Interpreting esxtop Statistics

Table of Contents

Section 1. Introduction

Esxtop allows monitoring and collection of data for all system resources: CPU, memory, disk and network. When used interactively, this data can be viewed on different types of screens; one each for CPU statistics, memory statistics, network statistics and disk adapter statistics. In addition to the disk adapter statistics in earlier versions, starting with ESX3.5, disk statistics at the device and VM level are also available. Starting with ESX 4.0, esxtop has an interrupt statistics screen. In the batch mode, data can be redirected to a file for offline uses.

Many esxtop statistics are computed as rates, e.g. CPU statistics %USED. A rate is computed based on the refresh interval, the time between successive snapshots. For example, *%USED = (CPU used time at snapshot 2 - CPU*

used time at snapshot 1) / time elapsed between snapshots. The default refresh interval can be changed by the command line option " *-d*", or the interactive command *'s'*. The return key can be pressed to force a refresh.

In each screen, data is presented at different levels of aggregation. It is possible to drill down to expanded views of this data. Each screen provides different expansion options.

It is possible to select all or some fields for which data collection is done. In the case of interactive use of esxtop, the order in which the selected fields are displayed can be selected.

In the following sections, this document will describe the esxtop statistics shown by each screen and their usage.

Section 2. CPU

Section 2.1 Worlds and Groups

Esxtop uses worlds and groups as the entities to show CPU usage. A **world** is an ESX Server VMkernel schedulable entity, similar to a process or thread in other operating systems. A **group** contains multiple worlds.

Let's use a VM as an example. A powered-on VM has a corresponding group, which contains multiple worlds. In ESX 4.0, there is one vcpu (hypervisor) world corresponding to each VCPU of the VM. The guest activities are represented mostly by the vcpu worlds. (In ESX 3.5, esxtop shows a vmm world and a vcpu world for each VCPU. The guest activities are represented mostly by the vmm worlds.) Besides the vcpu worlds, there are other assisting worlds, such as a MKS world and a VMX world. The MKS world assists mouse/keyboard/screen virtualization. The VMX world assists the vcpu worlds (the hypervisor). The usage of the VMX world is out of the scope of this document. In ESX 4.0, there is only one vmx world. (In ESX 3.5, there are two vmx worlds for each VM.)

There are other groups besides VM groups. Let's go through a few examples:

- The "idle" group is the container for the idle worlds, each of which corresponds to one PCPU.
- The "system" group contains the VMKernel system worlds.
- The "helper" group contains the helper worlds that assist VMKernel operations.
- In classic ESX, the "console" group is for the console OS, which runs ESX management processes. In ESXi, these ESX management processes are running as user worlds directly on VMKernel. So, on an ESXi box you can see much more groups than on a classic ESX, but not the "console" group.

Note that groups can be organized in a hierarchical manner in ESX. However, esxtop shows, in a flat form, the groups that contain some worlds. More detailed discussion on the groups are out of the scope.

Q: Why can't we find any vmm worlds for a VM in ESX 4.0?

A: Before ESX 4.0, each VCPU has two worlds "vmm" and "vcpu". In ESX 4.0, cpu scheduler merges their statistics to one vcpu world. So, CPU stats won't show vmm worlds. This is not a problem.

Section 2.2 Global Statistics

- **"up time"**

The elapsed time since the server has been powered on.

- **"number of worlds"**

The total number of worlds on ESX Server.

- **"CPU load average"**

The arithmetic mean of CPU loads in 1 minute, 5 minutes, and 15 minutes, based on 6-second samples. CPU load accounts the run time and ready time for all the groups on the host.

- **"PCPU(%)"**

The percentage CPU utilization per physical CPU.

Q: What does it mean if PCPU% is high?

A: It means that you are using lots of resource. (a) If all of the PCPUs are near 100%, it is possible that you are overcommiting your cpu resource. You need to check RDY% of the groups in the system to verify cpu overcommitment. Refer to RDY% below. (b) If some PCPUs stay near 100%, but others are not, there might be an imbalance issue. Note that you'd better monitor the system for a few minutes to verify whether the same PCPUs are using ~100% CPU. If so, check VM CPU affinity settings.

- **"used total"**

Sum(PCPU(%)) / number of PCPUs

- **"LCPU(%)"**

The percentage CPU utilization per logical CPU. The CPU used percentages for the logical CPUs belonging to a package add up to 100%. This line is displayed only if hyper-threading is present and enabled.

- **"CCPU(%)"**

Percentages of total CPU time as reported by the ESX Service Console. "us" is for percentage user time, "sy" is for percentage system time, "id" is for percentage idle time and "wa" is for percentage wait time. "cs/sec" is for the context switches per second recorded by the ESX Service Console.

Q: What's the difference of CCPU% and the console group stats?

A: CCPU% is measured by the COS. "console" group CPU stats is measured by VMKernel. The stats are related, but not the same.

Section 2.3 World Statistics

A group statistics is the sum of world statistics for all the worlds contained in that group. So, this section focuses on worlds. You may apply the description to the group as well, unless stated otherwise.

ESX can make use of the Hyperthreading technology, so, the performance counters takes Hyperthreading into consideration as well. But, to simplify this document, we will ignore HT related issues. Please refer to "Resource Management Guide" for more details.

- **"%USED"**

The percentage physical CPU time accounted to the world. If a system service runs on behalf of this world, the time spent by that service (i.e. %SYS) should be charged to this world. If not, the time spent (i.e. %OVRLP) should not be charged against this world. See notes on %SYS and %OVRLP.

%USED = %RUN + %SYS - %OVRLP

Q: Is it possible that %USED of a world is greater than 100%?

A: Yes, if the system service runs on a different PCPU for this world. It may happen when your VM has heavy I/O.

Q: For an SMP VM, why does VCPU 0 have higher CPU usage than others?

A: The system services are accounted to VCPU 0. You may see higher %USED on VCPU 0 than others, although the run time (%RUN) are balanced for all the VCPUs. This is not a problem for CPU scheduling, but only the way VMKernel does the CPU accounting.

Q: What is the maximum %USED for a VM group?

*A: The group stats is the sum of the worlds. So, the maximum %USED = NWLD * 100%. NWLD is the number of worlds in the group.*

*Typically, worlds other than VCPU worlds are waiting for events most of time, not costing too much CPU cycles. Among all the worlds, VCPU worlds represent best the guest. Therefore, %USED for a VM group usually do not exceed Number of VCPUs * 100%.*

Q: What does it mean if %USED of a VM is high?

A: The VM is using lots of CPU resource. You may expand to worlds to see what worlds are using most of them.

- **"%SYS"**

The percentage of time spent by system services on behalf of the world. The possible system services are interrupt handlers, bottom halves, and system worlds.

Q: What does it mean if %SYS is high?

A: It usually means that your VM has heavy I/O.

Q: Are %USED and %SYS similar to user time and system time in Linux?

A: No. They are totally different. For Linux OS, user (system) time for a process is the time spent in user (kernel) mode. For ESX, %USED is for the accounted time and %SYS is for the system service time.

- **"%OVRLP"**

The percentage of time spent by system services on behalf of other worlds. In more detail, let's use an example.

When World 'W1' is running, a system service 'S' interrupts 'W1' and services World 'W2'. The time spent by 'S', annotated as 't', is included in the run time of 'W1'. We use %OVRLP of 'W1' to show this time. This time 't' is accounted to %SYS of 'W2', as well.

Again, let's take a look at "%USED = %RUN + %SYS - %OVRLP". For 'W1', 't' is included in %RUN and %OVRLP, not in %SYS. By subtracting %OVRLP from %RUN, we do not account 't' in %USED of 'W1'. For 'W2', 't' is included in %SYS, not in %RUN or %OVRLP. By adding %SYS, we accounted 't' to %USED of 'W2'.

Q: What does it mean if %OVRLP of a VM is high?

A: It usually means the host has heavy I/O. So, the system services are busy handling I/O. Note that %OVRLP of a VM group may or may not be spent on behalf of this VM. It is the sum of %OVRLP for all the worlds in this group.

- **"%RUN"**

The percentage of total scheduled time for the world to run.

Q: What is the difference between %USED and %RUN?

