Networking Performance

VMware® ESX Server 3.5

There is a general belief that due to the extra layer introduced by virtualization code, networking performance in a virtualized environment cannot match the performance in a native environment. In this paper, we present the following results:

- For most network configurations, virtual machines running on ESX Server 3.5 achieve the same throughput as native machines.
- In 32-bit and 64-bit versions of Microsoft® Windows Server® 2003 and Red Hat® Enterprise Linux U4, VMware's near-native performance is not limited to an operating system or architecture type.
- Virtualization using ESX Server does not introduce unreasonable latencies in the network.

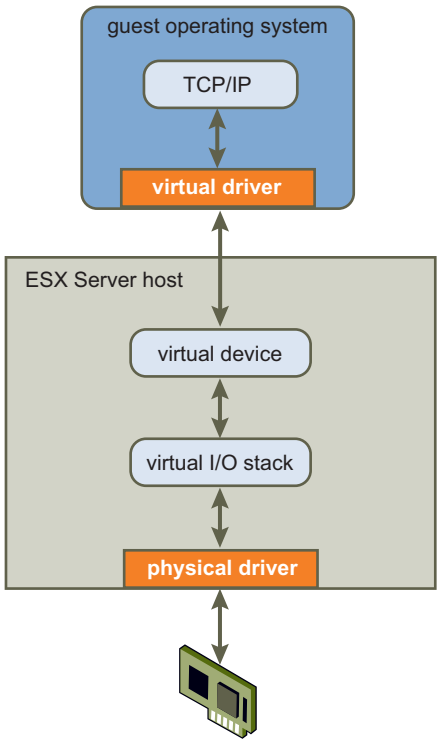
This study covers the following topics:

Network Virtualization in ESX Server 3.5

Packets that are sent by a guest application pass through multiple layers before they go on to the wire. The message from the application is first processed by TCP/IP in the guest operating system. After the required headers have been added to the packet, it is sent to the device driver in the virtual machine. Depending on the virtual device being presented by ESX Server, the device driver can be an unmodified Intel® e1000, AMD® PCNet32 (vlance), or a virtualization-aware vmxnet driver from VMware.

Once the packet has been received by the virtual device, it is passed on to the virtual I/O stack for additional processing, where the destination of the packet is also determined. If the packet is destined for another virtual machine connected to the same virtual switch, it is forwarded to that virtual machine and not sent on the wire. All other packets are sent to the physical NIC's device driver in the VMkernel to be sent out on the physical network as shown in Figure 1.

Figure 1. ESX Server 3.5 Network Stack



Performance Enhancements in ESX Server 3.5

Enhancements, including support for Jumbo Frames and TCP Segmentation Offload (TSO) for vmxnet devices, have been integrated into the networking code of ESX Server 3.5. Jumbo Frames are large Ethernet frames (up to 9000 bytes) and are bigger than the standard Ethernet frames that have a Maximum Transfer Unit (MTU) of 1500 bytes. Guest operating systems that use Jumbo Frames need fewer packets to transfer large amounts of data and can achieve higher throughputs and lower CPU utilization than guests that use a standard MTU size packet.

TSO, widely supported by today's network cards, allows the expensive task of segmenting large TCP packets of up to 64KB to be offloaded on to the hardware. ESX Server 3.5 allows guests to use TSO even when the underlying hardware does not have TSO capabilities. Because the guest operating system can now send packets larger than the MTU to the ESX Server, the processing overheads on the transmit path are reduced. The direct result of TSO is visible in our virtual machine-virtual machine results, which have improved between ESX Server 3.0.2 and ESX Server 3.5.0.

For the experiments presented in this paper, standard MTU sizes were used. Jumbo Frames were disabled, and TSO was enabled in the guest operating systems.

Benchmarking Methodology

The network benchmarking tool, netperf 2.4.2, was used for all the experiments. Netperf measures unidirectional network performance for TCP and UDP traffic. It includes support to measure TCP and UDP throughput, using bulk transfers, and end-to-end latencies. Netperf has a client-server model and comprises the following:

- Netperf client, which acts as a data sender
- Netserver process, which acts as a receiver

To check for network performance under different configurations, netperf allows you to specify parameters, such as the socket size and the message size, for the tests.

