collected the resource utilization statistics. If we assume for the sake of this example that the spike on the Windows server and on the Linux server happened at separate times, we would have even more resources available if we had other VMs running on the same physical hardware.

The same concept of sharing physical resources carry through the four major resources that we discussed in *Chapter 1*, *Understanding vSphere System Requirements*, that is, CPU, memory, disk, and networking. It's unlikely that a single OS and application are utilizing all of the physical resources available in a modern server. By virtualizing several OS and applications, we can maximize the resource utilization available in a physical server, providing a higher ROI and in many cases, a lower TCO for the organization.

CPU scheduling and the effect of multiple vCPU VMs

Modern CPUs are so powerful that we perceive the ability to multitask or use several programs at once. In reality, CPU cores act in a very serial fashion, that is to say, a single core is only ever really executing one task at a time. Because of the speed and logic in these modern processors, they are able to quickly switch between the requests from different applications, so when one request is finished, the CPU switches to the next task and so on.

A technology such as **Hyper-Threading** (**HT**) by Intel or AMD-V/RVI will make a single core appear as two cores to the operating system and can help process multiple requests at once on a single CPU core.

If you have HT enabled, you will see that your OS, if supported with a hypervisor such as ESXi, perceives this to be an available physical processor core; however, for purposes of designing your infrastructure, I would recommend staying within the physical capabilities of your processor. With an Intel Xeon CPU running HT on four cores, the ESXi hypervisor will be presented with eight available cores. Although HT increases number of the available virtual cores, you should design your clusters for the load based on the number of physical cores.

With this basic understanding of how CPUs function, you can see why it's important to understand the resource requirements for a specific server or application. Servers or applications with relatively low CPU resource requirements could potentially work fine with 10 or 12 vCPUs per physical core; however, if the application was processor intensive, you might find that you can only run two to three vCPUs per physical core. It's important to note here that generally, you are not assigning a VM or vCPU to a physical CPU; however, you could through the use of CPU affinity in ESXi. We are simply saying that depending on your workload and the number of physical cores in your CPU, you could expect to receive certain consolidation ratios.

When possible, you should try to perform the following actions:

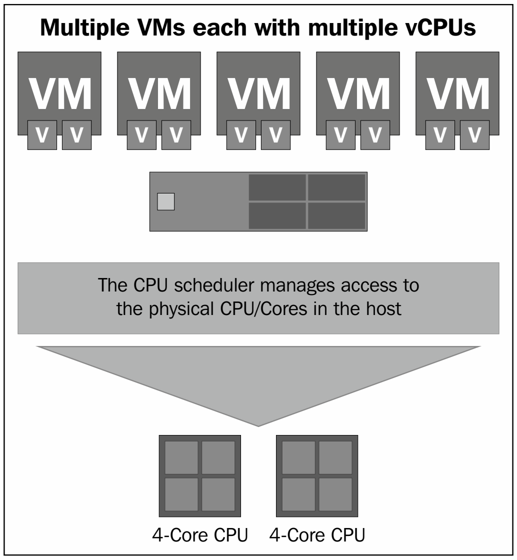
- Monitor resource utilization prior to any project—physical to virtual or virtual to virtual
- Do not overcommit the CPU; only assign what the VM and workload require
- Avoid CPU affinity as you will not be able to vMotion VMs with CPU affinity enabled until it is disabled

VMware has developed an advanced CPU scheduler to ensure the efficient operation of VMs in your environment. In fact, they have published an excellent technical white paper that you should read, as we don't have the space in this book to cover the CPU scheduler in that level of detail. You can find the white paper at `http://www.vmware.com/files/pdf/techpaper/VMware-vSphere-CPU-Sched-Perf.pdf`.

If your VM has one vCPU, and the host it is running on has four pCPU, then it's the job of the CPU scheduler to find an available pCPU for the VM to access when required. That is an easy translation to make to a physical server with an OS installed. However, what happens when your VM has multiple vCPUs?

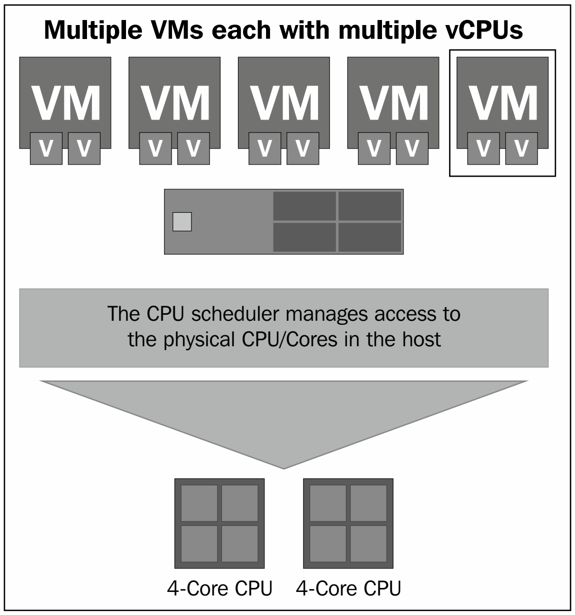
Using vCenter as an example, as we discussed in *Chapter 1*, *Understanding vSphere System Requirements*, vCenter recommends at least two vCPUs. However, at any given point, it may not require the processing power of both vCPUs. If vCenter was installed on its own physical server, the OS would expect to have access to all of the CPUs or cores on that server and would just operate as required. However, in a VM where you are sharing a pCPU between multiple VMs, the pCPU might already be busy processing a request from another VM. Since the OS expects all of the CPUs assigned to it to be available, it must have to wait for two pCPUs to be available since it is assigned two vCPUs. Let's look at another example visually.

