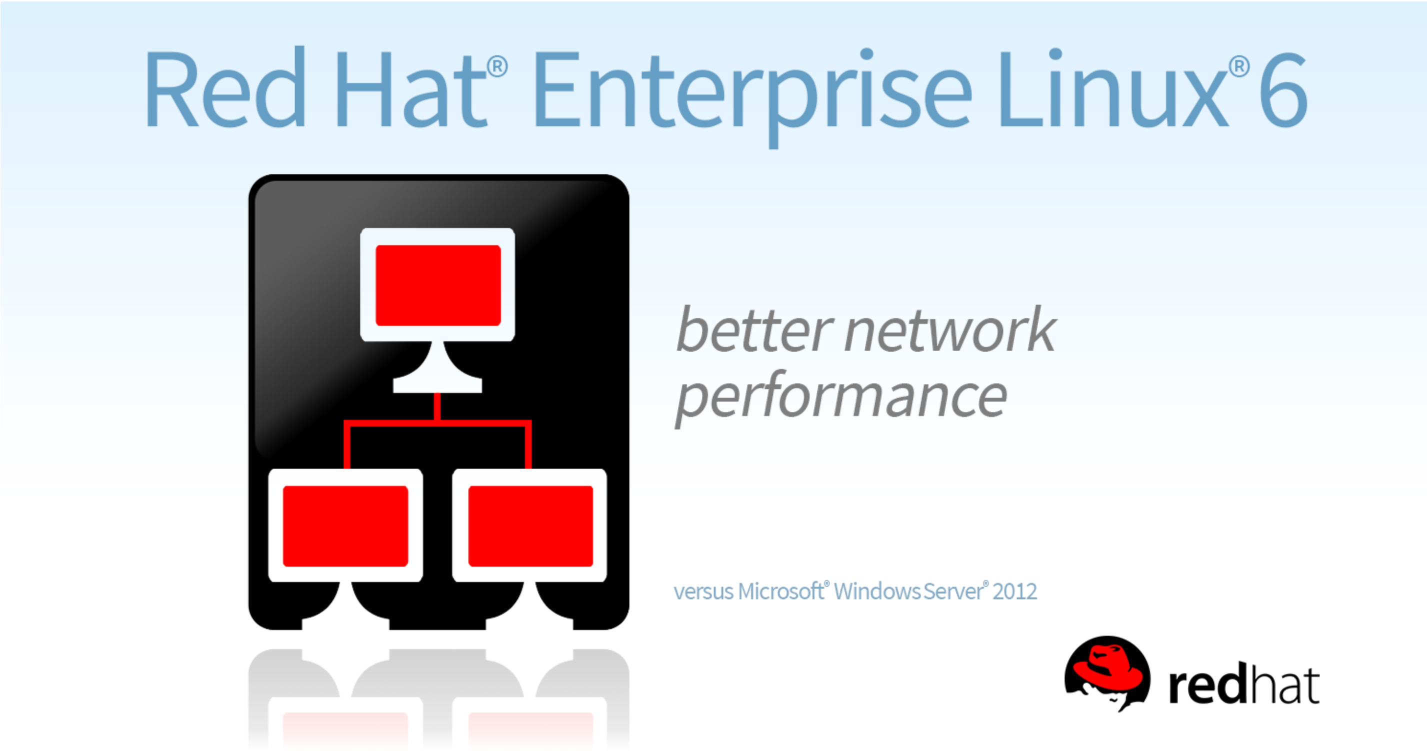
COMPARING NETWORK PERFORMANCE: RED HAT ENTERPRISE LINUX 6 VS. MICROSOFT WINDOWS SERVER 2012



For enterprises today, the ability to quickly and reliably transmit data can provide a competitive advantage. Machines, users, and applications constantly need to exchange information with one another, making the network a critical resource in a modern datacenter. Because applications do not typically manage networking resources directly and instead rely on operating systems to do so, the operating system you select may have a direct impact on TCP and UDP performance available to your applications and users.

To help you quantify the actual impact, we used the Netperf benchmark in the Principled Technologies labs to compare the TCP and UDP streaming network performance of Red Hat Enterprise Linux 6 and Microsoft Windows Server 2012.

We found that Red Hat Enterprise Linux 6 delivered better TCP and UDP network performance than Windows Server 2012 in most of our tests, in both out-of-box and optimized configurations. Because network performance is crucial in many business applications, selecting the appropriate operating system is critical to help you achieve your infrastructure's maximum potential.

MAY 2013
A PRINCIPLED TECHNOLOGIES TEST REPORT
Commissioned by Red Hat, Inc.

BETTER NETWORK PERFORMANCE

We used the Netperf benchmark to compare the network performance of two operating systems: Red Hat Enterprise Linux 6 and Microsoft Windows Server 2012. For the tests, we first configured both solutions with out-of-box (default) settings, and then we tested those solutions using multiple tuning parameters to deliver optimized results. We performed the test three times and report the results from the median run. For detailed system configuration information, see Appendix A. For details on how we tested, see Appendix B.

In our TCP throughput tests using the Netperf benchmark with out-of-box settings, we found that the server running Red Hat Enterprise Linux 6 could deliver over three times more throughput at message sizes ranging from 512B to 64KB than the Microsoft Windows Server 2012 solution. In the optimized configuration using the same benchmark, we recorded nearly two times more TCP throughput for Red Hat Enterprise Linux 6 at message sizes ranging from 512B to 64KB. Figures 1 and 2 show the TCP throughput that both solutions achieved throughout our test.

