

*Learn. Innovate. Grow.*

vmware  
**technology**exchange

## **VI Performance Monitoring**

Preetham Gopaldaswamy – Group Product Manager

Ravi Soundararajan – Staff Engineer

September 15, 2008

## Agenda

- ▶ **Introduction to performance monitoring in VI**
- ▶ **Common customer/partner questions (use cases)**
- ▶ **Tips and Tricks**

## Performance Metrics Primer

- ▶ The VI platform exposes over 150 performance counters.
- ▶ Using the VI API, counter values can be retrieved for the entire datacenter including hosts and VMs, or just for a user-defined resource pool of hosts and/or VMs.
- ▶ A counter is uniquely identified by a combination of its name, group and rollup type. It can be represented using a dotted notation: `<group>.<name>.<roll-up>`  
e.g. `cpu.usage.min` is the minimum CPU usage in the sample period.
- ▶ Every counter includes a description and unit of measure.
- ▶ Latest Information <http://vmware.com/developer>  
VMware Developer Center Blog

## Performance Metrics Primer

- ▶ **Use the VI API to ask the server what counters it exposes. A sample script to accomplish this is available on the VMware website.**
- ▶ **The counters are broadly divided into these categories:**
  - > CPU
  - > Management Agent
  - > Resource Group
  - > Network
  - > Disk
  - > Memory
  - > System
- ▶ **The rollup options over a sample period are:**
  - > none (instantaneous value)
  - > average (average over the sampling period)
  - > maximum (maximum value in the sampling period)
  - > minimum (minimum value in the sampling period)
  - > latest (last value in the sampling period)
  - > summation (sum of the values over the sampling period)

## Performance Metrics Primer

- ▶ **VirtualCenter collects performance metrics from the hosts that it manages and aggregates the data using consolidation algorithms based on MRTG. The algorithm is optimized to keep the database size constant over time.**
- ▶ **If the partner application is also aggregating the data, VMware recommends collecting the consolidated data from VC.**
- ▶ **Statistics collection levels (range 1-4) define the number of counters collected and aggregated by VC per provider. VMware recommends that normal operation should be Level 1 or 2. Higher values are for debugging i.e. for short periods of time.**
- ▶ **Default stat collection periods and how long they are stored are:**

Interval	Interval Period	Interval Length
Per day	5 minutes*	1 day*
Per week	30 minutes	1 week
Per month	2 hours	1 month
Per year	1 day	1 year*

*(Items with a \* next to them can be configured)*

## Performance Metrics Primer

- ▶ **The performance statistic collection level and aggregation are extremely configurable.**
- ▶ **Customers can tune the collection level based on the historical interval. Debugging statistics need not be retained for long periods of time.**  
*e.g. Per HBA statistics are important for a week but not a year*
- ▶ **The aggregation can also be turned off after a particular historical time level.**
- ▶ **Below is an example of a customer configuration**

Interval	Interval Period	Interval Length	Level	Aggregate
Per day	5 minutes*	1 day*	4	Yes
Per week	30 minutes	1 week	3	Yes
Per month	2 hours	1 month	2	No
Per year	1 day	1 year*	1	No

## Performance Metrics Primer

- ▶ **The minimum counter granularity to collect statistics is 20 seconds.**
- ▶ **If information is requested from Virtual Center at a frequency of 5 minutes or lower, that request is passed through directly to the host to get accurate real-time data.**
- ▶ **Virtual Center scalability for statistics is significantly improved in VC 2.5**

### **Partner quote:**

- > VC 2.0 could get Level 4 stats for up to 20 hosts in about 5 minutes.
- > VC 2.5 can get the same stats for up to 100 hosts (500 powered-on VMs) in 1.5 minutes

## Common Customer Questions



### Why can't I use (r)esxtop? How is it different from the counters?

- ▶ **I get different numbers from the API v/s esxtop in COS**
  - > Source of data is the same (VMkernel).
  - > Sampling frequencies may differ (esxtop: 5s, VirtualCenter 20s)
- ▶ **Are there other differences between the metrics?**
  - > esxtop contains some counters that VC does not (e.g. Disk ACTV)
  - > The unit of measure on some counters is different (% vs. ms)
- ▶ **esxtop has better interval granularity. I will use it all the time.**
  - > esxtop puts a very high load on the server. It should be used for interactive troubleshooting at best.
  - > The API counters are optimized for retrieval and aggregation and provide all the data that is necessary to debug problems.