In the following figure, we have an ESXi host with two quad core pCPU for a total of eight pCPU cores available to VMs with five VMs, each with two vCPUs assigned:


Multiple VMs each with multiple vCPUs
VM
V V
VM
V V
VM
V V
VM
V V
VM
V V
The CPU scheduler manages access to the physical CPU/Cores in the host
4-Core CPU
4-Core CPU

When all eight cores are available and the first VM makes a request, the CPU scheduler can provide the vCPUs with access to two pCPU cores, so the VM's operating system can operate as expected. The CPU scheduler can support simultaneous requests for only three more VMs running on this host. When the fifth VM makes a request, the CPU scheduler does not have access to any pCPU cores. Because the VM is unaware that it is virtual, it assumes it has access to the number of processors assigned.

In this scenario, which is shown in the following figure, the VM will have to wait until all of the assigned vCPUs have access to pCPUs to service the request. Even if only one vCPU is required, the OS has to wait until all vCPUs can be serviced.


Multiple VMs each with multiple vCPUs
VM
V V
The CPU scheduler manages access to the physical CPU/Cores in the host
4-Core CPU
4-Core CPU

This is where the concept of rightsizing comes into play. Prior to virtualization, the OS had access to all pCPUs in a server, whether or not it needed this access. So, a web server with dual quad core processors could generally be idle a majority of the time depending, of course, on its workload. If we used the same configuration in a virtual machine, that is, assigning eight vCPUs, the VM would have to wait until the CPU scheduler could provide access to eight pCPUs at once, whether or not it needed all eight pCPUs.

Going back again to the example at the end of the previous chapter, where we had a physical server with two cores utilizing only 35 percent of the CPU, you would be better off assigning only one vCPU to the VM for the same workload. This would help since the scheduler would only ever need to access one pCPU that could provide all the necessary computing power that the workload required.

To build off yet another topic that we covered in *Chapter 1*, *Understanding vSphere System Requirements*, it's important to understand the resource requirements for environment and specific applications. Once virtual, you want to make sure you provide the required resources to a VM and do not carry over the practice from the physical world of providing extra resources that are not required. There are quite a few products in the market today to help you identify VMs that are over-provisioned, some are even free of charge.

One item that you should be aware of once virtualized is a command called ESXTOP. ESXTOP is a command-line utility to monitor resource utilization for your ESXi host and the VMs on that host. If you are having performance problems with a VM and you want to see whether it's due to the pCPU resources that have not been made available to the VM, you can check **%RDY**.

%RDY displays the time span that your VM is ready to run for, but the scheduler was not able to provide pCPU resources. This should be under five percent in normal operation. A number higher than five percent could mean you have overcommitted your pCPU or assigned too many vCPUs to a VM.

If your VM has multiple vCPUs assigned, you should also monitor **%CSTP**, which is only relevant for multiple vCPU VMs. A VM with a %CSTP of 3 percent or higher suggests you should lower the number of vCPUs assigned to the VM. In the case of a VM that has four vCPUs assigned but operates at 30-40 percent CPU utilization, you might actually improve performance by removing two vCPUs and leaving the VM with two vCPUs. Here, the VM operates at 40 percent capacity, so removing 50 percent of the CPUs should leave enough compute power for the VM to operate.

Other useful statistics include (host or VM-based statistics) the following:

- **CPU load average**: This gives the average utilization of all pCPUs in the host. A reading of 1.0 in the host means you are fully utilized, 0.5 means 50 percent utilized, 0.02 means two percent utilized, and so on.
- **PCPU USED (%)**: This is the percent of all CPUs used by the host.
- **%RUN**: This is the amount of vCPUs utilized by the VM. This setting takes into account all vCPUs that are assigned. So, a value of 100 percent with one vCPU means that the vCPU is fully utilized. A value of 150 percent for a VM with two vCPUs means that it is roughly 75 percent utilized if you are looking within the guest OS (VM).
- **%WAIT**: This refers to how long the VM is waiting for other processes managed by ESXi to complete, such as I/O (VM).

To dive deeper into ESXTOP how-tos, check out VMware KB 2001003 at `http://kb.vmware.com/kb/2001003` and the VMware communities post at `https://communities.vmware.com/docs/DOC-9279`.

