DATABASE SCALING USING
RED HAT ENTERPRISE LINUX 6

Running multiple database instances on Red Hat Enterprise Linux® 6 more than doubled server database performance

OUR FINDINGS

As server capabilities increase, running a single database instance on a multi-socket, multi-core server can leave processor and memory resources unused. Adding database instances on Red Hat Enterprise Linux 6 can increase performance by fully utilizing all resources. Additionally, binding resources together using control groups (cgroups) can provide even more performance gains. In Principled Technologies' tests in our labs, a server running Red Hat Enterprise Linux 6 with four PostgreSQL database instances and cgroups, provided more than twice the database performance of the same server running a single database instance: 312,247 orders per minute (OPM) while running four database instances and 130,870 OPM running a single database instance.

OUR PROCESS

We tested the performance of one PostgreSQL database instance on a Dell™ PowerEdge™ R710 server running Red Hat Enterprise Linux 6 using the open-source DVD Store Version 2 (DS2) benchmark. We then used DS2 to test the performance of four database instances with cgroups. We compared the performance score from the single-instance test to that from the four-database-instances test to demonstrate the database scaling capabilities of Red Hat Enterprise Linux 6.

NOVEMBER 2010

A PRINCIPLED TECHNOLOGIES TEST REPORT

Commissioned by Red Hat, Inc.

PROJECT OVERVIEW

We tested the performance of a server running one PostgreSQL database instance on Red Hat Enterprise Linux 6, and then added three more instances to show the performance increases that occur when you add multiple instances. Running a single database instance on a multi-core system can leave memory and processor resources available, underutilizing the server. Our testing shows that adding database instances improves both server utilization and total performance.

We also show how the cgroups method of allocating systems resources to the database instances as a group of similar processes can yield further performance gains. Cgroups is a resource management method, available in Red Hat Enterprise Linux 6, that lets users allocate system resources such as CPU, disk I/O, network I/O, and memory to programs or tasks according to their importance. Users can group processes together and assign the aggregate CPU and memory resources. The Red Hat Enterprise Linux 6 operating system can often better schedule the running of the group, which can lead to performance gains compared to unaggregated processes. In our tests, we used cgroups to allocate the resources for the four database instances to improve performance.

As Figure 1 shows, the Dell PowerEdge R710 running Red Hat Enterprise Linux 6 achieved a 139 percent performance increase in the DS2 benchmark when running four database instances with cgroups over one database instance without cgroups. The single database instance could not effectively use the server's multi-core resources.

We present more detailed findings in the What We Found section.

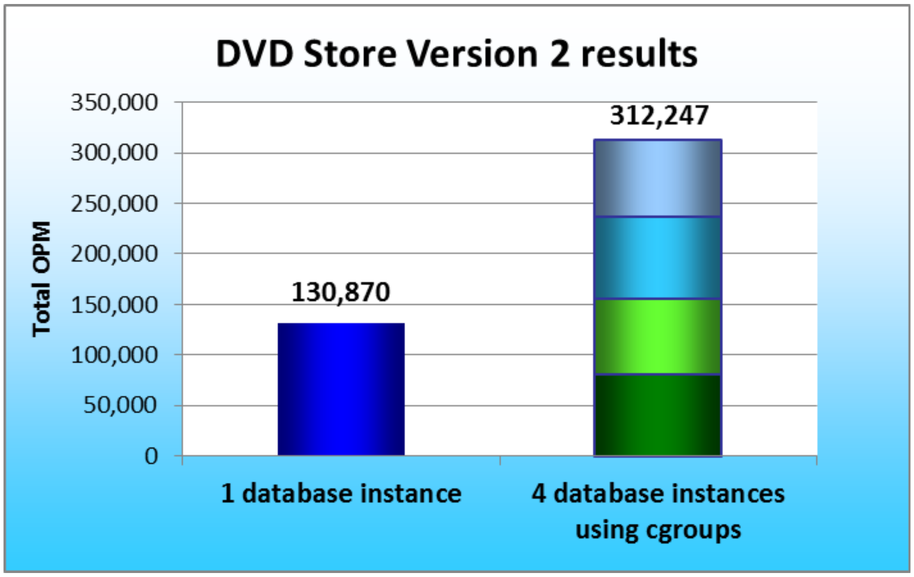

Figure 1: DVD Store Version 2 results for the server running Red Hat Enterprise Linux 6 with one database instance and four database instances using cgroups. Higher numbers are better.