## Common Customer Questions



**How can I validate that my virtual environment is better?**

- ▶ **System administrators are often under pressure to virtualize the datacenter to reduce TCO**
- ▶ **But there is always that nagging question:  
*Am I doing better than or at least as well as before? How is my system performing?***

## Virtual Environment v/s Physical Environment

### ▶ First, define “better” or “as well as”

- Better CPU utilization?  
1 server at 80% v/s 4 servers at 20% each
- Better memory utilization?  
1 server with 4GB RAM v/s 4 servers with 2GB each
- Lower power consumption?  
Fewer physical servers means less power and less cooling
- More scalable performance?  
4 UP VMs with better throughput than a 4-way native server

### ▶ How can our counters help?

- Counters are currently limited to resource utilization (cpu, memory, disk, network)
- Collect cpu usage %, memory consumed and compare to the physical
- VMware currently does not expose metrics for application performance or power consumed

## Common Customer Questions



### When do I need to add another host?

- ▶ I now have 30 Virtual Machines running on 3 hosts.
- ▶ I have to provision another 5 VMs in the next quarter.
- ▶ Can I leverage my existing infrastructure or should I be planning on bring in another host? Have I already maxed out my current CPU capacity?

## CPU capacity

### ▶ How do we know we are maxed out?

- If VMs are waiting for CPU time, maybe we need more CPUs.
- To measure this, look at CPU *ready time*.

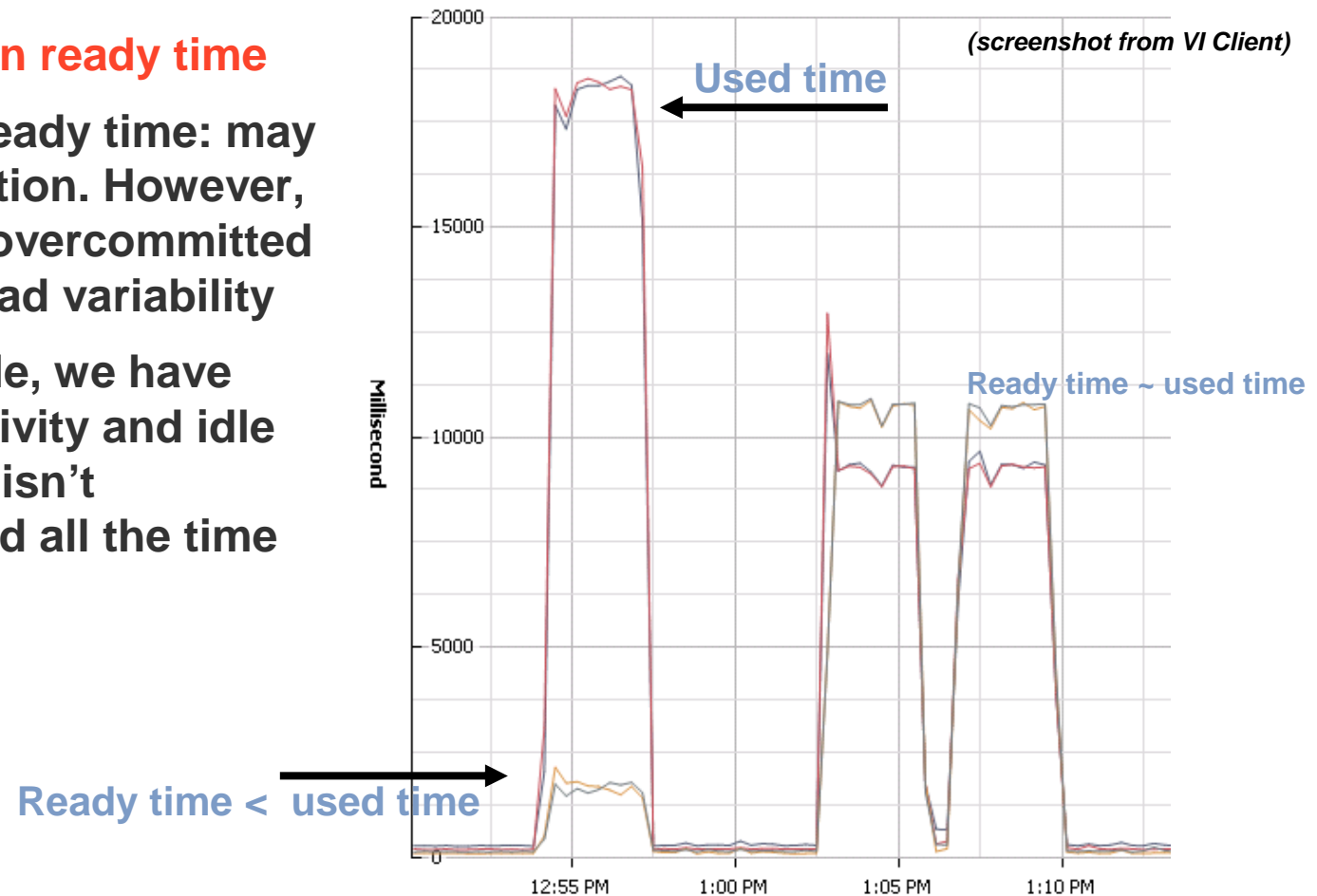
### ▶ What exactly am I looking for?