A: %USED = %RUN + %SYS - %OVRLP. (%USED takes care of the system service time.) Details above.

Q: What does it mean if %RUN of a VM is high?

A: The VM is using lots of CPU resource. It does not necessarily mean the VM is under resource constraint. Check the description of %RDY below, for determining CPU contention.

- **"%RDY"**

The percentage of time the world was ready to run.

A world in a run queue is waiting for CPU scheduler to let it run on a PCPU. %RDY accounts the percentage of this time. So, it is always smaller than 100%.

Q: How do I know CPU resource is under contention?

A: %RDY is a main indicator. But, it is not sufficient by itself.

If a "CPU Limit" is set to a VM's resource settings, the VM will be deliberately held from scheduled to a PCPU when it uses up its allocated CPU resource. This may happen even when there is plenty of free CPU cycles. This time deliberately held by scheduler is shown by "%MLMTD", which will be describe next. Note that %RDY includes %MLMTD. For, for CPU contention, we will use "%RDY - %MLMTD". So, if "%RDY - %MLMTD" is high, e.g., larger than 20%, you may experience CPU contention.

What is the recommended threshold? Well, it depends. As a try, we could start with 20%. If your application speed in the VM is OK, you may tolerate higher threshold. Otherwise, lower.

Q: How do we break down 100% for the world state times?

A: A world can be in different states, either scheduled to run, ready to run but not scheduled, or not ready to run (waiting for some events).

100% = %RUN + %READY + %CSTP + %WAIT.

Check the description of %CSTP and %WAIT below.

Q: What does it mean if %RDY of a VM is high?

A: It means the VM is possibly under resource contention. Check "%MLMTD" as well. If "%MLMTD" is high, you may raise the "CPU limit" setting for the VM. If "%RDY - %MLMTD" is high, the VM is under CPU contention.

- **"%MLMTD"**

The percentage of time the world was ready to run but deliberately wasn't scheduled because that would violate the "CPU limit" settings.

Note that %MLMTD is included in %RDY.

Q: What does it mean if %MLMTD of a VM is high?

A: The VM cannot run because of the "CPU limit" setting. If you want to improve the performance of this VM, you may increase its limit. However, keep in mind that it may reduce the performance of others.

- **"%CSTP"**

The percentage of time the world spent in ready, co-deschedule state. This co-deschedule state is only meaningful for SMP VMs. Roughly speaking, ESX CPU scheduler deliberately puts a VCPU in this state, if this VCPU advances much farther than other VCPUs.

Q: What does it mean if %CSTP is high?

A: It usually means the VM workload does not use VCPUs in a balanced fashion. The VCPU with high %CSTP is used much more often than the others. Do you really need all those VCPUs? Do you pin the guest application to the VCPUs?

- **"%WAIT"**

The percentage of time the world spent in wait state.

This %WAIT is the total wait time. I.e., the world is waiting for some VMKernel resource. This wait time includes I/O wait time, idle time and among other resources. Idle time is presented as %IDLE.

Q: How do I know the VCPU world is waiting for I/O events?

A: %WAIT - %IDLE can give you an estimate on how much CPU time is spent in waiting I/O events. This is an estimate only, because the world may be waiting for resources other than I/O. Note that we should only do this for VMM worlds, not the other kind of worlds. Because VMM worlds represent the guest behavior the best. For disk I/O, another alternative is to read the disk latency stats which we will explain in the disk section.

Q: How do I know the VM group is waiting for I/O events?

A: For a VM, there are other worlds besides the VCPUs, such as a mks world and a VMX world. Most of time, the other worlds are waiting for events. So, you will see ~100% %WAIT for those worlds. If you want to know whether the guest is waiting for I/O events, you'd better expand the group and analyze the VCPU worlds as stated above.

*Since %IDLE makes no sense to the worlds other than VCPUs, we may use the group stats to estimate the guest I/O wait by "%WAIT - %IDLE - 100% * (NWLD - NVCPU)". Here, NWLD is the number of worlds in the group; NVCPU is the number of VCPUs. This is a very rough estimate, due to two reasons. (1) The world may be waiting for resources other than I/O. (2) We assume the other assisting worlds are not active, which may not be true.*

Again, for disk I/O, another alternative is to read the disk latency stats which we will explain in the disk section.

Q: Why do I always see a high %WAIT for VMX/mks worlds?

A: This is normal. That means there are not too much activities on them.

Q: Why do I see a high %WAIT for a VM group?

A: For a VM, there are other worlds besides the VCPUs, such as a mks world and VMX worlds. These worlds are waiting for events most of time.

- **"%IDLE"**

The percentage of time the VCPU world is in idle loop. Note that %IDLE is included in %WAIT. Also note that %IDLE only makes sense to VCPU world. The other worlds do not have idle loops, so, %IDLE is zero for them.

- **"%SWPWT"**

The percentage of time the world is waiting for the ESX VMKernel swapping memory. The %SWPWT (swap wait) time is included in the %WAIT time. This is a new statistics added in ESX 4.0.

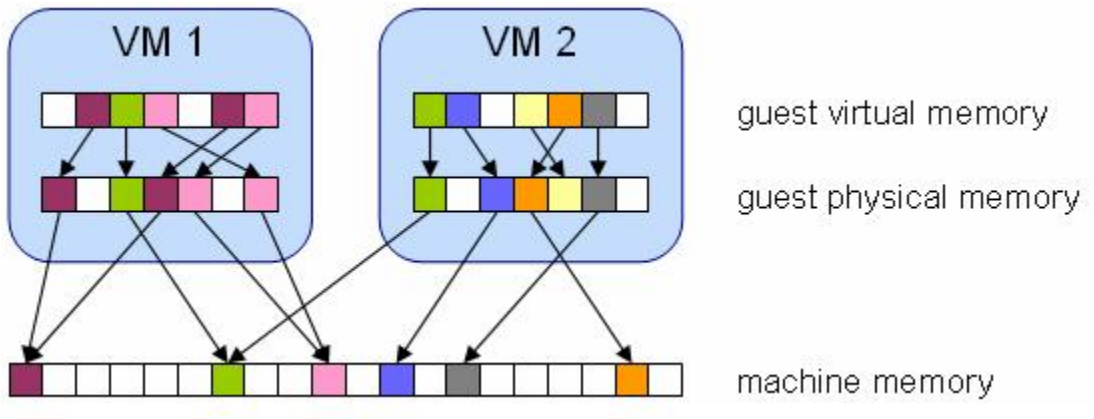
Q: Why do I see a high %SWPWT for a VM group?

A: The VM is swapping memory.

Section 3. Memory

Section 3.1 Machine Memory and Guest Physical Memory

It is important to note that some statistics refer to guest physical memory while others refer to machine memory. " **Guest physical memory**" is the virtual-hardware physical memory presented to the VM. " **Machine memory**" is actual physical RAM in the ESX host. Let's use the following figure to explain. In the figure, two VMs are running on an ESX host, where each block represents 4 KB of memory and each color represents a different set of data on a block.



Inside each VM, the guest OS maps the virutal memory to its physical memory. ESX Kernel maps the guest physical memory to machine memory. Due to ESX Page Sharing technology, guest physical pages with the same content can be mapped to the same machine page.

Section 3.2 Global Statistics

- " **MEM overcommit avg**"

Average memory overcommit level in 1-min, 5-min, 15-min (EWMA).

Memory overcommit is the ratio of total requested memory and the "managed memory" minus 1. VMKernel computes the total requested memory as a sum of the following components: (a) VM configured memory (or memory limit setting if set), (b) the user world memory, (c) the reserved overhead memory. (Overhead memory will be discussed in more detail for "OVHD" and "OVHDMAX" in Section 3.3.)

"managed memory" will be defined in "VMKMEM" section.

Q: What does it mean if overcommit is not 0?

A: It means that total requested guest physical memory is more than the machine memory available. This is fine, because ballooning and page sharing allows memory overcommit.

This metric does not necessarily mean that you will have performance issues. Use "SWAP" and "MEMCTL" to find whether you are experiencing memory problems.

Q: What's the meaning of overcommit?

A: See above description for details. Roughly speaking, it reflects the ratio of requested memory and the available memory.

- **"PMEM" (MB)**

The machine memory statistics for the host.