Memory assignment and management

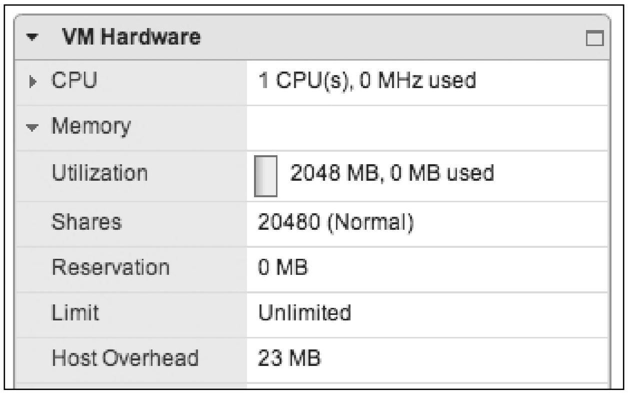
While physical memory is allocated to VMs through the hypervisor just like CPU, the calculations and methods to determine memory requirements are much simpler. Memory like vCPUs, are assigned to a VM, and the OS on the VM assumes it has that amount of memory available to it; however, in many scenarios, you are likely to overcommit the memory to achieve a greater consolidation ratio or increase the VM density per host. The benefits of overcommitment for memory are again along the same lines as overcommitting CPUs; your VMs are not likely to use the entire amount of memory assigned to them at all times. As with all resources in a virtual environment, you want to rightsize your memory configuration as well. While you could easily give all VMs 16 GB of memory, the amount of memory assigned to a VM has other impacts on your infrastructure and the vSphere cluster, which we will see shortly. VMware has several memory management techniques to reduce the likelihood of contention or in the event that you run into a period of contention techniques to reclaim memory from VMs.

A current trend, thanks to decreasing memory costs and the manufacturer's ability to increase the amount of memory available within a system, is actually to not overcommit on memory. Take for example, a modern yet mid-range physical server such as a Dell R520 or Cisco UCS B22M blade that can contain up to 384 GB of memory. If you were to configure the VMs on these servers each with 4 GB of memory, you would have enough memory capacity for around 92 to 94 VMs, leaving enough memory for ESXi itself. Depending on the CPU you selected, anywhere from four to eight cores per processor, you would potentially be pushing a 12:1 vCPU to pCPU overcommitment ratio if each VM had a single vCPU. In this scenario, you might actually find CPU cores to be your limiting resource instead of memory. In this example, however, we have not factored in the memory required to run ESXi or the memory overhead for the VM. What is memory overhead you ask? Well, we were just getting to that.

Memory overhead

Memory overhead is additional memory that each VM requires in order for the ESXi to manage it. It is generally a very small amount of memory based on the amount of memory assigned to the VM and the number of vCPUs. Using the vCenter requirements from *Chapter 1*, *Understanding vSphere System Requirements*, a VM with two vCPUs and 16 GB of memory would require about 144 MB of memory overhead. Another example, a VM with eight vCPUs and 16 GB of memory should only require around 169 MB of memory overhead.

You can see the memory overhead for any particular VM on the summary page by expanding **Memory**, as seen in the following screenshot from the vSphere 5.5 web client:



VM Hardware

CPU	1 CPU(s), 0 MHz used
Memory	
Utilization	2048 MB, 0 MB used
Shares	20480 (Normal)
Reservation	0 MB
Limit	Unlimited
Host Overhead	23 MB

You can also find a sample memory overhead charge in the *VMware Resource Management* guide at `http://pubs.vmware.com/vsphere-55/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-55-resource-management-guide.pdf`. The following chart from VMware is meant to serve as a guide to help you determine memory overhead based on the resources assigned to the VM. It is useful while designing your infrastructure, but there are other advanced settings that also factor into memory reservations, such as the VMX swap file.

Memory (MB)	1 vCPU	2 vCPUs	4 vCPUs	8 vCPUs
256	20.29	24.28	32.23	48.16
1024	25.90	29.91	37.86	53.82
4096	48.64	52.72	60.67	76.78
16384	139.62	143.98	151.93	168.60

Transparent page sharing and memory compression

Two memory management techniques that ESXi implements to improve memory utilization is **transparent page sharing** (**TPS**) and memory compression. As you might expect from its name, ESXi compresses memory pages during times of contention to prevent swapping. This is done because the process to compress and thus decompress the memory when required is still faster than swapping to a disk. Memory compression is enabled by default, and unless you have been instructed to do so by support, it's a setting best left enabled.

Transparent page sharing is where identical memory pages are shared among multiple virtual machines. There is some debate as to the usefulness of TPS as more guess, operating systems are leveraging large memory pages, typically 2 MB in size compared to 4 KB.