- For each host, collect *ready time* for each VM
- Compute *%ready time* for each VM (*ready time/sampling interval*)
- If average *%ready time* > 20% over an extended interval, probe further

## CPU capacity

### Some caveats on ready time

- ▶ Used time ~ ready time: may signal contention. However, might not be overcommitted due to workload variability
- ▶ In this example, we have periods of activity and idle periods: CPU isn't overcommitted all the time



## Further ready time examination

```
2:01:53pm up 4 days 29 min, 87 worlds; CPU load average: 0.16, 0.16, 0.09
PCPU(%): 13.20, 15.55, 10.71, 23.06 ; used total: 15.63
CCPU(%): 0 us, 0 sy, 99 id, 0 wa ; cs/sec: 98
```

ID	GID	NAME	NWLD	%USED	%RUN	%SYS	%WAIT	%RDY	%MLMTD
1	1	idle	4	337.41	338.34	0.00	0.00	61.66	0.00
2	2	system	6	0.02	0.02	0.00	599.97	0.00	0.00
6	6	helper	22	0.02	0.02	0.00	2199.96	0.11	0.00
7	7	drivers	11	0.01	0.01	0.00	1099.98	0.00	0.00
9	9	console	1	0.92	0.93	0.00	98.27	0.80	0.00
14	14	vmkapimod	2	0.00	0.00	0.00	200.00	0.00	0.00
15	15	vmware-vmkauthd	1	0.00	0.00	0.00	100.00	0.00	0.00
16	16	cpuBurn-CLONE	7	50.11	50.03	0.00	495.74	154.20	152.44
17	17	fakeDB	7	1.43	1.44	0.00	697.67	0.89	0.00
18	18	Windows 2003 SP	7	5.10	5.12	0.00	693.01	1.87	0.00
19	19	SQL2005	7	1.63	1.59	0.03	697.16	1.24	0.00
20	20	memhog-linux-CL	5	1.17	1.16	0.02	498.29	0.55	0.00
21	21	cpuBurn-CLONE2	7	1.31	1.29	0.03	698.01	0.70	0.00

High Ready Time

High MLMTD: there is a limit on this VM...

