Performance Evaluation of VMXNET3 Virtual Network Device

VMware vSphere 4 build 164009

Introduction

With more and more mission-critical networking-intensive workloads being virtualized and consolidated, virtual network performance has never been more important and relevant than today. VMware has been continuously improving the performance of its virtual network devices. VMXNET Generation 3 (VMXNET3) is the most recent virtual network device from VMware, and was designed from scratch for high performance and to support new features.

This paper compares the networking performance of VMXNET3 to that of enhanced VMXNET2 (the previous generation of high performance virtual network device) on VMware vSphere 4 to help users understand the performance benefits of migrating to this next-generation device. We conducted the performance evaluation on an RVI-enabled AMD Shanghai system using the netperf benchmark. Our study covers both Windows Server 2008 and Red Hat Enterprise Linux 5 (RHEL 5) guest operating systems and results show that overall VMXNET3 is on par with or better than enhanced VMXNET2 for both 1 Gig and 10 Gig workloads. Furthermore, VMXNET3 has laid the groundwork for further improvement by being able to take advantage of new advances in both hardware and software.

The overhead of network virtualization includes the virtualization of the CPU, the MMU (Memory Management Unit), and the I/O devices. Therefore, advances in any of these components will affect the overall networking I/O performance. In this paper, we will focus on device- and driver-specific features.

In the next section, we describe the new features introduced by VMXNET3. We then describe our experimental methodologies, benchmarks, and detailed results in subsequent sections. Both IPv4 and IPv6 TCP performance results are presented. Finally, we conclude by providing a summary of our performance experience with VMXNET3.

Background

Both enhanced VMXNET2 and VMXNET3 devices are virtualization-aware (para-virtualized): the device and the driver have been designed with virtualization in mind to minimize I/O virtualization overhead. To further close the performance gap between virtual network device and native device, a number of new enhancements have been introduced with VMXNET3.

With the advent of 10GbE NICs, networking throughput is often limited by the processor speed and its ability to handle high-volume network processing tasks. Multi-core processors offer opportunities for parallel processing that scale to more than one processor. Modern operating systems, including Windows Server 2008 and Linux, have incorporated support to leverage such parallelism by allowing networking processing on different cores simultaneously. Receive-Side Scaling (RSS) is one such technique used in Windows Server 2008 and Vista to support parallel packet receive processing. VMXNET3 supports RSS on those operating systems.

The VMXNET3 driver is NAPI-compliant on Linux guests. NAPI is an interrupt mitigation mechanism that improves high-speed networking performance on Linux by switching back and forth between interrupt mode and polling mode during packet receive. It is a proven technique to improve CPU efficiency and allows the guest to process higher packet loads. VMXNET3 also supports Large Receive Offload (LRO) on Linux guests. However, in ESX 4.0 the VMkernel backend supports large receive packets only if the packets originate from another virtual machine running on the same host.

VMXNET3 supports larger Tx/Rx ring buffer sizes compared to previous generations of virtual network devices. This feature benefits certain network workloads with bursty and high-peak throughput. Having a larger ring size provides extra buffering to better cope with transient packet bursts. VMXNET3 supports three interrupt modes: MSI-X, MSI, and INTx. Normally the VMXNET3 guest driver will attempt to use the interrupt modes in the order given above, if the guest kernel supports them. With VMXNET3, TCP Segmentation Offload (TSO) for IPv6 is supported for both Windows and Linux guests now, and TSO support for IPv4 is added for Solaris guests in addition to Windows and Linux guests.

A summary of the new features is listed in Table 1. To use VMXNET3, the user must install VMware Tools on a virtual machine with hardware version 7.

Table 1. Feature comparison table

Features	Enhanced VMXNET2	VMXNET3
TSO, Jumbo Frames, TCP/IP Checksum Offload	Yes	Yes
MSI/MSI-X support (subject to guest operating system kernel support)	No	Yes
Receive Side Scaling (RSS, supported in Windows 2008 when explicitly enabled)	No	Yes
IPv6 TCP Segmentation Offloading (TSO over IPv6)	No	Yes
NAPI (supported in Linux)	No	Yes
LRO (supported in Linux, VM-VM only)	No	Yes
IPv6 TCP Segmentation Offloading (TSO over IPv6)	No	Yes

Experimental Methodology

This section describes the experimental configuration and the benchmarks used in this study.