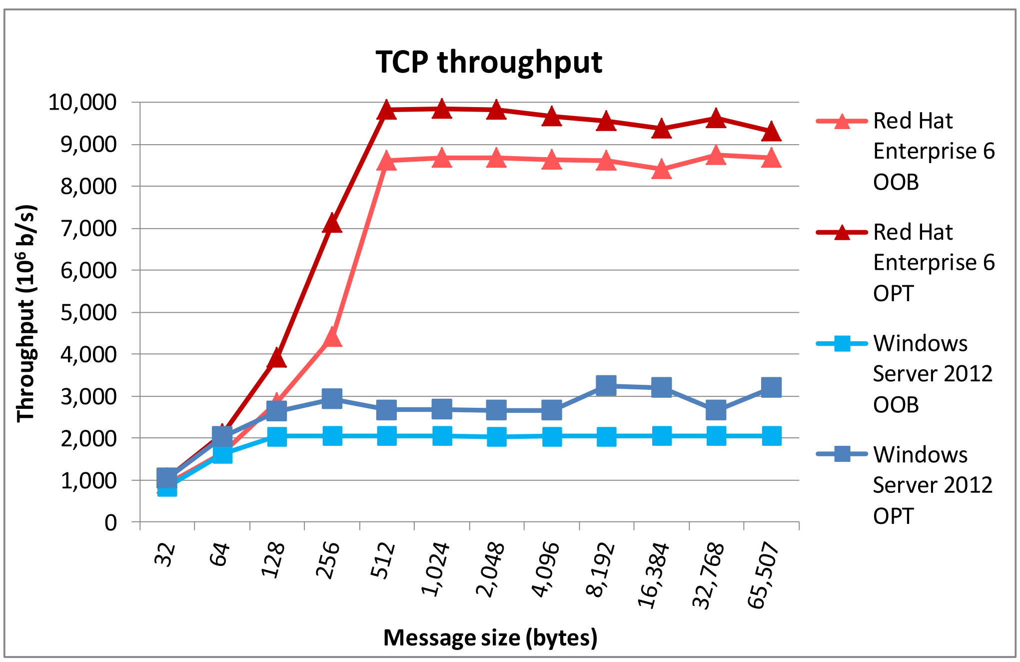


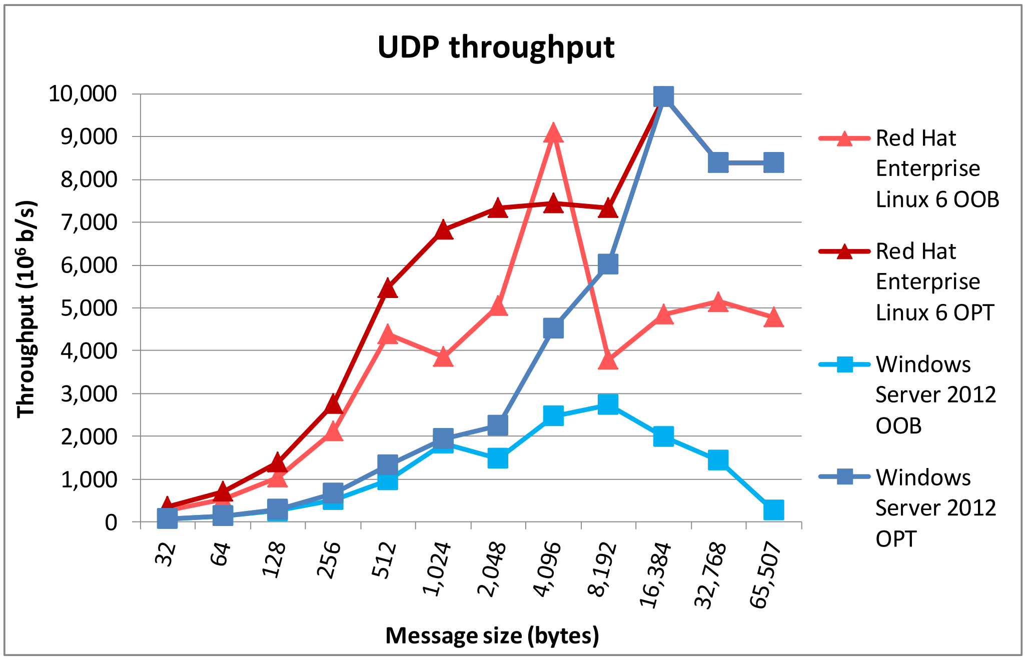
Figure 1: TCP throughput, in 10^6 b/s, using varying message sizes for both solutions, in the out-of-box (OOB) and optimized (OPT) configurations. Higher throughputs are better.

Message size (B)	TCP streaming throughput (10^6 b/s)					
	Out-of-box		Optimized		Red Hat advantage	
	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Out-of-box configuration	Optimized configuration
32	897.68	832.68	1,044.92	1,039.45	7.81%	0.53%
64	1,658.16	1,627.13	2,095.32	2,029.09	1.91%	3.26%
128	2,851.11	2,041.81	3,907.90	2,643.84	39.64%	47.81%
256	4,419.73	2,047.50	7,129.39	2,923.70	115.86%	143.85%
512	8,607.41	2,045.76	9,828.32	2,671.09	320.74%	267.95%
1,024	8,678.31	2,056.95	9,840.46	2,673.83	321.90%	268.03%
2,048	8,683.02	2,037.70	9,824.61	2,665.22	326.12%	268.62%
4,096	8,630.58	2,042.16	9,660.46	2,666.18	322.62%	262.33%
8,192	8,624.96	2,041.91	9,549.95	3,235.02	322.40%	195.21%
16,384	8,422.96	2,048.03	9,381.22	3,200.53	311.27%	193.11%
32,768	8,750.67	2,046.31	9,633.31	2,662.86	327.63%	261.77%
65,507	8,679.33	2,047.97	9,301.49	3,202.70	323.80%	190.43%

Figure 2: TCP streaming throughput in 10^6 bits per second for the two solutions, both out-of-box and optimized. Higher values are better.

In our UDP throughput tests using the Netperf benchmark with out-of-the-box settings, we found that the server running Red Hat Enterprise Linux 6 could deliver nearly three times as much throughput at the 4KB message size than the Microsoft Windows Server 2012 solution. In the optimized configuration using the same benchmark, we recorded more than twice as much UDP throughput at message sizes ranging from 32B to 2KB. Figures 3 and 4 show the UDP throughput that both solutions achieved throughout our test.

Figure 3: UDP throughput, in 10^6 b/s, using varying message sizes for both solutions, in the out-of-box (OOB) and optimized (OPT) configurations. Higher throughputs are better.



Message size (B)	UDP streaming throughput (10^6 b/s)					
	Out-of-box		Optimized		Red Hat advantage	
	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Out-of-box configuration	Optimized configuration
32	261.20	67.21	348.03	72.72	288.63%	378.59%
64	518.32	126.21	698.38	139.04	310.68%	402.29%
128	1,040.83	254.29	1,392.89	278.09	309.31%	400.88%
256	2,111.50	505.04	2,758.45	660.69	318.09%	317.51%
512	4,375.00	974.90	5,473.36	1,314.16	348.76%	316.49%
1,024	3,860.51	1,834.50	6,836.92	1,947.50	110.44%	251.06%
2,048	5,049.16	1,483.68	7,325.47	2,248.62	240.31%	225.78%
4,096	9,086.05	2,466.47	7,450.47	4,526.42	268.38%	64.60%
8,192	3,793.38	2,724.16	7,344.22	6,016.42	39.25%	22.07%
16,384	4,835.84	1,982.14	9,924.79	9,923.92	143.97%	0.01%
32,768	5,142.80	1,436.45	8,393.38	8,392.39	258.02%	0.01%
65,507	4,789.19	258.90	8,389.68	8,388.79	1749.82%	0.01%

Figure 4: UDP streaming throughput in 10^6 bits per second for the two solutions, both out-of-box and optimized. Higher values are better.

In our TCP throughput tests using auto-tuning features of both solutions, we found that the Windows Server 2012 solution delivered much less throughput than the Red Hat Enterprise Linux 6 solution (see Figure 1). To overcome this, we found that by adjusting application-level settings, specifically the receive socket size, Windows was able to achieve near 10Gbps performance. To demonstrate this behavior, we fixed the message size at 4KB and varied the receive socket size (see Figure 5). Although we used a specialized benchmark application that allowed this type of tuning, some applications do not have the necessary interfaces and mechanisms to enable it, which could result in sub-optimal performance. This highlights the importance of an operating system's capacity to manage networking resources efficiently.

