Greenplum Database Performance on VMware vSphere 5.5

Performance Study

TECHNICAL WHITEPAPER

vmware®

Table of Contents

Introduction

The rapid growth of today's data volumes and sources exceed that which a single, large computer could handle alone. Parallel processing across multiple computers has become a viable design for large-scale analytic processing and data warehousing. Greenplum's shared-nothing massively parallel processing (MPP) architecture is designed for Big Data Analytics with linear scaling and high availability. Paired with the VMware vSphere infrastructure, analytical workloads in a virtualized Greenplum database environment could be improved by running multiple Greenplum segment virtual machines on each host in a Greenplum cluster.

In this paper, we present the results of experiments which demonstrate the Pivotal Greenplum database performs well on vSphere 5.5, providing a reduction of 11% in analytic process time when compared to the same hardware configuration in a physical environment.

This paper addresses these performance characteristics:

- The performance implications of running Greenplum segment servers in a virtual environment versus on a physical environment.
- The performance impact of scaling out with multiple Greenplum segment server virtual machines on each host.

Performance best practice recommendations for the virtualized Greenplum database environment on vSphere 5.5 are also provided.

Experimental Configuration and Methodology

The performance studies were done in VMware's internal labs. The purpose of the tests was to measure, analyze, and understand the performance of Greenplum database in the vSphere 5.5 virtual environment. In the following sections, we describe in detail the test bed configuration used for the experiments and the test tools. Finally, we present a description of the experiments.

Test Bed Configuration

In our test configuration, we used six HP ProLiant DL380e Gen8 servers for the Greenplum database cluster. Each server was configured with two Intel Xeon E5-2470 processors and 96GB of physical memory. The virtual environment used VMware vSphere 5.5 to virtualize the Greenplum database cluster environment. The Greenplum database was installed on Red Hat Enterprise Linux (RHEL) 6.2.

Here is the detailed test bed configuration:

COMPONENT	DETAILS
Server	6 HP ProLiant DL380e Gen8 servers
Processors	2 Intel Xeon E5-2470 processors @ 2.3GHz, with hyper-threading and HP Dynamic Power Saving Mode
Memory	96GB DIMMs 1333MHz
Storage controller	HP H220 Host Bus Adapter
Local storage disks	2X 1TB STAT 7,200 RPM disks for VM storage and Native OS install
	16X 500GB SATA 7,200 RPM disks for Greenplum database

Local network adapter	HP Ethernet 10Gb 2-port 530SFP+ Adapter
Operating system	Red Hat Enterprise Linux (RHEL) 6.2 64-bit Detail in Appendix A: Operating System Configuration
Greenplum database	Greenplum database 4.2.2.0
Virtualization software	VMware vSphere 5.5

Table 1. Test bed configuration

Figure 1 below shows the physical layout of our Greenplum cluster testbed. Each of the six Greenplum cluster hosts is connected by one 1 gigabit (Gb) connection to a Summit X350 1 gigabit Ethernet (GbE) switch to outside of the cluster and two 10Gb connections to an Arista 7050S 10GbE switch as the gNet software interconnect between Greenplum hosts.

