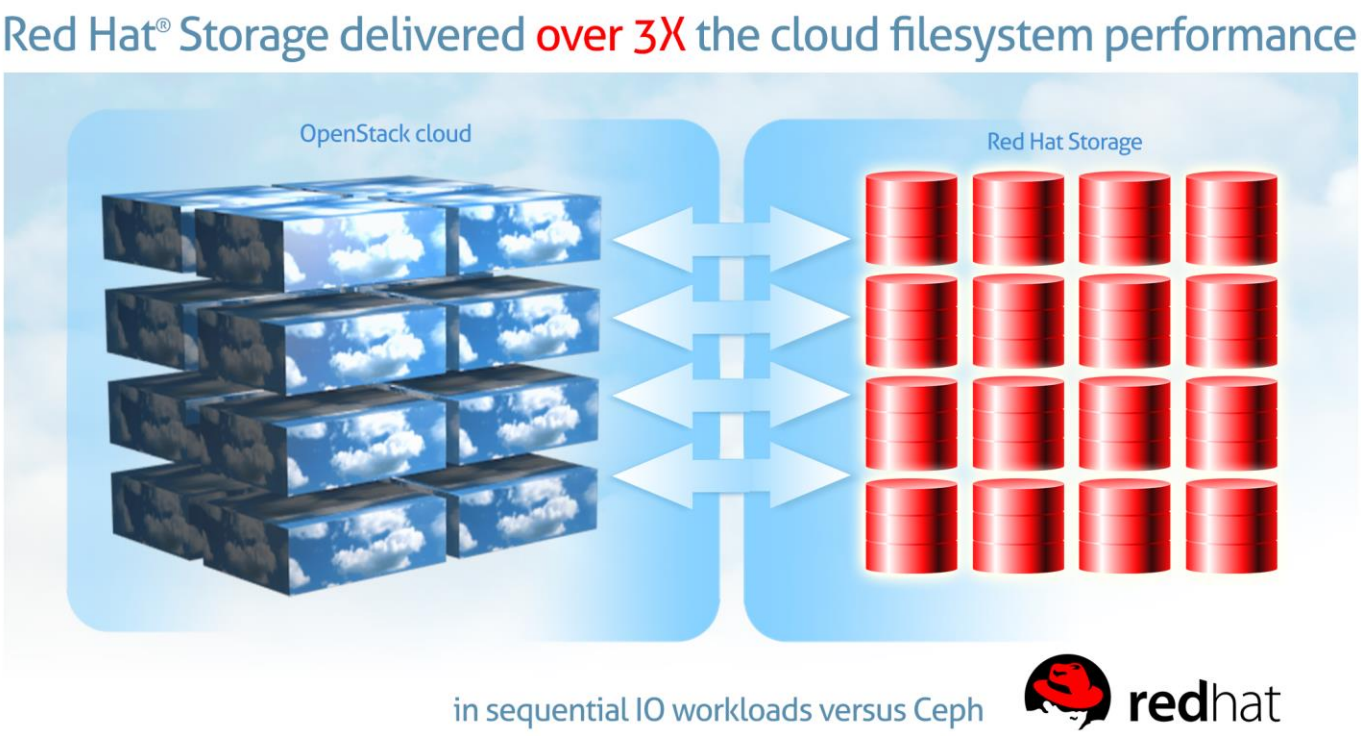
DISTRIBUTED STORAGE PERFORMANCE FOR OPENSTACK CLOUDS: RED HAT STORAGE SERVER VS. CEPH STORAGE



Distributed scale-out storage can provide a cost-effective, fluid storage environment for private and public OpenStack cloud environments. Software-defined storage systems that manage such environments allow you to pool storage capacity and resources across your cloud environment while letting you scale resources independently. However, not all scale-out software-defined storage solutions deliver the same performance or scale optimally, so choosing the right solution is critical to make the most of your cloud infrastructure.

In the Principled Technologies labs, we compared two distributed storage solutions, Red Hat Storage Server and Ceph Storage, in both performance and in scalability across multiple physical storage nodes, as well as multiple physical compute clients and virtual machines. In our tests with a single compute node, Red Hat Storage delivered as much as 3.5 times more read throughput than Ceph. With two compute nodes, Red Hat Storage outperformed Ceph Storage by as much as 3.8 times, and when we scaled to four compute nodes, Red Hat Storage delivered more than double the throughput of Ceph across all VM counts. With Red Hat Storage, performance also scaled better in our tests as we increased server nodes and virtual machines. This means that with similar IO profiles and node/VM configurations, Red Hat Storage could provide better throughput for your OpenStack cloud infrastructure.

OCTOBER 2013
A PRINCIPLED TECHNOLOGIES TEST REPORT

DISTRIBUTED STORAGE TESTING

OpenStack and distributed storage

An OpenStack cloud manages compute, storage, and networking resources. For the backing storage in an OpenStack cloud environment, organizations face the challenge of selecting cost-effective, flexible, and high-performing storage. By using distributed scale-out storage with open-source software, companies can achieve these goals. Software such as Red Hat Storage removes the high-cost and specialized skillset of traditional storage arrays and instead relies on servers as storage nodes, which means that datacenter staff can simply add servers to scale capacity and performance. Conversely, if storage needs decrease, administrators can repurpose those server storage nodes if necessary.

Software overview

In our tests of two leading open-source distributed storage solutions, we compared sequential read and write performance of Red Hat Storage Server and Ceph storage, along with the scalability of both solutions using one to four nodes and one to 16 VMs. We used RDO OpenStack for our OpenStack distribution, and we used the IOzone benchmark running within virtual machine instances on our OpenStack compute nodes to measure filesystem read and write throughput for multiple configurations using each storage solution. Using IOzone, we tested 64KB record sizes using a 100% sequential workload.

For testing, we used the same hardware for both solutions – four compute nodes running RDO OpenStack[1] and four storage nodes running either Red Hat Storage or Ceph Storage. For detailed system configuration information, see Appendix A. The tests and this report were commissioned by Red Hat.

Test goals

Because we wanted to test scalability as well as raw performance, we tested the solutions in a number of different compute node counts with differing VM counts as well. First, we tested with one compute node at three different VM counts: one, two, and four. Then, we added a second compute node and ran our tests using one VM per node, two VMs per node, three VMs per node, and four VMs per node – resulting in a total of two, four, six, and eight VMs across the servers. Finally, we scaled to four compute nodes, again running one VM per node, two VMs per node, three VMs per node, and four VMs per node – resulting in a total of 4, 8, 12, and 16 VMs. In each compute node and VM configuration, we used four server storage nodes. For details about our test setup and how we tested, see Appendix B and Appendix C.

[1] http://openstack.redhat.com

Red Hat Storage and Ceph filesystem differences

Red Hat Storage was configured in a distributed-replicated configuration, with a replication level of 2. This means data is spread across storage nodes with each file in its entirety residing at two separate storage nodes. Read IO for Red Hat Storage in this configuration will pull IO from only one of the two storage nodes where the data resides. Write IO for Red Hat Storage is sent from the client nodes directly to the storage nodes where each copy of the data object resides. See Figure 1 for a visual representation of read and write IO in Red Hat Storage.

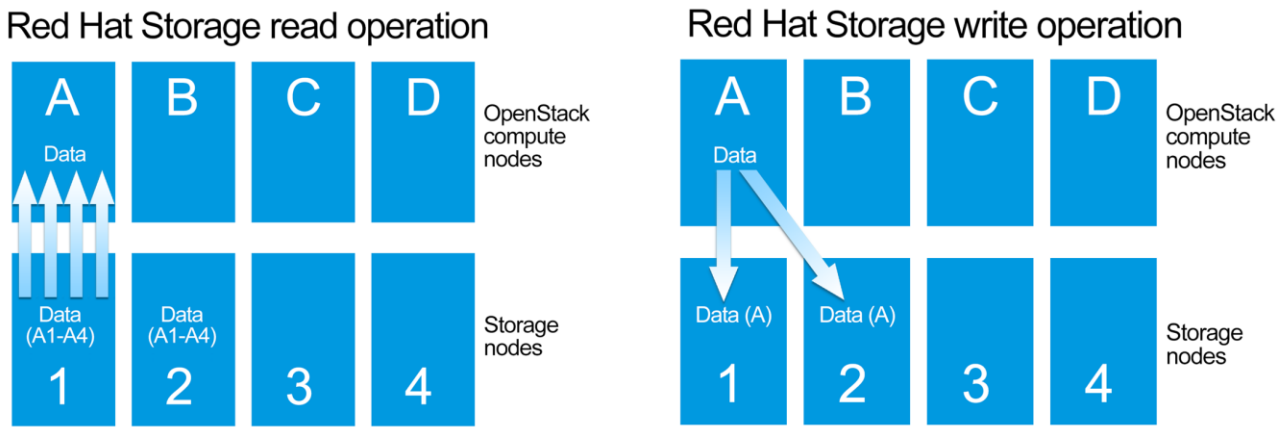


Figure 1: Read and write IO of Red Hat Storage.

Ceph is a distributed object storage platform with a filesystem layer on top.. Data written into the cluster is divided into objects, and each of those objects contain blocks. Those blocks are distributed across all storage nodes in the storage pool. Object storage devices (OSDs) make up these pools. We configured each storage node with multiple OSDs – one OSD for each individual disk on the storage node. A replication level of 2 was used, which ensured that all data blocks had at least two copies on two different storage nodes. Read IO for Ceph in this configuration is distributed across all storage nodes, because the blocks that make up the data objects are distributed across the OSDs. Write IO for Ceph is first sent to a single OSD at which point it is replicated to another OSD on a different storage node. See Figure 2 for a visual representation of read and write IO in Ceph Storage.