"total": the total amount of machine memory in the server. It is the machine memory reported by BIOS.

"cos" : the amount of machine memory allocated to the ESX Service Console.

"vmk" : the amount of machine memory being used by the ESX VMKernel. "vmk" includes kernel code section, kernel data and heap, and other VMKernel management memory.

"other": the amount of machine memory being used by everything other than the ESX Service Console and ESX VMKernel. "other" contains not only the memory used by VM but also the user worlds that run directly on VMKernel.

"free" : the amount of machine memory that is free.

Q: Why is total not the same as RAM size plugged in my memory slots?

A: This is because some memory range is not available for use. It is fine, if the difference is small. If the difference is big, there might be some hardware issue. Check your BIOS.

Q: Why can't I find the cos part?

A: COS is only available in classic ESX. You are using ESXi.

Q: How do I break down the total memory?

A: total = cos + vmk + other + free

Q: Which one contains the memory used by VMs?

A: "other" contains the machine memory that backs guest physical memory of VMs. Note that "other" also includes the overhead memory.

Q: How do I know my "free" memory is low? Is it a problem if it is low?

A: You could use the "state" field, which will be explained next, to see whether the free memory is low. Basically, it is fine if you do not experience memory swapping or ballooning. Check "SWAP" and "MEMCTL" to find whether you are experiencing memory problems.

- **"VMKMEM" (MB)**

The machine memory statistics for VMKernel.

"managed": the total amount of machine memory managed by VMKernel. VMKernel "managed" memory can be dynamically allocated for VM, VMKernel, and User Worlds.

"minfree": the minimum amount of machine memory that VMKernel would like to keep free. This is because VMKernel needs to keep some amount of free memory for critical uses.

"rsvd" : the amount of machine memory that is currently reserved. "rsvd" is the sum of three parts: (a) the reservation setting of the groups; (b) the overhead reservation of the groups; (c) "minfree".

"ursvd" : the amount of machine memory that is currently unreserved. It is the memory available for reservation.

Please note that the VM admission control is done at resource pool level. So, this statistics is not used directly by admission control. "ursvd" can be used

as a system level indicator.

"state" : the free memory state. Possible values are high, soft, hard and low. The memory "state" is "high", if the free memory is greater than or equal to 6% of "total" - "cos". If is "soft" at 4%, "hard" at 2%, and "low" at 1%. So, high implies that the machine memory is not under any pressure and low implies that the machine memory is under pressure.

While the host's memory state is not used to determine whether memory should be reclaimed from VMs (that decision is made at the resource pool level), it can affect what mechanisms are used to reclaim memory if necessary. In the high and soft states, ballooning is favored over swapping. In the hard and low states, swapping is favored over ballooning.

Please note that "minfree" is part of "free" memory; while "rsvd" and "ursvd" memory may or may not be part of "free" memory. "reservation" is different from memory allocation.

Q: Why is "managed" memory less than the sum of "vmk", "other" and "free" in the PMEM line? Is it normal?

+A: It is normal, just the way we do accounting. A more precise definition for "managed" is the free memory after VMKernel initialization. So, this amount of memory can be dynamically allocated for use of VMs, VMKernel, and user worlds. "managed" = "some part of vmk" + "other" + "free".+

+So, "managed" < "vmk" + "other" + "free". Or, in an equivalent form, "managed" < "total" - "cos".+

Q: How do I break down the managed memory in terms of reservation?

A: "managed" = "rsvd" + "ursvd" + "vmkernel usage"

VMKernel machine memory manager needs to use some part of memory, which should not be subject to reservation, so, it is not in "rsvd", nor in "ursvd". In the above equation, we put this part under "vmkernel usage". Unfortunately, it is not shown directly in esxtop.

Note that the vmkernel usage in managed memory is part of "vmk".

Q: What does it mean if "ursvd" is low?

A: VMKernel admission control prohibits a VM PowerOn operation, if it cannot meet the memory reservation of that VM. The memory reservation includes the reservation setting, a.k.a. "min", and the monitor overhead memory reservation. Note that even if "min" is not set, VMKernel still needs to reserve some amount

of memory for monitor uses.

So, it is possible that even though you have enough free memory, a new VM cannot power on due to the violation of memory reservation.

Q: Why do I fail admission control even though "ursvd" is high?

A: The VM admission control is done at resource pool level. Please check the "min" setting of all its parent resource pools.

Q: Why is "managed" greater than the sum of "rsvd" and "ursvd"? Is it normal?

A: It is normal. See above question. VMKernel may use some of the managed memory. It is not accounted in "rsvd" and "ursvd".

Q: What is the meaning of "state"?

A: See the description of "state" above.

Q: How do I know my ESX box is under memory pressure?

A: It is usually safe to say the ESX box is under memory pressure, if "state" is "hard" or "low". But, you need also check "SWAP" and "MEMCTL" to find whether you are experiencing memory problems. Basically, if there is not enough free memory and ESX are experiencing swapping or ballooning, ESX box is under memory pressure.

Note that ballooning does not have as big performance hit as swapping does. Ballooning may cause guest swapping. ESX swapping means host swapping.

Also note that A VM may be swapping or ballooning, even though there is enough free memory. This is due to the reservation setting.

- **"COSMEM" (MB)**

The memory statistics reported by the ESX Service Console.

"free" : the amount of idle machine memory.

"swap_t": the total swap configured.

"swap_f": the amount of swap free.

"r/s" : the rate at which memory is swapped in from disk.

"w/s" : the rate at which memory is swapped out to disk.

Note that these stats essentially come from the COS proc nodes.

Q: What does it mean if I see a high r/s or w/s?

A: Your console OS is swapping. It is highly likely that your COS free memory is low. You may either configure more memory for COS and restart your ESX box, or stop some programs running inside your COS.

Q: Why can't I see this COSMEM line?

A: You are using ESXi not classic ESX.

- **"NUMA" (MB)**

The ESX NUMA statistics. For each NUMA node there are two statistics: (1) the "total" amount of machine memory managed by ESX; (2) the amount of machine memory currently "free".

Note that ESX NUMA scheduler optimizes the uses of NUMA feature to improve guest performance. Please refer to "Resource Management Guide" for details.

Q: Why can't I see this NUMA line?

A: You are not using a NUMA machine, or your BIOS disables it.

Q: Why is the sum of NUMA memory not equal to "total" in the PMEM line?

A: The PMEM "total" is the memory reported by BIOS, while the NUMA "total" is the memory managed by VMKernel machine memory manager. There are two major parts of memory seen by BIOS but not given to machine memory manager: (1) COS uses, and (2) VMKernel uses during early initialization.

So, Sum("NUMA total") < "PMEM total" - "cos".

Note that the free memory on all the nodes can be added up as the "free" memory in the PMEM line.

- **"PSHARE" (MB)**

The ESX page-sharing statistics.

"shared": the amount of guest physical memory that is being shared.

"common": the amount of machine memory that is common across World(s).

"saving": the amount of machine memory that is saved due to page-sharing.

The monitor maps guest physical memory to machine memory. VMKernel selects to map guest physical pages with the same content to the same machine page. In other words, those guest physical pages are sharing the same machine page. This kind of sharing can happen within the same VM or among the VMs.

Since each VM's "shared" memory measures guest physical memory, the host's "shared" memory may be larger than the total amount of machine memory if memory is overcommitted. "saving" illustrates the effectiveness of page sharing for saving machine memory.

"shared" = "common" + "saving".

Note that esxtop only shows the pshare stats for VMs, excluding the pshare stats for user worlds.

- **"SWAP" (MB)**

The ESX swap usage statistics.

"curr" : the current swap usage. This is the total swapped machine memory of all the groups. So, it includes VMs and user worlds.

"target": the swap usage expected to be. This is the total swap target of all the groups. So, it includes VMs and user worlds.

"r/s" : the rate at which machine memory is swapped in from disk.

"w/s" : the rate at which machine memory is swapped out to disk.

Note that swap here is host swap, not guest swap inside the VM.

Q: What does it mean if "curr" is not the same as "target"?

A: It means ESX will swap memory to meet the swap target. Note that the actual swapping is done at the group level. So, you should check "SWCUR" and "SWTGT" for each group. We will discuss this in the next section.