The details of the experimental setup are presented in Table 1. All virtual machines in the experiments used a uniprocessor hardware abstraction layer (HAL)/kernel and were configured with one virtual CPU. All virtual machines were configured with 512MB of RAM and used the vmxnet virtual device.

Table 1. Experimental Setup

	Hardware	
ESX Server	AMAX Opteron 270	■ Two Dual-Core Opteron CPUs at 2.0GHz ■ 8GB RAM ■ Intel 82545 GM Ethernet Controller
Client Machine	Dell PowerEdge 1800	■ Two Intel Xeon CPUs at 2.80GHz ■ 4GB RAM ■ Intel 82535 GM Ethernet Controller
	Software	
ESX Server	ESX Server 3.0.2 and ESX Server 3.5.0	
Client machine	Suse Linux ver. 10, kernel ver. 2.6.16.1	
Virtual machine OS	■ Windows Server 2003 sp1 64-bit ■ RedHat Enterprise Linux u4 64-bit kernel: 2.6.9.22 ■ Windows Server 2003 sp1 32-bit ■ RedHat Enterprise Linux u4 32-bit kernel: 2.6.9.22	■ 1 VCPU ■ 512MB RAM ■ Virtual device: vmxnet
Native OS	■ Windows Server 2003 sp1 64-bit ■ RedHat Enterprise Linux u4 64-bit kernel: 2.6.9.22 ■ Windows Server 2003 sp1 32-bit ■ RedHat Enterprise Linux u4 32-bit kernel: 2.6.9.22	■ 1 CPU ■ 512MB RAM ■ Network driver: e1000

To understand the performance difference between e1000 and vmxnet virtual devices, see the *Performance Comparison of Virtual Network Devices* paper. Because ESX Server 3.0.2 does not officially support the vmxnet device for 64-bit operating systems, all the 64-bit experiments on ESX Server 3.0.2 were conducted using experimental support for vmxnet.

Figure 2 shows the experimental setup for the send experiments where the virtual machine sends data. For the receive experiment, the netperf and netserver processes were exchanged.

Figure 2. Experimental Setup for Virtual Machine to Native Send Test

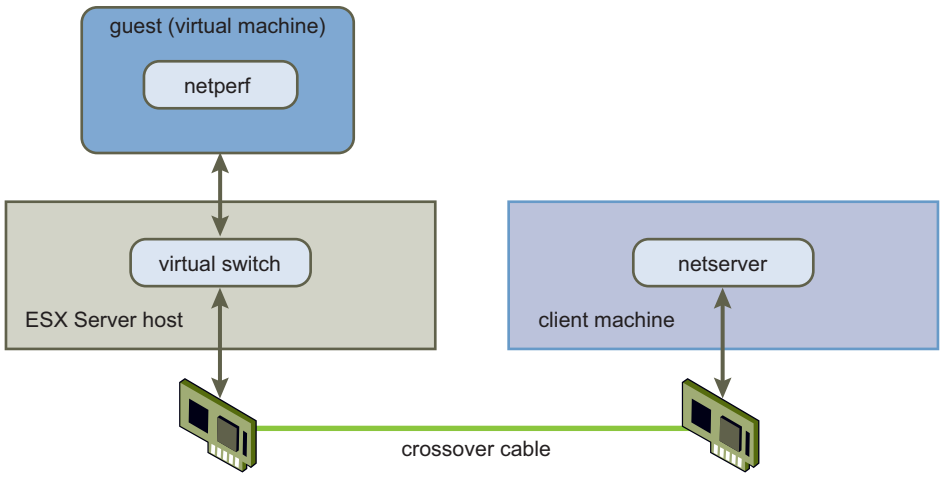
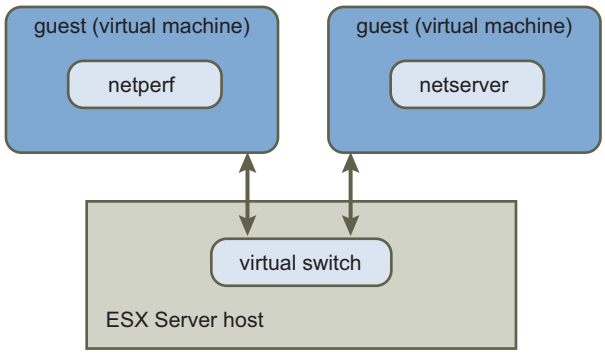


Figure 3 shows the experimental setup for the virtual machine to virtual machine send experiments.