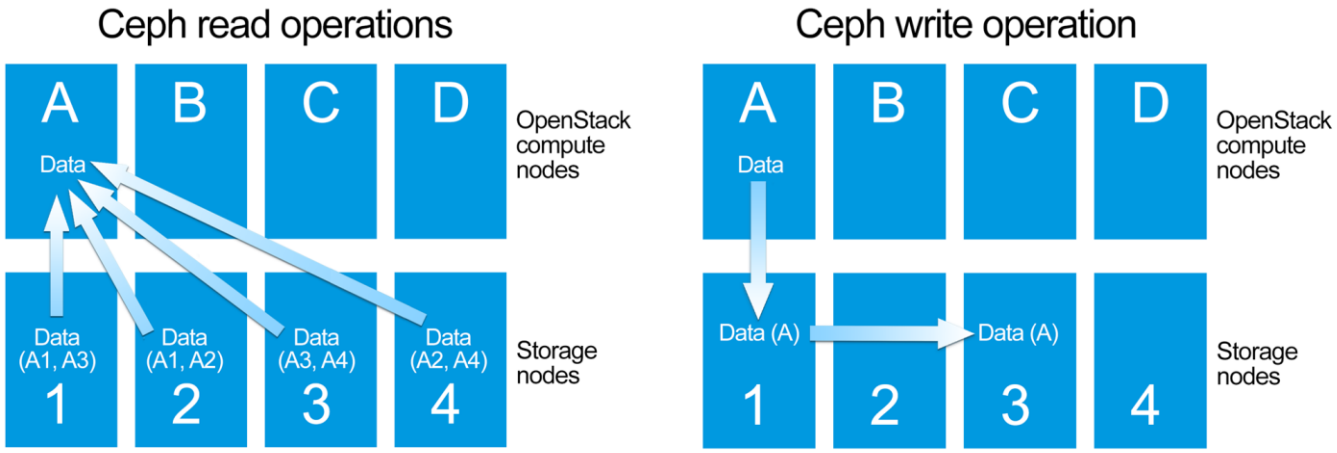


Figure 2: Read and Write IO of Ceph Storage.

TEST RESULTS

The clear winner in our performance and scalability tests was Red Hat Storage. In both sequential read and sequential write throughput, it dramatically outperformed Ceph on nearly every compute node/VM configuration we tested.

Single-node test results

With a single compute node, Red Hat Storage scaled better on sequential read throughput as we doubled the number of VMs (see Figure 3). For specific throughput data for this scenario, see Figure 6.

In the single compute node configuration, Red Hat Storage and Ceph had comparable write performance, with Ceph delivering slightly more write throughput at the higher VM counts. This is because Red Hat Storage sent all write requests twice (once to each storage node the data resides on) which limited write speeds to 600MB/s for each 10Gb network connection used. As we found in our tests, this did not occur when we added more compute nodes, as this increased the number of 10Gb network connections.

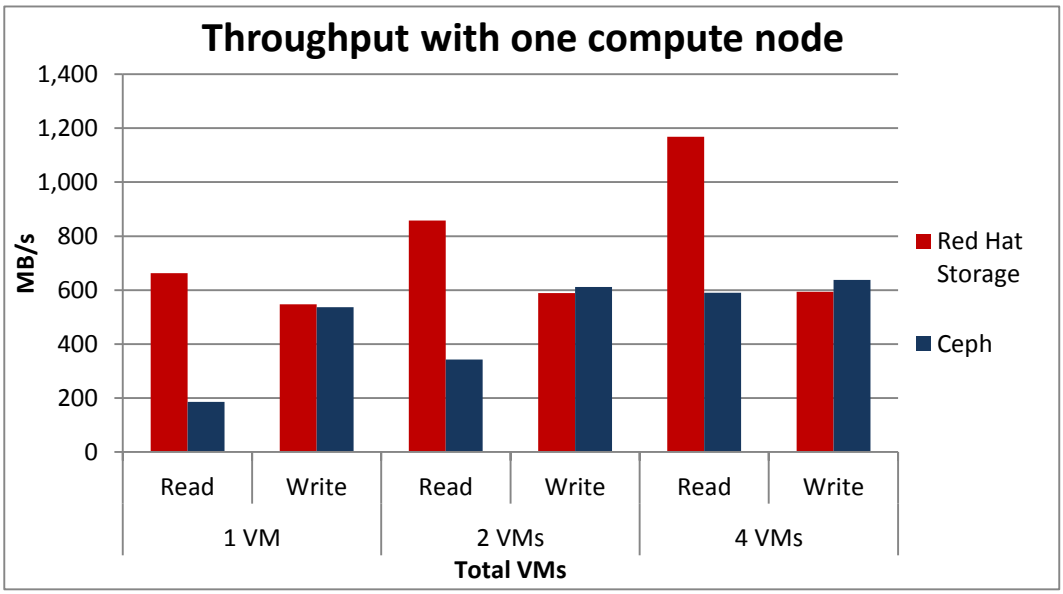


Figure 3: Throughput comparison of the storage solutions at varying VM counts on a single compute node.

Two-node test results

We added a second compute node to our next configuration, and we split the OpenStack VMs evenly across both compute nodes. When we added a second compute node, the configuration using Red Hat Storage delivered more sequential read and write throughput than the configuration using Ceph Storage at every VM count (see Figure 4). While Ceph sequential write performance increased slightly as we added VMs, Red Hat Storage write throughput scaled more significantly, and delivered between 111.9 percent and 286.4 percent more read throughput and 62.8 percent to 85.3 percent more write throughput than Ceph. For specific throughput data for this scenario, see Figure 7.

Again, this means that Red Hat Storage provided greater throughput to the OpenStack cloud than Ceph Storage, which means the end user or application could access more data in a shorter amount of time.

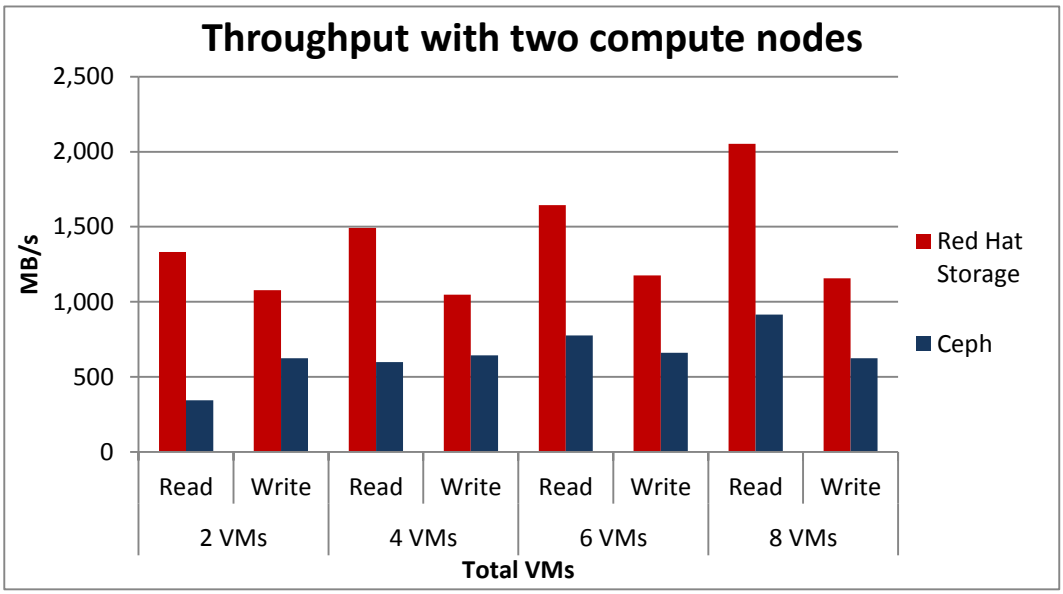


Figure 4: Throughput comparison of the storage solutions at varying VM counts across two compute nodes.

Four-node test results

Doubling the two compute nodes to four also showed dramatically higher sequential read and write throughput for Red Hat Storage over Ceph Storage (see Figure 5). Red Hat Storage achieved read throughput between 101.0 percent and 235.2 percent higher than Ceph. Write performance for the cloud using Red Hat Storage was between 105.9 percent and 187.5 percent higher than the cloud using Ceph Storage. Once more, we found that Red Hat Storage delivered consistently higher performance and also scaled better than Ceph did. For specific throughput data for this scenario, see Figure 8.

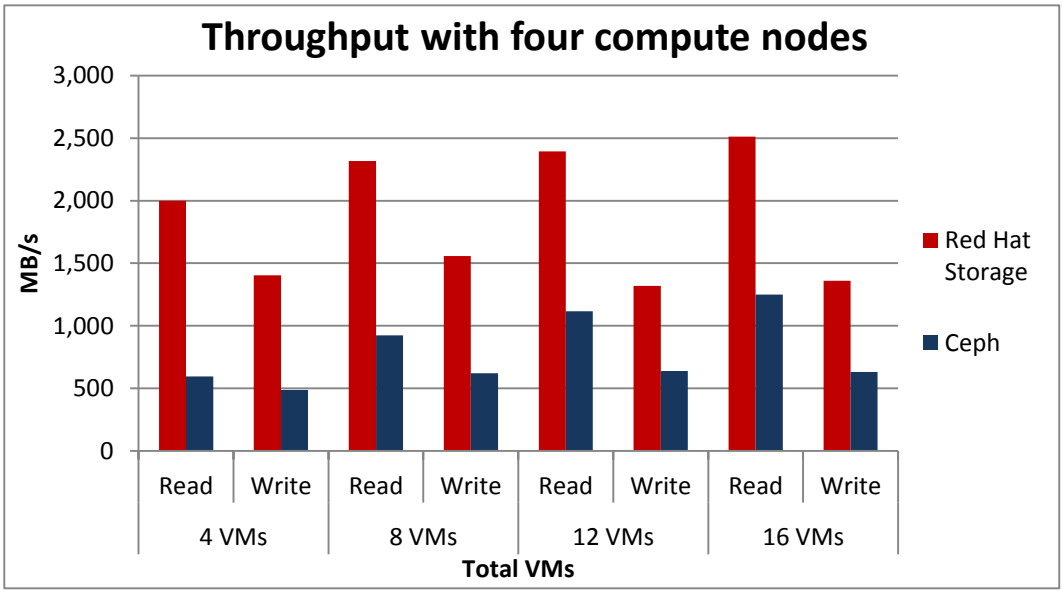


Figure 5: Throughput comparison of the storage solutions at varying VM counts across four compute nodes.