Q: Is it bad if "r/s" is high?

A: Yes, it is very bad. This usually means that you have memory resource contention. Because swapin is synchronous, it will hurt guest performance a lot.

Do two things: (1) Check your "free" memory or "state" as mentioned above. If free memory is low, you need to move VMs to other hosts or add more memory to the host. (2) If free memory is not low, check your resource setting of your VMs or user worlds. You may have set a low "limit", which causes swapping.

Q: Is it bad if "w/s" is high?

A: Yes, it is also very bad. This usually means that you have memory resource contention. Do the similar actions as mentioned above.

- **"MEMCTL" (MB)**

The memory balloon statistics.

"curr" : the total amount of physical memory reclaimed by balloon driver. This is the total ballooned memory by the VMs.

"target": total amount of ballooned memory expected to be. This is the total ballooned targets of the VMs.

"max" : the maximum amount of physical memory reclaimable.

Note that ballooning may or may not lead to guest swapping, which is decided by the guest OS.

Q: What does it mean if "curr" is not the same as "target"?

A: It means ESX will balloon memory to meet the balloon target. Note that the actual ballooning is done for the VM group. So, you should check "MCTLSZ" and "MCTLTGT" for each group. We will discuss this in the next section.

Q: How do I know the host is ballooning memory?

A: If the "curr" is changing, you can know it is ballooning. Since ballooning is done at VM level, a better way is to monitor "MCTLSZ" for each group. We will discuss this in the next section.

Q: Is it bad if we have lots of ballooning activities?

A: Usually it is fine. Ballooning tends to take unused memory from one VM and make them available for others. The possible side effects are (a) reducing the memory cache used by guest OS, (b) guest swapping. In either cases, it may hurt guest performance. Please note that (a) and (b) may or may not happen, depending on your workload inside VM.

On the other hand, under memory contention, ballooning is much better than swapping in terms of performance.

Section 3.3 Group Statistics

Esxtop shows the groups that use memory managed by VMKernel memory scheduler. These groups can be used for VMs or purely for user worlds running directly on VMKernel. You may see many pure user world groups on ESXi, not on classic ESX.

Tip: use 'V' command to show only the VM groups.

- **"MEMSZ" (MB)**

For a VM, it is the amount of configured guest physical memory.

For a user world, it includes not only the virtual memory that is backed by the machine memory, but also the reserved backing store size.

Q: How do I break down "MEMSZ" of a VM?

A: A VM's guest physical memory could be mapped to machine memory, reclaimed by balloon driver, or swapped to disk, or never touched. The guest physical memory can be "never touched", because (1) the VM has never used it since power on; or, (2) it was reclaimed by balloon driver before, but has not been used since the balloon driver releases it last time. This part of memory is not measured directly by VMKernel.

"MEMSZ" = "GRANT" + "MCTLSZ" + "SWCUR" + "never touched"

Please refer to "GRANT", "MCTLSZ", "SWCUR".

- **"GRANT" (MB)**

For a VM, it is the amount of guest physical memory granted to the group, i.e., mapped to machine memory. The overhead memory, "OVHD" is not included in GRANT. The shared memory, "SHRD", is part of "GRANT". This statistics is added to esxtop in ESX 4.0.

The consumed machine memory for the VM, not including the overhead memory, can be estimated as "GRANT" - "SHRDSVD". Please refer to "SHRDSVD".

For a user world, it is the amount of virtual memory that is backed by machine memory.

Q: Why is "GRANT" less than "MEMSZ"?

A: Some guest physical memory has never been used, or is reclaimed by balloon driver, or is swapped out to the VM swap file. Note that this kind of swap is host swap, not the guest swap by the guest OS.

"MEMSZ" = "GRANT" + "MCTLSZ" + "SWCUR" + "never touched"

Q: How do I know how much machine memory is consumed by this VM?

A: GRANT accounts the guest physical memory, it may not be the same as the mapped machine memory, due to page sharing.

The consumed machine memory can be estimated as "GRANT" - "SHRDSVD". Please note that this is an estimate. Please refer to "SHRDSVD".

Note that overhead memory, "OVHD", is not part of the above consumed machine memory.

- **"SZTGT" (MB)**

The amount of machine memory to be allocated. (TGT is short for "target".) Note that "SZTGT" includes the overhead memory for a VM.

This is an internal counter, which is computed by ESX memory scheduler. Usually, there is no need to worry about this. Roughly speaking, "SZTGT" of all the VMs is computed based on the resource usage, available memory, and the "limit/reservation/shares" settings. This computed "SZTGT" is compared against the current memory consumption plus overhead memory for a VM to determine the swap and balloon target, so that VMKernel may balloon or swap appropriate amount

of memory to meet its memory demand. Please refer to "Resource Management Guide" for details.

Q: How come my "SZTGT" is larger than "MEMSZ"?

A: "SZTGT" includes the overhead memory, while "MEMSZ" does not. So, it is possible for "SZTGT" be larger than "MEMSZ".

Q: How do I use "SZTGT"?

A: This is an internal counter. You don't need to use it.

This counter is used to determine future swapping and ballooning activities. Check "SWTGT" and "MCTLTGT".

- **"TCHD" (MB)**

The amount of guest physical memory recently used by the VM, which is estimated by VMKernel statical sampling.

VMKernel estimates active memory usage for a VM by sampling a random subset of the VM's memory resident in machine memory to detect the number of memory reads and writes. VMKernel then scales this number by the size of VM's configured memory and averages it with previous samples. Over time, this average will approximate the amount of active memory for the VM.

Note that ballooned memory is considered inactive, so, it is excluded from "TCHD".

Because sampling and averaging takes time, "TCHD" won't be exact, but becomes more accurate over time.

VMKernel memory scheduler charges the VM by the sum of (1) the "TCHD" memory and (2) idle memory tax. This charged memory is one of the factors that memory scheduler uses for computing the "SZTGT".

Q: What is the difference between "TCHD" and working set estimate by guest OS?

A: "TCHD" is the working set estimated by VMKernel. This number may be different from guest working set estimate. Sometimes the difference may be big, because (1) guest OS uses a different working set estimate algorithm, (2) guest OS has a different view of active guest physical memory, due to ballooning and host swapping,

Q: How is "TCHD" used?

A: "TCHD" is a working set estimate, which indicates how actively the VM is using its memory. See above for the internal use of this counter.

- **"%ACTV"**

Percentage of active guest physical memory, current value.

"TCHD" is actually computed based on a few parameters, coming from statistical sampling. The exact equation is out of scope of this document. Esxtop shows some of those parameters, %ACTV, %ACTVS, %ACTVF, %ACTVN. Here, this document provides simple descriptions without further discussion.

%ACTV reflects the current sample.

%ACTVS is an EWMA of %ACTV for long term estimate.

%ACTVF is an EWMA of %ACTV for short term estimate.

%ACTVN is a predict of what %ACTVF will be at next sample.

Since they are very internal to VMKernel memory scheduler, we do not discuss their usage here.

- **"%ACTVS"**

Percentage of active guest physical memory, slow moving average. See above.

- **"%ACTVF"**

Percentage of active guest physical memory, fast moving average. See above.

- **"%ACTVN"**

Percentage of active guest physical memory in the near future. This is an estimated value. See above.

- **"MCTL?"**

Memory balloon driver is installed or not.

If not, install VMware tools which contains the balloon driver.

- **"MCTLSZ" (MB)**

The amount of guest physical memory reclaimed by balloon driver.

This can be called "balloon size". A large "MCTLSZ" means lots of this VM's guest physical memory is "stolen" to decrease host memory pressure. This usually is not a problem, because balloon driver tends to smartly steal guest physical memory that cause little performance problems.

Q: How do I know the VM is ballooning?

A: If "MCTLSZ" is changing, balloon driver is actively reclaiming or releasing memory. I.e., the VM is ballooning. Please note that the ballooning rate for a short term can be estimated by the change of "MCTLSZ", assuming it is either increasing or decreasing. But, for a long term, we cannot do it this way, because that monotonically increase/decrease assumption may not hold.

Q: Does ballooning hurt VM performance?

A: If guest working set is smaller than guest physical memory after ballooning, guest applications won't observe any performance degradation. Otherwise, it may cause guest swapping and hurt guest application performance.

