The Top 20 VMware Performance Metrics You Should Care About

Why you can't ignore them and how they can help you find and avoid problems.

WHITEPAPER BY ALEX ROSEMBLAT

Table of Contents

Introduction

Virtualization introduces new challenges for managing the data center: Applications now compete for shared, dynamic infrastructure resources such as storage, CPU cycles, and memory. This creates a complex resource allocation problem that if not solved, only worsens as more virtual machines are added, causing performance bottlenecks, poor ROI and unhappy customers. This whitepaper provides descriptions of 20 metrics collected from VMWare's vCenter that indicate when resource-related bottlenecks and capacity issues are occurring. Specifically in the descriptions, the whitepaper will highlight what causes performance issues, how these metrics help to spot problems, what kinds of further analysis should be done to validate that a certain problem is causing VM issues, and how to fix a problem. Lastly, recommendations are given on solutions that will analyze the data from these 20 metrics to find VM performance issues and assist a data center in resolving them.

The 20 Metrics that Reveal VM Performance Issues

Capacity bottlenecks can occur in the disk, disk I/O, network, memory or CPU resources related to VM usage. When a VM or host is suffering from a bottleneck, the application running on the VM will react sluggishly and may be unable to complete instructions. Analysis of the metrics presented below will assist in pinpointing the cause of a VM performance problem and finding solutions for these issues. However, it should be noted that a thorough analysis to reveal problems can be quite intense. Sufficiently monitoring these metrics involves frequent collection of the data from each VM at least 10 times per hour, followed by analysis of this data for every VM, host, cluster, and resource pool. The amount of data alone is massive. For a simple 100 VM environment, nearly 17 million pieces of data get collected over a 30 day period that need to be analyzed to spot trends and predict future issues. Methods to perform this analysis will be discussed later in the paper. Let's start with a review of they key metrics.

CPU Metrics

CPU oriented metrics are the most commonly reviewed. They are also the most commonly misunderstood in virtual environments.

cpu.extra.summation

The cpu.extra.summation metric is measured at the VM level and indicates when a mismatch between I/O traffic and CPU processing speed is occurring. This mismatch causes VM processing to slow from a time lag known as I/O Wait, and is perceived as a performance problem. Analyzed by itself, this metric is not very informative and VMWare has stated that cpu.extra.summation is an unreliable measure as it can be skewed. In fact, cpu.extra.summation has been removed in vSphere 4.0 and any users of that software will see blank "n/a" cells when attempting to obtain a measurement for this metric.

However, cpu.extra.summation can still be helpful in finding performance issues. Cpu.extra.summation is an indicator of possible CPU Ready issues and can provide insights when used as a relative measure in finding problem areas among multiple virtual objects. High relative cpu.extra.summation values in a particular virtual infrastructure area are caused when a VM does not have enough processing power allocated, or is experiencing CPU Ready issues. VMs suffering CPU bottlenecks can be repaired by allocating more CPU resources or resolving CPU Ready issues with that VM.

cpu.ready.summation

Note: A value in this particular metric indicates that a bottleneck exists in the virtual environment.
The cpu.ready.summation metric is measured at the VM level on a real-time basis and assesses whether a VM is having CPU Ready issues. CPU Ready issues result from CPU overutilization and occur when VMs are contending with one another for use of limited physical cores and a wait time ensues. The wait time is caused by one VM waiting for a CPU transaction to finish with the other VM before it can carry on its transaction. The time lag that ensues from CPU Ready causes VM processing slowness which is perceived as a performance problem.

Per VMware best practices, a CPU Ready bottleneck is occurring when more than 5% of the time involved in a CPU transaction by a VM is in wait time for the physical CPU resource to be ready. Although VMware attempts to schedule a VM to use a physical core in such a way that loads will be balanced, this scheduling can be stretched thin if too many virtual cores are provisioned per physical cores. In order to avoid spreading CPU resources too thinly, many organizations will attempt to keep a ratio of 3 virtual CPUs to 1 physical CPU as a maximum resource allocation density.

CPU Ready can be a difficult problem to detect, and is a cause for many performance problems. The resolution for CPU ready issues is to rebalance VM loads to spread out physical CPU usage or to “right-size” a VM’s CPU resource allocations. Importantly, CPU ready issues can be avoided by monitoring CPU utilization with cpu.usagemhz.average and cpu.extra.summation.

cpu.usagemhz.average

The cpu.usagemhz.average measures physical CPU use and is measured at the VM level. A high measure in the cpu.usagemhz.average metric denotes that the CPU usage for a physical CPU resource is approaching or has hit full utilization. It is possible that VMs with a high measure in this metrics may be experiencing lag times in performance as the overutilization of CPU can lead to CPU Ready issues as commands wait for processing cycles to become available.