Figure 3. Experimental Setup for Virtual Machine to Virtual Machine Send Test



Virtual Machine to Native Performance Results

The most common deployment scenario is virtual machine to native, where a virtual machine communicates with a physical machine over the network. In Figure 4, the send case is important to developers who virtualize server-side applications, such as Web servers with large amounts of outgoing traffic. The receive case depicts the download speeds of clients that run in virtual machines. Although virtualization introduces an additional layer between the operating system and the hardware, ESX Server can achieve near-native throughput in both cases.

Figure 4. 64-Bit Windows Virtual Machine to Native TCP Throughput

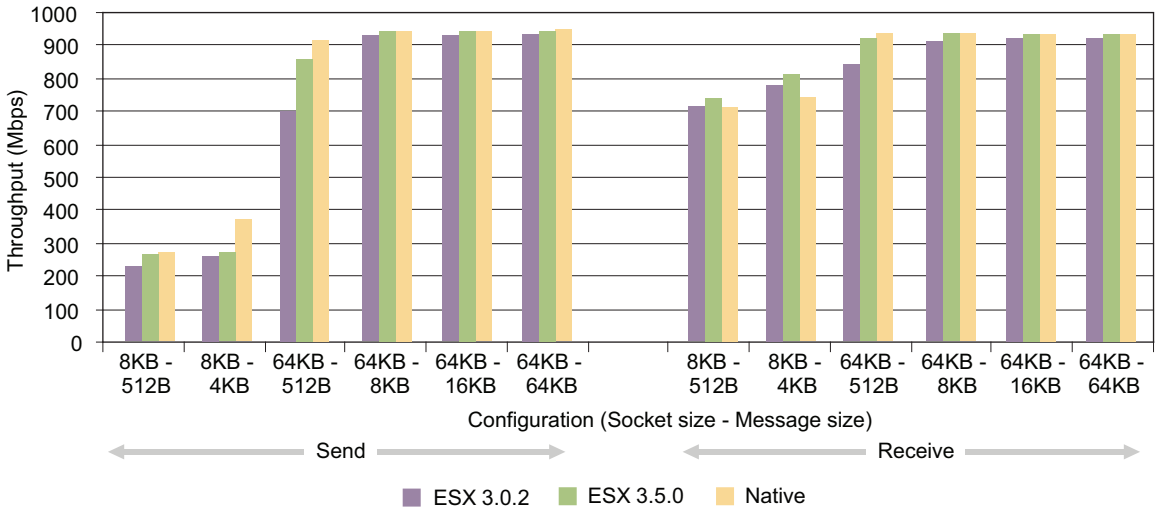


Figure 4 shows the TCP throughput for a 64-bit Windows Server 2003 virtual machine that runs on ESX Server 3.0.2 and ESX Server 3.5.0. The throughput is compared to Windows running natively. The points in the left side of the graph are for the send workload, and the points on the right side are for the receive workload. To keep the comparison fair, the Windows `boot.ini` file in the native setup was modified to boot the operating system with one CPU and 512MB RAM.

The TCP throughput for Windows running in a virtual machine on ESX Server 3.5 is similar to the throughput of Windows running natively. On ESX Server 3.5.0, compared to ESX Server 3.0.2, TCP performance improved by up to 20% in some cases. As a result, near-native performance was achieved across all network configurations.

In Figure 5, the results from experiments with a 64-bit RHEL4 virtual machine and a native RHEL4 installation are displayed. The boot options for the RHEL4-based machine were modified to load a uniprocessor kernel with memory restricted to 512MB. On the send path, ESX Servers 3.0.2 and 3.5.0 perform as well as a native machine and achieve a throughput of approximately 930 Mbps. On the receive path, the maximum throughput for ESX Server 3.5.0 is approximately 900 Mbps, which is within 4% of the native machine.