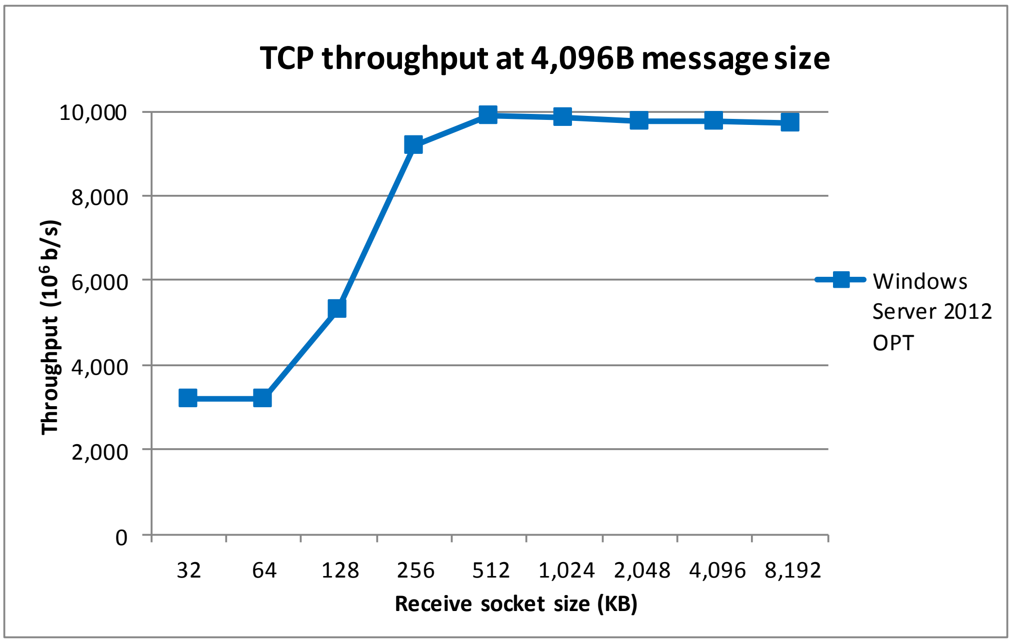
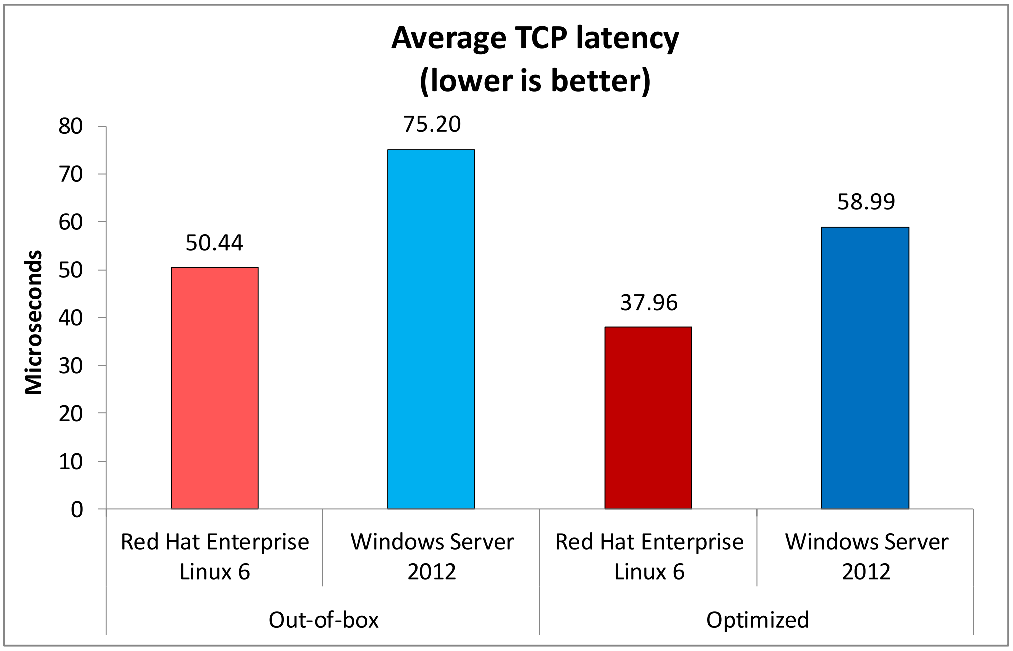


Figure 5: TCP throughput (10^6 b/s) using varying receive-socket sizes for Windows Server 2012 in the optimized configuration. Higher throughputs are better.

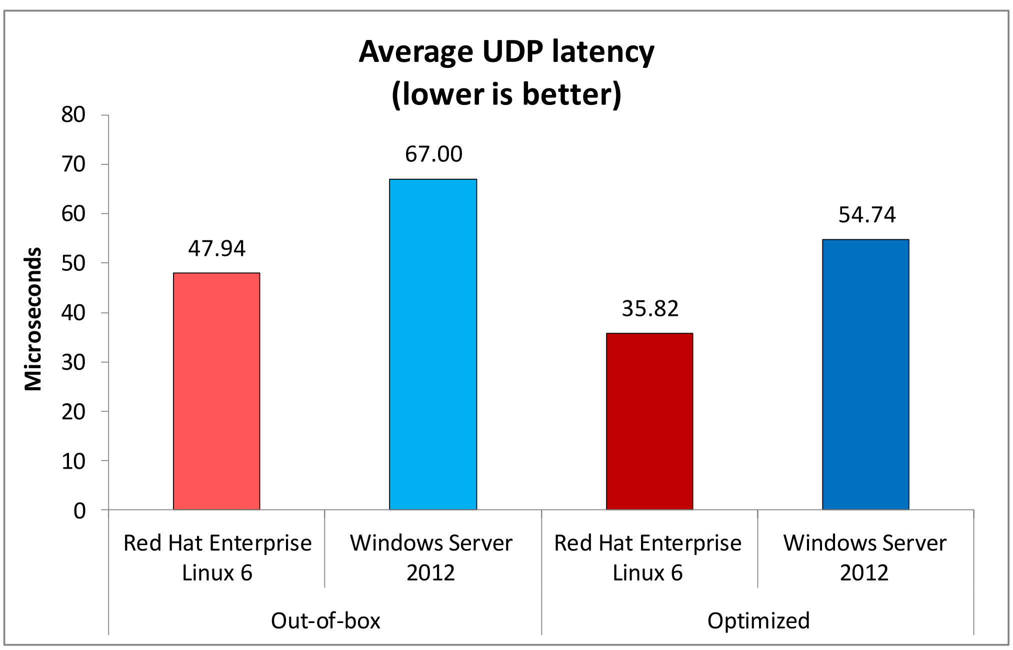
Network latency is another way to assess network performance. Quantity of data sent or received in a given amount of time is sometimes not as important as how long it takes your network to transmit and receive a fixed amount of data. A smaller latency is indicative of a more efficient network infrastructure. As Figure 6 shows, Red Hat Enterprise Linux 6 delivered significantly lower average round-trip TCP latencies than Windows Server 2012. Across varying message sizes, Red Hat Enterprise Linux 6 demonstrated 32.9 percent lower latency on average with an out-of-box configuration, and 35.6 percent lower latency on average with an optimized configuration.