Benchmarking Methodology

We used the network benchmarking tool netperf 2.4.3 for IPv4 experiments and 2.4.4 for Windows IPv6 experiments (due to problems with netperf 2.4.3 in supporting IPv6 traffic on Windows), with varying socket buffer sizes and message sizes. Testing different socket buffer sizes and message sizes allows us to understand TCP/IP networking performance under various constraints imposed by different applications. Netperf is widely used for measuring unidirectional TCP and UDP performance. It uses a client-server model and comprises a client program (netperf) acting as a data sender, and a server program (netserver) acting as a receiver.

We focus on TCP performance in this paper, with the relevant performance metrics defined as follows:

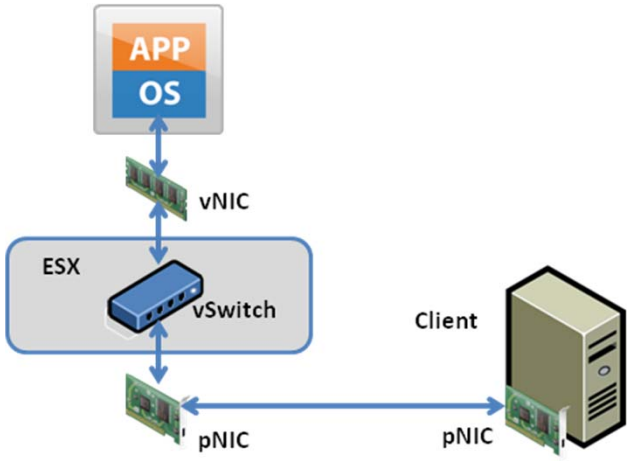
- **Throughput** – This metric was measured using the netperf TCP_STREAM test. It reports the number of bytes transmitted per second (Gbps for 10G tests and Mbps for 1G tests). We ran five parallel netperf sessions and reported the aggregate throughput for 10G tests and 1 netperf session for 1G tests.
- **Latency** – This metric was measured using the netperf TCP Request-Response (TCP_RR) test. It reports the number of request-response transactions per second, which is inversely proportional to end-to-end latency.
- **CPU usage** – CPU usage is defined as the CPU resources consumed to run the netperf test on the testing server. It is measured after the test enters the stable state after the test is started.

Details of the different TCP tests mentioned above can be found in the netperf documentation.

Experimental Configuration

Our testbed setup consisted of two physical hosts, with the server machine running vSphere 4 and the client machine running RHEL 5.1 natively. The 1GbE and 10GbE physical NICs on both hosts were connected using crossover cables. The testbed setup is illustrated in Figure 1.

Figure 1. Testbed setup



Configuration information is listed in Table 2. All virtual machines used are 64-bit with 1 vCPU. 2GB RAM is used for both Windows and Linux virtual machines. The default driver configuration is used for VMXNET3 unless mentioned otherwise.

Table 2. Configuration

Server	
CPUs	Two Quad-Core AMD Opteron 2384 Processors (“Shanghai”) @2.7GHz
Memory	8GB
Networking	Intel 82572EI GbE NIC Intel 82598EB 10GbE AF Dual Port NIC
Virtualization Software	VMware vSphere 4 (build 164009) Hardware virtualization, RVI enabled
Virtual Machine	
CPUs	1 virtual CPU
Memory	2GB
Operating Systems	Windows Server 2008 Enterprise Red Hat Enterprise Linux 5.2
Client	
CPUs	Two Quad-Core Intel Xeon X5355
Memory	16GB
Networking	Intel 82572EI GbE NIC Intel 82598EB 10GbE AF Dual Port NIC
Operating System	Red Hat Enterprise Linux 5.1 (64-bit)

Windows Server 2008 VM Experiments

In this section, we describe the performance of VMXNET3 as compared to enhanced VMXNET2 on a Windows Server 2008 Enterprise virtual machine, using both 1GbE and 10GbE uplink NICs. All reported data points are the average of three runs.

10G Test Results

Aggregated 10Gbps network load will be common in the near future due to high server consolidation ratios and high bandwidth requirements of today's applications. To efficiently handle throughput at this rate is very demanding. We used five concurrent netperf sessions for a 1-vCPU virtual machine 10G test.