Please check what causes ballooning and take appropriate actions to reduce memory pressure. There are two possible reasons: (1) The host does not have enough machine memory for use. (2) Memory used by the VM reaches the "limit" setting of itself or "limit" of the resource pools that contain this VM. In either case, ballooning is necessary and preferred over swapping.

- **"MCTLTGT" (MB)**

The amount of guest physical memory to be kept in balloon driver. (TGT is short for "target".)

This is an internal counter, which is computed by ESX memory scheduler. Usually, there is no need to worry about this.

Roughly speaking, "MCTLTGT" is computed based on "SZTGT" and current memory usage, so that the VM can balloon appropriate amount of memory. If "MCTLTGT" is greater than "MCTLSZ", VMKernel initiates inflating the balloon immediately, causing more VM memory to be reclaimed. If "MCTLTGT" is less than "MCTLSZ", VMKernel will deflate the balloon when the guest is requesting memory, allowing the VM to map/consume additional memory if it needs it. Please refer to "Resource Management Guide" for details.

Q: Why is it possible for "MCTLTGT" to be less than "MCTLSZ" for a long time?

A: If "MCTLTGT" is less than "MCTLSZ", VMKernel allows the balloon to deflate. But, balloon deflation happens lazily until the VM requests new memory. So, it is possible for "MCTLTGT" to be less than "MCTLSZ" for a long time, when the VM is not requesting new memory.

- **"MCTLMAX" (MB)**

The maximum amount of guest physical memory reclaimable by balloon driver.

This value can be set via vmx option "sched.mem.maxmemctl". If not set, it is determined by the guest operating system type. "MCTLTGT" will never be larger than "MCTLMAX".

If the VM suffers from ballooning, "sched.mem.maxmemctl" can be set to a smaller value to reduce this possibility. Remember that doing so may result in host swapping during resource contention.

- **"SWCUR" (MB)**

Current swap usage.

For a VM, it is the current amount of guest physical memory swapped out to the backing store. Note that it is the VMKernel swapping not the guest OS swapping.

It is the sum of swap slots used in the vswp file or system swap, and migration swap. Migration swap is used for a VMotioned VM to hold swapped out memory on the destination host, in case the destination host is under memory pressure.

Q: What does it mean if "SWCUR" of my VM is high?

A: It means the VM's guest physical memory is not resident in machine memory, but on disk. If those memory will not be used in the near future, it is not an issue. Otherwise, those memory will be swapped in for guest's use. In that case, you will see some swap-in activities via "SWR/s", which may hurt the VM's performance.

- **"SWTGT" (MB)**

The expected swap usage. (TGT is short for "target".)

This is an internal counter, which is computed by ESX memory scheduler. Usually, there is no need to worry about this.

Roughly speaking, "SWTGT" is computed based on "SZTGT" and current memory usage, so that the VM can swap appropriate amount of memory. Again, note that it is the VMKernel swapping not the guest swapping. If "SWTGT" is greater than "SWCUR", VMKernel starts swapping immediately, causing more VM memory to be swapped out. If "SWTGT" is less than "SWCUR", VMKernel will stop swapping. Please refer to "Resource Management Guide" for details.

Q: Why is it possible for "SWTGT" to be less than "SWCUR" for a long time?

A: Since swapped memory stays swapped until the VM accesses it, it is possible for "SWTGT" be less than "SWCUR" for a long time.

- **"SWR/s" (MB)**

Rate at which memory is being swapped in from disk. Note that this stats refers to the VMKernel swapping not the guest swapping.

When a VM is requesting machine memory to back its guest physical memory that was swapped out to disk, VMKernel reads in the page. Note that the swap-in operation is synchronous.

Q: What does it mean if SWR/s is high?

A: It is very bad for VM's performance. Because swap-in is synchronous, the VM needs to wait until the requested pages are read into machine memory. This happens when VMKernel swapped out the VM's memory before and the VM needs them now. Please refer to "SWW/s".

- **"SWW/s" (MB)**

Rate at which memory is being swapped out to disk. Note that this stats refers to the VMKernel swapping not the guest swapping.

As discussed in "SWTGT", if "SWTGT" is greater than "SWCUR", VMKernel will swap out memory to disk. It happens usually in two situations. (1) The host does not have enough machine memory for use. (2) Memory used by the VM reaches the "limit" setting of itself or "limit" of the resource pools that contain this VM.

Q: What does it mean if SWW/s is high?

A: It is very bad for VM performance. Please check the above two reasons and fix your problem accordingly.

If this VM is swapping out memory due to resource contention, it usually means VMKernel does not have enough machine memory to meet memory demands from all the VMs. So, it will swap out mapped guest physical memory pages to make room for the recent requests.

- **"SHRD" (MB)**

Amount of guest physical memory that are shared.

VMKernel page sharing module scans and finds guest physical pages with the same content and backs them with the same machine page. "SHRD" accounts the total guest physical pages that are shared by the page sharing module.

- **"ZERO" (MB)**

Amount of guest physical zero memory that are shared. Thisis an internal counter.

A zero page is simply the memory page that is all zeros. If a zero guest physical page is detected by VMKernel page sharing module, this page will be backed by the same machine page on each NUMA node. Note that "ZERO" is included in "SHRD".

- **"SHRDSVD" (MB)**

Estimated amount of machine memory that are saved due to page sharing.

Because a machine page is shared by multiple guest physical pages, we only charge "1/ref" page as the consumed machine memory for each of the guest physical pages, where "ref" is the number of references. So, the saved machine memory will be "1 - 1/ref" page."SHRDSVD" estimates the total saved machine memory for the VM.

The consumed machine memory by the VM can be estimated as "GRANT" - "SHRDSVD".

- **"COWH" (MB)**

Amount of guest physical hint pages for page sharing. This is an internal counter.

- **"OVHDUW" (MB)**

Amount of overhead memory reserved for the vmx user world of a VM group. This is an internal counter.

"OVHDUW" is part of "OVHDMAX".

- **"OVHD" (MB)**

Amount of overhead memory currently consumed by a VM.

"OVHD" includes the overhead memory consumed by the monitor, the VMkernel and the vmx user world.

- **"OVHDMAX" (MB)**

Amount of reserved overhead memory for the entire VM.

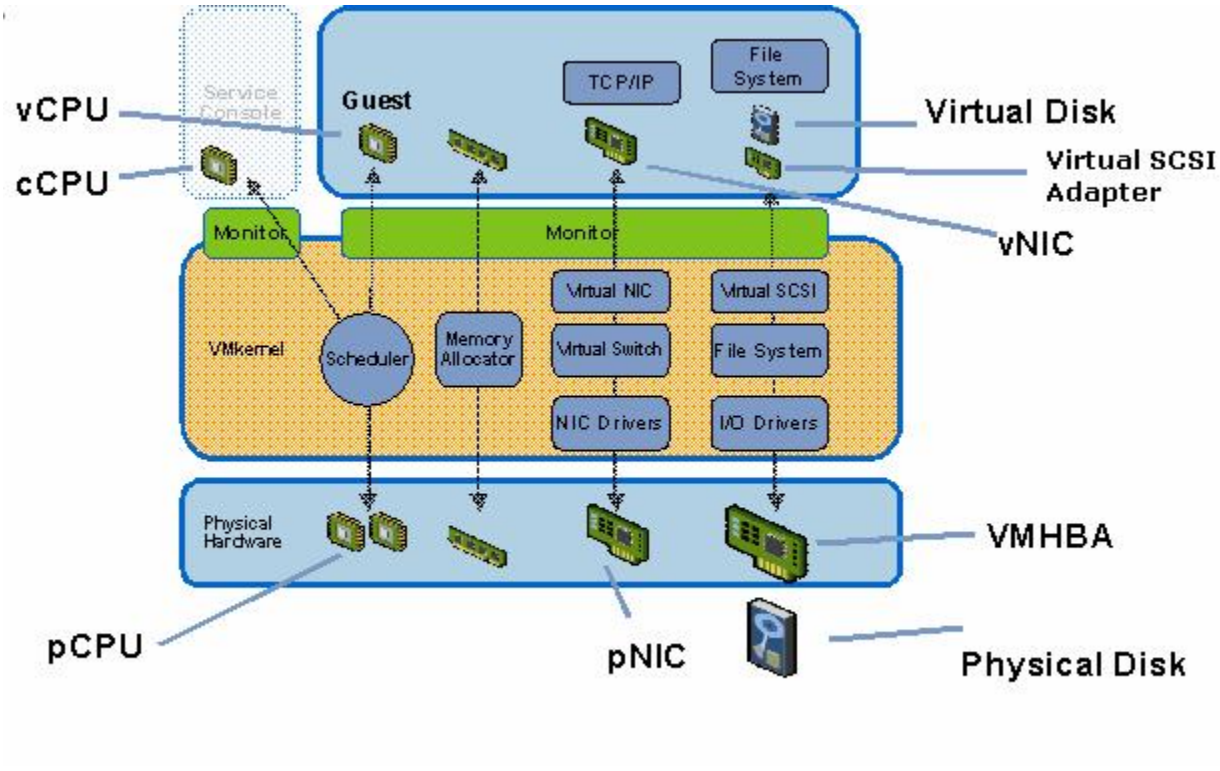
"OVHDMAX" is the overhead memory a VM wants to consume in the future. This amount of reserved overhead memory includes the overhead memory reserved by the monitor, the VMkernel, and the vmx user world. Note that the actual overhead memory consumption is less than "OVHDMAX". "OVHD" < "OVHDMAX".