Figure 6: Round-trip TCP latency times averaged over message sizes for the two solutions, out-of-box and optimized. Lower latencies are better.



As Figure 7 shows, Red Hat Enterprise Linux 6 delivered significantly lower average round-trip UDP latencies than Windows Server 2012. Across varying message sizes, Red Hat Enterprise Linux 6 demonstrated, on average, 28.5 percent lower latency in an out-of-box configuration and 34.6 percent lower latency in an optimized configuration.

Figure 7: Round-trip UDP latency times averaged over message sizes for the two solutions, out-of-box and optimized. Lower latencies are better.



See Appendix C for detailed latency results.

WHAT WE TESTED

About Red Hat Enterprise Linux 6

Red Hat Inc. positions Red Hat Enterprise Linux 6 as a prime competitor to proprietary operating systems found in enterprise data centers. Red Hat Enterprise Linux 6 is designed to deliver performance and scalability for both small and large servers with documented scalability up to 4,096 CPUs and 64 terabytes of RAM. It provides native support for the majority of the latest and most important enterprise datacenter technologies, such as 40Gb Ethernet networking and KVM virtualization as well as InfiniBand®, FCoE, and iSCSI protocols. According to Red Hat, Red Hat and its hardware partners are enabling reliability, availability, serviceability (RAS), and scalability features to help minimize downtime, increase availability, and protect data. Red Hat includes as part of its Linux offering tested, open source applications. For more information about Red Hat Enterprise Linux 6, see www.redhat.com/f/pdf/rhel/RHEL6_datasheet.pdf.

About Netperf

The Netperf benchmark measures different types of networking performance, most notably TCP or User Datagram Protocol (UDP) bulk-data transfer and request/response performance (a measure of network latency). We used Netperf to generate TCP and UDP messages of a fixed but tunable size for 120 seconds for the bulk-data transfer test, and 10 seconds for the request/response test.

Netperf is a client-server application. We ran the netperf server on the server under test, which ran either Red Hat Enterprise Linux or Windows, and the netperf client, or traffic generator, on the second Dell™ PowerEdge™ R720 server, which ran Red Hat Enterprise Linux 6. For more information about Netperf, visit www.netperf.org/netperf/.

IN CONCLUSION

Understanding how your choice of operating system affects network performance can be extremely valuable as you plan your infrastructure. Throughout our network tests, we found that the open-source Red Hat Enterprise Linux 6 solution delivered up to three times better TCP throughput than Microsoft Windows Server 2012 in an out-of-box configuration, and up to two times better throughput in an optimized configuration. In addition, Red Hat Enterprise Linux 6 delivered better UDP throughput at various message sizes. By choosing an operating system that can deliver strong network performance without manual tuning, and can increase network performance when tuned, you are giving your applications greater potential to perform well, which could translate to better user experience and improved productivity across your enterprise.

APPENDIX A – SYSTEM CONFIGURATION INFORMATION

Figure 8 shows the system configuration information for the servers we used in our tests.

System	Dell PowerEdge R720 #1 (Server under test)	Dell PowerEdge R720 #2 (traffic generator)
Power supplies		
Total number	2	2
Vendor and model number	Dell D750E-S1	Dell D750E-S1
Wattage of each (W)	750	750
Cooling fans		
Total number	6	6
Vendor and model number	AVC DBTC0638B2V	AVC DBTC0638B2V
Dimensions (h x w) of each	2.5” x 2.5”	2.5” x 2.5”
Volts	12	12
Amps	1.2	1.2
General		
Number of processor packages	2	2
Number of cores per processor	8	8
Number of hardware threads per core	2	2
System power management policy	Performance Per Watt (DAPC) or Performance Per Watt (OS) (see step 30 in section Installing Red Hat Enterprise Linux 6.4)	Performance Per Watt (OS)
CPU		
Vendor	Intel®	Intel
Name	Xeon®	Xeon
Model number	E5-2690	E5-2680
Stepping	6	6
Socket type	2011LGA	2011LGA
Core frequency (GHz)	2.90	2.70
Bus frequency	100	100
L1 cache	32 KB I + 32 KB D (per core)	32 KB I + 32 KB D (per core)
L2 cache	256 KB on chip (per core)	256 KB on chip (per core)
L3 cache	20 MB	20 MB
Platform		
Vendor and model number	Dell PowerEdge R720	Dell PowerEdge R720
Motherboard model number	OM1GCR	OM1GCR
BIOS name and version	Dell 1.5.1	Dell 1.5.1
BIOS settings	Default	Default, but without hyperthreads
Memory module(s)		
Total RAM in system (GB)	128	64
Vendor and model number	Samsung® M393B1K70BH1-CH9	Hynix® HMT31GR7BFR4A-H9
Type	PC3-10600	PC3-10600
Speed (MHz)	1,333	1,333

System	Dell PowerEdge R720 #1 (Server under test)	Dell PowerEdge R720 #2 (traffic generator)
Speed running in the system (MHz)	1,333	1,333
Timing/Latency (tCL-tRCD-tRP-tRASmin)	9-9-9-36	9-9-9-36
Size (GB)	8	8
Number of RAM module(s)	8	4
Chip organization	Double-sided	Double-sided
Rank	Dual	Dual
Operating system #1		
Name	Red Hat Enterprise Linux 6.4	Red Hat Enterprise Linux 6.4
Filesystem	Ext4	Ext4
Kernel	2.6.32-358.0.1.el6.x86_64, or 2.6.32-358.1.1.el6.x86_64 (see the section Optimizing operating system configurations)	2.6.32-358.1.1.el6.x86_64
Language	English	English
Operating system #2		
Name	Windows Server 2012 Datacenter Edition	N/A
Build number	9200	N/A
Filesystem	NTFS	N/A
Kernel	ACPI x64-based PC	N/A
Language	English	N/A
Graphics		
Vendor and model number	Matrox® G200eR	Matrox G200eR
Graphics memory (MB)	16	16
Driver	Matrox Graphics, Inc 2.3.3.0 (8/19/2011)	Matrox Graphics, Inc 2.3.3.0 (8/19/2011)
RAID controller		
Vendor and model number	Dell PERC H710P Mini	Dell PERC H710P Mini
Firmware version	21.1.0-007	21.1.0-007
Cache size	1 GB	1 GB
RAID configuration	OS #1 RAID 1 configuration of two disks OS #2 RAID 1 configuration of two disks	RAID 1 configuration of two disks
Hard drives type #1		
Vendor and model number	Fujitsu® MBB2073RC	Dell WD300BKHG-18A29V0
Number of drives	2	2
Size (GB)	73	300
Buffer size (MB)	16	16
RPM	10K	10K
Type	SAS	SAS

