Networking Performance in Multiple Virtual Machines

In a companion paper (*Multi-NIC Networking Performance in ESX 3.0.1 and XenEnterprise 3.2.0*, http://www.vmware.com/pdf/Multi-NIC_Performance.pdf) we looked at loading up a single virtual machine (VM) with multiple `netperf` instances, each running over its own NIC (Network Interface Controller). This effectively exposes the real virtualization overhead of high-throughput networking. While there are some real-world use cases that require this much network bandwidth in a single VM, a much more common scenario is spreading this bandwidth over many VMs running on one physical machine. This is a natural result of consolidating servers. For this paper we used the same hardware and software as in the multi-NIC paper, but performed a "scale-out" test: each of several VMs had a 1 Gbps physical NIC dedicated to it and each communicated to a similar dedicated NIC on the client machine through a `netperf`/`netserver` pair. The VMs did not share NICs. We hope this will lead to a better understanding of the performance issues involved with virtualizing networking.

By using up to four VMs/NICs here instead of the three NICs used in the earlier paper, we made better use of the 4-core server and were also able to test the scaling properties of each hypervisor. With dual- and quad-port network cards now commonly available, many users expect to be able to use four or more NICs even in low-end servers. Though the documentation included with XenEnterprise 3.2.0 indicates that the product supports only three physical NICs, the user interface had no problems configuring four NICs and issued no errors or warnings. Because of this ambiguity, and since this is such an important case, we included four-NIC benchmark results. Users should always verify support for their desired hardware configurations. VMware® ESX Server 3.0.1 supports 32 e1000 physical NICs.

There is no native test that would be analogous to this multi-VM test. Instead, we show results for the closest alternative: a native 4-CPU machine configured with all four NICs. This is a reasonable configuration if consolidation is being considered at the application level instead of the OS level. As before, "send" and "receive" cases were run. For send, each VM on the server ran one instance of `netperf` and the client ran the same total number of instances of `netserver`. For receive, the VMs each ran one instance of `netserver` and the client ran the appropriate number of `netperf` instances. Each `netperf`/`netserver` pair communicated over a unique subnet and port. In both the send and receive cases, all the `netperf` instances had to be started simultaneously. In the previous paper, and for the receive case here, this was easy since only one machine was involved. For the send case, we used Remote Execute (http://www.ibexsoftware.com) to launch all the `netperf` processes on the VMs remotely from the client machine.

The server VMs were running Windows Server 2003 Release 2 Enterprise Edition (32-bit) configured with one processor and 1GB memory running on a 4-core 3 GHz HP DL380G5 with 16GB of memory. The client machine was similar: a 4-core 2.6 GHz HP DL385G2 with the same adapters, same memory, and same operating system, but configured with the full hardware resources in an attempt to prevent it from becoming a bottleneck (more on this below). Each pair of server and client NICs was connected directly with a cross-over Ethernet cable. `Netperf` was configured for TCP stream, message size 8KB, and socket buffer size 64KB. This socket size yielded close to the maximum throughput in the single-NIC case. Full hardware and software configuration details are given in "Configurations," at the end of this Technical Note.

The hypervisors tested were ESX Server 3.0.1 (referred to as "ESX301") and XenEnterprise 3.2.0 (referred to as "XE320"). Both were installed with no modifications or tuning, except where otherwise noted. The corresponding "tools" packages containing paravirtualized (PV) network drivers were installed in all the guests under both hypervisors. As a baseline, the same guest OS was run natively. It was also configured with 1GB memory, but used four CPUs and an ACPI multiprocessor HAL instead of the ACPI uniprocessor HAL used in the guests.

Throughput results for send and receive are shown in the two figures below. "Number of NICs/VMs" refers both to the total number of NICs used in the physical machine and to the number of virtual machines for the hypervisor cases. In the native send case when using all four NICs the client started to become a bottleneck instead of the server. For just this case the fourth NIC was moved to a second client. The performance improvement with this change was about 2%. Use of a single client did not cause a bottleneck in the virtualized cases or in the native receive case.

The native tests achieved full wire speed using up to four NICs. The ESX301 tests stayed close to native, falling off to 88-90% of wire speed for four VMs. The XE320 tests achieved maximum throughput at three VMs and then slowed down at four VMs to 61% of wire speed for the send case and 57% for the receive case. One reason for this was CPU saturation in dom0 (Xen's privileged domain, where the backend drivers live), which is only uniprocessor. ESX Server is designed to avoid any CPU bottlenecks in the virtualization layer, and thus it scales much better.