"OVHDMAX" can be used as a conservative estimate of the total overhead memory.

Section 4 Disk

Section 4.1 Adapter, Device, VM screens

The ESX storage stack adds a few layers of code between a virtual machine and bare hardware. All virtual disks in virtual machines are seen as virtual SCSI disks. The ESX storage stack allows these virtual disks to be located on any of the multiple storage options available.



For performance analysis purposes, an IO request from an application in a virtual machine traverses through multiple levels of queues, each associated with a resource, in the guest OS, the VMkernel and the physical storage. (Note that physical storage could be an FC- or IP- SAN or disk array.) Each queue has an associated latency, dictated by its size and whether the IO load is low or high, which affects the throughput and latency seen by applications inside VMs.

Esxtop shows the storage statistics in three different screens: adapter screen, device screen, and vm screen. Interactive command *'d'* can be used to switch to the adapter screen, *'u'* for the device screen, and 'v' for the vm screen.

The main difference in the data seen in these three screens is the level at which it is aggregated, even though these screens have similar counters. By default, data is rolled up to the highest level possible for each screen. (1) On the adapter screen, by default, the statistics are aggregated per storage adapter but they can also be expanded to display data per storage channel, target, path or world using a LUN. See interacitive commands, *'e', 'E', 'P', 'a', 't', 'l'*, for the expand operations. (2) On the device screen, by default, statistics are aggregated per storage device. Statistics can also be viewed per path, world, or partition. See interactive commands, *'e', 'p', 't'*, for the expand operations. (3) On the VM screen, statistics are aggregated on a per-group basis by default. One VM has

one corresponding group, so they are equivalent to per-VM statistics. You can use interactive command *'V'* to show only statistics related to VMs. Statistics can also be expanded so that a row is displayed for each world or a per-world-per-device basis. See interactive commands, *'e' and 'l'*.

Please refer to esxtop man page for the details of the interactive commands.

Section 4.2 Disk Statistics

Due to the similarities in the counters of the three disk screens, this section discusses the counters without distinguishing the screens. Similar to other esxtop screens, the storage counters are also organized in different sets, each of which contains related counters. The counters can be selected as a set by selecting the appropriate field option in esxtop. If esxtop is used in batch mode, make sure that the esxtop configuration file includes all counters of interest.

Each group of counters in the following subsections corresponds to a particular field option.

Section 4.2.1 I/O Throughput Statistics

- **CMDS/s**

Number of commands issued per second.

- **READS/s**

Number of read commands issued per second.

- **WRITES/s**

Number of write commands issued per second.

- **MBREAD/s**

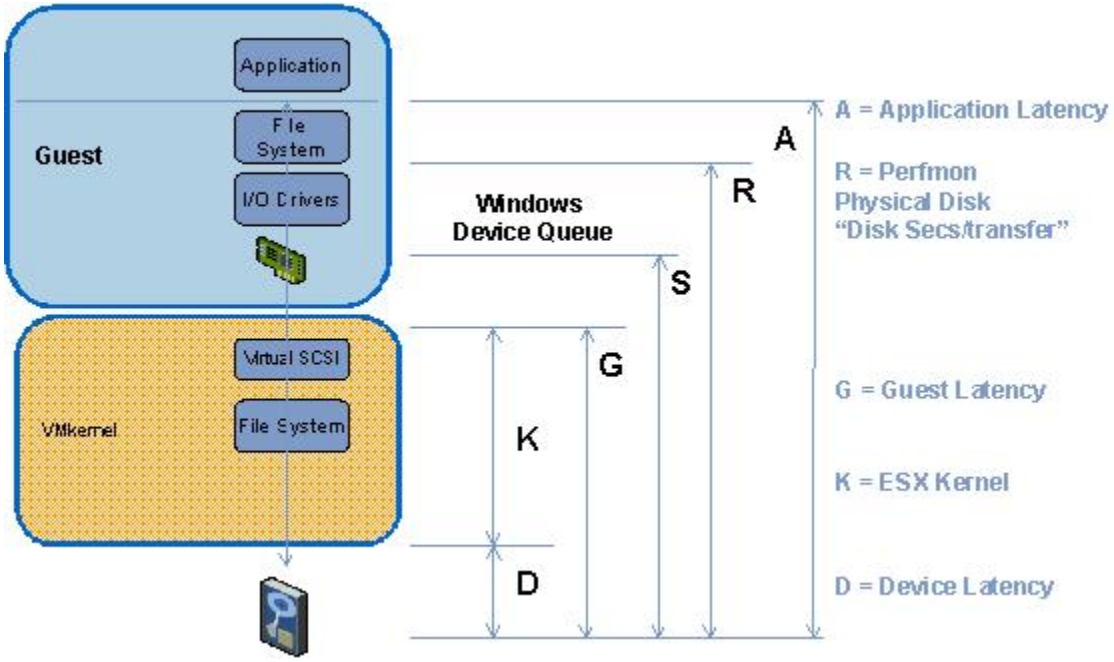
Megabytes read per second.

- **MBWRTN/s**

Megabytes written per second.

Section 4.2.2 Latency Statistics

This group of counters report latency values measured at three different points in the ESX storage stack. In the context of the figure below, the latency counters in esxtop report the Guest, ESX Kernel and Device latencies. These are under the labels GAVG, KAVG and DAVG, respectively. Note that GAVG is the sum of DAVG and KAVG counters.



Note that esxtop shows the latency statistics for different objects, such as adapters, devices, paths, and worlds. They may not perfectly match with each other, since their latencies are measured at the different layers of the ESX storage stack. To do the correlation, you need to be very familiar with the storage layers in ESX Kernel, which is out of our scope.

Latency values are reported for all IOs, read IOs and all write IOs. All values are averages over the measurement interval.

- All IOs: KAVG/cmd, DAVG/cmd, GAVG/cmd, QAVG/cmd
- Read IOs: KAVG/rd, DAVG/rd, GAVG/rd, QAVG/rd
- Write IOs: KAVG/wr, DAVG/wr, GAVG/wr, QAVG/wr

- **GAVG**

This is the round-trip latency that the guest sees for all IO requests sent to the virtual storage device.

GAVG should be close to the R metric in the figure.

Q: What is the relationship between GAVG, KAVG and DAVG?

A: GAVG = KAVG + DAVG

- **KAVG**

These counters track the latencies due to the ESX Kernel's command.

The KAVG value should be very small in comparison to the DAVG value and should be close to zero. When there is a lot of queuing in ESX, KAVG can be as high, or even higher than DAVG. If this happens, please check the queue statistics, which will be discussed next.

- **DAVG**

This is the latency seen at the device driver level. It includes the roundtrip time between the HBA and the storage.