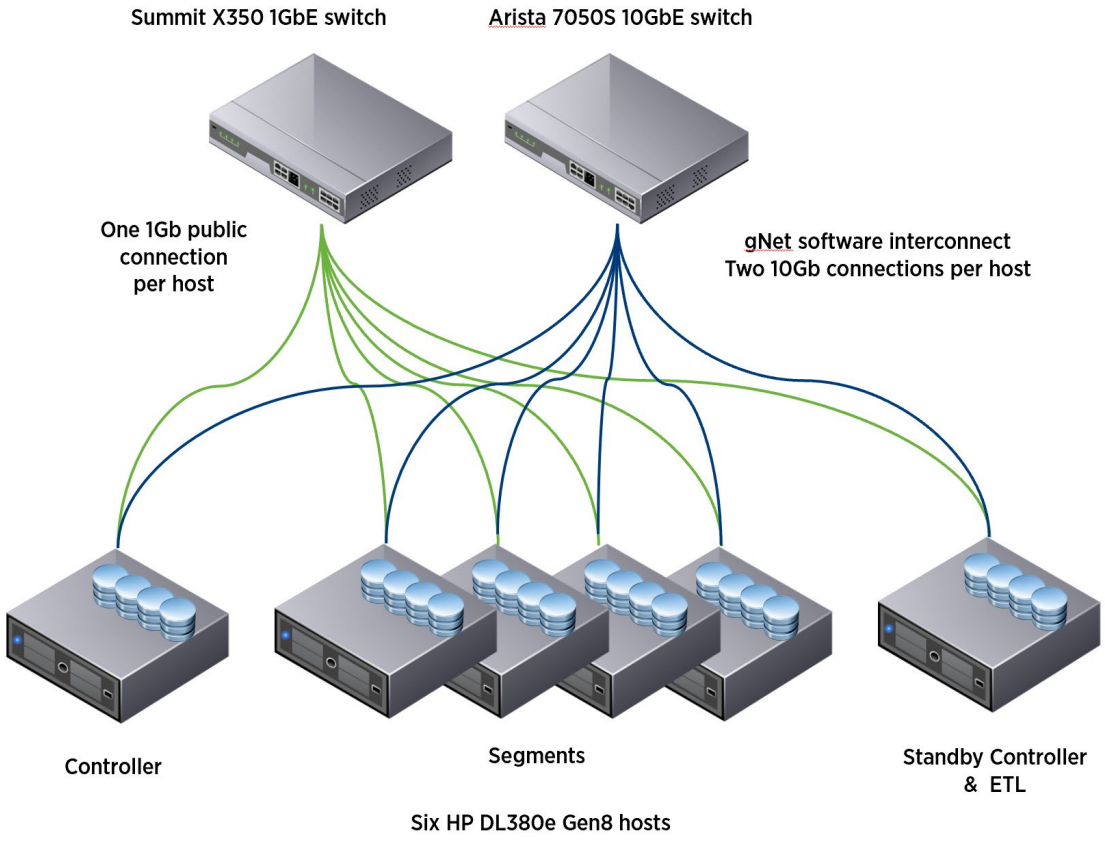

Figure 1. Greenplum cluster test-bed layout

All virtual machines ware configured with one 1Gb public connection and two 10Gb for the high speed Greenplum gNet software interconnect and used a single queue on the VMXNET3 virtual network driver. Local data disks were passed through to the virtual machines using physical raw device mappings (RDM) and a single partition was created on each disk. The guest RHEL operating system used the xfs high-performance file system with a deadline I/O scheduler for all data disks.

Below is a brief description of Greenplum components in the Greenplum cluster testbed layout.

For more details about Greenplum installation and technology, please refer the Greenplum database installation and administrator guides. Also see the Greenplum Database: Critical Mass Innovation architecture white paper at http://www.emc.com/collateral/hardware/white-papers/h8072-greenplum-database-wp.pdf.

The **controller** is the entry point to the Greenplum Database system. The controller does the work of authenticating client connections, processing the incoming SQL commands, distributing the workload between the segments, coordinating the results returned by each of the segments, and presenting the final results to the client program. The controller maintains the system catalog, but not the user data, which resides only on the segments.

The **segments** are where the data is stored and where the majority of query processing takes place. The Greenplum database shared-nothing architecture separates the physical storage of data into small units on individual segment servers. The segment servers are able to process every query in a fully parallel manner, use all disk connections simultaneously, and efficiently flow data between segments as query plans dictate because shared-nothing databases automatically distribute data and make query workloads parallel across all available hardware. Since the massively parallel data processing occurs in segment servers, they are resource-hungry components which require the appropriate amount of CPU and memory with high bandwidth of storage and network.

The **ETL** (extract-transform-load) is for data loading purposes. It runs Greenplum parallel file servers (gpfdist) to parallel streaming input data that simultaneously flow to all Greenplum database segments.

The **Standby controller** is a backup or mirror of the controller instance. A backup controller host serves as a warm standby in case the primary controller host becomes unavailable.

The **gNet software interconnect** is the networking layer of the Greenplum database. When a user connects to a database and issues a query, processes are created on each of the segments to handle the work of that query. The interconnect refers to the inter-process communication between the segments, as well as the network infrastructure on which this communication relies.