System	Dell PowerEdge R720 #1 (Server under test)	Dell PowerEdge R720 #2 (traffic generator)
Hard drives type #2		
Vendor and model number	Seagate Savvio® ST9146803SS	N/A
Number of drives	2	N/A
Size (GB)	146	N/A
Buffer size (MB)	16	N/A
RPM	10K	N/A
Type	SAS	N/A
Ethernet adapters #1		
Vendor and model number	Intel Gigabit 4P I350-t rNDC	Intel Gigabit 4P I350-t rNDC
Type	Internal	Internal
Ethernet adapters #2		
Vendor and model number	Intel Ethernet Server Adapter X520-SR1	Intel Ethernet Server Adapter X520-SR2
Type	PCIe	PCIe
Optical drive(s)		
Vendor and model number	TEAC DV-28SW	TEAC DV-28SW
Type	DVD-ROM	DVD-ROM
USB ports		
Number	4 external, 1 internal	4 external, 1 internal
Type	2.0	2.0

Figure 8: Configuration information for our test systems.

APPENDIX B – HOW WE TESTED

In this section, we present the step-by-steps for how we tested. First, we cover operating system installation and optimization settings (for optimized solution testing only) for both solutions. Then, we detail how to install and test using the Netperf benchmark.

Installing Red Hat Enterprise Linux 6.4

Perform steps 1 through 22 on both Netperf servers. We configured the Netperf traffic generator, which is not under test, as an optimized server. See those steps below.

1. Insert and boot from the Red Hat Enterprise Linux 6.4 x86_64 installation DVD.
2. At the welcome screen, select Install or upgrade an existing system, and press Enter.
3. At the Media test screen, select Skip, and press Enter.
4. At the Red Hat Enterprise Linux 6 title screen, click Next.
5. At the Choose an Installation Language screen, select English, and click Next.
6. At the Keyboard Type screen, select U.S. English, and click Next.
7. At the Storage Devices screen, select Basic Storage Devices, and click Next.
8. If a warning for device initialization appears, select Yes, discard any data.
9. At the Name the Computer screen, type the host name, and click Configure Network.
10. At the Network Connections screen, select the server's main or management network interface, and click Edit.
11. At the Editing network interface screen, check Connect Automatically.
12. On the same screen, select the IPv4 Settings tab, change the Method to Manual, and click Add.
13. On the same screen, enter the IP address, Netmask, Gateway, and DNS server. Click Apply.
14. Click Close on the Network Connections screen, and click Next on the Name the Computer screen.
15. At the Time zone selection screen, select the appropriate time zone, and click Next.
16. Enter the root password in the Root Password and Confirm fields, and click Next.
17. At the Assign Storage Devices screen, from the list in the left column, select the Linux disk, and click the arrow to copy the device to the right column. Next to the Linux disk, click the Boot radio button, and click Next.
18. At the Partition selection screen, select Replace Existing Linux System(s), and click Next.
19. If a warning appears, click Write changes to disk.
20. At the default installation screen, click Next to begin the installation.
21. At the Congratulations screen, click Reboot.
22. After the system reboots, log in as root.

Complete the following additional configuration steps for the second Dell PowerEdge R720 server, which is the netperf traffic generator.

1. Disable SELinux by editing the file /etc/selinux/config and change the line `SELINUX=enforcing` to `SELINUX=disabled`.
2. Update the Linux kernel to version 2.6.32-358.6.1.el6.x86_64 by installing these RPMs:
   ```
   yum localinstall kernel-2.6.32-358.1.1.el6.x86_64.rpm \
         kernel-firmware-2.6.32-358.1.1.el6.noarch.rpm
   ```
3. Replace irqbalance with version -0.55-34, and install version 0.2.19-12 of the tuned daemon and utilities.

```
yum localinstall irqbalance-0.55-34.el6.x86_64.rpm tuned-0.2.19-12.el6.noarch.rpm \
      tuned-utils-0.2.19-12.el6.noarch.rpm
```

4. Set the tuned profile to throughput-performance for the TCP and UDP throughput tests, and to latency-performance for the TCP and UDP latency test.
   ```
   tuned-adm profile throughput-performance        # throughput tests
   ```
   or
   ```
   tuned-adm profile latency-performance           # latency tests
   ```
5. Increase the size of the network queues from 100 to250,000 by adding the line `net.core.netdev_max_backlog=250000` to /etc/sysctl.conf and executing the command sysctl -p
6. Stop unneeded services by running the script **DisableSomeDefaultServices.sh** (see the Optimizing operating system configurations section below).
7. Reboot the server, and enter BIOS configuration.
8. Modify the BIOS settings to disable hyperthreading, and set the system power configuration to Performance per Watt Optimized (OS).
9. Exit BIOS configuration, and boot the server.

Installing Microsoft Windows Server 2012 Datacenter

1. Insert and boot from the Windows Server 2012 Datacenter installation DVD.
2. At the first Window Setup screen, keep the defaults for installation language, time/currency format, and keyboard input method. Click Next.
3. At the second Windows Setup screen, click Install now.
4. At the third Windows Setup screen, enter the Windows activation key, and click Next.
5. At the fourth Windows Setup screen, select the Windows Server 2012 Datacenter (Server with a GUI), and click Next.
6. At the fifth Windows Setup screen, select the checkbox to accept the license term, and click Next.
7. At the sixth Windows Setup screen, click Custom: Install Windows only (advanced).
8. At the seventh Windows Setup screen, select Drive 2 as the Windows installation drive, and click Next to start installation.
9. The system will reboot. At the Settings screen, enter the password for the Administrator (twice), and click Finish.

Optimizing operating system configurations

Before running tests for the optimized Red Hat Enterprise Linux 6 configuration, run the following bash scripts. Red Hat Enterprise Linux 6.4 uses the tuned utility with the throughput-performance and latency-performance profiles to configure the server for better performance and to run the CPUs and network devices at high performance.

First, perform the second set of steps (1 through 9) in the Installing Red Hat Enterprise Linux 6.4 section on the server under test.

DisableSomeDefaultServices.sh

```
#!/bin/bash
## For the optimized-configuration tests, disable unneeded services
## March 2013
for i in abrt-ccpp abrt-oops abrtd acpid atd auditd autofs \
    avahi-daemon cgconfig crond cups haldaemon irqbalance kdump\
    libvirt-guests mcelogd mdmonitor messagebus portreserve\
    postfix rhnsd rhsmcertd rpcbind rpcgssd rpcidmapd certmonger\
    netfs sysstat; do
  service $i stop
done
service lvm2-monitor force-stop
## end of DisableSomeDefaultServices.sh
```

Before running tests for the optimized Windows Server 2012 configuration, run the following batch script. In particular, the OS power profile is set to High performance and the desktop GUI is configured for high performance.