Although a high cpu.usagemhz.average metric alone itself does not necessarily signify that a bottleneck is occurring, it can be a warning indicator that should be closely followed. This metric can help a system administrator proactively avoid performance problems. Also, this metric can be useful in isolating the cause of bottleneck issues at the CPU resource.

Some VMs will occasionally use 100% of the resources allocated based on their normal processing needs which may lead to issues with other VMs that need to share those resources. VMs with high cpu.usagemhz.average values should be investigated on a longitudinal basis to assess how CPU usage evolves throughout VM active times. The only way to resolve a CPU related bottlenecks is to allocate more CPU resources to VMs experiencing issue resulting from CPU overutilization.

Disk Metrics

disk.busResets.summation

Note: A value in this particular metric indicates that a bottleneck exists in the virtual environment.
A disk bus reset is when all commands that have been queued up in an HBA or Disk Bus have been wiped out. The disk.busResets.summation metric measures when a disk bus reset has occurred. This metric is measured at the VM level in real time.

If the disk.busResets.summation metric has a value for a VM this can be indicative of severe disk-related issues. It is possible that:

- A disk has become overloaded from too much traffic from:
  - Too many VMs accessing that disk
  - Too many commands originating from the VMs accessing that disk
- There are other throughput-related issues
- There has been a hardware failure

VMs with a disk.busResets.summation value will suffer from sluggishness and may hang, or crash. System administrators can attempt to troubleshoot this issue by viewing VM traffic to the disk to see if there is a throughput issue from any of the VMs accessing the disk. Most likely, the assistance of a storage administrator will be necessary to identify the root cause. Resolving an issue identified with disk.busResets.summation may require a hardware fix or more typically, a rebalancing of VM traffic by moving VMs to other datastores with more available capacity.

disk.commandsAborted.summation

Note: A value in this particular metric indicates that a bottleneck exists in the virtual environment.
The disk.commandsAborted.summation is a metric that shows the number of times a request was sent to a disk and the command was aborted. This metric is measured at the VM level in real time. The values for this metric for every VM should be zero. If the value is anything but zero, it is indicative of a severe disk-related issue and should be investigated immediately. VMs with a value in disk.commandsAborted.summation will suffer from sluggishness and may hang, or crash. The causes for this metric being anything but zero are that:

- A disk has become overloaded from too much traffic from:
  - Too many VMs accessing that disk
  - Too many commands originating from the VMs accessing that disk
- There are other throughput-related issues
- There has been a hardware failure

A system administrator can attempt to troubleshoot this issue by viewing VM traffic to that disk to see if there is a throughput issue. Most likely, the assistance of a storage administrator will be necessary to identify the issue causing the disk.commandsAborted.summation to have a value. Resolving an issue with a disk.commandsAborted.summation may require a hardware fix or typically, a rebalancing of VM traffic by moving VMs to other datastores with more capacity available.

disk.totalLatency.average

Note: A value in this particular metric indicates that a bottleneck exists in the virtual environment.
The disk.totalLatency.average metric shows the amount of disk latency that is occurring on a disk. This metric is measured at the host level in real time. Disk latency is the amount of time it takes for a response to be generated after the delivery of a message to the disk. This metric is helpful in assessing whether an environment is experiencing a performance problem since if disk latency is high, there is an issue somewhere. However, because many issues can cause disk latency, the disk.totalLatency.average metric is not precise and further investigation is necessary to pinpoint what the exact problem causing the latency is. These problems can occur as a result of memory or disk throughput issues. VMs suffering disk latency issues will react sluggishly. Disk.totalLatency.average can be resolved by load balancing, finding out what top resource-consuming VMs are and right-sizing their resources and/or moving those VMs to other datastores.

disk.queueLatency.average

Note: A value in this particular metric indicates that a bottleneck exists in the virtual environment.
The disk.queueLatency.average metric denotes the amount of time a command waits in a queue to be processed by the disk. This metric is measured at the host level in real time. As the time a command is waiting in queue increases, VM performance decreases. A high disk.queueLatency.average metric is usually found in tandem with a high disk.totalLatency.average metric, as commands are likely waiting in a queue because of the increased time for a command to be processed by the disk. Thus, if disk.queueLatency.average reveals an issue, the disk.totalLatency.average metric should be checked as well. If disk latency is indeed an issue, as mentioned in the disk.totalLatency.average section, other performance bottlenecks are causing this issue, and a full investigation into memory and throughput metrics should also be enacted.