The results on the left side of the graph are for transmit workload, and the results on the right side for receive workload. To ensure a fair comparison, CPU usage is normalized by throughput and the ratio of VMXNET3 to enhanced VMXNET2 is presented. Any data point with a CPU ratio higher than 1 means that VMXNET3 uses more CPU resource for the given test to drive the same amount of throughput, while any data point with a CPU ratio lower than 1 means that VMXNET3 is more efficient in driving the same amount of throughput for the given test. The X-axis on the graphs represents the different network configuration parameters (socket buffer size and message size pairs) used for netperf.

Figure 2 illustrates that VMXNET3 achieves a higher throughput compared to enhanced VMXNET2 for the majority of the tests, with the receive workload achieving more significant gains. The most noticeable improvement is for large socket size/message size receive tests, with an improvement up to 92%.

Figure 2. Windows 10G throughput results (higher is better)

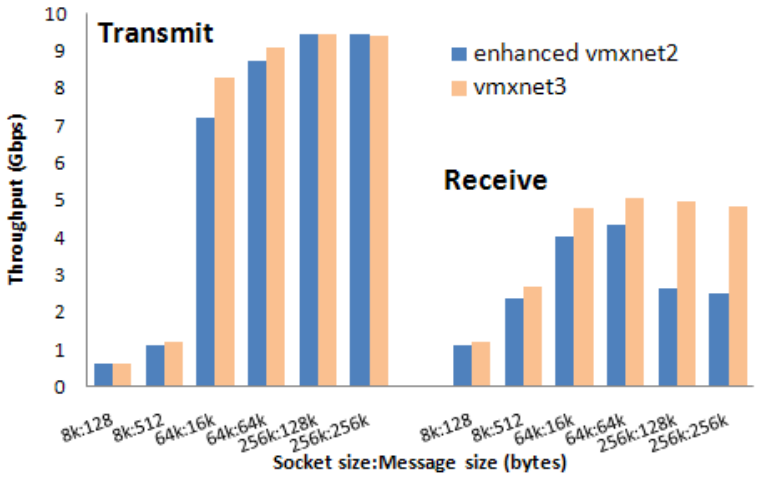
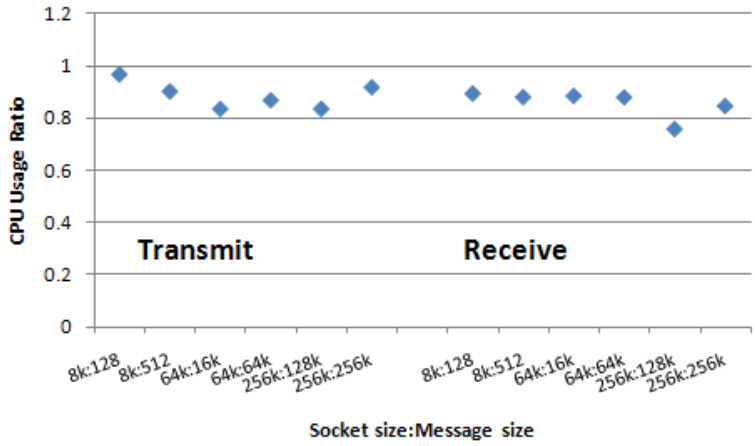


Figure 3 shows the CPU usage ratio of VMXNET3 to enhanced VMXNET2 and it shows that VMXNET3 is more CPU efficient than enhanced VMXNET2 across the board. This saves more CPU resource for other useful work.

Figure 3. Windows 10G CPU usage ratio comparison (lower than 1 means VMXNET3 is better)



1G Test Results

Figure 4 shows the throughput results using a 1GbE uplink NIC. We use one netperf session for this test as a single session will be able to reach line rate for both transmit and receive. Overall, VMXNET3 throughput is on par with enhanced VMXNET2 and is noticeably better for small socket/message size transmit tests. There is one exception, though, for 8k/512 receive test. However, the gap is quite small and falls into the range of noise.