Detailed test results

Here, we provide the detailed throughput data we found in our IOzone tests, along with the percentage greater sequential read and write throughput Red Hat Storage delivered for each node and VM count as we scaled.

Figure 6 compares the throughput of both solutions at varying VM counts on a single server node.

Single compute node						
	1 VM		2 VMs		4 VMs	
	Read	Write	Read	Write	Read	Write
Red Hat Storage	663	547	858	589	1,168	594
Ceph	186	537	343	612	590	638
Red Hat win	256.5%	1.9%	150.1%	-3.8%	98.0%	-6.9%

Figure 6: Throughput, in MB/s, for the storage solutions at varying VM counts on a single compute node.

Figure 7 compares the throughput of both solutions at varying VM counts across two server nodes.

Two compute nodes								
	2 VMs		4 VMs		6 VMs		8 VMs	
	Read	Write	Read	Write	Read	Write	Read	Write
Red Hat Storage	1,333	1,078	1,492	1,047	1,644	1,177	2,053	1,158
Ceph	345	625	599	643	776	661	915	625
Red Hat win	286.4%	72.5%	149.1%	62.8%	111.9%	78.1%	124.4%	85.3%

Figure 7: Throughput, in MB/s, for the storage solutions at varying VM counts across two compute nodes.

Figure 8 compares the throughput of both solutions at varying VM counts across four server nodes.

Four compute nodes								
	4 VMs		8 VMs		12 VMs		16 VMs	
	Read	Write	Read	Write	Read	Write	Read	Write
Red Hat Storage	1,998	1,403	2,316	1,557	2,394	1,318	2,513	1,359
Ceph	596	488	925	622	1,117	640	1,250	632
Red Hat win	235.2%	187.5%	150.4%	150.3%	114.3%	105.9%	101.0%	115.0%

Figure 8: Throughput, in MB/s, for the storage solutions at varying VM counts across four compute nodes.

WHAT WE TESTED

About Red Hat Storage Server

Red Hat Storage Server is a software-based, or according to Red Hat "software-defined," storage platform to manage big, semi-structured, and unstructured data growth while maintaining performance, capacity, and availability to meet demanding enterprise storage requirements. Running on open-source software, it collects compute and network resources in addition to storage capacity, on both physical infrastructure and cloud environments to independently scale beyond the limitations of each type of environment. Along with the ability to deploy on-premises or in a cloud environment, Red Hat Storage Server has flexible deployment options to meet various business needs. For more information about Red Hat Storage, visit http://www.redhat.com/products/storage-server/.

About Ceph Storage

Ceph Storage is an object-based storage system, separating objects from the underlying storage hardware using Reliable Autonomic Distributed Object Store (RADOS). According to Ceph, the RADOS foundation ensures flexibility in data storage by allowing applications object, block or file system interfaces simultaneously.

For more information about Ceph Storage, visit http://ceph.com/ceph-storage/.

About the IOzone Filesystem benchmark

The IOzone benchmark tests a system's file I/O performance by simulating file-access patterns that may be used in different enterprise applications, and by using operating-system specific heuristics for reading and writing files, such as direct and asynchronous I/O, as well as operating-system specific optimizations at the file system level. The read and write operations IOzone tests include:

- Write data to a new file
- Overwrite an existing file
- Write data to random locations of a file
- Write and immediately rewrite data to a fixed section of the file
- Write data to a new file using buffered I/O system routines
- Overwrite an existing file using buffered I/O system routines
- Read an entire file
- Read an entire, recently read file
- Read the entire file starting from the file's end and proceeding to the beginning
- Read data from sections separated by a fixed amount (stride)
- Read data from random locations of a file
- Read an entire file using buffered I/O system routines
- Read an entire, recently read file using buffered I/O

For more information about IOzone, visit http://www.iozone.org.

IN CONCLUSION

Demanding OpenStack cloud infrastructures require storage that is cost-effective, flexible, and high performing to meet customers' demands. Using distributed scale-out storage with a software-defined platform such as Red Hat Storage can help meet these needs by allowing hardware fluidity and minimizing datacenter costs.

In our performance tests at this scale, we found that open-source Red Hat Storage provided better sequential read and write throughput for OpenStack cloud filesystems than Ceph Storage did, achieving read throughputs of up to 3.8 times greater and write throughputs of up to 2.8 times greater. In our test configurations, performance with Red Hat Storage also scaled better than Ceph. Better performance and scalability, such as what Red Hat Storage displayed in our tests, is critical to an ever-changing cloud landscape.

APPENDIX A – SYSTEM CONFIGURATION INFORMATION

Figure 9 provides detailed configuration information for the systems we used in our tests.

System	Dell™ PowerEdge™ C8220X (storage node)	Dell PowerEdge C8220 (compute node)
Power supplies		
Total number	2	2
Vendor and model number	Dell B07B	Dell B07B
Wattage of each (W)	2800	2800
Cooling fans		
Total number	6 (chassis fans)	6 (chassis fans)
Vendor and model number	Delta Electronics, Inc. PFC1212DE	Delta Electronics, Inc. PFC1212DE
Dimensions (h x w) of each	5” x 5” x 1.5”	5” x 5” x 1.5”
Volts	12	12
Amps	4.80	4.80
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	N/A	N/A
CPU		
Vendor	Intel®	Intel
Name	Xeon®	Xeon
Model number	E5-2650	E5-2650
Stepping	C2	C2
Socket type	LGA2011	LGA2011
Core frequency (GHz)	2.00	2.00
Bus frequency	4,000	4,000
L1 cache	32 KB + 32 KB (per core)	32 KB + 32 KB (per core)
L2 cache	256 KB (per core)	256 KB (per core)
L3 cache	20 MB	20 MB
Memory module(s)		
Total RAM in system (GB)	16	128
Vendor and model number	Samsung® M393B5273DH0-CK0	Samsung M393B1K70DH0-CK0
Type	PC3-12800R	PC3-12800R
Speed (MHz)	1,600	1,600
Size (GB)	4	8
Number of RAM module(s)	4	8
Chip organization	Double-sided	Double-sided
Rank	Dual	Dual

System	Dell™ PowerEdge™ C8220X (storage node)	Dell PowerEdge C8220 (compute node)
Operating system		
Name	Red Hat Enterprise Linux®	Red Hat Enterprise Linux
Build number	6.4	6.5 Beta
File system	xfs	ext4
Kernel	2.6.32-358.18.1.el6.x86_64	2.6.32-415.el6.x86_64
Language	English	English
Graphics		
Vendor and model number	ASPEED AST2300	ASPEED AST2300
Graphics memory (MB)	16	16
RAID controller 1		
Vendor and model number	Intel C600	Intel C600
Cache size	N/A	N/A
RAID controller 2		
Vendor and model number	LSI 9265-8i	N/A
Cache size	1 GB	N/A
Hard drive		
Vendor and model number	Dell 9RZ168-136	Dell 9RZ168-136
Number of disks in system	2	2
Size (GB)	1,000	1,000
Buffer size (MB)	32	32
RPM	7,200	7,200
Type	SATA 6.0 Gb/s	SATA 6.0 Gb/s
Hard drive 2		
Vendor and model number	Dell 9TG066-150	N/A
Number of disks in system	8	N/A
Size (GB)	600	N/A
Buffer size (MB)	64	N/A
RPM	10,000	N/A
Type	SAS 6.0 Gb/s	N/A
Ethernet adapters		
First network adapter		
Vendor and model number	Intel I350-BT2	Intel I350-BT2
Type	Integrated	Integrated
Second network adapter		
Vendor and model number	Mellanox MAX383A	Mellanox MAX383A
Type	10/40 GbE	10/40 GbE
USB ports		
Number	2	2
Type	2.0	2.0

Figure 9: Configuration information for our test systems.

APPENDIX B – TEST SETUP OVERVIEW

Compute nodes and OpenStack controller

We installed four compute server nodes with Red Hat Enterprise Linux 6.5 Beta to be used for the OpenStack cloud. Each compute node contained two hard disks, which we configured in a RAID 1 mirror, where we installed the operating system. We used a separate server node to serve as our OpenStack controller, on which we ran all OpenStack services (Neutron, Cinder, Horizon, Keystone, MySQL) other than nova-compute, which ran on the compute nodes. Figure 10 shows our test configuration.