Figure 1. Netperf Send Results

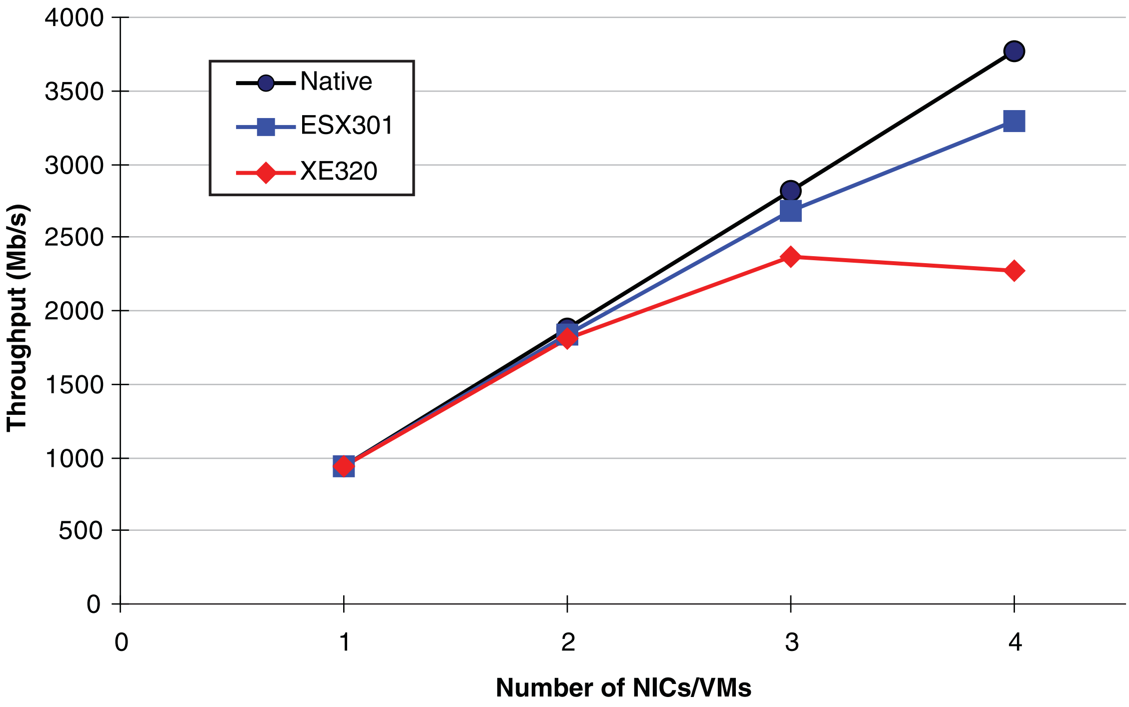
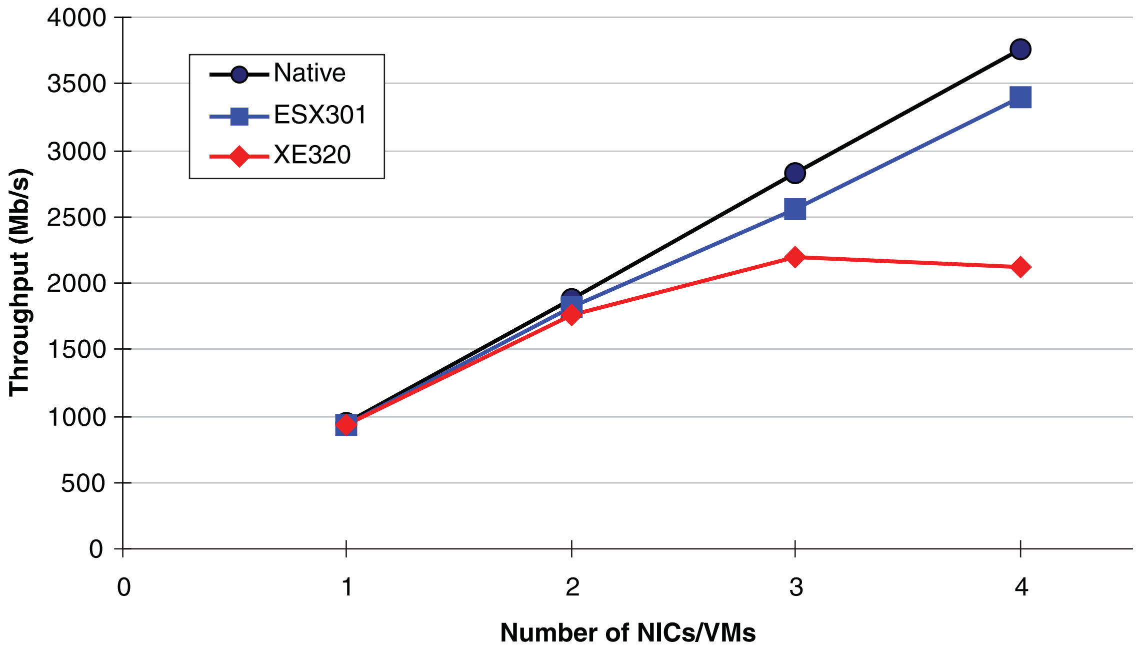