Figure 4. Windows 1G throughput results (higher is better)

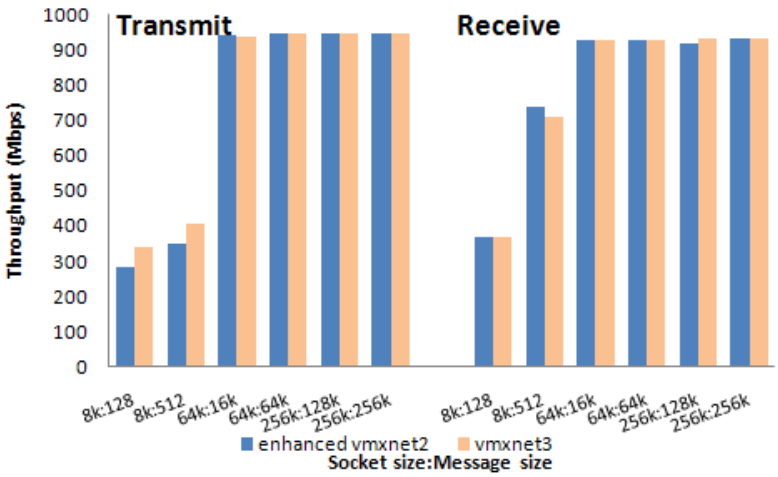
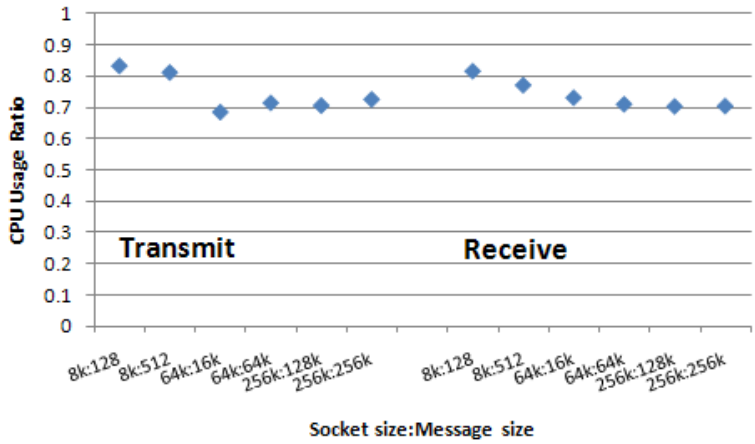


Figure 5 shows that the CPU usage ratio of VMXNET3 to enhanced VMXNET2 is consistently lower than 1, which means that VMXNET3 is more efficient than enhanced VMXNET2 to drive the same amount of throughput, with up to 25% savings.

Figure 5. Windows 1G CPU Usage Ratio Comparison (lower than 1 means VMXNET3 is better)

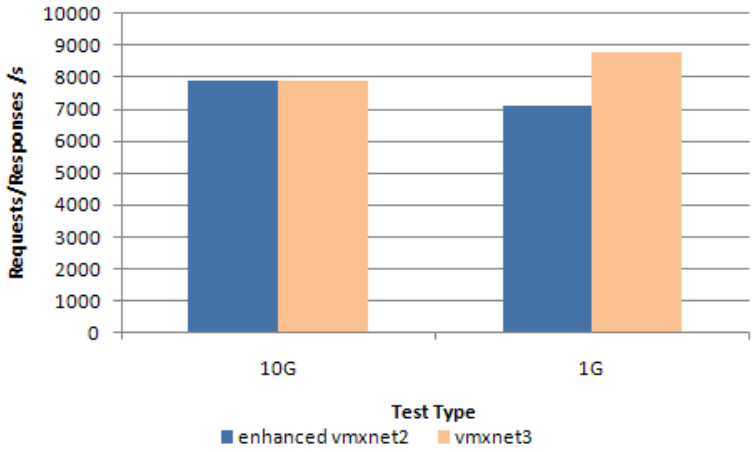


Latency Results

In addition to throughput, end-to-end latency is another metric to measure networking performance, which is particularly important for latency-sensitive applications such as Web servers, database applications, and interactive desktop workloads.

Figure 6 shows the number of request-responses per second for a Windows Server 2008 virtual machine for both 10G and 1G traffic. A message size of 1 byte was used for the latency experiments. VMXNET3 is as good as enhanced VMXNET2 using a 10G uplink and achieves a much higher RR rate with a 1G uplink. As requests/responses per second are inversely proportional to end-to-end message latency, this shows that VMXNET3 achieves a lower latency than enhanced VMXNET2 in general.