WHAT WE TESTED

We used DVD Store Version 2, an open-source application that models an online DVD store, to build the workload. DS2 has a back-end database component, a front-end Web application layer, and a driver layer that executes the workload. Because our goal was to test database server performance, we did not use the front-end Web client component of DS2, but ran the included driver program directly via the client machine's command-line interface. We used the default setup configuration for DS2, with exceptions we note in the DVD Store setup section in the How We Tested section of this report. We chose PostgreSQL version 8.4.4 running on Red Hat Enterprise Linux 6 as the back-end database.

DS2 reports results in orders per minute. One DS2 order consists of a customer login, a search for movies by title, actor, or category, and a purchase. The workload also performs other actions, such as adding new customers, to exercise a wide range of database functions.

We configured the Dell PowerEdge R710 server with a 4GB DS2 database. When running multiple instances of PostgreSQL, each instance contained a 4GB DS2 database. We used two clients against each instance of PostgreSQL.

Each client machine ran a single instance of the DS2 driver, with 20 threads, to simulate a heavily loaded environment. The client machines ran with no think time, processing requests as quickly as they were able. The workload ran for 15 minutes on each server, with the multiple instances on the Dell PowerEdge R710 running the workload simultaneously.

For more details about the DS2 tool, see http://www.delltechcenter.com/page/DVD+Store.

We used DS2 to test a single database instance to determine its baseline performance. To determine the performance increase with multiple database instances, we tested the performance of the server running four database instances with cgroups configured to bind each database instance to a certain processor, thus further increasing performance.

WHAT WE FOUND

Figure 2 shows the DVD Store Version 2 OPM results, by client, for the Dell PowerEdge R710 running one database instance and running four database instances using cgroups. For both configurations, we used two clients for each instance.

	1 database instance	4 database instances using cgroups
Client 1	65,338	40,378
Client 2	65,532	40,464
Client 3		37,399
Client 4		37,691
Client 5		40,180
Client 6		40,203
Client 7		37,905
Client 8		38,027
Total OPM	**130,870**	**312,247**

Figure 2: DVD Store Version 2 OPM results, by instance, for the Dell PowerEdge R710 running one database instance and running four database instances using cgroups. Higher numbers are better.

HOW WE TESTED

Setting up and configuring the storage

Internal storage configuration

We configured two volumes on the internal storage. We configured the first volume as a RAID 1 set of two 73GB drives, and we installed the host operating system on this volume. We configured the second volume on the remaining 146GB drive and used that volume for scripts and utility files.

Dell EqualLogic™ storage configuration

Each of the two Dell EqualLogic arrays had 16 drives, for a total of 32 drives. We configured the six Dell EqualLogic arrays in regular RAID 10 mode (14 active drives and 2 spares each), for a total of 28 active drives.

We connected the Dell PowerEdge R710 server to the Dell EqualLogic storage using a Dell PowerConnect™ 6248 switch. We connected the 10Gb network card on our server to the 10GbE uplink module on the switch via a Cat6 Ethernet cable. We created two storage pools and then alternated the instances between the two pools.

We dedicated two LUNS to each instance: one 40GB LUN for data, and one 20GB LUN for logging.

Setting up the storage

1. If the Dell EqualLogic PS5000XV needs to be reset, connect through the serial port, and issue the following command upon login:

```
# reset
```

2. Using the command line wizard, proceed with the reset process, and provide a group IP address, subnet mask, and login information.
3. Perform the same procedure on the second Dell EqualLogic PS5000XV array, but provide the group IP address from Step 2 as the group to join when the wizard prompts you.