Figure 5. 64-Bit Linux Virtual Machine to Native TCP Throughput

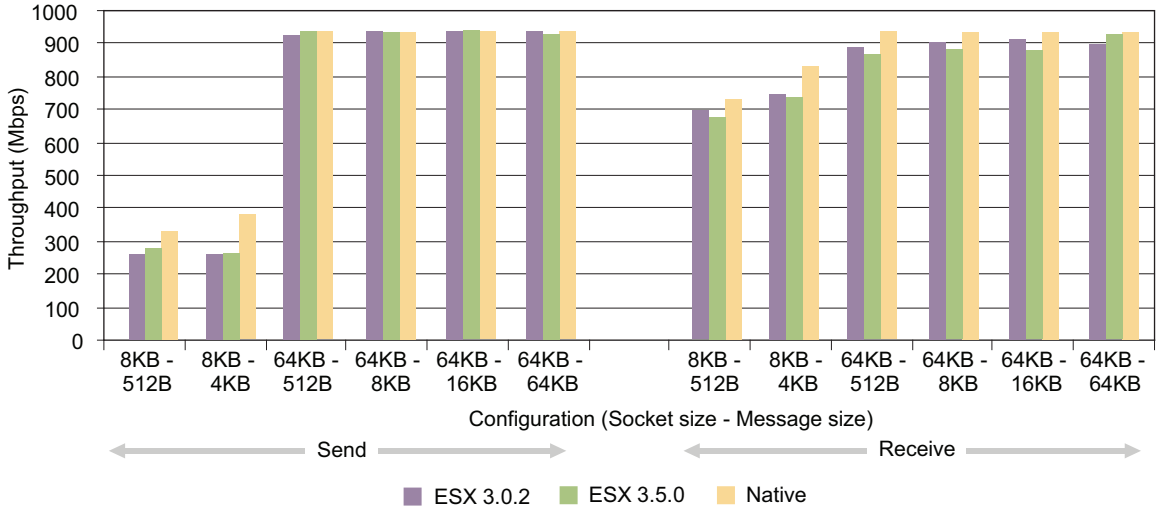
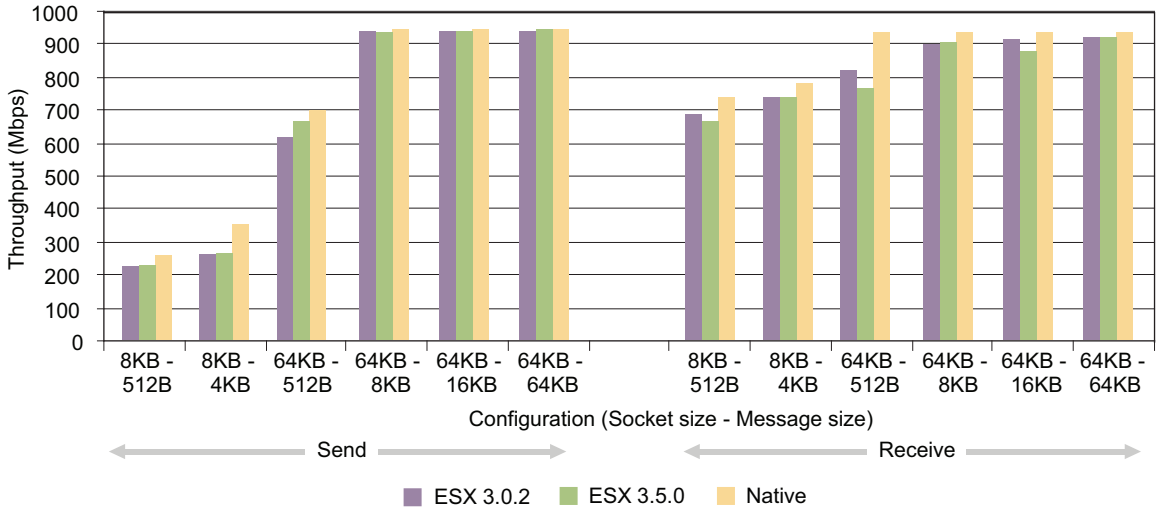