→ High ready time not always because of overcommitment

## Ready time in VI client



## 3 Possible reasons for high ready time

### ▶ Possible causes

#### > CPU over-commitment

#### > Workload variability

- A bunch of VMs wake up all at once
- Note: system may be mostly idle: not always overcommitted

#### > Reservation set on VM

- 4x2GHz host, 2 vcpu VM, limit set to 1GHz (VM can consume 1GHz)
- Without limit, max is 2GHz. With limit, max is 1GHz (50% of 2GHz)
- CPU all busy: %USED: 50%; %MLMTD & %RDY = 150% [total is 200%, or 2 CPUs]

### ▶ Possible solutions

#### > VMotion the VM or use DRS to optimize resources

#### > Change share allocations to de-prioritize less important VMs

#### > Check CPU limit settings

#### > More CPUs may be the solution

## Common Customer Questions



**Will adding more memory to my hosts help?**

- ▶ **I now have 30 Virtual Machines running on 3 hosts.**
- ▶ **The CPU utilization seems to be optimal but applications are a bit sluggish.**
- ▶ **Will adding more memory help solve the problem? Can I find that out by analyzing the performance statistics?**

## Memory capacity

### ▶ How do we identify host memory contention?

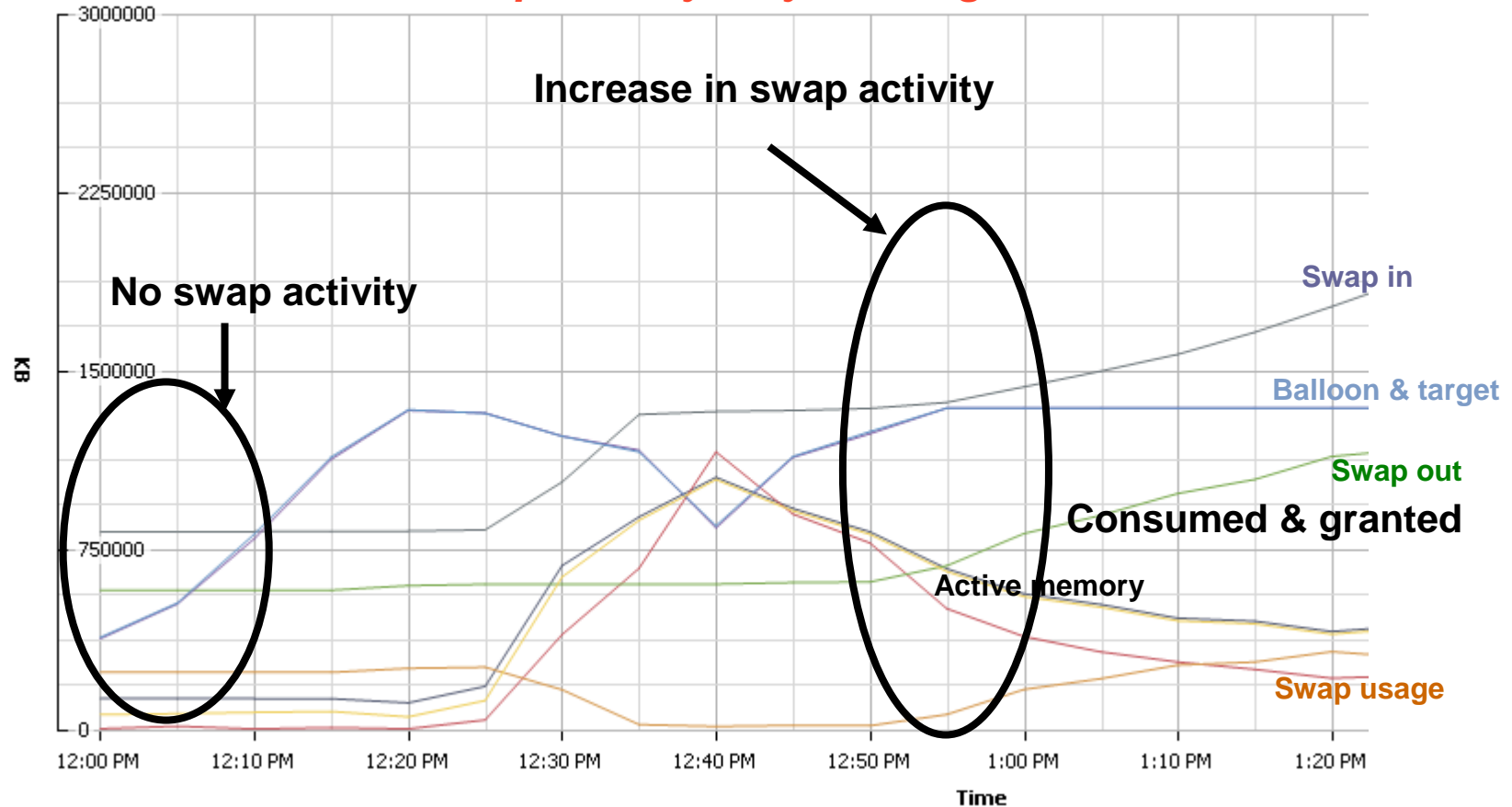
- Host-level swapping (e.g., robbing VM A to satisfy VM B).
- Active memory for all VMs > physical memory on host  
This could mean possible memory over-commitment