Figure 6. Windows netperf latency (TCP_RR) results (higher is better)



IPv6 Test Results

With VMXNET3, IPv6 support has been further enhanced with TSO6 (TCP Segmentation Offload over IPv6), which significantly helps to reduce transmit processing performed by the vCPUs and improves both transmit efficiency and throughput. If the uplink NIC supports TSO6, the segmentation work will be offloaded to the network hardware; otherwise, software segmentation will be conducted inside the VMkernel before passing packets to the uplink. Therefore, TSO6 can be enabled for VMXNET3 whether or not the hardware NIC supports it.

Next, we will look at IPv6 TCP performance inside a Windows Server 2008 virtual machine on a 10G setup. IPv6 performance on a Linux virtual machine shows similar improvement of throughput and CPU efficiency for transmit.

Figure 7 shows that transmit throughputs almost double that of enhanced VMXNET2 for large socket/message tests. Receive throughput shows a similar improvement observed with IPv4. Note that the difference between IPv4 and IPv6 transmit throughput for the same test is due to the lack of LRO support for IPv6 packets on the client host (native Linux with the ixgbe driver).

Figure 7. Windows 10G IPv6 throughput results (higher is better)

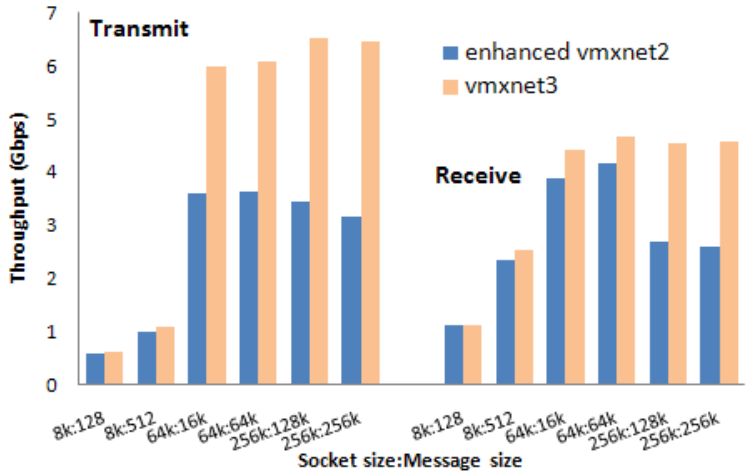
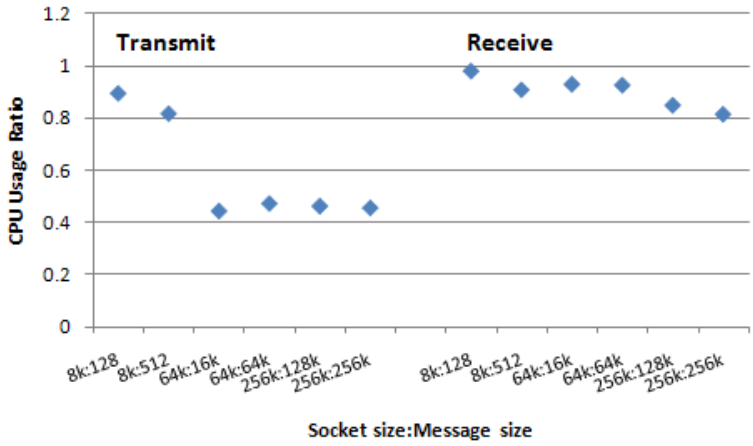


Figure 8 shows that VMXNET3 is as good as or better than enhanced VMXNET2 in terms of CPU efficiency for IPv6.

Figure 8. Windows 10G IPv6 CPU usage ratio comparison (lower than 1 means VMXNET3 is better)



Red Hat Enterprise Linux Virtual Machine Experiments

In this section, we present the performance of VMXNET3 as compared to enhanced VMXNET2 on a Red Hat Enterprise Linux 5.2 (RHEL 5.2) virtual machine.

10G Test Results

Figures 9 and 10 show the throughput and CPU efficiency respectively with a 10G uplink NIC. VMXNET3 is on par with enhanced VMXNET2 for most of the tests and slightly better for most receive tests.