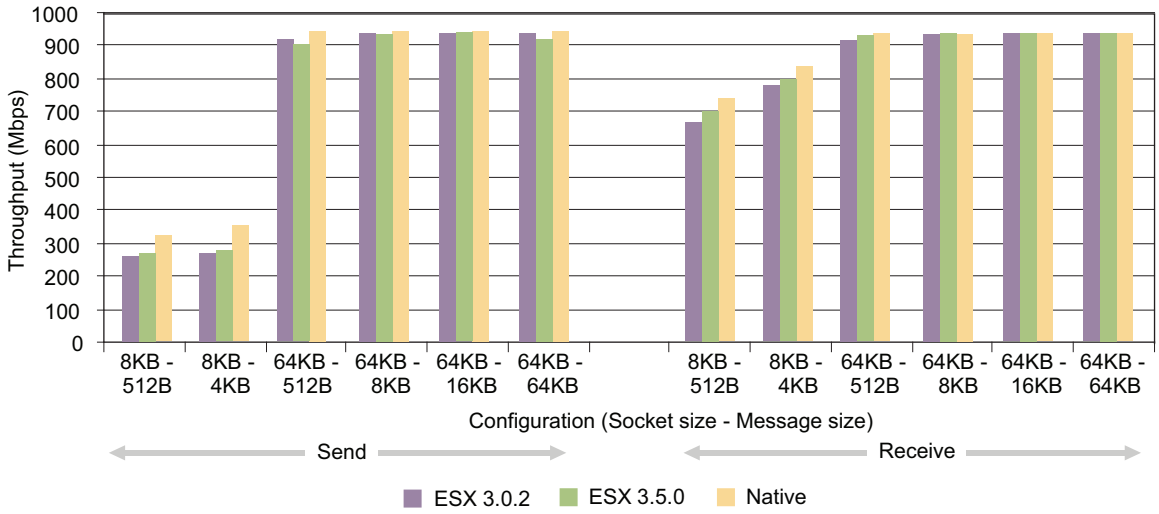
Figure 6 and Figure 7 show the results for 32-bit Windows and 32-bit Linux machines, respectively. For both operating systems, ESX Server network performance has remained constant across the releases and is at par with the 32-bit native results.

Figure 6. 32-Bit Windows Virtual Machine to Native TCP Throughput



For both operating systems, the maximum send and receive throughputs are above 900 Mbps.

Figure 7. 32-Bit Linux Virtual Machine to Native TCP Throughput



For 32-bit and 64-bit operating systems, the throughput achieved by a virtualized operating system on ESX Server 3.5.0 is similar to the throughput achieved by the same operating system running natively. Even though 64-bit vmxnet was not officially supported on ESX Server 3.0.2, near-native performance was achieved for Linux and Windows virtual machines.

Virtual Machine to Virtual Machine Performance Results

ESX Server allows multiple virtual machines to run on the same physical machine. Virtual machines that are expected to communicate with each other over the network should be placed on the same ESX server and connected to the same virtual switch. This test configuration measures the throughput and latency between two virtual machines connected to the same virtual switch.

Unlike the previous experiments, only the send test was performed with netperf running on the first virtual machine and netserver running on the second virtual machine. Because the generated traffic does not need to go on to the wire, the throughput between the two virtual machines should not be affected by the speed of the physical NIC.

Figure 8 shows the virtual machine to virtual machine TCP throughput for 64-bit and 32-bit Windows Server 2003 virtual machines. The maximum throughput in the 64-bit case is approximately 1350 Mbps, and the maximum throughput in the 32-bit scenario is close to 1600 Mbps. The minimal differences in performance could be attributed to the following:

- Different network stack implementations in the operating systems
- Differences in the virtual machine monitor in the 32-bit and 64-bit versions