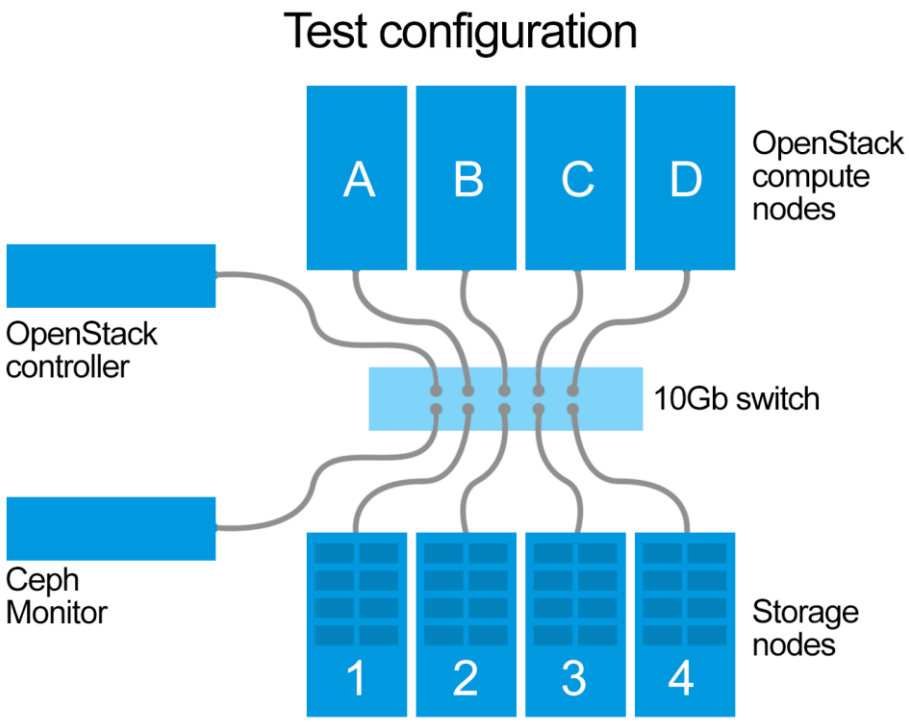


Figure 10: Test hardware configuration.

Storage server configuration

Each of the four storage nodes contained two 1TB 7,200RPM SATA disks, which we configured in a RAID 1 mirror. On this RAID 1 set, we created two logical volumes. On the first, we installed Red Hat Storage 2.1 Beta for Red Hat Storage tests. On the second logical volume, we installed Red Hat Enterprise Linux 6.4 and the necessary Ceph Storage packages for the Ceph Storage tests (ceph version 0.67.4). To switch between storage platforms, we used GRUB to choose the boot volume, and booted the storage nodes into the correct environment, either Red Hat Storage, or Ceph Storage. These configurations remained constant amongst the four storage nodes.

Each of the four storage nodes also contained eight 600GB 10K RPM SAS disks. We configured these disks to be our data disks for IOzone testing, and varied our approach based on each platform's best practices and recommendations. For Red Hat Storage, we configured these eight disks in an eight disk RAID 6 volume and presented the volume to Red Hat Storage. Figure 11 shows the node configuration for Red Hat Storage tests.

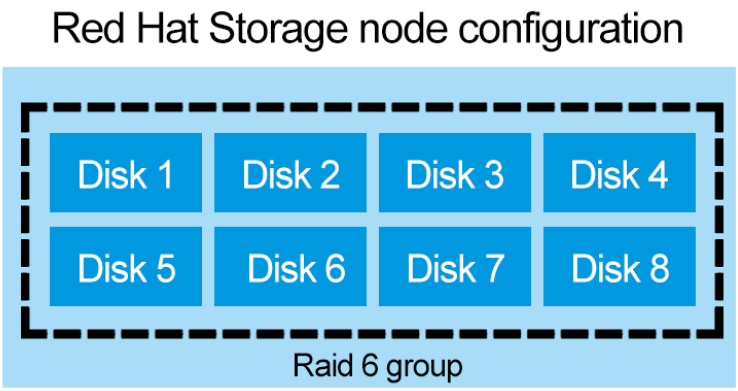


Figure 11: Red Hat Storage node configuration.

For Ceph Storage, we configured eight RAID 0 volumes (one for each physical disk) and presented all of them to Ceph Storage, whereby it could then use an independent OSD on each physical disk, per Ceph Storage best practices. These configurations remained constant amongst the four storage nodes. Figure 12 shows the node configuration for our Ceph tests.

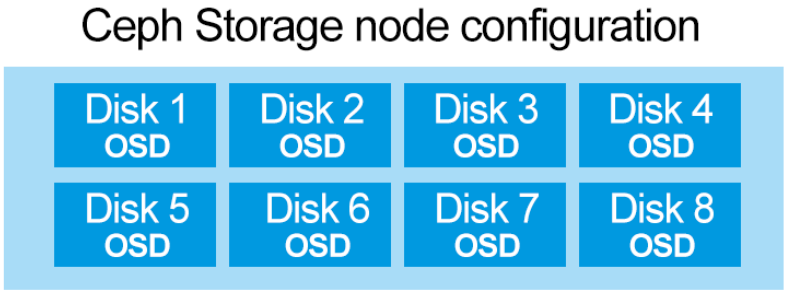


Figure 12: Ceph Storage node configuration.

Figure 13 details the software versions we used in our tests.

Servers	Operating system	Additional software
OpenStack Controller	Red Hat Enterprise Linux 6.5 Beta	RDO 2013.2 b3
OpenStack Compute nodes	Red Hat Enterprise Linux 6.5 Beta	qemu-kvm-rhev-0.12.1.2-2.411 glusterfs-api-3.4.0.34rhs-1 librbd1-0.67.4-0
Storage nodes (Red Hat Storage tests)	Red Hat Storage 2.1 Beta	glusterfs-server-3.4.0.19rhs-2
Storage nodes (Ceph Storage tests)	Red Hat Enterprise Linux 6.4	ceph-0.67.4-0

Figure 13: Software versions we used in our tests.

APPENDIX C – DETAILED CONFIGURATION STEPS

In this section, we review in detail the steps we followed on the various machines to install and configure the various components. Commands are presented with no shading, while file contents or output is presented with gray shading.

Configuring Red Hat Network Beta repositories

1. On each machine that will use Red Hat Enterprise Linux Beta, configure the RHN Beta repositories.

```
subscription-manager repos --enable=rhel-6-server-beta-rpms
subscription-manager repos --enable=rhel-6-server-optional-beta-rpms
```

Configuring networking on all servers

1. Install the necessary rpms by using the following commands:

```
yum install -y openssh-clients wget acpid cpuspeed tuned sysstat sysfsutils
```

2. Bring these devices down using the following commands:

```
ifconfig p2p1 down
ifconfig ib0 down
```

3. Remove the ifcfg files using the following commands:

```
cd /etc/sysconfig/network-scripts/
rm -f ifcfg-p[0-9]p[0-9] ifcfg-ib[0-9]
```

4. Configure the Mellanox components using the following commands:

```
modprobe -r mlx4_en mlx4_ib mlx4_core
sed -i '/mlx4_core/,+1d' /etc/udev/rules.d/70-persistent-net.rules
echo "install mlx4_core /sbin/modprobe --ignore-install mlx4_core msi_x=1
enable_64b_cqe_eqe=1 port_type_array=2 && /sbin/modprobe mlx4_en" >
/etc/modprobe.d/mlx4.conf
modprobe mlx4_core
```

5. Disable SELinux using the following commands:

```
sed -i 's/SELINUX=.*/SELINUX=disabled/' /etc/selinux/config
reboot
```

6. Edit the /etc/hosts file on every host using the following command. Run vi and edit the hosts file.

```
vi /etc/hosts
```

We used the following /etc/hosts file:

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.43.10     rdo-cont.test.lan rdo-cont
192.168.43.12     cephmon.test.lan cephmon

192.168.43.101    compute1.test.lan compute1
192.168.43.102    compute2.test.lan compute2
192.168.43.103    compute3.test.lan compute3
192.168.43.104    compute4.test.lan compute4

192.168.43.201    storage1.test.lan storage1
```

```
192.168.43.202    storage2.test.lan storage2
192.168.43.203    storage3.test.lan storage3
192.168.43.204    storage4.test.lan storage4
```

Configuring additional networking – OpenStack controller

1. Edit the network configuration for the first NIC using the following command. Run vi and edit the ifcfg-em1 file.

```
vi ifcfg-em1
```

We used the following settings:

```
DEVICE=em1
TYPE=Ethernet
ONBOOT=yes
IPADDR=192.168.43.10
PREFIX=24
MTU=9000
```

2. Edit the network configuration for the second NIC using the following command. Run vi and edit the ifcfg-em2 file.

```
vi ifcfg-em2
```

We used the following settings:

```
DEVICE=em2
TYPE=Ethernet
ONBOOT=yes
MTU=9000
```

3. Set up passwordless ssh access for all relevant nodes from the OpenStack controller using the following commands:

```
ssh-keygen
ssh-copy-id cephmon
ssh-copy-id compute1
ssh-copy-id compute2
ssh-copy-id compute3
ssh-copy-id compute4
ssh-copy-id storage1
ssh-copy-id storage2
ssh-copy-id storage3
ssh-copy-id storage4
```

4. Configure DNS using the following command:

```
echo "nameserver 192.168.43.1" > /etc/resolv.conf
service network restart
```

Configuring additional networking – OpenStack compute nodes

1. Edit the network configuration for the first NIC using the following command. Run vi and edit the ifcfg-em1 file.

```
vi ifcfg-em1
```

We used the following settings:

```
DEVICE=em1
TYPE=Ethernet
ONBOOT=no
IPADDR=192.168.43.101
PREFIX=24
MTU=9000
```

2. Edit the network configuration for the second NIC using the following command. Run vi and edit the ifcfg-em2 file.

```
vi ifcfg-em2
```

We used the following settings.

```
DEVICE=em2
TYPE=Ethernet
ONBOOT=yes
MTU=9000
```

3. Edit the network configuration for the third NIC using the following commands. Run vi and edit the ifcfg-eth0 file.