Test and Measurement Tools

Three types of common heavy workloads are run in any analytic processing system: an analytical SQL (FullScanSQL) scanning all of the data in a fact table, an analytical SQL (PartialScanSQL) scanning a subset of data in a fact table, and a massive amount of data loading into a fact table. The size of 6 terabytes (TB) of 5 year retail ordering data was generated in the Greenplum database for our performance study. The order_lineitems table was the fact table accessed by the three performance test cases. The order_lineitems table and the three performance test cases are described in detail.

The order_lineitems table stores each line item detail data in orders, one row per line item. The table was partitioned monthly for the first four years and weekly for the recent year, and contained 8.4 billion rows of data.

The performance test cases:

- **FullScanSQL:** A report of a list of customers with the date of their first and last electronic orders. The FullScanSQL executed a large amount of sequential read operations on disks by scanning the 8.4 billion of rows in the order_lineitems table; the sorting and aggregating took a small amount of filtered data in memory.

The FullScanSQL statement:

```
SELECT CUSTOMER_ID
,      PRODUCT_CATEGORY_NAME
,      FIRST_ORDER_DATETIME
,      FIRST_ORDER_ID
,      LAST_ORDER_DATETIME
,      LAST_ORDER_ID
FROM   (
SELECT CUSTOMER_ID
,      PRODUCT_CATEGORY_NAME
,      FIRST_VALUE(ORDER_DATETIME) OVER (PARTITION BY CUSTOMER_ID,
PRODUCT_CATEGORY_ID ORDER BY ORDER_DATETIME ASC) AS FIRST_ORDER_DATETIME
,      FIRST_VALUE(ORDER_ID) OVER (PARTITION BY CUSTOMER_ID,
PRODUCT_CATEGORY_ID ORDER BY ORDER_DATETIME ASC) AS FIRST_ORDER_ID
,      LAST_VALUE(ORDER_DATETIME) OVER (PARTITION BY CUSTOMER_ID,
PRODUCT_CATEGORY_ID ORDER BY ORDER_DATETIME ASC) AS LAST_ORDER_DATETIME
,      LAST_VALUE(ORDER_ID) OVER (PARTITION BY CUSTOMER_ID,
PRODUCT_CATEGORY_ID ORDER BY ORDER_DATETIME ASC) AS LAST_ORDER_ID
,      ROW_NUMBER() OVER (PARTITION BY CUSTOMER_ID, PRODUCT_CATEGORY_ID
ORDER BY NULL) AS RN
FROM   RETAIL_DEMO.ORDER_LINEITEMS
WHERE  PRODUCT_ID IN (20,25,29,50,38)
) BASE
WHERE  BASE.RN = 1
```

- **PartialScanSQL:** A report of the average number of items bought by all customers in the year of 2010, sorted by month. The PartialScanSQL performed a mixed of read and write disk operations simultaneously. The SQL was writing a sizeable amount of intermediate data to disks as a temporary storage of the sorting process while scanning 1.7 billion rows in the 2010 year partitions of the order_lineitems table.

  The PartialScanSQL statement:

```
SELECT ROUND(AVG(ITEM_COUNT), 2) AS AVERAGE_ITEM_COUNT
FROM   (SELECT CUSTOMER_ID,
               TO_CHAR(ORDER_DATETIME, 'YYYY/MM') AS ORDER_MONTH,
               SUM(ITEM_QUANTITY)                 AS ITEM_COUNT
        FROM  ORDER_LINEITEMS
        WHERE  ORDER_DATETIME BETWEEN TIMESTAMP '2010-01-01' AND DATE
                                      '2010-12-31'
        GROUP  BY CUSTOMER_ID,
                  TO_CHAR(ORDER_DATETIME, 'YYYY/MM')
        ORDER  BY TO_CHAR(ORDER_DATETIME, 'YYYY/MM'))LINEITEMS;
```

- **Data Loading:** Load 1TB, or 2.3 billion rows, of order_lineitems data from 16 of Greenplum parallel file servers on the ETL server. Each Greenplum parallel file server streamed 62.5GB (1TB divided by 16 parallel file servers) of data into the Greenplum database and generated a large amount of disk write on Greenplum data nodes with heavy networking traffic.

The performance of the workloads is measured in the response time reported by the controller server when the process was complete. The timing feature in the Greenplum database is enabled by setting "\timing" in the psqlrc file.

