Performance Analysis
March 2010

Virtualization Networking with VMware® ESX 4.0 on IBM® System x® Servers

Steve Worley
IBM Corporation

Abstract

This paper discusses the performance of various networking technologies as they relate to virtualization on IBM System x® servers running VMware® ESX 4.0. The trend toward virtualizing on larger and more powerful servers puts pressure on the I/O throughput needs of the virtualized environment. Virtualization provides challenges from a networking standpoint because the I/O from numerous Virtual Machines (VMs) is aggregated over the physical network interface cards (NICs) in the ESX host.

Introduction

As virtualization continues to gain market share in Information Technology shops and new virtualization technologies are developed, virtualization I/O performance is a key consideration. The VMware VMmark™ benchmark can be used to evaluate and compare various technologies that may be beneficial to users of virtualization technology.

Test Bed Configuration

VMware's VMmark benchmark was used to compare the performance of various networking configurations. A total of 17 VMmark tiles were used. More information on the VMmark benchmark can be found at http://www.vmmark.com.

The configuration for the system under test was an IBM System x3650 M2 server with two Intel® Xeon® X5570 (2.93GHz) processors with 96GB (12 x 8GB DIMMs) of memory. The baseline run used two QLogic QLA2462 4Gb Fibre Channel host bus adapters connected to an IBM System Storage™ DS4800 Fibre Channel SAN controller for the disk I/O. The baseline network adapters were Broadcom NetXtreme II BCM5709 1000-BaseT Dual Port. The complete configuration used can be found in the VMmark results for the IBM System x3650 M2 at: http://www.vmware.com/files/pdf/vmmark/VMmark-IBM-2009-04-27-3650M2.pdf

Overview of VMDq

Virtual Machine Device Queue (VMDq) is a hardware technology developed by Intel that offloads network I/O management processing from the hypervisor to the network adapter. Queues and intelligence in the hardware free processor cycles for application work. In a virtual environment the hypervisor is responsible for managing network I/O processing. As more virtual machines (VMs) are added and data traffic increases through the platform, the hypervisor requires more CPU cycles to process data packets and move them to the associated VM. VMDq thus improves network I/O performance by reducing the processing burden on the hypervisor.

I/O containers such as VMware's ESX I/O architecture provide abstractions of I/O subsytems so that VMs can process I/O requests generated by guest software running in a VM. Two major I/O subsystems are the storage and networking subsystems. The focus of the VMDq architecture is to accelerate the network subsystem.

A typical I/O container network subsystem is made up of several software components. The components include the physical network NIC driver, a virtual switch, and virtual network devices. The performance of an I/O container can be improved by utilizing a VMDq device. A VMDq device does this by offloading some of the most commonly repeated I/O container tasks. The VMDq architecture allows a VMDq device to offload the I/O container's virtual switch tasks such as packet sorting, moving data from the I/O container to the VM, routing packets to the most advantageous CPU core for processing, and supporting transmit fairness.

In VMware's ESX, the software to interface with the VMDq-supported hardware is called NetQueue. NetQueue provides a VMDq network adapter with multiple receive queues that allow

data-interrupt processing to be associated with individual CPU cores and VMs. This improves receive-side networking performance. Receive queues can be assigned to each virtual NIC and mapped to guest memory to avoid a copy. Interrupts can be moved to idle or optimal processor cores.

When data packets arrive at the network adapter, a Layer 2 sorter determines which VM each packet is destined for, based on MAC addresses and VLAN tags. The sorter then places the packet in a receive queue assigned to the particular VM. The hypervisor's virtual switch merely reroutes the packets to the VM instead of performing the processor-intensive work of sorting data.

When transmitting packets from the VMs to the adapter, the hypervisor places the transmit data packets in their proper queues. To prevent blocking and to ensure that each queue is fairly serviced, the network controller transmits queued packets to the wire in a round-robin manner. This guarantees some measure of Quality of Service (QOS) to the VMs.