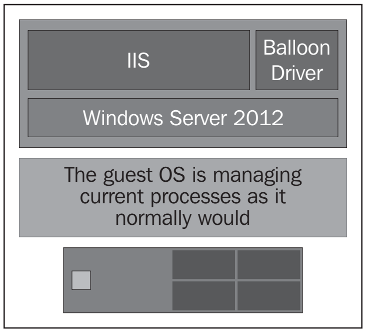
With 4 KB page sizes, it is theoretically easier to find a match among multiple memory pages, which ESXi can then basically deduplicate. If ESXi were to find 1000 4 KB pages, it could reduce that to a single memory page, thus making more memory available to VMs and other processes. With large pages, ESXi still scans the page but rather than trying to match an entire 2 MB page size, which is less likely, it scans 4 KB pages within the 2 MB page. If ESXi detects contention, it can then attempt to share, compress, or swap these smaller pages. For the purposes of this book, you should know that TPS attempts to deduplicate and share identical memory pages. For a deeper dive into TPS, VMware has published a white paper devoted entirely to TPS, which can be found at `http://download3.vmware.com/software/vmw-tools/papers/WP-2013-01E-FINAL.pdf`. You should also read *VMware vSphere 5.1 Clustering Deepdive*, *Duncan Epping* and *Frank Denneman*. This is hands down one of the best available technical books on VMware vSphere. I'll be suggesting this book again in the next chapter when we discuss HA and DRS.

> Unless required by your application, leaving large memory pages disabled allows TPS to more efficiently deduplicate memory.

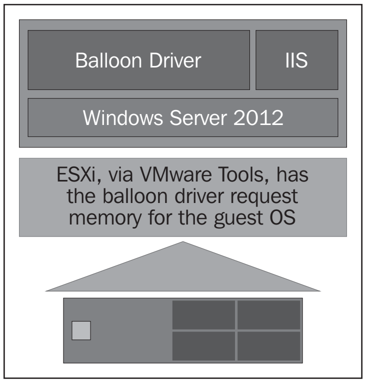
Ballooning

There is a driver installed on your guest OS when **VMware Tools** is installed. This allows ESXi to request memory from VMs that might otherwise be idle. ESXi asks the balloon driver, a process that is running in the guest OS, to request memory from the OS as any other application might. This allows the guest OS, which is aware of its current workload, to allocate memory without negatively impacting the performance of active processes.

Take, for example, a Windows server running IIS as shown in the following figure; an increase in demand may cause IIS and its related processes to require a specific amount of memory.


IIS
Balloon Driver
Windows Server 2012
The guest OS is managing current processes as it normally would

The memory that IIS previously requested may not be returned to the OS immediately even though it is no longer required. If ESXi is experiencing memory contention, ESXi makes a request to the balloon driver to ask the guest OS for additional memory. The guest OS can then determine what, if any, memory can be reclaimed from other processes. The OS can reclaim the memory not needed from other applications and provide it to the balloon driver, which then notifies ESXi of how much memory can be reclaimed from the guest. By default, ESXi is configured to reclaim up to 65 percent of the assigned memory, and this can be alerted with advanced settings at both the ESXi hosts, thus affecting all VMs or individual VMs. Make sure VMware Tools is installed on all VMs to allow ballooning to function.


Balloon Driver
IIS
Windows Server 2012
ESXi, via VMware Tools, has the balloon driver request memory for the guest OS

The VSWP swap file

The VSWP swap file is a file equal to the amount of memory assigned to the VM minus the amount of memory reserved (we will discuss reservations in *Chapter 3*, *Advanced Resource Management Features*, but it's very literal). The swap file is one of the methods used to reclaim memory during times of contention. When this happens, ESXi swaps the running memory to disk, much like what you would see with the Windows swap file or Linux swap partition. It should be noted that OS swapping is separate from the VSWP swap file, and your OS will still utilize its native swapping techniques if the guest OS is low on memory. ESXi will utilize the VM swap file if it is low on memory, even if the VM itself is not low on memory.

The VSWP swap file is stored by default in the same location as your virtual hard drive. When you think about this, as it relates to rightsizing, you can see why assigning too much memory for no reason can have an impact on your environment. If you had 100 VMs running on your host, each with 16 GB of memory and no memory reservations, you would need an additional 1.6 TB of storage space to store all of your VSWP swap files. If those VMs really needed 4 GB of memory, then you would only need an additional 400 GB of storage, which is not a trivial amount but much less than the 1.6 TB.

> Like CPU, it is important to rightsize the VM memory configuration.

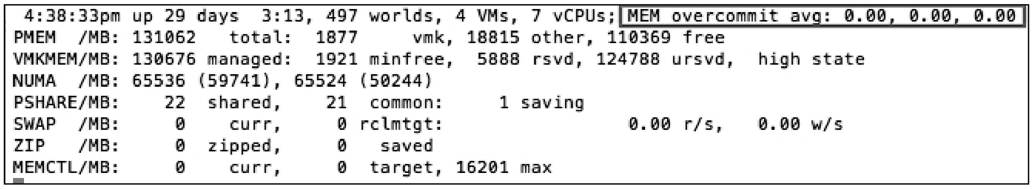
Do not forget to plan for memory overhead as well as additional storage requirements for the VSWP swap file.