Test Cases and Test Method

The burden of workload was in the Greenplum segment servers where the data is stored and where the majority of query processing takes place. Our performance experiments were done on the segment servers and kept the same for any other Greenplum components. The controller, standby controller, and ETL server were installed in

VMs running on their own host for our performance experiments, so apples-to-apples comparable configurations were maintained. The configurations of controller, standby controller, and ETL configuration:

The Controller host contains only the controller VM.

- **Controller VM:** 8vCPUs, 46GB memory

The Standby Controller host contains the Standby Controller VM and an ETL VM.

- **Standby Controller VM:** 8vCPUs, 46GB memory
- **ETL server VM:** 8vCPUs, 46GB memory, 16 data disks

We had two primary objectives in performing these experiments: 1) Compare the performance of physical Greenplum installations with their virtual counterparts, and 2) Understand the performance of scaling out Greenplum. For this, the following experiments were conducted.

Each of 4 segment hosts was configured

- **Physical:** A Greenplum segment server operated 16 segments in each of 4 hosts
- **Virtual:** Number of segment VMs scaled out per host with 1, 2, and 4 VMs
  - 1VM per host: 16 vCPUs, 92GB memory, 16 data disks
  - 2VMs per host: 8 vCPUs, 46GB memory, 8 data disks
  - 4VMs per host: 4vCPUs, 23GB memory, 4 data disks

The total of number Greenplum segments was 64 (32 primaries + 32 mirrors) for all of the test cases. Each host was running 16 segments (8 primaries + 8 mirrors). The primary and mirror segments were evenly divided into each VM in a host. For example, of the 2VMs per host case, each VM was running 8 segments (4 primaries + 4 mirrors).

Experimental Results

Experiments are presented to show the performance of a single virtual machine and multiple virtual machines per host in a Greenplum database environment. Single virtual machine performance provides insight into the characteristics of the workload, and multiple virtual machines shows vShpere's capabilities and feasibility for a Greenplum deployment.

Performance Comparison: Physical to Virtual

For an apples-to-apples comparison, identical software stacks were configured in both the physical and virtual environments. The results are discussed next.

Response Time

The results of the virtual Greenplum environments are very similar to the comparable physical environment. The response times of all the workloads in the virtual environment are within 6% of those measured in the physical environment.

With a large amount of disk read or write workloads in the FullScanSQL and Data Loading workloads, the overhead of virtualizing was only between a 5% to 6% range in the response time. For the overhead of the PartialScanSQL workload with mixed disk read and write: this case was reduced to 3% as the increased latency of mixed I/O in physical disks had the same effect on both physical and virtual environments.