Additional information about VMDq can be found at:
http://www.intel.com/network/connectivity/vtc_vmdq.htm

Requirements

One of the following adapters is required for using VMDq with IBM System x servers:

- Intel 82599EB 10 Gigabit Ethernet Controller
- Intel 82575 Gigabit Ethernet Controller

The Intel 82599EB 10 Gigabit Ethernet Controller was used for the analysis in this paper.

Configuration of VMDq

1. Enable NetQueue in VMkernel using VI3 Client.

    a. Choose **Configuration > Advanced Settings > VMkernel**.

    b. Select the checkbox for **VMkernel.Boot.netNetqueueEnabled**.

2. Enable the ixgbe module in the service console of the ESX Server host:

    # esxcfg-module -e ixgbe

3. Set the required NetQueue options for the ixgbe module:

    - MSI-X enables the Ethernet controller to direct interrupt messages to multiple processor cores. This feature is required for NetQueue to function with VMDq. In the examples below, the parameter InterruptType with a value of 2 specifies MSI-X.

    - A value for VMDQ must exist to indicate the number of receive queues. A value of 16 for VMDQ sets the number of receive queues to the maximum. The Intel 82598 10 Gigabit Ethernet Controller provides 32 transmit queues and 64 receive queues per port, which can be mapped to a maximum of 16 processor cores. The range of values for the VMDQ parameter is 1 to 16.

For a single port and the maximum number of receive queues, use the command:

esxcfg-module -s "InterruptType=2 VMDQ=16" ixgbe

For two ports, add the values in a comma-separated list for each parameter as shown in the following example:

esxcfg-module -s "InterruptType=2,2 VMDQ=16,16" ixgbe

4. Reboot the ESX Server system.

To verify that VMDq has been successfully enabled, follow these steps:

1. Verify NetQueue has been enabled:

   # cat /etc/vmware/esx.conf

   Confirm the following line has been added into the file:

   /vmkernel/netNetqueueEnabled = "TRUE"

2. Verify the options configured for the ixgbe module:

   # esxcfg-module -g ixgbe

   The output should be similar to the following example:

   ixgbe enabled = 1 options = 'InterruptType=2,2 VMDQ=16,16'

   The enabled value should be equal to 1, which indicates the ixgbe module is enabled. InterruptType should be equal to 2 and include an entry for each port. (This is a 2-port example.) VMDQ should be 16 to enable the maximum number of receive queues and should also have an entry for each port.

3. Query which ports have loaded the driver using esxcfg-nics -l. Then query the statistics using ethtool. If VMDq is enabled, statistics for multiple receive queues will be shown (rx_queue_0 through rx_queue_15 in the example below).

   # esxcfg-nics –l

   # ethtool -S vmnic2

When using more than one port in this analysis, the VMmark webservers were on one port while the rest of the VMs were on the second port. The webservers are the most network-intensive part of each of the VMmark tiles, so dedicating a port to them provides a performance advantage. Configuring 16 queues would provide a separate queue for almost all of the webservers.

VMDq Results

Results for the 82599EB 10 Gigabit Ethernet Controller are shown in Figure 1. NetQueue is the software in ESX 4.0 that allows the functioning of VMDq. These results were obtained using the ixgbe driver 100.2.0.38.2.3-1.0.4.164009.

The first comparison was running VMDq on a single port of the two port adatper. With NetQueue disabled, the results were similar to the baseline results. With NetQueue enabled, the results on a 17-tile Vmmark benchmark showed no better results than when NetQueue was disabled.