DAVG is a good indicator of performance of the backend storage. If IO latencies are suspected to be causing performance problems, DAVG should be examined. Compare IO latencies with corresponding data from the storage array. If they are close, check the array for misconfiguration or faults. If not, compare DAVG with corresponding data from points in between the array and the ESX Server, e.g., FC switches. If this intermediate data also matches DAVG values, it is likely that the storage is under-configured for the application. Adding disk spindles or changing the RAID level may help in such cases.

- **QAVG**

The average queue latency. QAVG is part of KAVG.

Response time is the sum of the time spent in queues in the storage stack and the service time spent by each resource in servicing the request. The largest component of the service time is the time spent in retrieving data from physical storage. If QAVG is high, another line of investigation is to examine the queue depths at each level in the storage stack.

Section 4.2.3 Queue Statistics

- **AQLEN**

The storage adapter queue depth. This is the maximum number of ESX Server VMKernel active commands that the adapter driver is configured to support.

- **LQLEN**

The LUN queue depth. This is the maximum number of ESX Server VMKernel active commands that the LUN is allowed to have. (Note that, in this document, the terminologies of LUN and Storage device can be used interchangeably.)

- **WQLEN**

The World queue depth. This is the maximum number of ESX Server VMKernel active commands that the World is allowed to have. Note that this is a per LUN maximum for the World.

- **ACTV**

The number of commands in the ESX Server VMKernel that are currently active. This statistic is only applicable to worlds and LUNs.

Please refer to %USD.

- **QUED**

The number of commands in the VMKernel that are currently queued. This statistic is only applicable to worlds and LUNs.

Queued commands are commands waiting for an open slot in the queue. A large number of queued commands may be an indication that the storage system is overloaded. A sustained high value for the QUED counter signals a storage bottleneck which may be alleviated by increasing the queue depth. Check that LOAD < 1 after increasing the queue depth. This should also be accompanied by improved performance in terms of increased cmd/s.

Note that there are queues in different storage layers. You might want to check the QUED stats for devices, and worlds.

- **%USD**

The percentage of queue depth used by ESX Server VMKernel active commands. This statistic is only applicable to worlds and LUNs.

%USD = ACTV / QLEN * 100%

For world stats, WQLEN is used as the denominator. For LUN (aka device) stats, LQLEN is used as the denominator.

%USD is a measure of how many of the available command queue "slots" are in use. Sustained high values indicate the potential for queueing; you may need to adjust the queue depths for system’s HBAs if QUED is also found to be consistently > 1 at the same time. Queue sizes can be adjusted in a few places in the IO path and can be used to alleviate performance problems related to latency. For detailed information on this topic please refer to the VMware whitepaper entitled "Scalable Storage Performance".

- **LOAD**

The ratio of the sum of VMKernel active commands and VMKernel queued commands to the queue depth. This statistic is only applicable to worlds and LUNs.

The sum of the active and queued commands gives the total number of outstanding commands issued by that virtual machine. The LOAD counter values is the ratio of this value with respect to the queue depth. If LOAD > 1, check the value of the QUED counter.

Section 4.2.4 Error Statistics

- **ABRTS/s**

The number of commands aborted per second.

It can indicate that the storage system is unable to meet the demands of the guest operating system. Abort commands are issued by the guest when the storage system has not responded within an acceptable amount of time, e.g. 60 seconds on some windows OS’s. Also, resets issued by a guest OS on its virtual SCSI adapter will be translated to aborts of all the commands outstanding on that virtual SCSI adapter.

- **RESETS/s**

The number of commands reset per second.

Section 4.2.5 PAE Statistics

- **PAECMD/s**

The number of PAE commands per second.

It may point to hardware misconfiguration. When the guest allocates a buffer, the vmkernel assigns some machine memory, which might come from a “highmem” region. If you have a driver that is not PAE-aware, then this counter is updated if accesses to this memory region result in copies by the vmkernel into a lower memory location before issuing the request to the adapter. This might happen if you do not populate the DIMMs with low memory first, then you may artificially cause “highmem” memory accesses.

- **PAECP/s**

The number of PAE copies per second.

Section 4.2.6 Split Statistics

- **SPLTCMD/s**

The number of split commands per second.

Commands can be split when they reach the vmkernel. This might impact perceived latency to the guest. The guest may be issuing commands of large block sizes which have to be broken down by the vmkernel. For ESX3.0.x, guest requests greater than 128KB are split into 128KB chunks. Since few applications do larger than 128KB ops, this is unlikely to be an issue. Splitting can also occur when IOs fall across partition boundaries but these are easily differentiated from the splitting as a result of the IO size.

- **SPLTCP/s**

The number of split copies per second.

Section 4.3 Batch Mode Output

Esxtop batch mode output can be loaded in perfmon directly. It uses a csv (comma separated values) format. The instance type can be ideitified via its name. Because there are quite a number of instances related to disk statistics, let's list a few examples below. You may easily match the format in your own environment.

- LUN (aka device): "\\<host>\Physical Disk(DEV-vmhba0:0:0)\<counter>"
- Partition: "\\<host>\Physical Disk(PN-vmhba0:0:0-1)\<counter>"
- Path: "\\<host>\Physical Disk(PH-vmhba0:C0:T0:L0)\<counter>"
- Per-World-Per-Device: "\\<host>\Physical Disk(WD-vmhba0:0:0-1024)\<counter>"
- Adapter: "\\<host>\Physical Disk(vmhba0)\<counter>"

Section 5 Network

Section 5.1 Port

We arrange the network stats per port of a virtual switch. "PORT-ID" identifies the port and "DNAME" shows the virtual switch name. A port can be linked to a physical NIC as an uplink, or can be connected by a virtual NIC. "UPLINK" indicates whether the port is an uplink.

If the port is an uplink, i.e., "UPLINK" is 'Y', "USED-BY" shows the physical NIC name.

If the port is connected by a virtual NIC, i.e., "UPLINK" is 'N', "USED-BY" shows the port client name. (a) If the port is used by a virtual machine, the client name contains a world id and the VM name. The world id identifies the leader world of the VM group. Note that "vswif" is used by COS (on classic ESX). (b) If the port is used by VMKernel system, there is no world id. The client name can be used to identify the use of the port. To give two examples.

- "vmk" is a port used by vmkernel. Users can create vmk NICs for their uses, such as VMotion. On ESXi, there will be at least one vmk NIC to communicate with outside of the host.
- "Management" is a management port for a portset. This is internal. Usually no need to worry about it.

For each non-uplink port, the NIC teaming policy determines which physical NIC is in charge of the port. "TEAM-PNIC" shows the physical NIC name, if valid. Please refer to NIC teaming documentation for details.

Section 5.2 Port Statistics

- **"SPEED" (Mbps)**

The link speed in Megabits per second. This information is only valid for a physical NIC.

- **"FDUPLX"**

'Y' implies the corresponding link is operating at full duplex. 'N' implies it is not. This information is only valid for a physical NIC.

- **"UP"**

'Y' implies the corresponding link is up. 'N' implies it is not. This information is only valid for a physical NIC.

- **"PKTTX/s"**

The number of packets transmitted per second.

- **"PKTRX/s"**

The number of packets received per second.

- **"MbTX/s" (Mbps)**

The MegaBits transmitted per second.

- **"MbRX/s" (Mbps)**

The MegaBits received per second.

Q: Why does MbRX/s not match PKTRX/s for different workloads?

A: This is because the packet size may not be the same. The average packet size can be computed as follows: average_packet_size = MbRX/s / PKTRX/s . A large packet size may improve CPU efficiency of processing the packet. However, it may potentially increase latency.

- **"%DRPTX"**

The percentage of transmit packets dropped.

"%DRPTX" = "dropped Tx packets" / ("success Tx packets" + "dropped Tx packets")

Q: What does it mean if %DRPTX is high?

A: This usually means the network transmit performance is bad. Please check whether the phsycial NICs are fully utilizing their capacity. You probably need physical NICs with better performance. Or, you may add more physical NICs and use a good NIC teaming load balancing policy.

- **"%DRPRX"**

The percentage of receive packets dropped.

"%DRPRX" = "dropped Rx packets" / ("success Rx packets" + "dropped Rx packets")

Q: What does it mean if %DRPRX is high?

A: This usally means the network recieve performance is bad. Try to give more CPU resource to the impacted VM, or increase the ring buffer size.