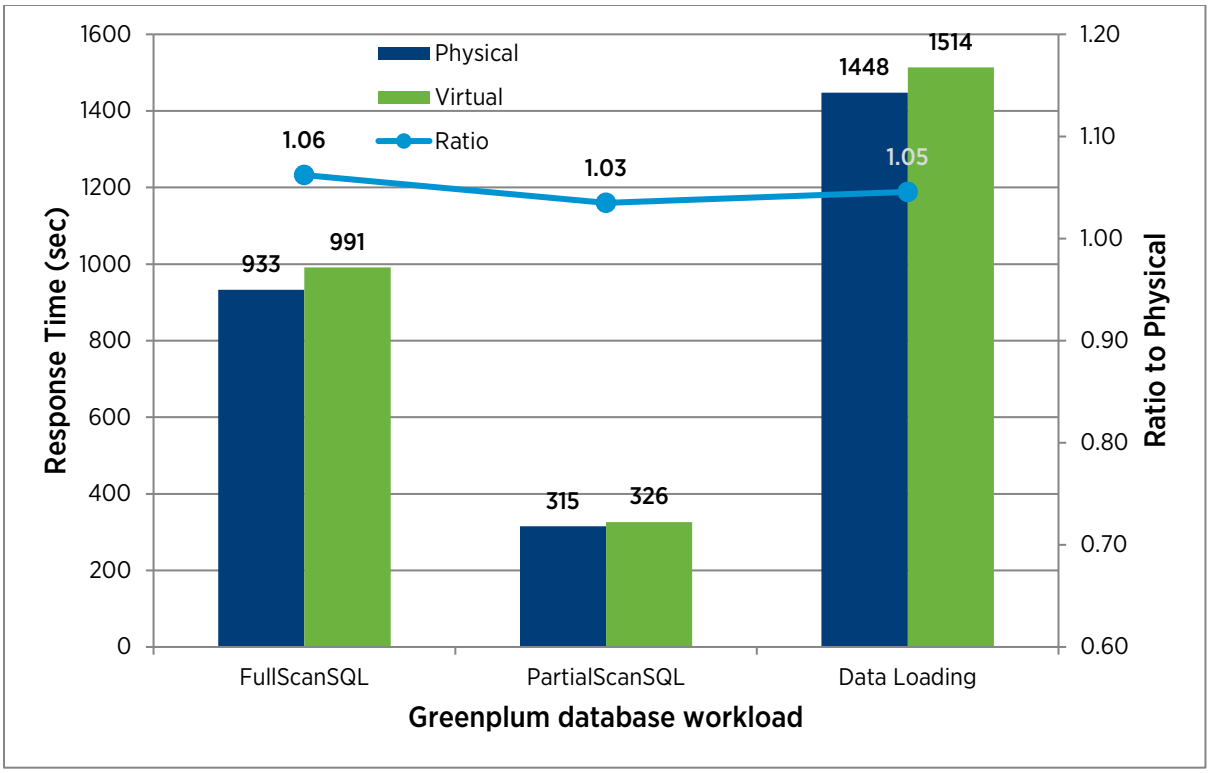

Figure 2. Response time for physical and virtual

Performance of Multiple Greenplum Virtual Machines

We now consider how effectively vSphere scales multiple Greenplum VMs running on each host for the same 6TB size of Greenplum database. The multiple Greenplum VMs test uses multiple two and four VMs per host by evenly partitioning the CPU, memory, and disks of the host into each VM.

Response Time

Figure 3 below depicts the response time results of 1, 2, and 4 VMs per host cases.