Figure 9. Linux 10G throughput results (higher is better)

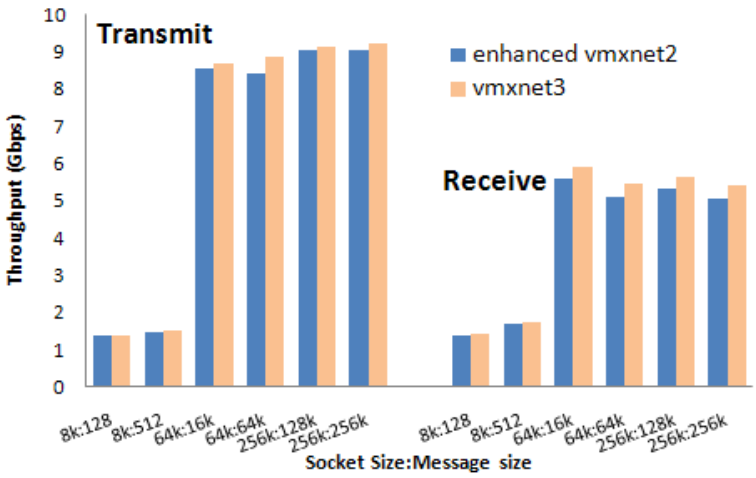
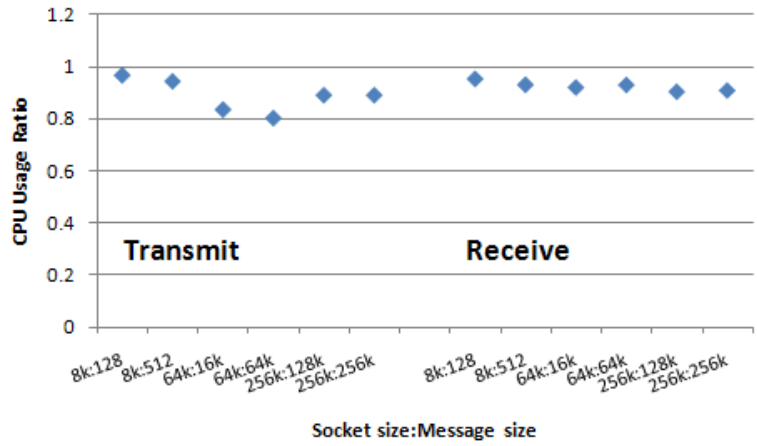


Figure 10. Linux 10G CPU Usage Ratio Comparison (lower than 1 means VMXNET3 is better)



1G Test Results

For 1G tests, VMXNET3 is in general as good as enhanced VMXNET2. There are a few exceptions though, where VMXNET3 spends more CPU resource to drive a similar amount of throughput. This can be explained by the larger ring size used by VMXNET3 by default and the working set of the Linux virtual machine.

Ring buffer size can be tuned to ensure enough queuing at the device level to handle the peak loads generated by the system and the network. With a large ring size, the device can handle higher peak loads without dropping any packets. However, this also increases the memory size used by the device and, as a result, a larger cache footprint. This ends up with a smaller cache size for other useful data/codes and slight performance degradation.

The default receive (Rx) ring size is **150** for enhanced VMXNET2 and **256** for VMXNET3. We used the default ring size for the results presented in both Figures 11 and 12. After reducing the receive ring size to the same used by enhanced VMXNET2, VMXNET3's receive CPU efficiency is on par with enhanced VMXNET2 again, with no loss in throughput. So depending on the workload, users may opt to tune the ring size to best fit their requirements. (Users can change the default receive ring size for VMXNET3 from within the Linux guest operating system: **`ethtool -G eth<x> rx <new_value>`**, where <x> refers to the Ethernet interface ID in the guest, and <new_value> refers to the new value for the Rx ring size.)

Figure 11. Linux 1G throughput results (higher is better)