CommandsNoPersonaManagement.bat

```
rem Note: script closely adapted from
rem http://mtellin.com/2010/09/13/creating-a-windows-7-template-for-vmware-
view/
rem Version dated 2012-02-05
reg load "hku\temp" "%USERPROFILE%\..\Default User\NTUSER.DAT"
reg ADD "hku\temp\Software\Policies\Microsoft\Windows\Control Panel\Desktop"
/v SCRNSAVE.EXE /d "%windir%\system32\scrnsave.scr" /f
reg ADD "hku\temp\Software\Policies\Microsoft\Windows\Control Panel\Desktop"
/v ScreenSaveTimeOut /d "600" /f
reg ADD "hku\temp\Software\Policies\Microsoft\Windows\Control Panel\Desktop"
/v ScreenSaverIsSecure /d "1" /f
reg ADD "hku\temp\Software\Microsoft\Windows\CurrentVersion\Policies\System"
/v Wallpaper /d " " /f
reg ADD "hku\temp\Software\Microsoft\Windows\CurrentVersion\Internet
Settings\Cache" /v Persistent /t REG_DWORD /d 0x0 /f
reg ADD "hku\temp\Software\Microsoft\Feeds" /v SyncStatus /t REG_DWORD /d
0x0 /f
reg ADD
"hku\temp\Software\Microsoft\WIndows\CurrentVersion\Policies\Explorer" /v
HideSCAHealth /t REG_DWORD /d 0x1 /f
reg unload "hku\temp"

reg ADD "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet
Explorer\Main" /v DisableFirstRunCustomize /t REG_DWORD /d 0x1 /f
reg ADD "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session
Manager\Memory Management\PrefetchParameters" /v EnableSuperfetch /t
REG_DWORD /d 0x0 /f
reg ADD
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU" /v
NoAutoUpdate /t REG_DWORD /d 0x1 /f
reg ADD "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows
NT\SystemRestore" /v DisableSR /t REG_DWORD /d 0x1 /f
reg ADD "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Disk" /v
TimeOutValue /t REG_DWORD /d 200 /f
```

```
reg ADD "HKEY_LOCAL_MACHINE\SOFTWARE\Image" /v Revision /t REG_SZ /d 1.0 /f

reg ADD "HKEY_LOCAL_MACHINE\SOFTWARE\Image" /v Virtual /t REG_SZ /d Yes /f
reg ADD
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\Application"
/v MaxSize /t REG_DWORD /d 0x100000 /f
reg ADD
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\Application"
/v Retention /t REG_DWORD /d 0x0 /f
reg ADD
"HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Network\NewNetworkWindo
wOff" /f
reg ADD
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\System" /v
MaxSize /t

REG_DWORD /d 0x100000 /f
reg ADD
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\System" /v
Retention /t REG_DWORD /d 0x0 /f
reg ADD
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\Security" /v
MaxSize /t REG_DWORD /d 0x100000 /f
reg ADD
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\Security" /v
Retention /t REG_DWORD /d 0x0 /f
reg ADD "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\CrashControl"
/v CrashDumpEnabled /t REG_DWORD /d 0x0 /f
reg ADD
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\Explo
rer" /v NoRecycleFiles /t REG_DWORD /d 0x1 /f
reg ADD "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server" /v fDenyTSConnections /t REG_DWORD /d 0x0 /f
reg ADD "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server\WinStations\RDP-Tcp" /v UserAuthentication /t REG_DWORD /d 0x0 /f
reg ADD
"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\policies\syste
m" /v EnableLUA /t REG_DWORD /d 0x0 /f
reg Add "HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Sideshow" /v
Disabled /t REG_DWORD /d 0x1 /f

Powershell Set-Service 'BDESVC' -startuptype "disabled"
Powershell Set-Service 'wbengine' -startuptype "disabled"
Powershell Set-Service 'DPS' -startuptype "disabled"
Powershell Set-Service 'UxSms' -startuptype "disabled"
Powershell Set-Service 'Defragsvc' -startuptype "disabled"
Powershell Set-Service 'HomeGroupListener' -startuptype "disabled"
Powershell Set-Service 'HomeGroupProvider' -startuptype "disabled"
Powershell Set-Service 'iphlpsvc' -startuptype "disabled"
Powershell Set-Service 'MSiSCSI' -startuptype "disabled"
Powershell Set-Service 'swprv' -startuptype "disabled"
Powershell Set-Service 'CscService' -startuptype "disabled"
```

```
Powershell Set-Service 'SstpSvc' -startuptype "disabled"
Powershell Set-Service 'wscsvc' -startuptype "disabled"
Powershell Set-Service 'SSDPSRV' -startuptype "disabled"
Powershell Set-Service 'SysMain' -startuptype "disabled"
Powershell Set-Service 'TabletInputService' -startuptype "disabled"
Powershell Set-Service 'Themes' -startuptype "disabled"
Powershell Set-Service 'upnphost' -startuptype "disabled"
Powershell Set-Service 'VSS' -startuptype "disabled"
Powershell Set-Service 'SDRSVC' -startuptype "disabled"
Powershell Set-Service 'WinDefend' -startuptype "disabled"
Powershell Set-Service 'WerSvc' -startuptype "disabled"
Powershell Set-Service 'MpsSvc' -startuptype "disabled"
Powershell Set-Service 'ehRecvr' -startuptype "disabled"
Powershell Set-Service 'ehSched' -startuptype "disabled"
Powershell Set-Service 'WSearch' -startuptype "disabled"
Powershell Set-Service 'wuauserv' -startuptype "disabled"
Powershell Set-Service 'Wlansvc' -startuptype "disabled"
Powershell Set-Service 'WwanSvc' -startuptype "disabled"
bcdedit /set BOOTUX disabled
vssadmin delete shadows /All /Quiet
Powershell disable-computerrestore -drive c:\
netsh advfirewall set allprofiles state off
powercfg -H OFF
powercfg -setactive 8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c
net stop "sysmain"
fsutil behavior set DisableLastAccess 1
schtasks /change /TN "\Microsoft\Windows\Defrag\ScheduledDefrag" /Disable
schtasks /change /TN "\Microsoft\Windows\SystemRestore\SR" /Disable
schtasks /change /TN "\Microsoft\Windows\Registry\RegIdleBackup" /Disable
schtasks /change /TN "\Microsoft\Windows Defender\MPIdleTask" /Disable
schtasks /change /TN "\Microsoft\Windows Defender\MP Scheduled Scan"
/Disable
schtasks /change /TN "\Microsoft\Windows\Maintenance\WinSAT" /Disable
rem End of CommandsNoPersonaManagement.bat
```

TESTING WITH NETPERF

We used Netperf version 2.4.5 from ftp://ftp.netperf.org/netperf/archive/netperf-2.4.5.tar.bz2. We compiled it for Red Hat Enterprise Linux 6 and for Windows Server 2012 as follows.