4. Using the Group Manager web application, set up a storage group with the following RAID policies:
   - PS5000XV-1: RAID-10
   - PS5000XV-2: RAID-10
   - PS5000XV-3: RAID-10
   - PS5000XV-4: RAID-10
   - PS5000XV-5: RAID-10
   - PS5000XV-6: RAID-10
5. Create a storage pool name of `pool1` with the following members: PS5000XV-1, PS5000XV-2, and PS5000XV-3.
6. Create a storage pool name of `pool2` with the following members: PS5000XV-4, PS5000XV-5, and PS5000XV-6.
7. Enable all network connections on each Dell EqualLogic PS5000XV using the following IP address scheme:
   - IP Address: `192.168.1.##` (## being 11, 13, and 15 for array 1; and 12, 14, 16 for array 2)
   - Subnet Mask: `255.255.255.0`
8. Create four 40GB volumes with no snapshot reserve, alternating them between storage pools, beginning with pool1.
9. Create four 20GB volumes with no snapshot reserve, alternating them between storage pools, beginning with pool2.
10. Create an access control record for every volume without specifying any limitations.
11. Create a read-only SNMP community name to use for group monitoring.

Setting up the host server with Red Hat Enterprise Linux 6 RC1

We installed Red Hat Enterprise Linux 6 RC1 on the host server.

Installing Red Hat Enterprise Linux 6 RC1

1. Insert and boot from the Red Hat Enterprise Linux 6 install DVD.
2. Press Enter to install using graphical mode.
3. At the media test screen, select Skip.
4. At the Red Hat Enterprise Linux 6 title screen, click Next.
5. At the Choose a Language screen, select English, and click Next.
6. At the Keyboard Type screen, select U.S. English, and click Next.
7. At the Installation Devices screen, select Basic Storage Devices, and click Next.
8. Enter the hostname dbhost.ds2.lan, and click Next.
9. At the Time zone selection screen, select the appropriate time zone, and click Next.
10. Enter the root password in the Root Password and Confirm fields, and click Next.
11. At the type of installation screen, select Use All Space, check Review and modify partitioning layout, and click Next.
12. At the partitioning layout screen, select lv_home, and click Delete. At the Confirm Delete warning, click Delete.
13. Select lv_root and click Edit. In the Size field, enter the Max size displayed. Click OK.
14. Click Next. If a warning appears, click Write changes to disk.
15. At the Boot loader screen, click Next.

16. At the default installation screen, select Software Development Workstation, Customize now, and click Next.
17. Under the Base System software list, select iSCSI Storage Client.
18. Under the Databases software list, select PostgreSQL Database client and PostgreSQL Database server, and click Next.
19. Installation will begin automatically.
20. At the Congratulations screen, click Reboot.
21. After the system reboots and the Welcome screen appears, click Forward.
22. Agree to the EULA, and click Forward.
23. At the Set Up Software Updates screen, select No, I prefer to register at a later time, and click Forward.
24. If a pop-up screen appears asking if you are sure, click No thanks.
25. At the Finish Updates Setup, click Forward.
26. At the Create User Screen, enter username `testuser` and password `Password1` and click Forward.
27. If a warning appears, click Yes.
28. At the Date and Time screen, set the time and date, and click Forward.
29. At the Kdump screen, uncheck Enable Kdump, and click Finish.
30. If a warning appears, click Yes, and click OK.

Setting up the network adapters

1. Log into the host, right-click the desktop, and choose Open Terminal.
2. Type the following command to edit the network configuration settings, where X is the relevant host NIC to modify:

```
# vi /etc/sysconfig/network-scripts/ifcfg-ethX
```

3. Modify the following lines to set the static IP address and netmask, where X is the remaining part of the relevant IP address:

```
BOOTPROTO=static
IPADDR=192.168.XXX.XXX
NETMASK=255.255.255.0
```

4. Save the file, and exit vi.
5. Type the following command to modify the remaining network settings:

```
# vi /etc/sysconfig/network
```

6. Modify the following lines to disable IPv6 and set the hostname, where “hostname” is the hostname of this server:

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=hostname.local.domain
```

7. Save the file, and exit vi.
8. Repeat steps 2 through 7 for the NIC dedicated to iSCSI traffic.
9. Reboot the server.

Configuring the iSCSI software initiator and multipathing in Red Hat Enterprise Linux 6

1. Log into the host, right-click the desktop, and choose Open Terminal.
2. Using a text editor, edit /etc/iscsi/iscsid.conf, and modify the following settings:

```
node.session.cmds_max = 1024
node.session.queue_depth = 128
```

3. Type the following sequence of commands to configure the iSCSI interface, configure iSCSI to start on boot, and to start it immediately:

```
# modprobe scsi_transport_iscsi
# modprobe bnx2i
# chkconfig iscsi on
# chkconfig iscsid on
# iscsiadm -m iface -o new -I iface1
# iscsiadm -m iface -o new -I iface2
# iscsiadm -m iface -o new -I iface3
# iscsiadm -m iface -o update -n iface.net_ifacename -v eth4 -I
iface1
# iscsiadm -m iface -o update -n iface.net_ifacename -v eth4 -I
iface2
# iscsiadm -m iface -o update -n iface.net_ifacename -v eth4 -I
iface3
# iscsiadm -m discovery -t st -p 192.168.10.10:3260
# service iscsi start
```

Adding the storage volumes to Red Hat Enterprise Linux 6

1. Issue the mkfs.ext4 command to make the file system:

```
# mkfs.ext4 -L log1 /dev/mapper/log1
# mkfs.ext4 -L log2 /dev/mapper/log2
# mkfs.ext4 -L log3 /dev/mapper/log3
# mkfs.ext4 -L log4 /dev/mapper/log4
# mkfs.ext4 -L db1 /dev/mapper/db1
# mkfs.ext4 -L db2 /dev/mapper/db2
# mkfs.ext4 -L db3 /dev/mapper/db3
# mkfs.ext4 -L db4 /dev/mapper/db4

# mkdir /vol/pglog1
# mkdir /vol/pglog2
# mkdir /vol/pglog3
# mkdir /vol/pglog4
# mkdir /vol/ds2data1
# mkdir /vol/ds2data2
# mkdir /vol/ds2data3
# mkdir /vol/ds2data4
```

2. Edit the fstab to mount the volumes on boot. Add a line for each volume, such as below.

```
/dev/mapper/log1 /vol/pglog1 ext4 _netdev,defaults 0 0
/dev/mapper/log2 /vol/pglog2 ext4 _netdev,defaults 0 0
/dev/mapper/log3 /vol/pglog3 ext4 _netdev,defaults 0 0
/dev/mapper/log4 /vol/pglog4 ext4 _netdev,defaults 0 0

/dev/mapper/db1 /vol/ds2data1 ext4 _netdev,defaults 0 0
/dev/mapper/db2 /vol/ds2data2 ext4 _netdev,defaults 0 0
/dev/mapper/db3 /vol/ds2data3 ext4 _netdev,defaults 0 0
/dev/mapper/db4 /vol/ds2data4 ext4 _netdev,defaults 0 0
```

3. Issue the following command to further configure the storage:

```
# mount -a
# tuned-adm profile enterprise-storage
```

Setting up cgroups for the database instances

1. Edit the /etc/cgconfig.conf file and add the following:

```
group cgnode0 {
     perm {
          task {
               uid = postgres;
               gid = postgres;
          }
          admin {
               uid = root;
               gid = root;
          }
     }
     cpuset {
          cpuset.cpus=0,2,4,6,8,10,12,14,16,18,20,22;
          cpuset.mems=0;
     }
}
group cgnode1 {
     perm {
          task {
               uid = postgres;
               gid = postgres;
          }
          admin {
               uid = root;
               gid = root;
          }
     }
     cpuset {
          cpuset.cpus=1,3,5,7,9,11,13,15,17,19,21,23;
          cpuset.mems=1;
     }
}
```

2. Start the four instances of PostgreSQL using the following commands:

```
# mkdir /tmp/pg{1,2,3,4}
# runuser -l postgres -c "cgexec -g cpuset:cgnode0 postgres -D
/vol/pglog1/data -k /tmp/pg1 &"
# runuser -l postgres -c "cgexec -g cpuset:cgnode1 postgres -D
/vol/pglog2/data -k /tmp/pg2 &"
# runuser -l postgres -c "cgexec -g cpuset:cgnode0 postgres -D
/vol/pglog3/data -k /tmp/pg3 &"
# runuser -l postgres -c "cgexec -g cpuset:cgnode1 postgres -D
/vol/pglog4/data -k /tmp/pg4 &"
```

Tuning the database

1. Edit the following lines of the postgresql.conf file:

```
shared_buffers = 4GB
effective_cache_size = 8GB
checkpoint_segments = 1024
checkpoint_timeout = 1h
```

Setting up DVD Store

Data generation overview

We built the database schema using the scripts in the DS2 distribution package, although we modified several as we note below. The DS2 stress tool provides options to generate 10MB, 1GB, or 100GB datasets, but not 4 GB of user data that we wished to use in the test. Accordingly, we modified the data-generation source code and the DVD Store application's scripts to generate our user data. Note: We created our test data on a Linux system to take advantage of the larger RAND MAX.

Editing the ds2_create_orders.c module

The module ds2_create_orders.c defines constants that specify the maximum values for the customer ID and the product ID. Because the constants for the 4GB database size did not exist, we added the constants for this size.

The available command-line options for specifying the size were S (small), M (medium), and L (large). We added the option R to create the 4GB database. In the switch statement that sets the values for the variables max_cust_id and max_prod_id, we added cases that assigned them the proper values for the 4GB database size.

We recompiled the ds2_create_orders.c module on Linux, following the instructions in the header comments. We used the following command line:

```
gcc -o ds2_create_orders ds2_create_orders.c -lm
```

Editing the ds2_create_cust.c module

We modified the ds2_create_cust.c module to permit 4GB sizes, following the changes we made to the ds2_create_orders.c module. We added the command-line option R, for creating 4 GB databases, to the existing options, S (small), M (medium), and L (large). In the switch statement that sets the values for the variables max_cust_id and max_prod_id, we added cases that assigned them the proper values for the 4GB database size.

We recompiled the ds2_create_cust.c module on Linux, following the instructions in the header comments. We used the following command line:

```
gcc -o ds2_create_cust ds2_create_cust.c -lm
```

Generating the data for the 4GB database

We used shell scripts to run the four executables that generate user data. Because the distribution did not include shell scripts for the 4GB size, we created shell scripts based on the ds2_create_cust_large.sh and ds2_create_orders_large.sh scripts. The ds2_create_prod and ds2_create_inv executables did not ship with associated shell scripts, so we created shell scripts using the instructions in the readme files. We ran the shell scripts in the following order to generate the data for the 4GB database:

1. `ds2_create_orders_4gb.sh`
2. `ds2_create_inv_4gb.sh`
3. `ds2_create_prod_4gb.sh`
4. `ds2_create_cust_4gb.sh`

We waited until the processes finished before we moved onto the next step.

Creating the database

After creating the PostgreSQL user as we note above, we placed the generated CSV data into the appropriate folders in the DVD Store distribution and ran the pgsqlds2_create_all.sh shell script to create the database.

After configuring and creating the PostgreSQL DS2 database, we performed a cold backup. We stopped the PostgreSQL service, copied all relevant files to our utility partition, and restarted the PostgreSQL service.

Editing the workload script - ds2xdriver.cs module

To use the 4GB database we created earlier, we changed the following constants in the DVD Store client application:

- In the routine Controller(), we changed the string sizes. We added the R option for the 4GB database size. DS2 uses the sizes string to interpret the db_size_str option.
- In the class Controller, we changed the arrays MAX_CUSTOMER and MAX_PRODUCT. To each, we added values specifying the bounds for the customer and product IDs. The Controller() routine uses these arrays.

Recompiling the ds2pgsqldriver.exe executable

We recompiled the ds2xdriver.cs and ds2pgsqlfns.cs module on the Windows server by following the instructions in the header comments. Because the DS2 instructions were for compiling from the command line, we performed the following steps:

1. Install the Npgsql connector on the development workstation.
2. Copy the Npgsql.dll and Mono.Security.dll files to the same directory as the DVD Store source files.
3. Open a command prompt.
4. Use the cd command to change directories to that containing our sources.
5. Execute the following command:

```
C:\Windows\Microsoft.NET\Framework\v2.0.50727\csc.exe
/out:ds2pgsqlserverdriver.exe ds2xdriver.cs ds2pgsqlserverfns.cs
/d:USE_WIN32_TIMER /d:GEN_PERF_CTRS /r:Npgsql.dll
/r:Mono.Security.dll /debug
```

Creating the DS2 client machines

We used eight Dell PowerEdge M600 blade servers for the DS2 clients. Each blade server had two Intel® Xeon® Processor E5345s 2.33GHz, 4 GB of PC2-5300 memory, two 73GB 10K SAS drives in RAID 1 configuration, and Microsoft® Windows Server® 2008 R2 Enterprise Edition.

For the DVD Store scripts, we used a number of client machines to simulate a number of users putting a load on the server. For our client machines, we installed Windows Server 2008 R2 Enterprise Edition, and copied the DVD Store files to the appropriate folders. We created a folder we called `C:\ClientShare` to store workload scripts, and shared this folder for access from our controller machine. We followed this process for each installation.

Installing Windows Server 2008 R2 Enterprise Edition

1. Boot the server, and insert the Windows Server 2008 R2 installation DVD in the DVD-ROM drive.
2. At the Language Selection Screen, click Next.
3. Click Install Now.
4. Select Windows Server 2008 R2 Enterprise (Full Installation), and click Next.
5. Click the I accept the license terms check box, and click Next.
6. Click Custom.
7. Click Drive options (advanced).
8. Ensure you select the proper drive, and click New.
9. Click Apply.
10. Click Next.
11. At the User's password must be changed before logging on warning screen, click OK.
12. Type `Password1` as the new password in both fields, and click the arrow to continue.
13. At the Your password has been changed screen, click OK.

Configuring the IP Address

1. Click Start→Control Panel→Network Connection→Local Area Connection.
2. Double-click Internet Protocol (TCP/IP).
3. Select Use the following IP address.
4. Type `192.168.137.X` for the IP address, `X` being the appropriate IP address.
5. Type `255.255.255.0` for the Subnet mask, and click OK.
6. Click OK to close the Local Area Connection Properties.

Testing procedure

To perform the test, we used a series of batch files and shell scripts, which we executed from a controller machine. We stored batch files respective to each system on that system; and we used the PsExec and Plink utilities to coordinate the test procedures amongst the client machine, server machine, and controller. We used simple file sharing and WinSCP to copy files between machines.

The testing procedure consisted of the following steps:

1. Execute batch files and shell scripts to clean up prior outputs on client machines, the server physical machine, and the controller.
2. Stop the PostgreSQL service.
3. Delete all prior database files.
4. Copy all original database files from the backup utility partition.

5. Reboot the client machines.
6. Reboot the server under test.
7. Wait for a ping response from all client machines.
8. Wait for a ping response from the physical server machine.
9. Mount all necessary partitions.
10. Start the PostgreSQL services (either one or four instances, depending on the test) on the server.
11. Start the workload connections.
12. Start the workload ramp-up period.
13. Start the workload.
14. Stop the workload.
15. Copy all output files to the controller.

APPENDIX A – SERVER CONFIGURATION INFORMATION

Figure 3 provides detailed configuration information for the test server.

System	Dell PowerEdge R710
Power supplies	
Total number	2
Vendor and model number	Dell A570P-00
Wattage of each (W)	570
Cooling fans	
Total number	5
Dimensions (h x w) of each	2-1/2” x 2-1/2”
Volts	12
Amps	1.6
General	
Number of processor packages	2
Number of cores per processor	6
Number of hardware threads per core	2