### ▶ What do I do?

- Check *swpin* (cumulative), *swapout* (cumulative) and *swapped* (“instantaneous”) for the host. Ballooning (*vmmemctl*) is also useful.
- If *swpin* and *swapout* are increasing, it means that there is possible memory over-commitment
- Another possibility: sum up active memory for each VM. See if it exceeds host physical memory.

## Memory capacity

*Increased swap activity may be a sign of over-commitment*



# Troubleshooting memory related problems

## ← Swapping

```
6:54:20am up 53 days 19:49, 87 worlds; MEM overcommit avg: 0.98, 1.15, 1.51
PMEM /MB: 4095 total: 272 cos, 175 vmk, 1461 other, 2186 free
VMKMEM/MB: 3735 managed: 224 minfree, 976 rsvd, 2648 ursvd, high state
COSMEM/MB: 9 free: 541 swap_t, 541 swap_f: 0.00 r/s, 0.00 w/s
PSHARE/MB: 5338 shared, 184 common: 5154 saving
SWAP /MB: 1295 curr, 677 target: 0.75 r/s, 0.01 w/s
MEMCTL/MB: 652 curr, 652 target, 4645 max
```

NAME	MEMSZ	SZTGT	MCTL?	MCTLSZ	MCTLTGT	MCTLMAX	SWCUR	SWTGT
Windows 2003 SP	1024.00	385.53	Y	0.00	0.00	665.60	119.82	0.00
SQL2005	2048.00	456.73	Y	0.00	0.00	1331.20	215.91	0.00
vc server	1024.00	284.19	Y	0.00	0.00	665.60	78.65	0.00
fakeDB	2048.00	483.75	Y	0.00	0.00	1331.20	203.79	0.00
memhog-linux-sm	1024.00	376.21	N	0.00	0.00	0.00	18.47	620.38
memhog-linux-CL	1024.00	347.39	Y	652.21	652.21	652.21	55.10	57.07

Memory Hog VMs

MCTL: N - Balloon driver not active, tools probably not installed

Ballooning active

Swapped in the past but not actively swapping now

More swapping since balloon driver is not active

## Common Customer Questions



**Is the problem with my network or disk configuration?**

- ▶ **I think that my problems are with my network or disk bandwidth.**
- ▶ **Should I consider reconfiguring my network or perhaps it is my storage network .....**

## Disk and network capacity

- ▶ **Identifying network or disk problems**
  - Check bandwidth of each and compare with expectations
  - Check disk latency and compare with expectations
  