1. On the second server, running Red Hat Enterprise 6.4, make sure the GCC compile environment is installed.
   ```
   yum install gcc make
   ```
2. Extract the source code from the archive.
   ```
   tar jzf netperf-2.4.5.tar.bz2
   cd netperf-2.4.5
   ```
3. Configure and compile netperf for Red Hat Enterprise Linux.
   ```
   ./configure
   make all test
   ```
4. Copy the netperf and netserver binaries on each Red Hat server.
   ```
   mkdir /root/np24
   cp src/netperf src/netserver /root/np24
   ```

For Windows Server 2012, we chose to compile Netperf using the Cygwin environment rather than using the Windows Driver Kit. We installed version Cygwin 1.7.18-1 under Windows Server 2012 on the server under test, and compiled Nerperf in a Cygwin bash shell by following steps 2 and 3 above. We copied the binaries to the folder Netperf\np24 under the Desktop.

- To run the TCP and UDP throughput tests for the Red Hat Enterprise Linux server, we ran the bash script **run_netserver_server1.sh** (see below)on the first server, the server under test. We ran the bash script **run_netperf_STREAM_server2.sh (**see below) on the second server, the traffic generator and captured its output. We rebooted both servers between runs.

- To run the TCP and UDP latency tests for the Red Hat Enterprise Linux server, we ran the bash script **run_netserver_server1.sh** (see below)on the first server, the server under test. We ran the bash script **run_netperf_RR_server2.sh (**see below) on the second server, the traffic generator and captured its output. We rebooted both servers between runs.

- To run the TCP and UDP throughput tests for the Windows Server 2012 server, we ran the bash script **run_netserver_windows.bat** (see below) on the first server, the server under test. We ran the bash script **run_netperf_STREAM_windows.bat (**see below) on the second server, the traffic generator and captured its output. We rebooted both servers between runs.

- To run the TCP and UDP latency tests for the Windows Server 2012 server, we ran the bash script **run_netserver_windows.bat** (see below)on the first server, the server under test. We ran the bash script **run_netperf_RR_windows.bat (**see below) on the second server, the traffic generator and captured its output. We rebooted both servers between runs.

run_netserver_server1.sh

```
#!/bin/bash

tuna  --socket 0 --include
```

```
##tuna  --irq p6p1\* --cpus 2,4,6,8,10,12,14 --spread
tuna  --irq p6p1\* --cpus 2,4,6,8,10,12,14,16,18,20,22,24,26,28,30 --spread
tuna  --irq em1\*  --socket 1 --spread
tuna  --show_irqs
tuna  --cpu 0 --include

numactl -l -N 0 nice -n -10 /root/np24/netserver -4 -L 192.168.41.234
tuna --cpu 0 --isolate -t netserver –move
## End of file run_netserver_server1.sh
```

run_netperf_STREAM_server2.sh

```
#!/bin/bash

## tuna: performance settings to move the NIC’s IRQs and
## place netperf on CPU 1
tuna --socket 1 --isolate
tuna --irq p4p2\* --cpus 3,5,7,9,11,13,15 --spread
tuna --irq em1\*  --socket 0 --spread
tuna --show_irqs --show_threads
tuna --cpu 1 --include

TEST_TIME=120
TESTS="TCP_STREAM UDP_STREAM"
SIZE="32 64 128 256 512 1024 2048 4096 8192 16384 32768 65507"
ME="236"
HOST="234"
BACKCHANNEL="10.41.4."
SUBNET="192.168.41."
EXE="numactl -l -N 1 /bin/nice -n -10 /root/np24/netperf"

function run_loop {
  #Pull the first arg to be the traffic generator
  GENSSH=$BACKCHANNEL$1
  GEN=$SUBNET$1
  #Pull the first arg to be the traffic target
  targ=$2

  for big in $SIZE; do
    for test in $TESTS; do
      echo
      echo " ********************************************************** "
      echo " **************    Starting new test     ****************** "
      echo " ********************************************************** "
      printf "Start of test: "; date

      echo "$EXE -P0 -l $TEST_TIME -H $SUBNET$targ -t $test" \
            "-B \"running $test between $GEN and $SUBNET$targ -- -m $big\" "
      $EXE -P0 -l $TEST_TIME -H $SUBNET$targ -t $test \
            -B "running $test between $GEN and $SUBNET$targ -- -m $big" -- -
m $big &
      tuna --cpu 1 --isolate -t netperf --move
```

```
        # Since everything is in the background
        wait
        printf "End of test: "; date

        # Let things quiesce
        sleep 15

      done
    done

    echo ; echo
  }

  printf "Start of run: "; date

  run_loop $ME $HOST

  printf "End of run: "; date
  exit
  ## End of file run_netperf_STREAM_server2.sh
```

run_netperf_RR_server2.sh

```
#!/bin/bash

## tuna: performance settings to move the NIC's IRQs and
## place netperf on CPU 1
tuna --socket 1 --isolate
tuna --irq p4p2\* --cpus 3,5,7,9,11,13,15 --spread
tuna --irq em1\* --socket 0 --spread
tuna --show_irqs --show_threads
tuna --cpu 1 --include

TEST_TIME=10
TESTS="TCP_RR UDP_RR"
SIZE="1 2 4 8 16 32 64 128 256 512 1024"
ME="236"
HOST="234"
BACKCHANNEL="10.41.4."
SUBNET="192.168.41."
EXE="numactl -l -N 1 nice -n -10 /root/np24/netperf"

function run_loop {
  #Pull the first arg to be the traffic generator
  GENSSH=$BACKCHANNEL$1
  GEN=$SUBNET$1
  #Pull the first arg to be the traffic target
  targ=$2

  for big in $SIZE; do
    for test in $TESTS; do
      echo
      echo " ********************************************************* "
```

```
        echo " *************    Starting new test     ***************** "
        echo " ****************************************************** "
        printf "Start of test: "; date

        echo "$EXE -P0 -l $TEST_TIME -H $SUBNET$targ -t $test" \
              "-B \"running $test between $GEN and $SUBNET$targ -- -m $big\" "
        $EXE -P0 -l $TEST_TIME -H $SUBNET$targ -t $test \
              -B "running $test between $GEN and $SUBNET$targ -- -m $big" -- -
m $big &
        tuna --cpu 1 --isolate -t netperf --move

        # Since everything is in the background
        wait
        printf "End of test: "; date

        # Let things quiesce
        sleep 15

      done
    done

    echo ; echo
}

printf "Start of run: "; date

run_loop $ME $HOST

##run_loop $HOST $ME


printf "End of run: "; date

exit
## End of file run_netperf_RR_server2.sh
```