```
cp -p ifcfg-em1 ifcfg-eth0
vi ifcfg-eth0
```

We used the following settings:

```
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
IPADDR=192.168.43.101
PREFIX=24
MTU=9000
```

4. Configure DNS using the following command:

```
echo "nameserver 192.168.43.1" > /etc/resolv.conf
service network restart
```

Installing OpenStack

1. On the OpenStack controller machine, install the RDO rpms using the following commands:

```
yum install -y http://rdo.fedorapeople.org/openstack-havana/rdo-release-havana.rpm
yum install -y http://download.fedoraproject.org/pub/epel/6/i386/epel-release-6-
8.noarch.rpm
```

2. On the OpenStack controller machine, install PackStack using the following commands:

```
yum install -y openstack-packstack
packstack --gen-answer-file=packstack-answer-havana.txt
cp packstack-answer-havana.txt packstack-answer-havana.txt.orig
```

3. Edit the PackStack configuration file. Below we show the revisions we made to our packstack configuration file from the original default file.

```
vi packstack-answer-havana.txt

#### revisions
diff packstack-answer-havana.txt packstack-answer-havana.txt.orig
43c43
< CONFIG_NTP_SERVERS=<NTP SERVER HERE>
---
> CONFIG_NTP_SERVERS=
142c142
< CONFIG_NOVA_COMPUTE_HOSTS=192.168.43.101,192.168.43.102,192.168.43.103,192.168.43.104
---
> CONFIG_NOVA_COMPUTE_HOSTS=192.168.43.10
220c220
< CONFIG_NEUTRON_L3_EXT_BRIDGE=provider
---
> CONFIG_NEUTRON_L3_EXT_BRIDGE=br-ex
249c249
```

```
< CONFIG_NEUTRON_OVS_TENANT_NETWORK_TYPE=vlan
---
> CONFIG_NEUTRON_OVS_TENANT_NETWORK_TYPE=local
253c253
< CONFIG_NEUTRON_OVS_VLAN_RANGES=inter-vlan:1200:1205
---
> CONFIG_NEUTRON_OVS_VLAN_RANGES=
257c257
< CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=inter-vlan:br-inst
---
> CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=
261c261
< CONFIG_NEUTRON_OVS_BRIDGE_IFACES=br-inst:em2
---
> CONFIG_NEUTRON_OVS_BRIDGE_IFACES=
```

4. On the OpenStack controller, run PackStack using the following command:

```
packstack --answer-file=packstack-answer-havana.txt
```

The output should be similar to the following:

```
Welcome to Installer setup utility

Installing:
Clean Up...                                             [ DONE ]
Adding pre install manifest entries...                  [ DONE ]
Installing time synchronization via NTP...              [ DONE ]
Setting up ssh keys...                                  [ DONE ]
Adding MySQL manifest entries...                        [ DONE ]
Adding QPID manifest entries...                         [ DONE ]
Adding Keystone manifest entries...                     [ DONE ]
Adding Glance Keystone manifest entries...              [ DONE ]
Adding Glance manifest entries...                       [ DONE ]
Installing dependencies for Cinder...                   [ DONE ]
Adding Cinder Keystone manifest entries...              [ DONE ]
Adding Cinder manifest entries...                       [ DONE ]
Checking if the Cinder server has a cinder-volumes vg...[ DONE ]
Adding Nova API manifest entries...                     [ DONE ]
Adding Nova Keystone manifest entries...                [ DONE ]
Adding Nova Cert manifest entries...                    [ DONE ]
Adding Nova Conductor manifest entries...               [ DONE ]
Adding Nova Compute manifest entries...                 [ DONE ]
Adding Nova Scheduler manifest entries...               [ DONE ]
Adding Nova VNC Proxy manifest entries...               [ DONE ]
Adding Nova Common manifest entries...                  [ DONE ]
Adding Openstack Network-related Nova manifest entries...[ DONE ]
Adding Neutron API manifest entries...                  [ DONE ]
Adding Neutron Keystone manifest entries...             [ DONE ]
Adding Neutron L3 manifest entries...                   [ DONE ]
Adding Neutron L2 Agent manifest entries...             [ DONE ]
Adding Neutron DHCP Agent manifest entries...           [ DONE ]
Adding Neutron Metadata Agent manifest entries...       [ DONE ]
Adding OpenStack Client manifest entries...             [ DONE ]
Adding Horizon manifest entries...                      [ DONE ]
Adding Ceilometer manifest entries...                   [ DONE ]
Adding Ceilometer Keystone manifest entries...          [ DONE ]
Preparing servers...                                    [ DONE ]
Adding post install manifest entries...                 [ DONE ]
```

```
Installing Dependencies...                                [ DONE ]
Copying Puppet modules and manifests...                   [ DONE ]
Applying Puppet manifests...
Applying 192.168.43.104_prescript.pp
Applying 192.168.43.103_prescript.pp
Applying 192.168.43.10_prescript.pp
Applying 192.168.43.101_prescript.pp
Applying 192.168.43.102_prescript.pp
192.168.43.104_prescript.pp :                  [ DONE ]
192.168.43.10_prescript.pp :                    [ DONE ]
192.168.43.102_prescript.pp :                  [ DONE ]
192.168.43.103_prescript.pp :                  [ DONE ]
192.168.43.101_prescript.pp :                  [ DONE ]
Applying 192.168.43.104_ntpd.pp
Applying 192.168.43.103_ntpd.pp
Applying 192.168.43.10_ntpd.pp
Applying 192.168.43.101_ntpd.pp
Applying 192.168.43.102_ntpd.pp
192.168.43.101_ntpd.pp :                             [ DONE ]
192.168.43.104_ntpd.pp :                             [ DONE ]
192.168.43.10_ntpd.pp :                               [ DONE ]
192.168.43.103_ntpd.pp :                             [ DONE ]
192.168.43.102_ntpd.pp :                             [ DONE ]
Applying 192.168.43.10_mysql.pp
Applying 192.168.43.10_qpid.pp
192.168.43.10_mysql.pp :                             [ DONE ]
192.168.43.10_qpid.pp :                               [ DONE ]
Applying 192.168.43.10_keystone.pp
Applying 192.168.43.10_glance.pp
Applying 192.168.43.10_cinder.pp
192.168.43.10_keystone.pp :                       [ DONE ]
192.168.43.10_glance.pp :                           [ DONE ]
192.168.43.10_cinder.pp :                           [ DONE ]
Applying 192.168.43.10_api_nova.pp
192.168.43.10_api_nova.pp :                       [ DONE ]
Applying 192.168.43.10_nova.pp
Applying 192.168.43.101_nova.pp
Applying 192.168.43.102_nova.pp
Applying 192.168.43.103_nova.pp
Applying 192.168.43.104_nova.pp
192.168.43.10_nova.pp :                               [ DONE ]
192.168.43.103_nova.pp :                             [ DONE ]
192.168.43.104_nova.pp :                             [ DONE ]
192.168.43.102_nova.pp :                             [ DONE ]
192.168.43.101_nova.pp :                             [ DONE ]
Applying 192.168.43.10_neutron.pp
Applying 192.168.43.104_neutron.pp
Applying 192.168.43.103_neutron.pp
Applying 192.168.43.102_neutron.pp
Applying 192.168.43.101_neutron.pp
192.168.43.103_neutron.pp :                       [ DONE ]
192.168.43.101_neutron.pp :                       [ DONE ]
192.168.43.104_neutron.pp :                       [ DONE ]
192.168.43.102_neutron.pp :                       [ DONE ]
192.168.43.10_neutron.pp :                         [ DONE ]
Applying 192.168.43.10_osclient.pp
Applying 192.168.43.10_horizon.pp
```

```
Applying 192.168.43.10_ceilometer.pp
192.168.43.10_osclient.pp :                  [ DONE ]
192.168.43.10_ceilometer.pp :              [ DONE ]
192.168.43.10_horizon.pp :                   [ DONE ]
Applying 192.168.43.104_postscript.pp
Applying 192.168.43.103_postscript.pp
Applying 192.168.43.10_postscript.pp
Applying 192.168.43.101_postscript.pp
Applying 192.168.43.102_postscript.pp
192.168.43.10_postscript.pp :              [ DONE ]
192.168.43.102_postscript.pp :           [ DONE ]
192.168.43.104_postscript.pp :           [ DONE ]
192.168.43.101_postscript.pp :           [ DONE ]
192.168.43.103_postscript.pp :           [ DONE ]
                           [ DONE ]

 **** Installation completed successfully ******


Additional information:
 * Did not create a cinder volume group, one already existed
 * To use the command line tools you need to source the file /root/keystonerc_admin
created on 192.168.43.10
 * To use the console, browse to http://192.168.43.10/dashboard
 * The installation log file is available at: /var/tmp/packstack/20131001-030053-
rzecgC/openstack-setup.log
```

Configuring OpenStack

Configuring Neutron

```
#### NOTE: THIS IS WORKAROUND FOR RDO AND RHEL6.5 AT THE TIME OF WRITING
#### DO ON ALL OPENSTACK SERVERS
yum downgrade iproute
#### NOTE: THIS FIXES A BUG WITH UNSUPPORTED HARDWARE VLAN OFFLOAD
ovs-vsctl set interface em2 other-config:enable-vlan-splinters=true