- ▶ **What do I do?**
  - Check requests per sampling interval and bytes *transferred/received* per sampling interval
  - For disks, check latencies
  - Compare with specs for the network or disk subsystems

## SAN Performance Rough Estimation

### ← From the perspective of a single VMware ESX, roughly:

Throughput (in MBps) = (Outstanding IOs \* Block size in KB) / latency in msec

Effective Link Bandwidth = ~80% of Real Bandwidth

Effective (2Gbps) = 200 MBps

Effective (4Gbps) = 400 MBps

### ← In a clustered Fiber-channel environment:

Throughput per host = (Effective Link Bandwidth / No. of IO intensive hosts)

### ← To achieve the effective link bandwidth:

Latency in msec  $\leq$  (Outstanding IOs \* Block size in KB) / Throughput per host

*Source: VMworld '07: IP42 "ESX Storage Performance – A Scalability Study"*

## Desired Latency Per Host

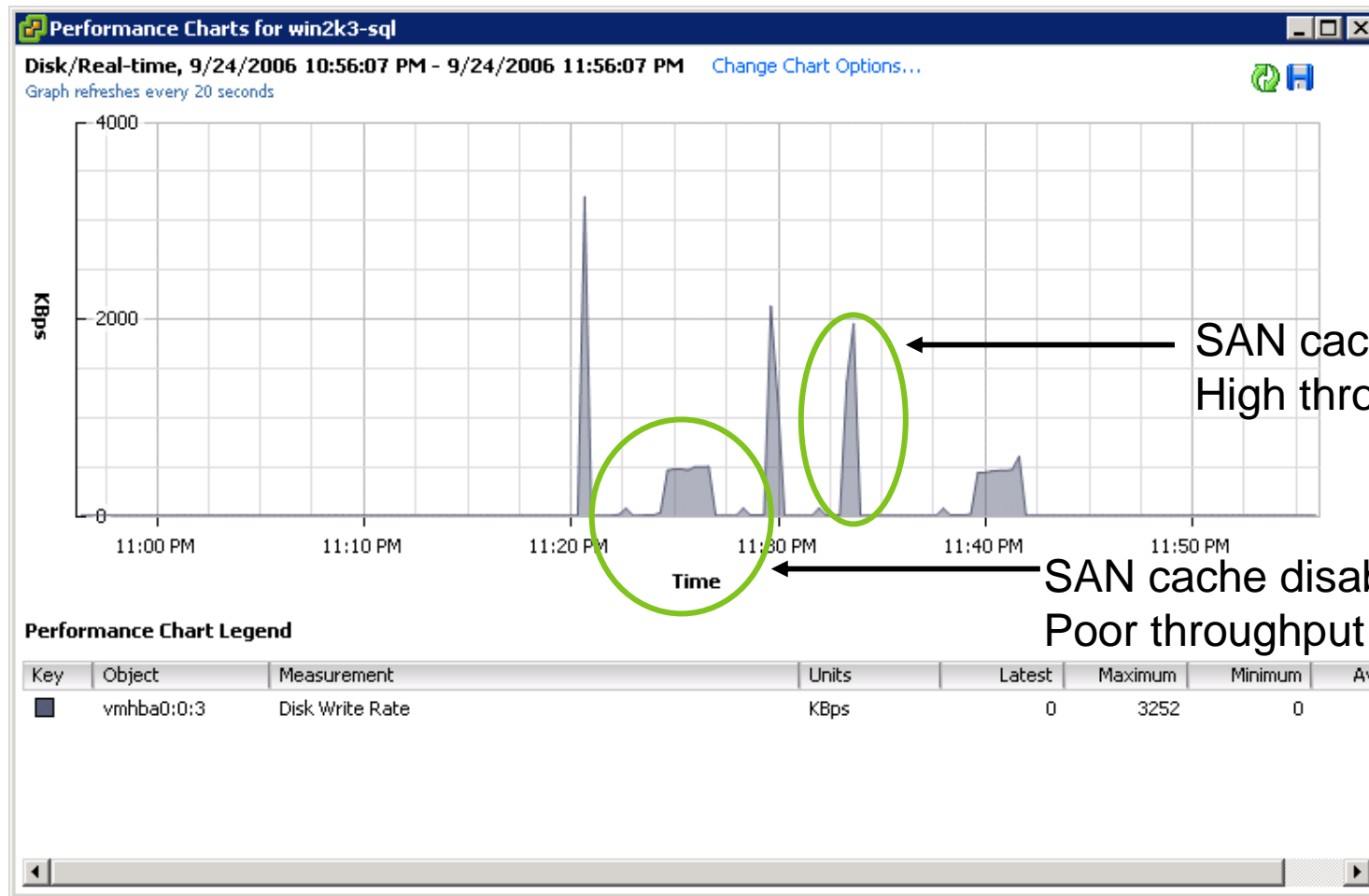
← **Desired Latency in msec  $\leq$  (Outstanding IOs \* Block size in KB) / Throughput per host**