run_netserver_windows.bat

```
@echo off
echo Starting netserver on node 0
c
Rem start /high /node 0 /affinity 0x1 /b /w np24\netserver -4 -L
192.168.41.234
start /high /node 0 /b /w np24\netserver -4 -L 192.168.41.234
@echo on
rem End of file run_netserver_windows.bat
```

run_netperf_RR_windows.bat

```
@echo off

SETLOCAL EnableDelayedExpansion

set file="lp-wins-%1.txt"
```

```
set test_time=10
set targ=192.168.41.236
set serv=192.168.41.234
echo Data written to %file%

for %%b in (1 2 4 8 16 32 64 128 256 512 1024) do (
   for %%t in (TCP_RR UDP_RR) do (
      echo " ********************************************************** "
      echo " **************    Starting new test (%%b, %%t) *********** "
      echo " ********************************************************** "
      date /t
      time /t

      call np %test_time% %%t %%b %targ% %serv% >> %file%

      echo End of run:
      date /t
      time /t
      choice /t 15 /c yn /d y > NUL 2>&1 )
   echo " "
   echo " ")
echo End of test >> %file%
date /t >> %file%
time /t >> %file%
ENDLOCAL
rem End of file run_netperf_RR_windows.bat
```

run_netperf_STREAM_windows.bat

```
@echo on
SETLOCAL EnableDelayedExpansion

set file="np-wins-%1.txt"

set test_time=120
set targ=192.168.41.236
set serv=192.168.41.234
echo Data written to %file%

for %%b in (32 64 128 256 512 1024 2048 4096 8192 16384 32768 65507) do (
   for %%t in (TCP_STREAM UDP_STREAM) do (
      echo " ********************************************************** "
      echo " **************    Starting new test (%%b, %%t) *********** "
      echo " ********************************************************** "
      date /t
      time /t

      call np %test_time% %%t %%b %targ% %serv% >> %file%

      echo End of run:
      date /t
      time /t
      choice /t 15 /c yn /d y > NUL 2>&1 )
```

```
   echo " "
   echo " ")
echo End of test >> %file%
date /t >> %file%
time /t >> %file%
ENDLOCAL
@echo on
rem End of file run_netperf_STREAM_windows.bat
```

np.bat

```
@echo off
goto Start

arg 1 = test time
arg 2 = test type
arg 3 = size
arg 4 = target IP
arg 5 = host IP

:Start
date /t
time /t

echo np24\netperf -P0 -l %1 -H %4 -t %2 -B "running %2 between %5 and %4 --
-m %3"

rem Use only one of the next two “start” commands.  The first is for the
rem     optimized configuration, the second for the out-of-box
configuration.
rem start /high /node 0 /affinity 0x1 /b /w np24\netperf -P0 -l %1 -4 -H %4
-t %2 -B "running %2 between %5 and %4 -- -m %3" -- -m %3

start /high /node 0 /b /w np24\netperf -P0 -l %1 -4 -H %4 -t %2 -B "running
%2 between %5 and %4 -- -m %3" -- -m %3

date /t
time /t
rem End of file np.bat
```

APPENDIX C – DETAILED TEST RESULTS

Figures 9 and 10 present detailed round-robin trip times (latency) results for our Netperf tests.

Message size (B)	TCP round-trip latency (µs)					
	Out-of-box		Optimized		Red Hat advantage	
	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Out-of-box configuration	Optimized configuration
1	50.46	73.46	37.95	57.79	31.31%	34.33%
2	50.70	74.75	37.98	57.32	32.17%	33.74%
4	50.33	74.67	37.99	59.23	32.60%	35.86%
8	50.36	75.33	37.99	59.38	33.15%	36.02%
16	50.54	75.96	37.97	58.99	33.46%	35.63%
32	50.44	76.71	37.94	59.40	34.25%	36.13%
64	50.68	74.33	37.96	59.27	31.82%	35.95%
128	50.12	75.08	37.97	59.49	33.24%	36.17%
256	50.30	74.97	37.96	59.18	32.91%	35.86%
512	50.48	75.98	37.96	59.64	33.56%	36.35%
1,024	50.40	76.01	37.94	59.20	33.69%	35.91%
Average	**50.44**	**75.20**	**37.96**	**58.99**	**32.93%**	**35.65%**

Figure 9: TCP round-trip latency in microseconds for the two solutions, both out-of-box and optimized. Lower latencies (or higher percentage advantages) are better.

Message size (B)	UDP round-trip latency (µs)					
	Out-of-box		Optimized		Red Hat advantage	
	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Red Hat Enterprise Linux 6	Microsoft Windows Server 2012	Out-of-box configuration	Optimized configuration
1	47.93	66.92	35.79	54.76	28.38%	34.64%
2	47.79	66.62	35.85	54.71	28.26%	34.47%
4	48.06	66.35	35.81	55.14	27.57%	35.06%
8	47.96	66.94	35.82	54.47	28.35%	34.24%
16	48.07	67.26	35.83	54.99	28.53%	34.84%
32	47.93	66.92	35.81	54.23	28.38%	33.97%
64	47.98	68.32	35.79	54.99	29.77%	34.92%
128	47.87	66.70	35.84	54.54	28.23%	34.29%
256	47.80	66.82	35.80	54.60	28.46%	34.43%
512	47.99	67.16	35.83	54.90	28.54%	34.74%
1,024	47.92	66.97	35.81	54.79	28.45%	34.64%
Average	**47.94**	**67.00**	**35.82**	**54.74**	**28.45%**	**34.56%**

Figure 10: UDP round-trip latency in microseconds for the two solutions, both out-of-box and optimized. Lower latencies (or higher percentage advantages) are better.

ABOUT PRINCIPLED TECHNOLOGIES

Principled Technologies, Inc.
1007 Slater Road, Suite 300
Durham, NC, 27703
www.principledtechnologies.com

We provide industry-leading technology assessment and fact-based marketing services. We bring to every assignment extensive experience with and expertise in all aspects of technology testing and analysis, from researching new technologies, to developing new methodologies, to testing with existing and new tools.

When the assessment is complete, we know how to present the results to a broad range of target audiences. We provide our clients with the materials they need, from market-focused data to use in their own collateral to custom sales aids, such as test reports, performance assessments, and white papers. Every document reflects the results of our trusted independent analysis.

We provide customized services that focus on our clients' individual requirements. Whether the technology involves hardware, software, Web sites, or services, we offer the experience, expertise, and tools to help our clients assess how it will fare against its competition, its performance, its market readiness, and its quality and reliability.

Our founders, Mark L. Van Name and Bill Catchings, have worked together in technology assessment for over 20 years. As journalists, they published over a thousand articles on a wide array of technology subjects. They created and led the Ziff-Davis Benchmark Operation, which developed such industry-standard benchmarks as Ziff Davis Media's Winstone and WebBench. They founded and led eTesting Labs, and after the acquisition of that company by Lionbridge Technologies were the head and CTO of VeriTest.