#### DO ALL THESE ON THE CONTROLLER
source ~/keystonerc_admin
neutron net-create priv_net
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 1cf06d0d-7bda-4665-90ea-92faf11071ca |
| name                      | priv_net                             |
| provider:network_type     | vlan                                 |
| provider:physical_network | inter-vlan                           |
| provider:segmentation_id  | 1200                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 1fbb0466632b42008ffc9e7a2f7f0f6f     |
+---------------------------+--------------------------------------+

neutron subnet-create priv_net 10.0.0.0/24 --name priv_subnet
```

```
Created a new subnet:
c
| Field             | Value                                          |
+-------------------+------------------------------------------------+
| allocation_pools  | {"start": "10.0.0.2", "end": "10.0.0.254"}     |
| cidr              | 10.0.0.0/24                                    |
| dns_nameservers   |                                                |
| enable_dhcp       | True                                           |
| gateway_ip        | 10.0.0.1                                       |
| host_routes       |                                                |
| id                | 81523105-f9e0-40c4-9b6f-6eefd216c712           |
| ip_version        | 4                                              |
| name              | priv_subnet                                    |
| network_id        | 1cf06d0d-7bda-4665-90ea-92faf11071ca           |
| tenant_id         | 1fbb0466632b42008ffc9e7a2f7f0f6f               |
+-------------------+------------------------------------------------+

neutron net-create ext_net --provider:network_type vlan --provider:physical_network
inter-vlan --provider:segmentation_id 1000 --router:external=True
Created a new network:
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | aab3138e-dbc6-452a-ba00-4c615a928988 |
| name                      | ext_net                              |
| provider:network_type     | vlan                                 |
| provider:physical_network | inter-vlan                           |
| provider:segmentation_id  | 1000                                 |
| router:external           | True                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 1fbb0466632b42008ffc9e7a2f7f0f6f     |
+---------------------------+--------------------------------------+

neutron subnet-create ext_net --allocation-pool start=10.35.1.100,end=10.35.1.150 --
gateway 10.35.1.1 10.35.1.0/24 -- --enable_dhcp=False
Created a new subnet:
+-------------------+--------------------------------------------------+
| Field             | Value                                            |
+-------------------+--------------------------------------------------+
| allocation_pools  | {"start": "10.35.1.100", "end": "10.35.1.150"}   |
| cidr              | 10.35.1.0/24                                     |
| dns_nameservers   |                                                  |
| enable_dhcp       | False                                            |
| gateway_ip        | 10.35.1.1                                        |
| host_routes       |                                                  |
| id                | 93788a04-e9c2-4752-86a9-59a201925230             |
| ip_version        | 4                                                |
| name              |                                                  |
| network_id        | aab3138e-dbc6-452a-ba00-4c615a928988             |
| tenant_id         | 1fbb0466632b42008ffc9e7a2f7f0f6f                 |
+-------------------+--------------------------------------------------+


neutron router-create router1
Created a new router:
+-----------------------+--------------------------------------+
```

```
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 0cab02ce-800b-43b0-90c9-cc202dd19b72 |
| name                  | router1                              |
| status                | ACTIVE                               |
| tenant_id             | 1fbb0466632b42008ffc9e7a2f7f0f6f     |
+-----------------------+--------------------------------------+

neutron router-gateway-set router1 ext_net
Set gateway for router router1

neutron router-interface-add router1 priv_subnet
Added interface 51ce34ea-3980-47bf-8ece-f4b2ce17fdcd to router router1.

neutron security-group-rule-create --protocol icmp --direction ingress default
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | ac2e33ca-741e-4b59-aec7-830a6332b79a |
| port_range_max    |                                      |
| port_range_min    |                                      |
| protocol          | icmp                                 |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | ff8c2ff5-a9db-4ec0-bf3e-d0ac249d8fe4 |
| tenant_id         | 1fbb0466632b42008ffc9e7a2f7f0f6f     |
+-------------------+--------------------------------------+

neutron security-group-rule-create --protocol tcp --port-range-min 22 --port-range-max 22
--direction ingress default
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 29e3394f-6e53-4b9d-87cc-e0cbf6c3dbb7 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | ff8c2ff5-a9db-4ec0-bf3e-d0ac249d8fe4 |
| tenant_id         | 1fbb0466632b42008ffc9e7a2f7f0f6f     |
+-------------------+--------------------------------------+


rm -f floatingip_list.txt; for i in `seq 1 16`; do neutron floatingip-create ext_net |
awk "/floating_ip_address/{print \$4\"\tvm$i\"}" | tee -a floatingip_list.txt ; done
10.35.1.101 vm1
10.35.1.102 vm2
10.35.1.103 vm3
```

```
10.35.1.104 vm4
10.35.1.105 vm5
10.35.1.106 vm6
10.35.1.107 vm7
10.35.1.108 vm8
10.35.1.109 vm9
10.35.1.110 vm10
10.35.1.111 vm11
10.35.1.112 vm12
10.35.1.113 vm13
10.35.1.114 vm14
10.35.1.115 vm15
10.35.1.116 vm16

cat floatingip_list.txt >> /etc/hosts

#### PREPARE KEYS (on the controller only)
nova keypair-add GUEST_KEY > GUEST_KEY.pem && chmod 600 GUEST_KEY.pem
```

Configuring Availability Zones

```
#### DO ALL THESE ON THE CONTROLLER
nova availability-zone-list

nova aggregate-create compaggr1 compzone1
nova aggregate-create compaggr2 compzone2
nova aggregate-create compaggr3 compzone3
nova aggregate-create compaggr4 compzone4

nova aggregate-add-host compaggr1 compute1.test.lan
nova aggregate-add-host compaggr2 compute2.test.lan
nova aggregate-add-host compaggr3 compute3.test.lan
nova aggregate-add-host compaggr4 compute4.test.lan

nova availability-zone-list
```

Upload image and create flavor

```
glance image-create --name 'rhel-6.4-iozone' --disk-format qcow2 --container-format bare
--min-disk 4 --min-ram 256 --is-public True --copy-from
http://<imgserver>/share/Linux/RHEL/6.4/rhel64-iozone.img --checksum
0c115efc9dca2fc157b03b990522ab0f

nova flavor-create m1.iozone 100 4096 5 1
```

Updates

KVM Update

Update these packages on the compute nodes:

```
yum install -y kvm_update/qemu-*.rpm

Installed:
  qemu-img-rhev.x86_64 2:0.12.1.2-2.411.el6
  qemu-kvm-rhev.x86_64 2:0.12.1.2-2.411.el6
  qemu-kvm-rhev-tools.x86_64 2:0.12.1.2-2.411.el6

Replaced:
```

```
  qemu-img.x86_64 2:0.12.1.2-2.398.el6
  qemu-kvm.x86_64 2:0.12.1.2-2.398.el6
```

Gluster Client Update

Update these packages on the compute nodes:

```
umount -a -t fuse.glusterfs
yum install -y gluster_update/glusterfs*rpm

Installed:
  glusterfs-libs-3.4.0.34rhs-1.el6.x86_64
  glusterfs-3.4.0.34rhs-1.el6.x86_64
  glusterfs-api-3.4.0.34rhs-1.el6.x86_64
  glusterfs-fuse-3.4.0.34rhs-1.el6.x86_64
```

Configuring Cinder for Red Hat Storage

```
#### ON CONTROLLER
source ~/keystonerc_admin
openstack-config --set /etc/cinder/cinder.conf DEFAULT enabled_backends GLUSTER
openstack-config --set /etc/cinder/cinder.conf GLUSTER volume_backend_name GLUSTER
openstack-config --set /etc/cinder/cinder.conf GLUSTER volume_driver
cinder.volume.drivers.glusterfs.GlusterfsDriver
openstack-config --set /etc/cinder/cinder.conf GLUSTER glusterfs_shares_config
/etc/cinder/glusterfs_shares

echo "storage1:/rhosfs" > /etc/cinder/glusterfs_shares

#### Do these 2 commands on all compute nodes and controller/cinder ####
openstack-config --set /etc/nova/nova.conf DEFAULT qemu_allowed_storage_drivers gluster
openstack-config --set /etc/nova/nova.conf DEFAULT debug False

#### ON CONTROLLER
for i in api scheduler volume; do sudo service openstack-cinder-${i} stop; done
for i in api scheduler volume; do sudo service openstack-cinder-${i} start; done

cinder type-create gluster
cinder type-key gluster set volume_backend_name=GLUSTER
cinder extra-specs-list
```

Installing Ceph Storage and configuring Cinder for Ceph Storage

```
#### Install repos on ceph monitor and all storage nodes then install main ceph packages
yum install -y http://ceph.com/rpm-dumpling/el6/noarch/ceph-release-1-0.el6.noarch.rpm
yum install -y ceph

#### on ceph monitor (hostname: cephmon)
yum install -y ceph-deploy

ceph-deploy new cephmon
ceph-deploy mon create cephmon
ceph-deploy gatherkeys cephmon

ceph-deploy disk list storage{1,2,3,4}
ceph-deploy disk zap storage{1,2,3,4}:sd{a,b,c,d,e,f,g,h}
ceph-deploy osd create storage{1,2,3,4}:sd{a,b,c,d,e,f,g,h}
```