While swapping is generally considered a bad thing, after all, it means we are running low on memory in our ESXi host; the memory management techniques used in advanced can typically eliminate the need for swapping if planned properly. If possible, place the VSWP swap file on a separate storage, preferably flash, for the best possible performance during memory contention.

Monitoring memory usage

As we discussed in case of the CPU, there are several useful `ESXTOP` statistics for monitoring memory. When you run `ESXTOP`, you are given the CPU view, which is the default view. To switch to memory statistics, just hit *M* on your keyboard and the view will change. Refer to the VMware communities list again and the KB article that we mentioned earlier if you need a refresher (`https://communities.vmware.com/docs/DOC-9279` and `http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=2001003`). The first item I tend to look at on a host is **MEM overcommit avg**. It shows how much memory you have overcommitted on your host.

A host that is not overcommitted (which is when the host has more memory than the VMs running on the host are assigned plus memory overhead) will have a return value of 0 as shown in the following screenshot:



```
 4:38:33pm up 29 days  3:13, 497 worlds, 4 VMs, 7 vCPUs; MEM overcommit avg: 0.00, 0.00, 0.00
PMEM  /MB: 131062   total:  1877     vmk, 18815 other, 110369 free
VMKMEM/MB: 130676 managed:  1921 minfree,  5888 rsvd, 124788 ursvd,  high state
NUMA  /MB: 65536 (59741), 65524 (50244)
PSHARE/MB:    22  shared,    21  common:     1 saving
SWAP  /MB:     0    curr,     0 rclmtgt:                 0.00 r/s,   0.00 w/s
ZIP   /MB:     0  zipped,     0   saved
MEMCTL/MB:     0    curr,     0  target, 16201 max
```

You have probably noticed there are three values of **0.00**, **0.00**, and **0.00** returned in the preceding screenshot. These are the averages over a 1, 5, and 15 minute window. The other statistics that you should make note of are as follows:

- **PSHARE**: This is a count of the number of shared pages and memory saving due to TPS.
- **SWAP**: This is the total amount of memory being swapped to the disk. This will most likely cause noticeable performance problems and should relate to a high **MEM overcommit avg**. Consider adding more memory or reducing the number of VMs on the host if this is consistently high.
- **MEMCTL**: This shows how much memory has been reclaimed by the balloon driver. A consistently high value here could lead to future disk swapping; however, this is a valid memory overcommitment management feature. You should also consider monitoring your individual VM OS for swapping if this value is high, to ensure VM performance is optimal.

There are also statistics to monitor at the VM level. For example, if **MEMCTL** is high, you may want to add the **MCTLSZ** column to see which VMs are returning memory via the balloon driver. To add this column that is not visible by default, perform the following actions:

1. Hit the *F* key on your keyboard.
2. Hit the *J* key on your keyboard to select **MCTL**.
3. Hit *Esc* and you will be returned to the `ESXTOP` statistics view with the new columns.

Now you can see how much memory has been reclaimed (if any) or could be reclaimed by the balloon driver by examining the **MCTLSZ** and **MCTLMAX** columns.

Storage considerations and their effects on performance

Storage is the one area that you cannot skip while evaluating your environment and workload, and no amount of CPU or RAM will make up for a misconfigured or underconfigured storage system. Storage in many environments is the biggest resource bottleneck due to its cost, general misunderstanding about how it works, and what is important. Outside of IT, storage is typically thought of in terms of capacity which, while a valid point, is only a small piece of the storage equation.

From a business perspective, cost is the driving factor on hard drives. Our job as administrators, engineers, and architects is to understand the workload so that we can select the storage appliances, drives, and connectivity that will support the current environment as well as future growth. You could spend an entire career learning about and mastering storage, but we have only few pages in this book. For a deeper dive into storage, I would suggest picking up *Storage Implementation in vSphere*, *Mostafa Khalil* and *Troubleshooting vSphere Storage*, *Mike Preston*.

What is IOPS?

For the purposes of our discussion, let's start at the storage appliance. In most vSphere environments, storage is provided by hardware and networks separate from the physical server known as the storage fabric. Two of the main metrics that we will look at is **I/O per Second** (**IOPS**) and throughput (the amount of time it takes for data to be transmitted). Let's start with IOPS, which is mostly determined by the hard drive, which also makes it one of the most important factors when it comes to performance. Typically, you will see four types of hard drives that are available (from the slowest and largest to the fastest and smallest); for example, SATA (or NL-SAS), SAS, FC, and SSD.

While SATA drives can generally provide enough storage capacity, they cannot generally provide enough IOPS, and SSDs can provide enough IOPS but generally not enough capacity. The following is a chart of generally accepted average hard drive IOPS:

Drive Type	**IOPS**	**Capacity**
SATA 7200 RPM	Up to 100 IOPS	Up to 4 TB
SATA 10,000 RPM	Up to 150 IOPS	Up to 1 TB
SAS 10,000 RPM	Up to 140 IOPS	Up to 1 TB
SAS 15,000 RPM	Up to 210 IOPS	Up to 600 GB
SSD	Up to 400 to 120,000 IOPS	Up to 1.2 TB

RAID

Unlike in a laptop or desktop where you usually have a single hard drive, storage appliances are composed of multiple drives grouped together called arrays and more specifically into a **redundant array of independent disks** (**RAID**), though you may also see **Just of Bunch of Disks** (**JBOD**), which does not provide any redundancy.