Figure 8. Windows Virtual Machine to Virtual Machine TCP Throughput

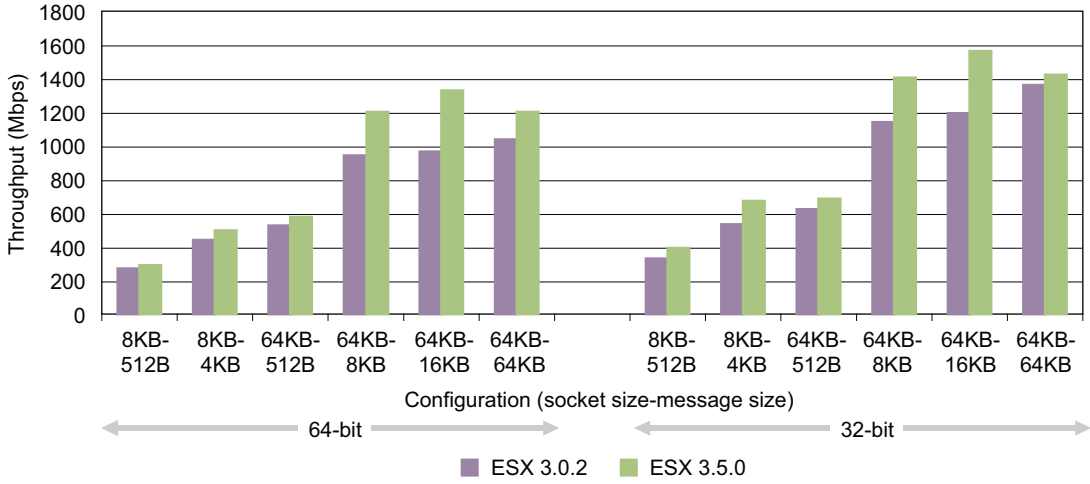
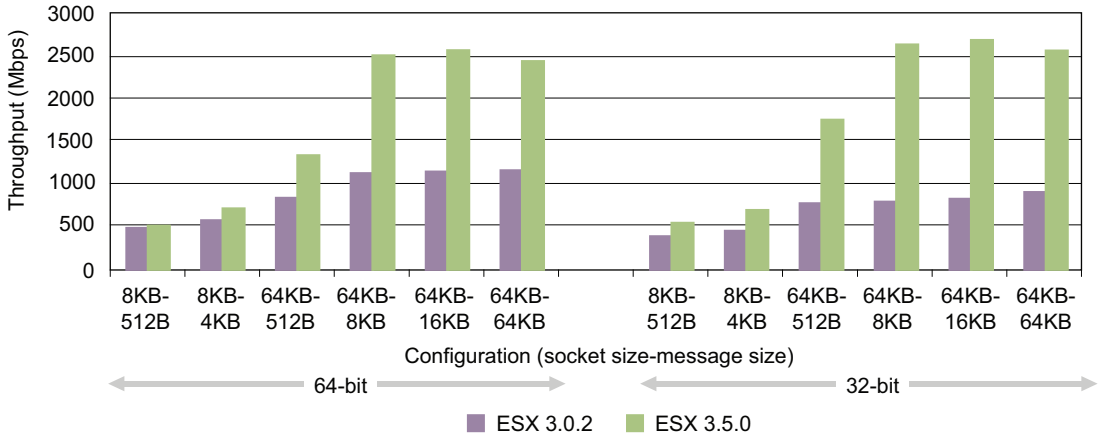


Figure 9 shows the results for the virtual machine to virtual machine experiments for Linux, where the 32-bit and 64-bit versions achieved throughputs greater than 2.5 Gbps. There is a significant difference between the performance of ESX Server 3.0.2 and ESX Server 3.5.0. As a result of the improvements in the ESX Server networking and scheduler code, some network configurations have shown a three-fold improvement in ESX Server 3.5.0 over ESX Server 3.0.2.

Figure 9. Linux Virtual Machine to Virtual Machine TCP Throughput



Thus, the virtual machine to virtual machine TCP throughput on ESX Server 3.5 can exceed 2.5 Gbps for some operating systems while speeds of physical networks with 1 Gbps NICs are limited to approximately 950 Mbps.

Latency Results

In addition to throughput, end-to-end latency is an important metric to gauge networking performance. Most applications in virtual machines are not expected to have high bandwidth requirements. However, there are certain applications, such as database applications and interactive desktop workloads, that are latency sensitive. The experiments show that ESX Server does not introduce any significant delays in the network path.

Figure 10 shows the number of request-responses per second for the 64-bit and 32-bit versions of Windows and Linux. A message size of one byte was used for all virtual machine to virtual machine latency experiments. The end-to-end latency is inversely proportional to the number of request-responses per second. The latencies observed in a 32-bit Windows Server 2003 virtual machine are within 25% of those observed on native machines. For all other operating systems, the latency observed in a virtual machine on ESX Server 3.5.0 is nearly the same as the latency observed on a native machine.