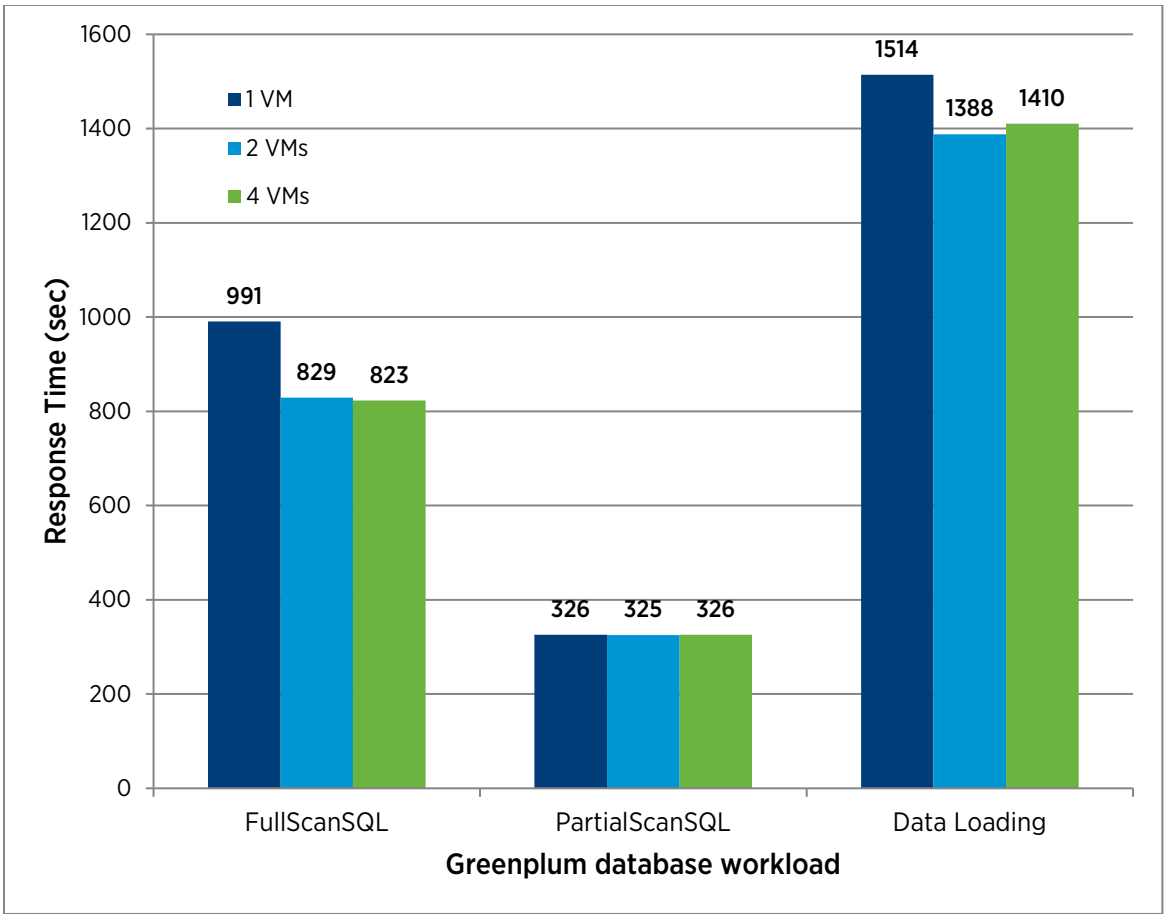

Figure 3. Response time for 1 VM, 2 VMs, and 4 VMs per host

The response time results of the 2 VMs case were significantly reduced 16% (162 seconds, from 991 to 829) in the FullScanSQL workload and 8% (126 seconds, from 1514 to 1388) in the Data Loading workload when compared with the 1 VM case. These reductions in response time of the 2 VMs case also present 11% in the FullScanSQL and 4% in the Data Loading workloads faster than the physical environment.

Both the FullScanSQL and Data Loading workloads used the available free memory as disk cache while they were processing a large amount of sequential disk read or disk write. The latency of memory access became a key performance element as prominent as the speed of the data stream to and from the storage. In the 2 VMs and 4 VMs per host cases, the NUMA-aware resource management in vSphere placed each smaller Greenplum database VM in a NUMA node to benefit lower latency in local NUMA memory access. To understand how much performance benefits from the local memory access, we conducted the Data Loading workload with memory node interleaving (disabled NUMA) setting in BIOS for the both of 1 VM and 2 VMS cases. The memory of the VM was interleaved in an equal amount into 2 memory nodes. The vCPU of VMs had affinity as in the NUMA case. The 16 vCPUs of the 1 VM case were pinned as follows: vCPUs 0-7 were pinned to the first processor and vCPUs 8-15 were pinned to the second processor. For the 2 VMs case, each VM's 8 vCPUs were pinned to its own processor. Figure 4 shows the results in ratio to the 1 VM with NUMA baseline case.

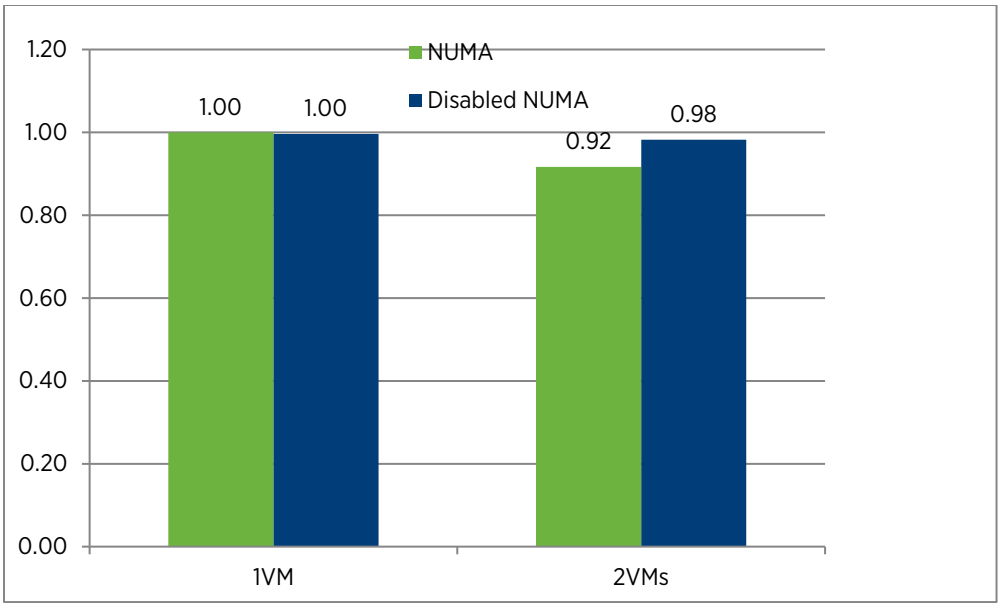

Figure 4. Ratio of response time for 1TB data loading to the 1 VM with NUMA setting, Number of VMs is per host, Lower is better

The response time of the 1 VM case was the same between the NUMA and disabled NUMA cases, which indicates the workload was randomly accessing all of its memory that resided in both local and remote memory. The response time of the 2VMs configuration was improved only by 2% in the disabled NUMA case and the improvement jumped to 8% with NUMA enabled. This implies the lower latency in local memory access provides a major part of the performance improvement.