← **Example:**

- > Number of Hosts = 64
- > Effective link bandwidth = 400 MBps
- > Throughput per host =  $400 / 64 = 6.25$  MBps
- > Desired latency =  $(32 * 32) / (6.25) = 163.84$  msec

Workload	Cached Sequential Read	Cached Sequential Write
Desired latency (msec)	163.84	163.84
Observed latency (msec)	~310	~163
Throughput drop ?	YES	NO
Throughput (MBps)	~270	~400

# Disk throughput



## Disk capacity – Looking at Disk latency

(screenshot of esxtop)

ADAPTR	CID	TID	LID	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	DAVG/cmd	KAVG/cmd	GA
vmhba0	-	-	-	1.18	0.00	1.18	0.00	0.01	13.07	0.02	
vmhba1	-	-	-	0.99	0.20	0.79	0.00	0.00	0.37	0.02	

Latency seems high

ADAPTR	CID	TID	LID	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	DAVG/cmd	KAVG/cmd	GA
vmhba0	-	-	-	2.76	0.00	2.76	0.00	0.01	2.42	0.02	
vmhba1	-	-	-	1.38	0.00	1.38	0.00	0.00	0.33	0.01	

After enabling the SAN cache, latency is much better

## Common Customer Questions



### **So many counters, so little time**

- ▶ **You said earlier that VMware exposes 150 counters**
- ▶ **Well, which ones do I care about?**
- ▶ **Which ones make sense to look at daily? Which ones will give me interesting trends that I should consider?**
- ▶ **Do I care about the rest?**

## Counters of interest

### ▶ If you are looking at real-time statistics ....

- **CPU:** *usage (% or MHz), used time, ready time, wait time*
- **Memory:** *consumed, active, swapused, swapin, swapout, vmmemctl*
- **Disk:** *diskReadLatency, diskWriteLatency, commands, commandsAborted, bytes transferred/received, disk bus resets*
- **Network:** *packets transmitted/received*

### ▶ Dig deeper if you see issues

- For example, on disks
  - *deviceLatency, kernellatency, queueLatency, totalLatency*
  - Disk bus resets may signal failing LUNs.

## Counters of interest

Counter Name	Description
cpu.usage.average	CPU usage (%)
cpu.used.summation	Used time (ms)
cpu.ready.summary	ready to run, no resources available (ms)
cpu.wait.summation	blocked waiting (e.g., for I/O) (ms)
mem.consumed.average	Machine pages taken by VM
mem.active.average	“working set” of VM
mem.swapused.average	“instantaneous” swapped memory for VM
mem.swapin.average	Cumulative swapped-in memory for VM
mem.swapout.average	Cumulative swapped-out memory for VM
mem.vmmemctl.average	Ballooned memory for VM