Next a second 10G port was utilized in order to confirm that there were no bandwidth limitations. The results with NetQueue disabled and NetQueue enabled did not show improved results over the baseline results. In addition to the configuration with 16 queues, additional benchmark runs with 8 and 4 queues where completed. Neither showed improvement over the baseline results. While VMDq did not show any benefit at low levels of VMmark throughput, it is expected to show some benefit at higher throughput levels. The network throughput running VMmark with 17 tiles was approximately 700 Mbps. As a comparison Intel results with VMDq use throughput levels between 4000 Mbps and 10,000 Mbps. Intel results can be found at: http://download.intel.com/network/connectivity/products/whitepapers/321968.pdf

Configuration	VMmark	% Difference
Baseline - Published Result	23.89	
VMDq – 1 port - NetQueue Disabled	23.52	-0.01549
VMDq – 1 port - NetQueue Enabled (16 Queues)	23.17	-0.03014
VMDq – 1 port - NetQueue Enabled (16 Queues)	23.40	-0.02051
VMDq – 2 port - NetQueue Disabled	23.08	-0.03391
VMDq – 2 port - NetQueue Disabled	23.57	-0.01339
VMDq – 2 port - NetQueue Enabled (16 Queues)	22.78	-0.04646
VMDq – 2 port - NetQueue Enabled (16 Queues)	23.09	-0.03349
VMDq – 2 port - NetQueue Enabled (8 Queues)	23.02	-0.03642
VMDq – 2 port - NetQueue Enabled (4 Queues)	22.86	-0.04311

Figure 1: VMmark results using Intel VMDq

Overview of Fibre Channel over Ethernet – FCoE

Fibre Channel over Ethernet (FCoE) is an encapsulation of Fibre Channel frames over an Ethernet network. The encapsulation allows Fibre Channel to use 10 Gigabit Ethernet networks while still preserving the Fibre Channel protocol.

Since classical Ethernet has no flow control, FCoE requires changes to the Ethernet standard to support a flow control mechanism. FCoE also requires Jumbo Frame support because the Fibre Channel payload is 2,112 bytes and cannot be fragmented to 1,500-byte Ethernet frames.

Because of the encapsulation and the limitation of flow control, it is not expected that performance of a 10G Ethernet and Fibre Channel VMmark configuration will benefit from a converged network.

Due to difficulties in configuring a Brocade 8000 FCoE switch to match the Cisco Catalyst 6500 switch VLAN configuration, results are not presented for FCoE. It is expected that FCoE would not be beneficial for the particular VMmark configuration we were looking at. FCoE could be an area for future investigation once improvements are made to the switch software to improve VLAN compatibility issues.

SR-IOV – Is it ready for Prime Time?

Single Root I/O Virtualization (SR-IOV) is another technology that has the potential to benefit virtualization network performance. SR-IOV defines extensions to the PCI Express (PCIe) specification suite to enable multiple guest operating systems to directly access subset portions of physical I/O resources. An SR-IOV device presents single or multiple Physical Functions (PFs), which are standard PCIe functions. Each PF can have Virtual Functions (VFs) associated with it. The VFs surface as PCI devices that are backed on the physical PCI device by resources such as queues and register sets. Virtual Functions give the same performance benefit of assigning a physical PCI device to a guest while eliminating the need to have a server filled with many physical PCI devices. Since the guest OS is effectively driving the hardware directly, the I/O performance is on par with native performance. SR-IOV is expected to have improved performance over emulated or para-virtual I/O devices.

Currently VMware has not provided support for SR-IOV on 10G devices in their ESX 4.0 virtualization engine.

SR-IOV is supported on Red Hat Enterprise Linux® 5.4 for the Intel 82575 Gigabit Ethernet Controller.

Conclusion

There are many methodologies being used to improve the performance of virtualization I/O technologies. Based on the data presented above, most of the I/O technologies have not yet proved to be a benefit in real-world customer scenarios that can be simulated with VMmark. While VMDq did not show any benefit at low levels of VMmark throughput, it is expected to show some benefit at higher throughput levels.

IBM Systems and Technology Group
3039 Cornwallis Road
Research Triangle Park, NC 27709

Produced in the USA
March 2010