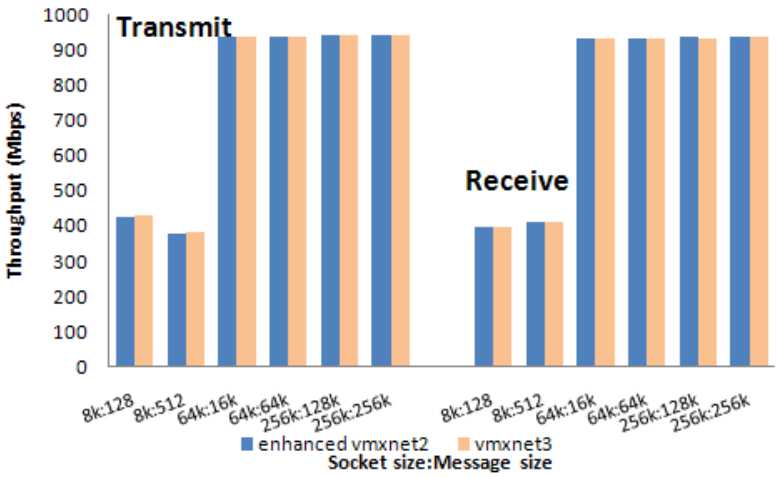
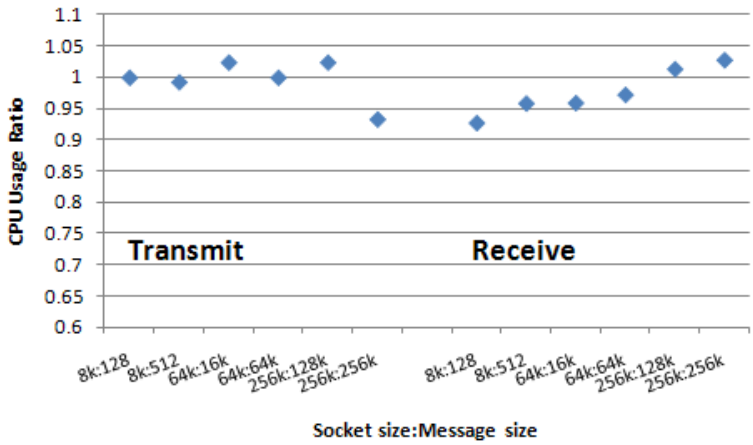


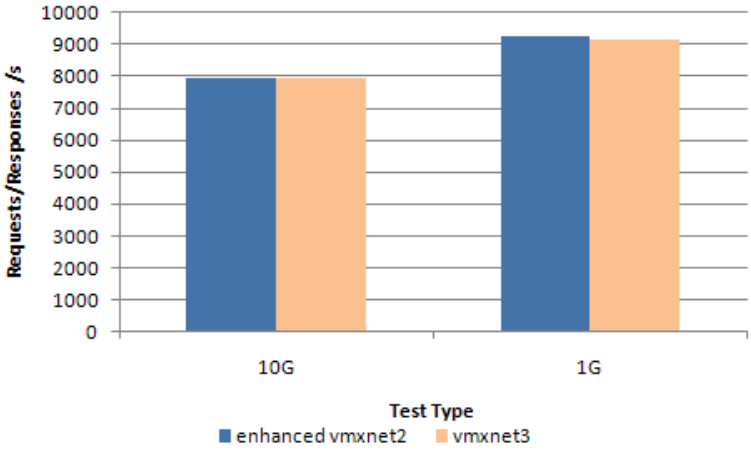
Figure 12. Linux 1G CPU usage ratio comparison (lower means VMXNET3 is better)



Latency Results

Figure 13 shows netperf latency results for both 10G and 1G setup with an RHEL 5.2 virtual machine. VMXNET3 is again as good as enhanced VMXNET2.

Figure 13. Linux netperf latency (TCP_RR) results (higher is better)



Conclusion

VMXNET3, the newest generation of virtual network adapter from VMware, offers performance on par with or better than its previous generations in both Windows and Linux guests. Both the driver and the device have been highly tuned to perform better on modern systems. Furthermore, VMXNET3 introduces new features and enhancements, such as TSO6 and RSS. TSO6 makes it especially useful for users deploying applications that deal with IPv6 traffic, while RSS is helpful for deployments requiring high scalability. All these features give VMXNET3 advantages that are not possible with previous generations of virtual network adapters. Moving forward, to keep pace with an ever-increasing demand for network bandwidth, we recommend customers migrate to VMXNET3 if performance is of top concern to their deployments.

References

- The netperf Web site (http://www.netperf.org/netperf)
- Receive-Side Scaling Enhancements in Windows Server 2008 (http://www.microsoft.com/whdc/device/network/ndis_rss.mspx)
- NAPI (http://www.linuxfoundation.org/en/Net:NAPI)
- VMware, Inc. white paper. “Performance Comparison of Virtual Network Devices,” (http://www.vmware.com/files/pdf/perf_comparison_virtual_network_devices_wp.pdf)
- VMware, Inc. white paper, “Performance Evaluation of AMD RVI Hardware Assist,” (http://www.vmware.com/pdf/RVI_performance.pdf)

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

Item: EN-000295-00 If you have comments about this documentation, submit your feedback to: docfeedback@vmware.com