```
ceph-deploy disk list storage{1,2,3,4}

[ceph_deploy.cli][INFO  ] Invoked (1.2.7): /usr/bin/ceph-deploy disk list storage1
storage2 storage3 storage4
[ceph_deploy.sudo_pushy][DEBUG ] will use a remote connection without sudo
[ceph_deploy.osd][INFO  ] Distro info: RedHatEnterpriseServer 6.4 Santiago
[ceph_deploy.osd][DEBUG ] Listing disks on storage1...
[storage1][INFO  ] Running command: ceph-disk list
[storage1][INFO  ] /dev/sda :
[storage1][INFO  ]  /dev/sda1 ceph data, active, cluster ceph, osd.0, journal /dev/sda2
[storage1][INFO  ]  /dev/sda2 ceph journal, for /dev/sda1
[storage1][INFO  ] /dev/sdb :
[storage1][INFO  ]  /dev/sdb1 ceph data, active, cluster ceph, osd.1, journal /dev/sdb2
[storage1][INFO  ]  /dev/sdb2 ceph journal, for /dev/sdb1
[storage1][INFO  ] /dev/sdc :
[storage1][INFO  ]  /dev/sdc1 ceph data, active, cluster ceph, osd.2, journal /dev/sdc2
[storage1][INFO  ]  /dev/sdc2 ceph journal, for /dev/sdc1
[storage1][INFO  ] /dev/sdd :
[storage1][INFO  ]  /dev/sdd1 ceph data, active, cluster ceph, osd.3, journal /dev/sdd2
[storage1][INFO  ]  /dev/sdd2 ceph journal, for /dev/sdd1
[storage1][INFO  ] /dev/sde :
[storage1][INFO  ]  /dev/sde1 ceph data, active, cluster ceph, osd.4, journal /dev/sde2
[storage1][INFO  ]  /dev/sde2 ceph journal, for /dev/sde1
[storage1][INFO  ] /dev/sdf :
[storage1][INFO  ]  /dev/sdf1 ceph data, active, cluster ceph, osd.5, journal /dev/sdf2
[storage1][INFO  ]  /dev/sdf2 ceph journal, for /dev/sdf1
[storage1][INFO  ] /dev/sdg :
[storage1][INFO  ]  /dev/sdg1 ceph data, active, cluster ceph, osd.6, journal /dev/sdg2
[storage1][INFO  ]  /dev/sdg2 ceph journal, for /dev/sdg1
[storage1][INFO  ] /dev/sdh :
[storage1][INFO  ]  /dev/sdh1 ceph data, active, cluster ceph, osd.7, journal /dev/sdh2
[storage1][INFO  ]  /dev/sdh2 ceph journal, for /dev/sdh1
[storage1][INFO  ] /dev/sdi other, isw_raid_member
[storage1][INFO  ] /dev/sdj other, isw_raid_member
[ceph_deploy.sudo_pushy][DEBUG ] will use a remote connection without sudo
[ceph_deploy.osd][INFO  ] Distro info: RedHatEnterpriseServer 6.4 Santiago
[ceph_deploy.osd][DEBUG ] Listing disks on storage2...
[storage2][INFO  ] Running command: ceph-disk list
[storage2][INFO  ] /dev/sda :
[storage2][INFO  ]  /dev/sda1 ceph data, active, cluster ceph, osd.8, journal /dev/sda2
[storage2][INFO  ]  /dev/sda2 ceph journal, for /dev/sda1
[storage2][INFO  ] /dev/sdb :
[storage2][INFO  ]  /dev/sdb1 ceph data, active, cluster ceph, osd.9, journal /dev/sdb2
[storage2][INFO  ]  /dev/sdb2 ceph journal, for /dev/sdb1
[storage2][INFO  ] /dev/sdc :
[storage2][INFO  ]  /dev/sdc1 ceph data, active, cluster ceph, osd.10, journal /dev/sdc2
[storage2][INFO  ]  /dev/sdc2 ceph journal, for /dev/sdc1
[storage2][INFO  ] /dev/sdd :
[storage2][INFO  ]  /dev/sdd1 ceph data, active, cluster ceph, osd.11, journal /dev/sdd2
[storage2][INFO  ]  /dev/sdd2 ceph journal, for /dev/sdd1
[storage2][INFO  ] /dev/sde :
[storage2][INFO  ]  /dev/sde1 ceph data, active, cluster ceph, osd.12, journal /dev/sde2
[storage2][INFO  ]  /dev/sde2 ceph journal, for /dev/sde1
[storage2][INFO  ] /dev/sdf :
[storage2][INFO  ]  /dev/sdf1 ceph data, active, cluster ceph, osd.13, journal /dev/sdf2
[storage2][INFO  ]  /dev/sdf2 ceph journal, for /dev/sdf1
[storage2][INFO  ] /dev/sdg :
```

```
[storage2][INFO  ]  /dev/sdg1 ceph data, active, cluster ceph, osd.14, journal /dev/sdg2
[storage2][INFO  ]  /dev/sdg2 ceph journal, for /dev/sdg1
[storage2][INFO  ] /dev/sdh :
[storage2][INFO  ]  /dev/sdh1 ceph data, active, cluster ceph, osd.15, journal /dev/sdh2
[storage2][INFO  ]  /dev/sdh2 ceph journal, for /dev/sdh1
[storage2][INFO  ] /dev/sdi other, isw_raid_member
[storage2][INFO  ] /dev/sdj other, isw_raid_member
[ceph_deploy.sudo_pushy][DEBUG ] will use a remote connection without sudo
[ceph_deploy.osd][INFO  ] Distro info: RedHatEnterpriseServer 6.4 Santiago
[ceph_deploy.osd][DEBUG ] Listing disks on storage3...
[storage3][INFO  ] Running command: ceph-disk list
[storage3][INFO  ] /dev/sda :
[storage3][INFO  ]  /dev/sda1 ceph data, active, cluster ceph, osd.16, journal /dev/sda2
[storage3][INFO  ]  /dev/sda2 ceph journal, for /dev/sda1
[storage3][INFO  ] /dev/sdb :
[storage3][INFO  ]  /dev/sdb1 ceph data, active, cluster ceph, osd.17, journal /dev/sdb2
[storage3][INFO  ]  /dev/sdb2 ceph journal, for /dev/sdb1
[storage3][INFO  ] /dev/sdc :
[storage3][INFO  ]  /dev/sdc1 ceph data, active, cluster ceph, osd.18, journal /dev/sdc2
[storage3][INFO  ]  /dev/sdc2 ceph journal, for /dev/sdc1
[storage3][INFO  ] /dev/sdd :
[storage3][INFO  ]  /dev/sdd1 ceph data, active, cluster ceph, osd.19, journal /dev/sdd2
[storage3][INFO  ]  /dev/sdd2 ceph journal, for /dev/sdd1
[storage3][INFO  ] /dev/sde :
[storage3][INFO  ]  /dev/sde1 ceph data, active, cluster ceph, osd.20, journal /dev/sde2
[storage3][INFO  ]  /dev/sde2 ceph journal, for /dev/sde1
[storage3][INFO  ] /dev/sdf :
[storage3][INFO  ]  /dev/sdf1 ceph data, active, cluster ceph, osd.21, journal /dev/sdf2
[storage3][INFO  ]  /dev/sdf2 ceph journal, for /dev/sdf1
[storage3][INFO  ] /dev/sdg :
[storage3][INFO  ]  /dev/sdg1 ceph data, active, cluster ceph, osd.22, journal /dev/sdg2
[storage3][INFO  ]  /dev/sdg2 ceph journal, for /dev/sdg1
[storage3][INFO  ] /dev/sdh :
[storage3][INFO  ]  /dev/sdh1 ceph data, active, cluster ceph, osd.23, journal /dev/sdh2
[storage3][INFO  ]  /dev/sdh2 ceph journal, for /dev/sdh1
[storage3][INFO  ] /dev/sdi other, isw_raid_member
[storage3][INFO  ] /dev/sdj other, isw_raid_member
[ceph_deploy.sudo_pushy][DEBUG ] will use a remote connection without sudo
[ceph_deploy.osd][INFO  ] Distro info: RedHatEnterpriseServer 6.4 Santiago
[ceph_deploy.osd][DEBUG ] Listing disks on storage4...
[storage4][INFO  ] Running command: ceph-disk list
[storage4][INFO  ] /dev/sda :
[storage4][INFO  ]  /dev/sda1 ceph data, active, cluster ceph, osd.24, journal /dev/sda2
[storage4][INFO  ]  /dev/sda2 ceph journal, for /dev/sda1
[storage4][INFO  ] /dev/sdb :
[storage4][INFO  ]  /dev/sdb1 ceph data, active, cluster ceph, osd.25, journal /dev/sdb2
[storage4][INFO  ]  /dev/sdb2 ceph journal, for /dev/sdb1
[storage4][INFO  ] /dev/sdc :
[storage4][INFO  ]  /dev/sdc1 ceph data, active, cluster ceph, osd.26, journal /dev/sdc2
[storage4][INFO  ]  /dev/sdc2 ceph journal, for /dev/sdc1
[storage4][INFO  ] /dev/sdd :
[storage4][INFO  ]  /dev/sdd1 ceph data, active, cluster ceph, osd.27, journal /dev/sdd2
[storage4][INFO  ]  /dev/sdd2 ceph journal, for /dev/sdd1
[storage4][INFO  ] /dev/sde :
[storage4][INFO  ]  /dev/sde1 ceph data, active, cluster ceph, osd.28, journal /dev/sde2
[storage4][INFO  ]  /dev/sde2 ceph journal, for /dev/sde1
[storage4][INFO  ] /dev/sdf :
```

```
[storage4][INFO  ]  /dev/sdf1 ceph data, active, cluster ceph, osd.29, journal /dev/sdf2
[storage4][INFO  ]  /dev/sdf2 ceph journal, for /dev/sdf1
[storage4][INFO  ] /dev/sdg :
[storage4][INFO  ]  /dev/sdg1 ceph data, active, cluster ceph, osd.30, journal /dev/sdg2
[storage4][INFO  ]  /dev/sdg2 ceph journal, for /dev/sdg1
[storage4][INFO  ] /dev/sdh :
[storage4][INFO  ]  /dev/sdh1 ceph data, active, cluster ceph, osd.31, journal /dev/sdh2
[storage4][INFO  ]  /dev/sdh2 ceph journal, for /dev/sdh1
[storage4][INFO  ] /dev/sdi other, isw_raid_member
[storage4][INFO  ] /dev/sdj other, isw_raid_member
```

Configuring Ceph pools for OpenStack