> The IOPS of your array is the number of drives of that array multiplied by the IOPS of the drive minus any RAID write penalties.

You should check with your storage appliance manufacturer to determine the recommended number of drives to place in a single array, which may vary based on the type of array you created.

Once the RAID or JBOD is created, you can then create a **LUN**. Depending on your storage appliance, you could create a single LUN on that RAID, or you could create multiple LUNS. Whenever possible, I prefer to create a single LUN on a single RAID or JBOD that fulfills the workload requirements being run by the VMs. By doing this, you can easily identify the workload and the associated VMs running that workload for a particular LUN. If you were to create multiple LUNS on a single RAID or JBOD, you may run into contention when a workload on one LUN is utilizing all of the I/O resources available from that RAID, which may not be immediately obvious while observing perceived performance problems on the other LUN.

RAID has a major impact on your available IOPS and adds another layer of complexity to account for workload patterns. Depending on your application, you could have read a heavy workload, that is, the one that is reading more data than it is writing, such as a reporting application, or a write heavy workload, that is, the one that performs more writes, such as applications that import large amounts of data.

You may even have applications where your workload changes depending on business cycles. One such example is a financial system which may have a write-heavy workload at the end of a month when the information is being imported and which then changes to a read-heavy workload at the beginning of the month when the reports are being run.

While we don't have enough space to dive deep into RAID types, the following chart should give you what you need to begin your research. Depending on your storage appliance, you may be able to assign what is called a hot spare drive, which will be used to automatically replace a failed drive in a RAID, though generally it's only useful where there is already a redundancy.

RAID Type	Redundancy	Read or Write Favorable
RAID 0	No	Both
RAID 1	Yes; 1 disk	Both, limited by the disk type
RAID 5	Yes; 1 disk	Read
RAID 6	Yes; 2 disks	Read
RAID 0+1	Yes; depends on RAID 0	Both

RAID 5 is one of the more common RAID types deployed because of its ability to maximize storage capacity and provide failure of a single disk without losing data. In nonvirtualized environments, RAID 5 could typically provide enough IOPS for even a write-heavy workload since it was the only workload utilizing those drives. The downfall of RAID 5 (and RAID 6) is that in order to achieve the drive redundancy, extra data is written that creates what is known as a write penalty since a single write has to be performed on two drives: the drive that the write was originally intended for and a parity write on a separate disk, which provides the redundancy. In a virtual environment where the LUN may be servicing multiple workload, the write penalty is magnified even further.

Selecting the drive type as well as the RAID type go hand in hand. Most modern storage appliances employ a data tiering system where you can perform writes on faster drives such as SAS or SSD, and the data is then moved to the most cost-effective SATA drives after a certain period of inactivity. Additionally, an increasing number of vendors are offering all-flash arrays that can provide extremely high levels of IOPS but with lower storage capacity; depending on your workload, this may be sufficient.

VMware vSphere Storage APIs – Array Integration (VAAI)

Most storage appliance vendors will partner VMware to offer what is called VMware vSphere Storage APIs – Array Integration or VAAI. VAAI allows VMware vSphere to offload certain tasks that would otherwise consume host resources such as cloning a VM to the storage appliance. These tasks, called primitives, can be managed more efficiently by the storage appliance by itself. When you are researching storage options for your vSphere environment, you should definitely consider appliances that support VAAI. You can find more information about VAAI at `http://www.vmware.com/files/pdf/techpaper/VMware-vSphere-Storage-API-Array-Integration.pdf`.

The following are some points to consider while determining your storage requirements:

- Spend extra time monitoring I/O and throughput requirements
- Select drives and RAID options that will fit your IOPS requirements today as well as allow for future expansion
- Understand your VM and application workload profile, that is, read or write-heavy and possibly both at different times
- Use storage appliances and vendors that support VAAI

Connectivity and throughput

Once you have your drive type and RAID type designed to support your workload, you then need to consider how you will connect your storage appliance(s) to your physical servers. There are three options for connecting your storage appliance: Ethernet, which is the same as your typical networking equipment, that supports iSCSI, NFS and **Fiber Channel over Ethernet** (**FCoE**); **Infiniband**; and traditional **Fiber Channel** (**FC**). Infiniband currently provides the fastest connectivity, which will improve throughput but it is also the most costly. Having said that, nothing costs more than outages and negative business impact. If your workload calls for Infiniband, you should advocate it. FC has been generally provided the best balance between cost and performance, but as 10 GbE (10 Gigabit Ethernet) becomes more cost effective, you might see a shift towards FCoE, iSCSI, and NFS.