Similar to issues with disk latency, disk.queueLatency.average can be resolved by load balancing, finding out what top resource-consuming VMs are and right-sizing their resources and/or moving those VMs to other datastores, or lastly if hardware issues are present, by replacing damaged hardware.

Throughput: An average of disk.read.average and disk.write.average

The disk.read.average metric refers to the amount of traffic coming from the disk in read commands and the disk.write.average metric refers to the amount of traffic going to the disk in write commands. These metrics are measured at the VM level in real time. Averaged together, the disk.read.average and disk.write.average become the disk throughput measure. Disk throughput refers to the average traffic capacity that a VM has in its connection to the disk. Seeing the relative measures of this metric across all VMs help identify which VMs are generating the greatest amount of traffic to the disk.

Throughput usage can be graphed over time to see if a VM has had performance problems at different activity levels and if this increase may be causing performance problems for other VMs by taking up bandwidth to the disk. VMs suffering throughput issues, or on hosts where throughput is constricted will react sluggishly. Importantly, high throughput levels will point to disk latency issues, as well as may be the cause of other disk issues such as Commands Aborted or Bus Resets. Throughput issues can be resolved by rebalancing loads and/or moving VMs to other areas where they will have enough capacity or will not take up the capacity that other VMs require.

Memory Metrics

mem.active.average

The mem.active.average metric is collected at the VM level and measures the amount of memory pages that are being actively used by a VM at a given time. A VM typically does not actively use all of its allocation at a given time. Much of the allocation holds other data that has been recently accessed but is not being actively worked on. Because a VM has more memory allocated than it actually uses at a given time, the mem.active.average metric is not representative of how much memory a VM has consumed. In fact, the mem.consumed.average metric is much more accurate to gauge a VM's total memory footprint. However, mem.active.average should still be evaluated to get a better idea of how much memory is being actively used by a VM at any given time. Evaluating this metric together with mem.consumed.average will help to assess whether a VM has had adequate amount of memory allocated. If issues are found related to the mem.active.average metric, the only way to resolve them is by adding or allocating more memory to the VM, or moving a VM to a host with more memory.

mem.consumed.average

The mem.consumed.average metric is collected at the VM level and measures the amount of memory that is being consumed in total by a VM. A VM uses more than just the memory that is being actively used as memory pages by the VM at the moment. A portion of consumed memory holds memory that has been used recently, but is not being actively accessed. The active memory and this "held memory" added together equal the total memory footprint for a VM, which is what mem.consumed.average measures.

Assessing mem.consumed.average values is useful when examining if memory shortages are affecting the performance of a VM and will show some memory constraints. It is important to assess mem.active.average at the same time to see if a VM is truly suffering from memory shortages. Closely examining this metric and the changes in resource utilization as a VM functions over a peak time period will also yield insights into how much memory a VM needs allocated. Importantly, some VMs will consume all resources assigned to them and if a VM is showing high mem.consumed.average values, the VM should be assessed to see if it is one of these "memory hog" applications. For those kinds of VMs, using the mem.consumed.average metric as a threshold for impending memory shortages will not work. The only ways to resolve a memory shortage which mem.consumed.average and mem.active.average will show is by adding or allocating more memory to the VM, or moving a VM to a host with more memory.

mem.overhead.average

The mem.overhead.average metric measures the amount of memory that is being used to manage allocated memory and is collected at the VM level per host. Because of the way that VMs, and computers in general use their memory, some memory must be used to manage itself. This process is the way that a computer keeps track of what its own resources are doing. Important to note with this behavior is that the additional overhead adds a requirement for more memory to be used by a VM than simply what has been allocated by a system administrator. The larger the amount of memory that has been allocated for a VM, the more memory that is also needed in overhead. The table below produced by VMware gives an accurate reading for how much memory overhead is needed per VM based on memory and CPU count for a VM. As memory is typically the resource with the most constraints, keeping an eye on the mem.overhead.average metric is essential to avoid what can lead to non-obvious memory shortages.

The mem.overhead.average should be monitored when deciding memory allocations for all VMs in a host. As the memory measured by the mem.overhead.average metric will be taken from the host's total memory store, a host will actually have less memory available than the difference between the host's specifications and the sum of all memory allocated to VMs on that host. To resolve a memory shortage which could affect all VMs on the host, VMs can be right-sized if they have had more memory allocated to them than is needed. Also, more memory can be added or VMs can be moved to hosts with more memory available.