- **"ACTN/s"**

Number of actions per second. The actions here are VMkernel actions. It is an internal counter. We won't discuss it further here.

Section 6. Interrupt

Interrupt screens are under development for our next release.

Section 7. Batch Mode

Esxtop batch mode output uses a csv (comma separated values) format. The first line contains the names of the performance counters and their instances. Each of the following lines contains the performance data for those counter instances in one snapshot.

One way to read the batch mode output file is to load it in Windows perfmon. (1) Run perfmon; (2) Type "Ctrl + L" to view log data; (3) Add the file to the "Log files" and click OK; (4) Choose the counters to show the performance data. Each batch mode counter has a category name (listed as a performance object in perfmon) and a counter name (listed in the counter list in perfmon).

The counter names in esxtop batch mode are different from the ones in interactive mode listed in the sections above. The tables below describe their relationships. The first column is the interactive mode counter name; the second column is the batch mode counter category; the last column is the batch mode counter name.

- **Table 7-1 CPU Batch Mode Counters**

Counter Name	Batch Mode Category	Batch Mode Counter Name
CPU load average	Physical Cpu Load	Cpu Load (1 Minute Avg)
		Cpu Load (5 Minute Avg)
		Cpu Load (15 Minute Avg)
PCPU(%)	Physical Cpu	% Processor Time
LCPU(%)	Logical Cpu	% Processor Time
CCPU(%) us	Console Physical Cpu	% User Time
CCPU(%) sy	Console Physical Cpu	% System Time
CCPU(%) id	Console Physical Cpu	% Idle Time
CCPU(%) wa	Console Physical Cpu	% I/O Wait Time

CCPU(%) cs/sec	Console Physical Cpu	% Context Switches/sec
%USED	Group Cpu (or Vcpu)	% Used
%SYS	Group Cpu (or Vcpu)	% System
%OVRLP	Group Cpu (or Vcpu)	% Overlap
%RUN	Group Cpu (or Vcpu)	% Run
%RDY	Group Cpu (or Vcpu)	% Ready
%MLMTD	Group Cpu (or Vcpu)	% Max Limited
%CSTP	Group Cpu (or Vcpu)	% CoStop
%WAIT	Group Cpu (or Vcpu)	% Wait
%IDLE	Group Cpu (or Vcpu)	% Idle
%SWPWT	Group Cpu (or Vcpu)	% Swap Wait

- **Table 7-2 Memory Batch Mode Counters**

Counter Name	Batch Mode Category	Batch Mode Counter Name
MEM overcommit avg	Memory	Memory Overcommit (1 Minute Avg)
		Memory Overcommit (5 Minute Avg)
		Memory Overcommit (15 Minute Avg)
PMEM total	Memory	Machine MBytes
PMEM cos	Memory	Console MBytes
PMEM vmk	Memory	Kernel MBytes
PMEM other	Memory	NonKernel MBytes
PMEM free	Memory	Free MBytes
VMKMEM managed	Memory	Kernel Managed MBytes
VMKMEM minfree	Memory	Kernel MinFree MBytes
VMKMEM rsvd	Memory	Kernel Reserved MBytes
VMKMEM ursvd	Memory	Kernel Unreserved MBytes
VMKMEM state	Memory	Kernel State (0: high, 1: soft, 2:hard, 3: low)
COSMEM free	Console Memory	Free MBytes
COSMEM swap_t	Console Memory	Swap Total MBytes
COSMEM swap_f	Console Memory	Swap Free MBytes

COSMEM r/s	Console Memory	Swap MBytes Read/sec
COSMEM w/s	Console Memory	Swap MBytes Write/sec
NUMA	Numa Node	Total MBytes
		Free MBytes
PSHARE shared	Memory	PShare Shared MBytes
PSHARE common	Memory	PShare Common MBytes
PSHARE saving	Memory	PShare Savings MBytes
SWAP curr	Memory	Swap Used MBytes
SWAP target	Memory	Swap Target MBytes
SWAP r/s	Memory	Swap MBytes Read/sec
SWAP w/s	Memory	Swap MBytes Write/sec
MEMCTL curr	Memory	Memctl Current MBytes
MEMCTL target	Memory	Memctl Target MBytes
MEMCTL max	Memory	Memctl Max MBytes
MEMSZ	Group Memory	Memory Size MBytes
GRANT	Group Memory	Memory Granted Size MBytes
SZTGT	Group Memory	Target Size MBytes
TCHD	Group Memory	Touched MBytes
%ACTV	Group Memory	% Active Estimate
%ACTVS	Group Memory	% Active Slow Estimate
%ACTVF	Group Memory	% Active Fast Estimate
%ACTVN	Group Memory	% Active Next Estimate
MCTL?	Group Memory	Memctl?
MCTLSZ	Group Memory	Memctl MBytes
MCTLTGT	Group Memory	Memctl Target MBytes
MCTLMAX	Group Memory	Memctl Max MBytes
SWCUR	Group Memory	Swapped MBytes
SWTGT	Group Memory	Swap Target MBytes
SWR/s	Group Memory	Swap Read MBytes/sec
SWW/s	Group Memory	Swap Written MBytes/sec
SHRD	Group Memory	Shared MBytes

ZERO	Group Memory	Zero MBytes
SHRDSVD	Group Memory	Shared Saved MBytes
COWH	Group Memory	Copy On Write Hint MBytes
OVHDUW	Group Memory	Overhead UW MBytes
OVHD	Group Memory	Overhead MBytes
OVHDMAX	Group Memory	Overhead Max MBytes

- **Table 7-3 Disk Batch Mode Counters**

Counter Name	Batch Mode Category	Batch Mode Counter Name
CMDS/s	Physical Disk	Commands/sec
READS/s	Physical Disk	Reads/sec
WRITES/s	Physical Disk	Writes/sec
MBREAD/s	Physical Disk	MBytes Read/sec
MBWRTN/s	Physical Disk	MBytes Written/sec
KAVG/cmd	Physical Disk	Average Kernel MilliSec/Command
DAVG/cmd	Physical Disk	Average Driver MilliSec/Command
GAVG/cmd	Physical Disk	Average Guest MilliSec/Command
QAVG/cmd	Physical Disk	Average Queue MilliSec/Command
KAVG/rd	Physical Disk	Average Kernel MilliSec/Read
DAVG/rd	Physical Disk	Average Driver MilliSec/Read
GAVG/rd	Physical Disk	Average Guest MilliSec/Read
QAVG/rd	Physical Disk	Average Queue MilliSec/Read
KAVG/wr	Physical Disk	Average Kernel MilliSec/Write
DAVG/wr	Physical Disk	Average Driver MilliSec/Write
GAVG/wr	Physical Disk	Average Guest MilliSec/Write
QAVG/wr	Physical Disk	Average Queue MilliSec/Write
AQLEN	Physical Disk	Adapter Q Depth
LQLEN	Physical Disk	Lun Q Depth
DQLEN	Physical Disk	Device Q Depth
WQLEN	Physical Disk	World Q Depth
ACTV	Physical Disk	Active Commands
QUED	Physical Disk	Queued Commands

%USD	Physical Disk	% Used
LOAD	Physical Disk	Load
LOAD	Physical Disk	Load
ABRTS/s	Physical Disk	Aborts/sec
RESETS/s	Physical Disk	Resets/sec
PAECMD/s	Physical Disk	PAE Commands/sec
PAECP/s	Physical Disk	PAE Copies/sec
SPLTCMD/s	Physical Disk	Split Commands/sec
SPLTCP/s	Physical Disk	Split Copies/sec

- **Table 7-4 Network Batch Mode Counters**

Counter Name	Batch Mode Category	Batch Mode Counter Name
SPEED	Network Port	Link Speed (Mb/s)
FDUPLX	Network Port	Full Duplex?
UP	Network Port	Link Up?
PKTTX/s	Network Port	Packets Transmitted/sec
PKTRX/s	Network Port	Packets Received/sec
MbTX/s	Network Port	MBits Transmitted/sec
MbRX/s	Network Port	MBits Received/sec
%DRPTX	Network Port	% Outbound Packets Dropped
%DRPRX	Network Port	% Received Packets Dropped
ACTN/s	Network Port	Actions Posted/sec