Figure 2. Netperf Receive Results



We investigated the poor throughput in XE320 a little further to see if we could find an easy fix. As noted above, the uniprocessor dom0 becomes CPU saturated. The number of processors available to dom0 is controlled by the `dom0-cpus` parameter in one of the Xen configuration files (`xend-config.sxp`). The kernel for dom0 is built with SMP enabled and it is booted with no limit on the number of processors. It is only when `xend` is started that the configuration file is evaluated and the number of processors available to dom0 becomes limited. In XenEnterprise `dom0-cpus` is set to 1. But in open-source Xen 3.0.4 it is set to 0 (i.e., dom0 uses all CPUs available). So even though there is no explicit support in XenEnterprise for changing this parameter, it is not unreasonable to try the open-source value to see if that is the source of the bottleneck. We did this for the four VM case. However, the throughput for both send and receive dropped by 9%; simply allowing dom0 to be SMP is not the fix. The throughput could not improve because an SMP dom0 does not take advantage of any more resources than does a UP dom0. In particular, all the interrupts are funneled through one processor in both cases. This was confirmed by noting that `/proc/irq/*/smp_affinity`, which binds interrupts to specific processors, is always set to 1 (CPU0). We then tried to distribute the interrupt processing by manually pinning half the interrupts to a different CPU. This relieved the one-processor bottleneck but resulted in no improvement for the receive case and a further 20% reduction in throughput for the send case. ESX301 does not have these throughput or scaling issues since interrupt processing and other work in the virtualization layer is appropriately spread and scheduled across the processors.

Configurations

Server Configuration (Virtualized System)

Hardware

Base hardware: HP DL380G5, 4-core Intel Woodcrest Xeon 5160 3 GHz

Memory: 16GB DDR2 667

Disks: Four 146GB 10,000 RPM Serial SCSI

NICs: Two Intel PRO/1000 PT Dual Port Server Adapters, Intel 82571EB Gigabit Ethernet Controller (rev 06)

Hypervisor Configuration

ESX301: ESX Server 3.0.1 GA release using BT monitor, 32 bit

XE320: XenEnterprise 3.2.0. Based on open-source Xen 3.0.4-1, Intel VT, 32 bit
Dom0: XenLinux 2.6.16.38, 32 bit, 512 MB

Guest Configuration

ESX301:

Virtual hardware: One processor, 1GB memory, VMware Tools installed (including the vmxnet network driver with default parameters). One virtual NIC, one vSwitch, and one physical NIC per VM.

Operating system: Windows Server 2003 Release 2 Enterprise Edition (32-bit), ACPI uniprocessor HAL

XE320:

Virtual hardware: One processor, 1GB memory, XenSource Windows Tools installed (including version 3.2.1.1 of the XenSource PV Ethernet Adapter with default parameters). One virtual NIC, one virtual bridge, and one physical NIC per VM.

Operating system: Windows Server 2003 Release 2 Enterprise Edition (32-bit), ACPI uniprocessor HAL

Server Configuration (Native System)

Base hardware: HP DL380G5, 4-core Intel Woodcrest Xeon 5160 3 GHz

Memory: 16GB DDR2 667 (limited to 1GB through `boot.ini` parameters)

Disks: Four 146GB 10,000 RPM Serial SCSI

NICs: Two Intel PRO/1000 PT Dual Port Server Adapters, Intel 82571EB Gigabit Ethernet Controller (rev 06)

Operating system: Windows Server 2003 Release 2 Enterprise Edition (32-bit), ACPI multiprocessor HAL

Client Configuration

Base hardware: HP DL385G2, 4-core AMD Opteron 2218 Rev. F 2.6 GHz

Operating system: Windows Server 2003 Release 2 Enterprise Edition (32-bit), ACPI multiprocessor HAL

Memory: 16GB DDR2 667

Disks: Two 146GB 10,000 RPM Serial SCSI

NICs: Two Intel Pro/1000 PT Dual Port Server Adapter, Intel 82571EB Gigabit Ethernet Controller (rev 06)

Netperf Configuration

Test: TCP_STREAM

Socket size: 64KB

Message size: 8KB

Each `netperf` instance is associated with a dedicated NIC, uses a unique port and subnet, and communicates with a dedicated `netserver` process on the other machine listening on the same port.

VMware, Inc. 3145 Porter Drive Palo Alto, CA 94304 www.vmware.com

Revision 20070604 Item: ESX-ENG-Q207-391