Figure 10. Virtual Machine to Native Latency

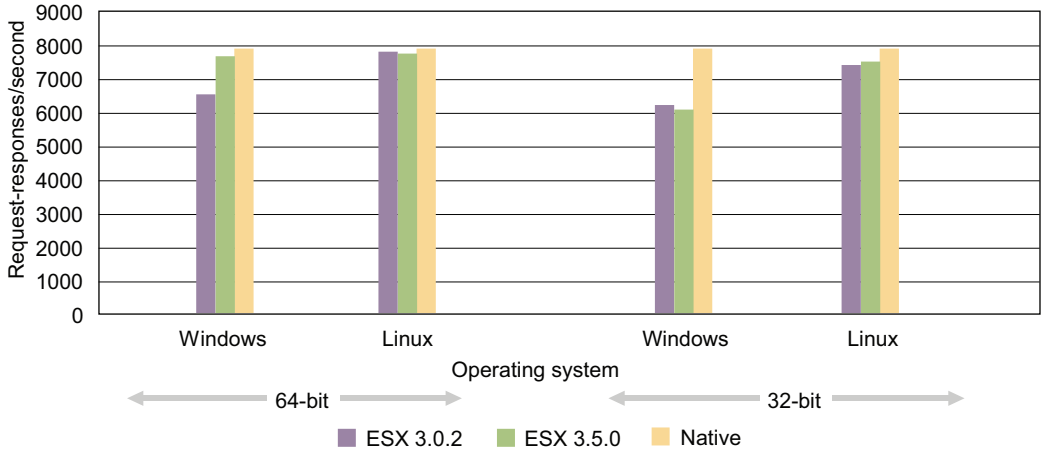
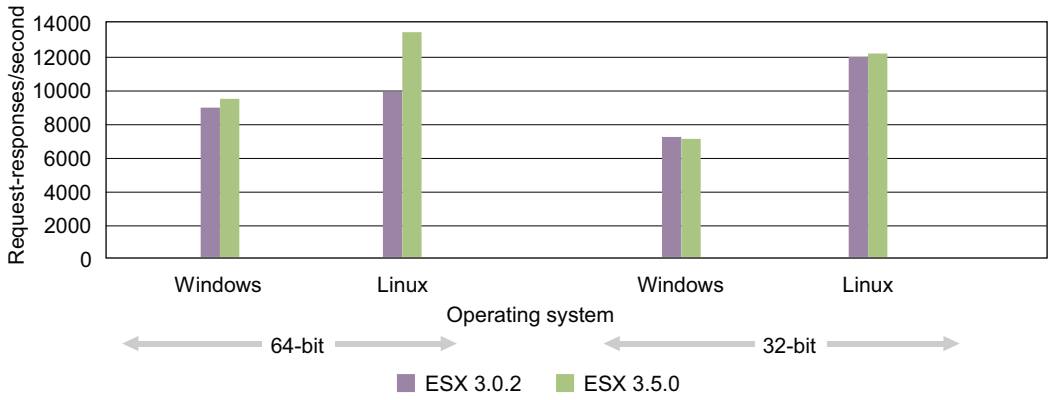


Figure 11 shows the results of latency experiments in virtual machine to virtual machine setups. Latencies between virtual machines located on the same ESX Server host are expected to be lower than latencies observed on virtual machine to native setups because the packets do not go on to the wire and go though fewer layers of code. On ESX Server 3.5, for operating systems tested, the virtual machine to virtual machine latencies are 15% to 40% lower than the corresponding virtual machine to native latencies. Windows Server 2003 virtual machines can exchange up to 10,000 requests and responses per second, and RHEL4 virtual machines can exchange up to 13,000 messages a second. Thus, ESX Server does not introduce large latencies in the networking path for most operating systems.

Figure 11. Virtual Machine to Virtual Machine Latency



Conclusion

The results in this study clearly show that the virtualized applications running in virtual machines on ESX Server 3.5.0 can achieve the same network throughput and latencies that are achieved by applications running natively. In fact, virtual machines that are connected to the same virtual switch can communicate at rates that are up to two-and-a-half times the rates supported by physical 1Gbps Ethernet networks.

References

- The netperf Web site at http://www.netperf.org/netperf/
- Sugerman Jeremy, Venkitachalan Ganesh, Lim Beng-Hong. Virtualizing I/O devices on VMware Workstation's Hosted Virtual Machine Monitor. Usenix, June 2001. http://www.usenix.org/publications/library/proceedings/usenix01/sugerman/sugerman.pdf.
- VMware Virtual Networking Concepts (VMTN) http://www.vmware.com/resources/techresources/997

VMware, Inc. 3401 Hillview Ave., Palo Alto, CA 94304 www.vmware.com

Revision 20080117 Item: PS-045-PRD-01-01