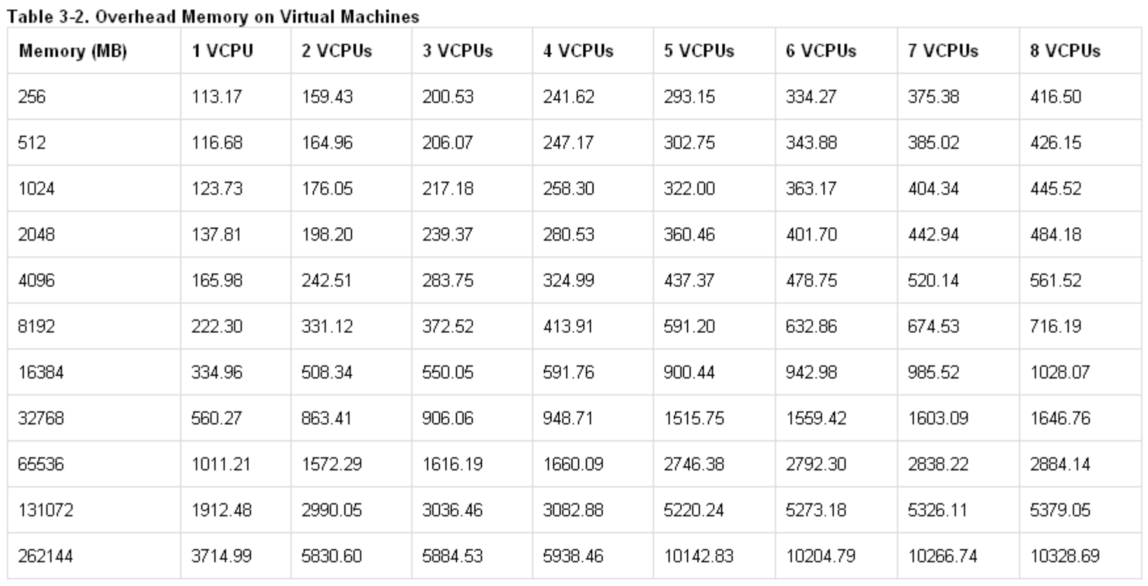

Table 3-2. Overhead Memory on Virtual Machines

Memory (MB)	1 VCPU	2 VCPUs	3 VCPUs	4 VCPUs	5 VCPUs	6 VCPUs	7 VCPUs	8 VCPUs
256	113.17	159.43	200.53	241.62	293.15	334.27	375.38	416.50
512	116.68	164.96	206.07	247.17	302.75	343.88	385.02	426.15
1024	123.73	176.05	217.18	258.30	322.00	363.17	404.34	445.52
2048	137.81	198.20	239.37	280.53	360.46	401.70	442.94	484.18
4096	165.98	242.51	283.75	324.99	437.37	478.75	520.14	561.52
8192	222.30	331.12	372.52	413.91	591.20	632.86	674.53	716.19
16384	334.96	508.34	550.05	591.76	900.44	942.98	985.52	1028.07
32768	560.27	863.41	906.06	948.71	1515.75	1559.42	1603.09	1646.76
65536	1011.21	1572.29	1616.19	1660.09	2746.38	2792.30	2838.22	2884.14
131072	1912.48	2990.05	3036.46	3082.88	5220.24	5273.18	5326.11	5379.05
262144	3714.99	5830.60	5884.53	5938.46	10142.83	10204.79	10266.74	10328.69

Source: From VMWare.com (http://pubs.vmware.com/vsp40_i/wwhelp/wwhimpl/common/html/wwhelp.htm#href=resmgmt/r_overhead_memory_on_virtual_machines.html#1_7_9_9_10_1&single=tru)

Memory Swapping: mem.swapin.average, mem.swapout.average and mem.swapped.average

Note: Values in these particular metrics indicate that a bottleneck exists in the virtual environment

The mem.swapin.average, mem.swapout.average and mem.swapped.average metrics are assessed together to gauge if any memory swapping bottlenecks are occurring. These metrics are measured at the VM level and are averaged together to make a percentage value. Memory swapping is an action that computers undertake to manage their memory. If a computer's memory becomes full, and there is no more capacity to process more information, the computer will take part of the contents of its memory and "swap" it out to storage on a disk so that it can take new information into the now cleaned up memory. If anything from the swapped out memory load is needed, the VM will request the old memory load from storage and swap it in to be worked on. This action is extremely time-consuming, and can result in processing times that can grow to a 3 to 5 order of magnitude as the information must move through the network to storage, and must then be processed by the disk. Likewise, when a computer swaps in data back to memory, there is an additional lag time which adds to the total processing transaction. As a result of these massive lag times, if VMs are swapping memory, performance will begin to slow down dramatically. Memory swapping may also begin to cause other bottlenecks, as the swapping of large chunks of data can clog up throughput. Increased throughput will tax the disk which can increase disk latency and also cause other disk issues.