For the PartialScanSQL workload, the response time of all of the cases displayed a very small amount of change. The latency of physical disk I/O from simultaneous read/write requests was dominant throughout the time of the workload. The response time is not impacted by a different number of VM configurations.

Best Practices

This section describes some recommendations for improving virtualized Greenplum database performance:

- Size Greenplum virtual machines to fit on one NUMA node in a NUMA system. The NUMA-aware resource management in vSphere achieves fast local memory accesses with much lower latency.
- Configure a single partition per data disk inside the VM to reduce randomization of sequential I/O streams.
- Assign data disks from different enclosures to virtual machines to avoid reaching the bandwidth limit from the I/O channel, if this applies to your hardware.
- Make sure the system has enough memory to avoid ESX host swapping; otherwise, performance in the virtual machines is significantly reduced. Pay attention to the balloon driver's inflation, which may induce guest swapping. Detailed information is available at *Understanding Memory Management in VMware vSphere 5* at http://www.vmware.com/resources/techresources/10206.
- Avoid multi-queue on the VMXNET3 virtual network device. A single queue utilizes less CPU cost and has better efficiency for most cases of the Greenplum workload.
- Follow the performance recommendations in the following guides: *Greenplum Installation Guide*, *Greenplum Database Installation Guide*, *Greenplum Database Administrator Guide*.

Conclusion

The results in this paper show that a Greenplum database running on vSphere 5.5 achieves very good performance. It shows that the performance of a single virtual machine per host in a virtualized Greenplum database cluster is near to that of a physical deployment with very little virtualization overhead.

The performance of a virtualized Greenplum database environment can be improved significantly by deploying multiple Greenplum database virtual machines per host that allow smaller virtual machines to fit into a NUMA node to gain lower latency in local memory access. As the result of this configuration, the response time of analytical workloads in a virtualized environment could reach up to 11% faster than a physical deployment.

A multiple Greenplum database virtual machine deployment not only is a simple way to leverage the hardware architecture of your system to achieve optimal performance, but also limits the impact of failed hardware and software maintenance inside an individual Greenplum database virtual machine.

Appendix A: Operating System Configuration

- Operating System: Linux RHEL 6.2 x86_64
- Kernel parameters in `/etc/security/limits.conf`

```
soft nofile 65536
hard nofile 65536
soft nproc 131072
hard nproc 131072
```

- Kernel parameters in `/etc/sysctl.conf`

```
xfs_mount_options = rw,noatime,inode64,allocsize=16m
kernel.shmmax = 500000000
kernel.shmmni = 4096
kernel.shmall = 4000000000
kernel.sem = 250 512000 100 2048
kernel.sysrq = 1
kernel.core_uses_pid = 1
kernel.msgmnb = 65536
kernel.msgmax = 65536
kernel.msgmni = 2048
net.ipv4.tcp_syncookies = 1
net.ipv4.ip_forward = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.tcp_tw_recycle = 1
net.ipv4.tcp_max_syn_backlog = 4096
net.ipv4.conf.all.arp_filter = 1
net.ipv4.ip_local_port_range = 1025 65535
net.core.netdev_max_backlog = 10000
vm.overcommit_memory = 2
```

- Data disk
  - Single Linux partition per hard disk
  - Partition formatted: xfs
  - I/O Scheduler: deadline
  - read-ahead (blockdev) value: 16384

Synchronize the system clocks: NTP (Network Time Protocol)

About the Author

Vincent Lin is a Performance Engineer at VMware. In this role, his primary focus is to evaluate and help improve the performance of VMware products in better supporting key enterprise applications. Prior to VMware, Vincent was a principal performance engineer at Oracle. Vincent received a Controller of Science degree from the University of West Florida.

vmware

VMware, Inc. **3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com**