CPU	
Vendor	Intel
Name	Xeon
Model number	X5670
Stepping	01
Socket type	LGA1366
Core frequency (GHz)	2.93
Bus frequency (GT/s)	6.40
L1 cache (KB)	32 + 32 (per core)
L2 cache (KB)	256 (per core)
L3 cache (MB)	12
Platform	
Vendor and model number	Dell PowerEdge R710
Motherboard model number	PWB9YY69
Motherboard chipset	Intel 5520
BIOS name and version	Dell Incorporated 2.2.2 (9/21/2010) Revision 2.2
BIOS settings	Default
Memory module(s)	
Total RAM in system (GB)	24
Vendor and model number	Crucial CT51272BB1339.36SFD1
Type	PC3-10600R
Speed (MHz)	1,333
Speed running in the system (MHz)	1,333
Timing/Latency (tCL-tRCD-tRP-tRASmin)	9-9-9-24
Size (GB)	4

System	Dell PowerEdge R710
Number of RAM module(s)	6
Chip organization	Double-sided
Rank	Dual
Hard disk	
Vendor and model number	Seagate ST9146802SS
Number of disks in system	2
Size (GB)	146
Buffer size (MB)	16
RPM	10,000
Type	SAS
Disk controller	
Vendor and model	LSI Logic / Symbios Logic SAS1068E
Controller driver (module)	2.6.32-71.el6.x86_64
Controller driver version	0B4D557979D0BC8F39D9984
Controller firmware	0.25.47.00-IR
RAID configuration	RAID 1
Operating system	
Name	Red Hat Enterprise Linux 6
File system	ext4
Kernel	2.6.32-71.el6.x86_64
Language	English
Graphics	
Vendor and model number	Matrox® G200eW
Ethernet	
Vendor and model number	Broadcom® NetXtreme® II BCM57710
Type	PCI Express
Driver (Module)	2.6.32-71.el6.x86_64
Driver Version	1.52.1-7
Optical drive(s)	
Vendor and model number	TEAC DV-28S
Type	DVD ROM
USB ports	
Number	4
Type	2.0

Figure 3: Detailed configuration information for the test server.

APPENDIX B – TEST STORAGE INFORMATION

Figure 4 presents detailed information for the test storage we used in this report.

Enclosure	Dell EqualLogic PS5000XV
General dimension information	
Height (inches)	5.25
Width (inches)	19.0
Depth (inches)	21.5
U size in server rack	3
Power supplies	
Total number	2
Wattage of each	450
Cooling fans	
Total number	4
Dimensions (H x W) of each	4” x 4”
Voltage (V)	12.0
Amps (A)	1.8
Hard disk	
Vendor and model number	Seagate ST3146855SS Maxtor 8k147S0087511
Number of disks in system	16
Size (GB)	146
Buffer size (MB)	16
RPM	15,000
Type	SAS
Storage connectivity (Fibre Channel, SAS, SATA, iSCSI, SCSI)	iSCSI
Firmware revision	V5.0.2
Storage cache (GB)	1
Number of storage controllers	2 (1 active)
Number of storage ports	3 per controller
Maximum network bandwidth of storage connectivity to host	1 Gb x 3
Switch type/model/firmware revision	Dell PowerConnect 6248 V3.2.0.7

Figure 4: Detailed configuration information for the test storage.

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.

Disclaimer of Warranties; Limitation of Liability:
PRINCIPLED TECHNOLOGIES, INC. HAS MADE REASONABLE EFFORTS TO ENSURE THE ACCURACY AND VALIDITY OF ITS TESTING, HOWEVER, PRINCIPLED TECHNOLOGIES, INC. SPECIFICALLY DISCLAIMS ANY WARRANTY, EXPRESSED OR IMPLIED, RELATING TO THE TEST RESULTS AND ANALYSIS, THEIR ACCURACY, COMPLETENESS OR QUALITY, INCLUDING ANY IMPLIED WARRANTY OF FITNESS FOR ANY PARTICULAR PURPOSE. ALL PERSONS OR ENTITIES RELYING ON THE RESULTS OF ANY TESTING DO SO AT THEIR OWN RISK, AND AGREE THAT PRINCIPLED TECHNOLOGIES, INC., ITS EMPLOYEES AND ITS SUBCONTRACTORS SHALL HAVE NO LIABILITY WHATSOEVER FROM ANY CLAIM OF LOSS OR DAMAGE ON ACCOUNT OF ANY ALLEGED ERROR OR DEFECT IN ANY TESTING PROCEDURE OR RESULT.

IN NO EVENT SHALL PRINCIPLED TECHNOLOGIES, INC. BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH ITS TESTING, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL PRINCIPLED TECHNOLOGIES, INC.'S LIABILITY, INCLUDING FOR DIRECT DAMAGES, EXCEED THE AMOUNTS PAID IN CONNECTION WITH PRINCIPLED TECHNOLOGIES, INC.'S TESTING. CUSTOMER'S SOLE AND EXCLUSIVE REMEDIES ARE AS SET FORTH HEREIN.