VM swaps are an indication that there aren't enough memory resources allocated to a VM, or that ballooning is occurring which is downsizing the memory to VMs on a host. To resolve a memory swapping bottleneck, a VM must be given more memory and further analysis must be conducted to find if the swapping is being caused by ballooning.

mem.vmmemctl.average(balloon)

Note: A value in this metric indicates that a bottleneck may be about to occur in the virtual environment.

A value in the mem.vmmemctl.average metric is an indicator that ballooning is occurring. Ballooning occurs when VMs use more memory resources and begin to come close to a limit, either physical or set through a resource limit in VMWare. With this spike in activity in a VM, all other VMs sharing resources are evaluated, and if there is non-active consumed memory in other VMs, VMWare appropriates that resource and gives it to the VM whose memory is spiking. Ballooning is significant in that it is directly connected to memory swapping which is a major cause of performance problems. VMWare engages in ballooning as a way to avoid memory shortages, but is essentially affecting "innocent bystander" VMs. Problems arise when a VM that has had its memory resources taken away begins to increase its use and the now minimized VM begins swapping memory to the disk to make space for the information which it must work on. This action leads to previously covered memory swapping issues. As mentioned before, memory swapping can then lead to throughput and disk issues. These resulting bottlenecks will vastly slow down VM performance. Values in the mem.vmmemctl.average metric should be investigated immediately. Also, it is possible with ballooning that VMWare limits that a system administrator may not be aware of have been set for a VM.

The way to resolve bottlenecks caused by ballooning is to correctly right-size memory for all VMs, add more memory if necessary, redeploy VMs to rebalance shared resource use, and importantly, to check if limits have been set in VMWare that are not appropriate for the VMs that are being managed.

Network Metrics

net.received.average, net.transmitted.average and net.usage.average

The net.received.average, net.transmitted.average and net.usage.average metrics measure network traffic and usage related to VM commands. These metrics are measured at the VM level and are similar to the information captured in disk throughput, but disk throughput directly relates to traffic going from the data store to the server. This metric can give insights if ISCSI or NFS storage is used, as the communication between the data store hardware here is captured in network traffic.

These metrics report on areas that are usually so minute in terms of bottlenecks, that significant values can only be seen in a noticeable quantity at the cluster or data center level. Even at these levels however, net.received.average, net.transmitted.average and net.usage.average represent minuscule amounts of traffic compared to other metrics. Network metrics are very rarely if ever the cause of performance bottlenecks.

Finding and Resolving Issues with Analysis of these Metrics

Finding Issues with VMs

To find issues within the data provided by these metrics, a data center requires the ability to pull out the raw metric data and process it with advanced analytical abilities in order to compare values, find outliers, and see if issues may arise at specific times. Additionally, some metrics need to be analyzed at the VM, host, cluster, and resource pool level, and even more importantly, a problem in a metric for one VM may be caused by actions occurring from another VM. These calculations can be complicated to set up and require large amounts of processing capacity to complete. Importantly, as resources are shared, the analysis must assess how VMs are interacting with one another in addition to the analysis on each individual VM, host, cluster, and resource pool.

Resolving Issues found through Analysis of the 20 Metrics

Resolving the issues that the 20 metrics reveal through analysis of data extracted from vCenter usually involves making configuration settings to an environment, right-sizing resource allocations of a VM, rebalancing VMs across equipment to better distribute loads, or examining hardware if a bottleneck is possibly being caused by a hardware issue. After a bottleneck is spotted, additional analysis of the data is required to find what the constraining resource is, or if a pattern is uncovered that hints at a certain issue causing the problem.

Conclusion

Sharing resources in a virtualized data center adds complexity to ensuring that every server, now a VM, has the resources it requires to function well. The 20 metrics described above should be actively monitored to manage an environment and ensure that all VMs remain high performing. Because of the sheer amount of information that is generated in real-time, the data from these 20 metrics must be analyzed in as timely a fashion as possible to not just spot bottlenecks, but also to prevent them from happening when early warning indicators are identified.