Since you are typically connecting your storage appliance to multiple physical servers, a switch is used just as you would use a switch for networking purposes. Most designs implement a completely separate storage switch; however, many modern switches support a modular architecture where you can combine Ethernet for both networking and storage as well as FC or Infiniband into the same switch. What you choose for your environment should largely depend on your workload requirements; if you are looking to cut budget, you should probably look at areas other than your storage fabric.

Throughput, which will be affected by the type of connectivity you use, is the measure of how much data can be written. While IOPS measures the number of read/write operations per second, throughput measures how many KBs, MBs, or GBs can be written in a certain amount of time, typically measured in MB per second. There are many factors within your storage appliance which will also affect throughput, so check with your storage vendor and ask them to help you determine what your workload profile is, and test that profile against their storage array.

VMFS

With most connectivity selections, you will format your storage in the Virtual Machine File System format (VMFS); however, if you have gone down the NFS route, then NFS will be your filesystem format. There are a few items to be aware of with NFS. First, you will thin provision your virtual hard drives (VMDK); thin provisioning consumes only the amount of space actually required by the virtual hard disk. If you assign a 100 GB virtual hard drive to a VM but it only uses 20 GB, you will only consume 20 GB of actual storage. We will touch on the pros and cons of thin provisioning shortly. Also, depending on your NFS network and uplink configuration, you may not be taking advantage of multiple uplinks for better performance. This is because the default load balancing method uses both the source and destination IP addresses, even with multiple bonded uplinks; these will always be the same if the NFS targets are all presented on the same IP address from your storage appliance. There are a few options to achieve better load balancing, such as creating multiple NFS servers with unique IP addresses. The VMware *Best Practices for Running VMware vSphere on Network-Attached Storage* document has more information on NFS considerations at (`http://www.vmware.com/files/pdf/techpaper/VMW-WP-vSPHR-NAS-USLET-101-WEB.pdf`).

While the items mentioned previously may seem like limitations, they are simply considerations that you will also have to plan for and manage with block-based protocols such as FC, FCoE, and iSCSI. With these protocols, you still need to design your storage fabric to provide the necessary throughput and redundancy. The debate between block and NFS, which for many years favored block, has come down to requirements and supportability. What your infrastructure requires and how well can you support the infrastructure should ultimately dictate which storage protocol you select.

VM disk provisioning

While NFS makes the decision on thin provisioning for you, this is still an option with VMFS, and it's important to understand the differences. As mentioned in the previous section, thin provisioning only allocates the amount of space actually used within the VMDK. This allows you to overcommit your storage appliance, like we have discussed with the CPU and memory. This can be useful where growth is expected over a long period of time or when the VM is not easily taken offline for maintenance. While the overhead of expanding a thin provisioned disk has been tested to be negligible, applications with high write workload may still be impacted. In these scenarios, you may wish to thick provision your VMDK.

Thick provisioning, as you may have guessed, allocates the full amount of assigned storage immediately. If you create a VMDK with 100 GB of space, that 100 GB of space is allocated on your storage appliance immediately. There are two options available for thick provisioned VMDKs: Eager Zeroed and Lazy Zeroed. Lazy Zeroed allocates the entire amount of space immediately to the VMDK but does not wipe or zero the blocks until the first write request. With Eager Zeroed disks, all blocks are wiped when the disk is created. This causes the disk creation time to take longer but provides the best possible performance for write-intensive applications. For Windows administrators, you can draw a parallel to quick formatting of an NTFS partition to thick Lazy Zeroed versus a full format to an Eager Zeroed drive.

Monitoring storage

Troubleshooting storage can be quite difficult; however, like CPU and memory, there are some key ESXTOP statistics to look at if you suspect a storage bottleneck. To access disk statistics, hit *v* on your keyboard while in ESXTOP. Unlike CPU and memory, this will be strictly VM-focused. The following counters are focused specifically on VMs:

- **READS/s**: This is the number of reads per second
- **WRITES/s**: This is the number of writes per second

The sum of these statistics should match the expected I/O from your datastore and LUN/array. For example, if you had seven SATA drives in a RAID 5, and your READS/s and WRITES/s were equal, that is a 50/50 split, then you could expect to get somewhere around 224 IOPS from that datastore/LUN/array (if you kept to a single datastore per array, otherwise you need to view all VMs on all datastores/LUNs created on that array). If the combination of reads and writes exceeds 224, then you have more I/O than the LUN can handle.

As we mentioned earlier, I/O is not the only factor in performance, you also need to consider throughput. To view storage adapter-related statistics, press the *d* key on your keyboard while in ESXTOP. The statistics to monitor include the following:

- **GAVG**: This is a measure of the round trip latency seen by the VM.
- **KAVG**: This is the latency related to the ESXi kernel.
- **DAVG**: This is the latency seen at the device driver level. It includes the roundtrip time between the HBA and the storage.

These statistics would likely need to be related to similar statistics from your storage array to determine whether the bottleneck is host-related, a specific array, a storage appliance, or a storage network.