## Counters of interest

Counter Name	Description
disk.commands.summation	Disk commands issued
disk.usage.average	Disk Bandwidth consumed
disk.commandsAborted.summation	Disk commands aborted
disk.busResets.summation	SCSI bus resets
disk.deviceLatency.average	Latency at the device
disk.kernelLatency.average	Latency within the vmkernel
net.usage.average	Network bandwidth consumed
net.packetsRx.summation	Packets received in sample interval
net.packetsTx.summation	Packets transmitted in sample interval

## Tips and Tricks

- ▶ **Use view API to monitor inventory**
- ▶ **Use CSV format**
- ▶ **Go multi-threaded**
- ▶ **Statically specify metrics to collect**
- ▶ **Query over small time increments**
- ▶ **Choose correct stats levels**
- ▶ **Historical vs. real-time retrieval (To DB or not to DB)**
- ▶ **Watch your serialization and DB costs**
- ▶ **Optimize your metric gathering code**

## Tips and Tricks: Serialization and Database costs

### ► How much data are we sending?

- 4-way host, 2 NICs, 1 datastore  
QueryAvailablePerfMetrics → 173 metrics!
- 2-way VM, 1 NIC, 1 datastore  
QueryAvailablePerfMetrics → 99 metrics!
- Assume 4 chars per metric  
~700B per host, ~400B per VM
- Assume 100 hosts, 1000 VMs  
~460KB to get 1 data point
- For 12 data points (1 hour of 5-minute stats): **5.4MB**  
**Things add up, don't they**
- 5.4MB → serialization cost becomes significant

## Tips and Tricks: Serialization and Database costs

### ► Sample latency breakdown for a subset of stats

- Single query for a 24 hours of data from a host
- Total query: 1.75s
  - SSL handshake 180ms (~ fixed latency)
  - Server deserialization/transfer: 500ms (scales with # of points selected)
  - DB access 270ms (scales with dataset)
  - call to DB 100ms (~ fixed latency)
  - client deserialization/transfer: 600ms (scales with # of points selected)
- Bottom line:
  - serialization is important: pick metrics wisely
  - As DB grows, its latency becomes significant

(Tools used: wireshark, SQL profiler, logging in SDK code)

## Tips and Tricks: Query VC v/s Querying each host

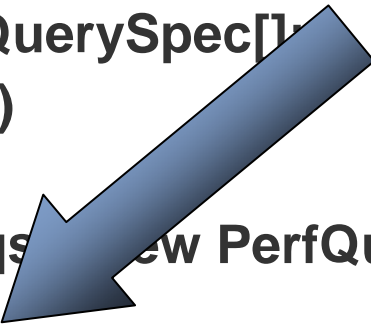
Threads	Query through VC(s)	Query directly to host(s)
1	251	242
2	131	153
4	81	77
6	60	70
8	52	48

- ▶ 64 hosts, 1233 powered-on VMs, real-time stats, VIPerl toolkit used
- ▶ Querying through VC can be ~ Querying through hosts  
(inventory monitoring easier with VC, though...consider views)
- ▶ Different client implementations may yield different results (# threads?)

## Tips and Tricks: Writing efficient code

### Code that will not scale

```
pqsArray = new PerfQuerySpec[1]; One element array
for (i = 0; i < 1000; i++ )
{
    PerfQuerySpec pqs = new PerfQuerySpec( ... );
    pqsArray[0] = pqs;
    PerfEntityMetricBase[ ] pemb =
        service.queryPerf(perfManager, pqsArray);
}
```



## Tips and Tricks: Writing efficient code

### Code that does it right

```
pqsArray = new PerfQuerySpec[];  
for (i = 0; i < 1000; i++ )  
{  
    PerfQuerySpec pqs = new PerfQuerySpec( ... );  
    pqsArray[i] = pqs;  
}  
PerfEntityMetricBase[ ] pemb =  
    service.queryPerf(perfManager, pqsArray);
```

### Remember:

- ▶ Collect only what you will use
- ▶ Use everything that you collect

# VMware Developer Center

<http://vmware.com/developer>

SDK, Toolkit Downloads, Sample Code, Forums, FAQs, Knowledge Base