Networking

Networking in a vSphere environment is not all that different from physical networking. VMs are assigned virtual network cards which are attached to switches. A virtual switch has an uplink to a physical switch.

Uplinks

An uplink is a physical NIC on the ESXi hosts that connects the server to the network. In a typical ESXi deployment, you typically have multiple physical NICs to support various types of network traffic. It is also considered best practice to have redundancy for your various uplinks with multiple NICs supporting a single traffic type. Depending on your requirements, this may be a single active NIC with a standby NIC, which becomes active in the event of a failure, or you may bond multiple interfaces together and create an LACP group on your physical switch to support additional throughput. In extreme cases, I have even heard of architects using NICs from different manufacturers so that a driver failure or crash would not affect all NICs on a host.

Having multiple physical NICs/uplinks is only part of the equation. You also need to determine where these uplinks will connect to. If you have used multiple physical nicks for redundancy, but only one uplink to a single physical switch, then you still have a single point of failure in the network stack.

Wherever possible, consider the following requirements:

- A minimum of two physical NIC ports, preferably on two physical NICs for each type of traffic uplink (for example, one on board the NIC port and one on the PCI card port)
- Each pair of NIC ports should connect to separate physical switches or at least separate line cards in a modular switch
- If bonding multiple NIC ports together for throughput, verify the recommended configuration from your switch vendor

At a minimum, there are two types of traffic from an ESXi host: VM kernel and VM network traffic. VM network traffic is just that—the network traffic from your VMs to the physical network. VM kernel or management traffic is the interface used to manage the ESXi host, connect it to vCenter, and for advanced features such as vMotion or Fault Tolerance; we will cover vMotion and Fault Tolerance in the next section.

As we have said before, the network design ultimately depends on your specific requirements. You could certainly run your management and VM network traffic over the same uplink, or you could just easily have separate physical uplinks for management, vMotion, FT, and VM network and storage; beyond that, you may even have multiple vSwitches for spreading out the same type of traffic.

What is a vSwitch?

A vSwitch is quite similar to a physical switch. A vSwitch has ports that VMs connect to as well as various settings that you might also configure on a physical switch, such as VLANs, or port groups on a vSwitch, and MTU settings if you need to configure jumbo frames. There are two types of vSwitches: a Standard Switch (vSS) and Distributed Switch (vDS)—think of a vSS like a typical 1U 24 or 48 port switch that has some management features. A vDS is more like an advanced enterprise grade switch that can support multiple line cards with more advanced features. Having said that, there is no reason why a vSS cannot be used in large environments. As we have said many times, it comes down to your requirements and whether the features you would like to use are available on a vSS or on a vDS.

Like physical switches, vSwitches support VLAN tagging to allow you to separate different types of network traffic. On a vSwitch, you would create a port group to support different VLANs. If you are using a vSS, you need to create vSwitches and port groups on each host. With a vDS, all port groups are managed centrally via vCenter and pushed to each ESXi host. In the case of a vDS, each ESXi host acts in a similar fashion to line cards in a modular switch. While vCenter is critical for managing a vDS, it will continue to operate if vCenter is unavailable, but you will not be able to make any changes until vCenter is brought back online.

Monitoring network connectivity with ESXTOP

Like CPU, memory, and disk, ESXTOP can monitor network connectivity of your physical NICS (uplinks) and virtual NICs associated with VMs. You can see typical data transmission statistics; in the case of ESXTOP, this is tracked in MegaBits sent and received per second in the MbTX/s (transmitted/sent) and MbRX/s (received). These statistics can be useful to understand the existing traffic and whether a physical NIC can continue to support additional VMs or whether a particular VM is producing excessive amounts of traffic.

Two statistics that can immediately show whether a physical NIC is overcommitted or even connected to a faulty physical switch port or cable is %DRPTX and %DRPRX. On a physical NIC, you should expect these values to be at or near 0.00. On a VM, %DRPRX could also indicate a VM does not have sufficient CPU, memory, or disk I/O resources; though more typically, CPU.

Summary

In this chapter, we looked how assigning resources to VMs affect all of the resources on the host as well as other VMs. In addition, we looked at the features implemented by VMware vSphere to manage overcommitment such as the CPU scheduler, transparent page sharing, and ballooning. We then discussed storage and the effect drive type and RAID selection has on performance; reviewed important ESXTOP statistics for each of them and wrapped up an overview of networking, typical practices with uplinks and vSwitches, and finally, tips on how to monitor network connectivity. Next, let's look at the advanced settings that help further manage and provide availability to VMs and their resources.

Where to buy this book

You can buy VMware vSphere Resource Management Essentials from the Packt Publishing website: `http://www.packtpub.com/vmware-vsphere-resource-management-essentials/book`.

Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

[PACKT] PUBLISHING enterprise
professional expertise distilled

www.PacktPub.com

For More Information:
www.packtpub.com/vmware-vsphere-resource-management-essentials/book