```
ceph osd pool create volumes 1600
ceph osd pool create images 128
```

Configuring Cinder for Ceph Storage

```
#### all openstack systems - controller, compute
yum install -y http://ceph.com/rpm-dumpling/el6/noarch/ceph-release-1-0.el6.noarch.rpm

#### on cinder server
yum install -y ceph
mkdir /usr/lib64/qemu
ln -s /usr/lib64/librbd.so.1 /usr/lib64/qemu/librbd.so.1

#### on compute nodes
yum install -y librbd1
mkdir /usr/lib64/qemu
ln -s /usr/lib64/librbd.so.1 /usr/lib64/qemu/librbd.so.1

#### on ceph monitor
ssh rdo-cont tee /etc/ceph/ceph.conf </etc/ceph/ceph.conf
for i in `seq 1 4`; do ssh compute$i tee /etc/ceph/ceph.conf </etc/ceph/ceph.conf ; done
ceph auth get-or-create client.volumes mon 'allow r' osd 'allow class-read object_prefix
rbd_children, allow rwx pool=volumes, allow rx pool=images'
ceph auth get-or-create client.images mon 'allow r' osd 'allow class-read object_prefix
rbd_children, allow rwx pool=images'

ceph auth get-or-create client.images | ssh rdo-cont tee
/etc/ceph/ceph.client.images.keyring
ssh rdo-cont chown glance:glance /etc/ceph/ceph.client.images.keyring
ceph auth get-or-create client.volumes | ssh rdo-cont tee
/etc/ceph/ceph.client.volumes.keyring
ssh rdo-cont chown cinder:cinder /etc/ceph/ceph.client.volumes.keyring

ceph auth get-key client.volumes | tee client.volumes.key

uuidgen > secret.uuid
cat > secret.xml <<EOF
<secret ephemeral='no' private='no'>
  <uuid>`cat secret.uuid`</uuid>
  <usage type='ceph'>
    <name>client.volumes secret</name>
  </usage>
</secret>
EOF
```

```
for i in `seq 1 4`; do cat secret.xml | ssh compute$i "tee secret.xml ; virsh secret-
define --file secret.xml ; virsh secret-set-value --secret $(cat secret.uuid) --base64
$(cat client.volumes.key) && rm client.volumes.key secret.xml" ; done


#### on cinder server
source ~/keystonerc_admin
for i in api scheduler volume; do sudo service openstack-cinder-${i} stop; done

openstack-config --set /etc/cinder/cinder.conf DEFAULT enabled_backends CEPH
openstack-config --set /etc/cinder/cinder.conf CEPH volume_backend_name CEPH
openstack-config --set /etc/cinder/cinder.conf CEPH volume_driver
cinder.volume.drivers.rbd.RBDDriver
openstack-config --set /etc/cinder/cinder.conf CEPH rbd_pool volumes
openstack-config --set /etc/cinder/cinder.conf CEPH glance_api_version 2
openstack-config --set /etc/cinder/cinder.conf CEPH rbd_user volumes
openstack-config --set /etc/cinder/cinder.conf CEPH rbd_secret_uuid `cat secret.uuid`

openstack-config --set /etc/nova/nova.conf DEFAULT disk_cachemodes
"'network=writethrough'"
openstack-config --del /etc/nova/nova.conf DEFAULT disk_cachemodes
service openstack-nova-compute restart


openstack-config --set /etc/ceph/ceph.conf client rbd_cache_max_dirty 0

for i in api scheduler volume; do sudo service openstack-cinder-${i} start; done

cinder type-create ceph
cinder type-key ceph set volume_backend_name=CEPH
cinder extra-specs-list
```

Ceph crush map

```
# begin crush map

# devices
device 0 osd.0
device 1 osd.1
device 2 osd.2
device 3 osd.3
device 4 osd.4
device 5 osd.5
device 6 osd.6
device 7 osd.7
device 8 osd.8
device 9 osd.9
device 10 osd.10
device 11 osd.11
device 12 osd.12
device 13 osd.13
device 14 osd.14
device 15 osd.15
device 16 osd.16
device 17 osd.17
device 18 osd.18
device 19 osd.19
```

```
device 20 osd.20
device 21 osd.21
device 22 osd.22
device 23 osd.23
device 24 osd.24
device 25 osd.25
device 26 osd.26
device 27 osd.27
device 28 osd.28
device 29 osd.29
device 30 osd.30
device 31 osd.31

# types
type 0 osd
type 1 host
type 2 rack
type 3 row
type 4 room
type 5 datacenter
type 6 root

# buckets
host storage1 {
      id -2       # do not change unnecessarily
      # weight 4.320
      alg straw
      hash 0      # rjenkins1
      item osd.0 weight 0.540
      item osd.1 weight 0.540
      item osd.2 weight 0.540
      item osd.3 weight 0.540
      item osd.4 weight 0.540
      item osd.5 weight 0.540
      item osd.6 weight 0.540
      item osd.7 weight 0.540
}
host storage2 {
      id -3       # do not change unnecessarily
      # weight 4.320
      alg straw
      hash 0      # rjenkins1
      item osd.8 weight 0.540
      item osd.9 weight 0.540
      item osd.10 weight 0.540
      item osd.11 weight 0.540
      item osd.12 weight 0.540
      item osd.13 weight 0.540
      item osd.14 weight 0.540
      item osd.15 weight 0.540
}
host storage3 {
      id -4       # do not change unnecessarily
      # weight 4.320
      alg straw
      hash 0      # rjenkins1
      item osd.16 weight 0.540
```

```
      item osd.17 weight 0.540
      item osd.18 weight 0.540
      item osd.19 weight 0.540
      item osd.20 weight 0.540
      item osd.21 weight 0.540
      item osd.22 weight 0.540
      item osd.23 weight 0.540
}
host storage4 {
      id -5       # do not change unnecessarily
      # weight 4.320
      alg straw
      hash 0      # rjenkins1
      item osd.24 weight 0.540
      item osd.25 weight 0.540
      item osd.26 weight 0.540
      item osd.27 weight 0.540
      item osd.28 weight 0.540
      item osd.29 weight 0.540
      item osd.30 weight 0.540
      item osd.31 weight 0.540
}
root default {
      id -1       # do not change unnecessarily
      # weight 17.280
      alg straw
      hash 0      # rjenkins1
      item storage1 weight 4.320
      item storage2 weight 4.320
      item storage3 weight 4.320
      item storage4 weight 4.320
}

# rules
rule data {
      ruleset 0
      type replicated
      min_size 1
      max_size 10
      step take default
      step chooseleaf firstn 0 type host
      step emit
}
rule metadata {
      ruleset 1
      type replicated
      min_size 1
      max_size 10
      step take default
      step chooseleaf firstn 0 type host
      step emit
}
rule rbd {
      ruleset 2
      type replicated
      min_size 1
      max_size 10
```

```
        step take default
        step chooseleaf firstn 0 type host
        step emit
}

# end crush map
```

Starting VMs and creating cinder disks

Below, we include a sample script to start VMs and create cinder volumes.

```
#!/bin/bash
COUNT=0
COMPUTE_NODES=4
if [ "${2}" = "" ]; then
  COUNT=${1:-0}
else
  COMPUTE_NODES=${1:-1}
  COUNT=${2}
fi

DELAY=1
#VOLTYPE=gluster
VOLTYPE=ceph
VOLSIZE=40

PRIV_NET_ID=`neutron net-list -F id -F name -f csv --quote none | awk -F',' 
'/priv_net/{print $1}'`

for i in `seq 1 $COUNT`;
do
  VM=vm$i
  nova boot --image rhel-6.4-iozone --flavor m1.iozone --key_name GUEST_KEY --
availability-zone compzone`expr \( ${i} - 1 \) % ${COMPUTE_NODES} + 1`  --nic net-
id=${PRIV_NET_ID} ${VM}
  sleep $DELAY
  DEVICE_ID=`nova list --name $VM | awk "/$VM/{ print \\$2 }"`
  #sleep $DELAY
  PORT_ID=`neutron port-list -- --device_id ${DEVICE_ID} | awk '/ip_address/{print $2}'`
  FLOATIP=`resolveip -s $VM`
  FLOATING_ID=`neutron floatingip-list | awk "/$FLOATIP/{ print \\$2 }"`
  until [ `nova list --name $VM | awk "/$VM/{ print \\$6 }"` = ACTIVE ]; do
    sleep $DELAY
  done
  neutron floatingip-associate $FLOATING_ID $PORT_ID
  sleep $DELAY
  echo -n "Pinging ${VM}..."
  until ping -qc 1 ${VM} 1> /dev/null 2> /dev/null ;
  do
    sleep 1
    echo -n "."
  done
  echo "success!"
done

sleep 30
```

```
for i in `seq 1 $COUNT`;
do
  VM=vm$i
  sleep $DELAY
  until [ `jobs | wc -l` -lt 2 ]; do
    sleep 1
  done
  if [ "`cinder list --display-name ${VM}_${VOLTYPE}vol | awk '/available/{print $2}'`" =
"" ]; then
    cinder create --volume_type $VOLTYPE --display_name ${VM}_${VOLTYPE}vol $VOLSIZE
    sleep $DELAY
    while [ "`cinder list --display-name ${VM}_${VOLTYPE}vol | awk '/available/{print
$2}'`" = "" ]; do
      sleep 1
    done
    nova volume-attach ${VM} `cinder list --display-name ${VM}_${VOLTYPE}vol | awk
'/available/{print $2}'` /dev/vdb
    sleep $DELAY
    BLOCKS=`expr $VOLSIZE \* 1024`
    sleep $DELAY
    ssh -i GUEST_KEY.pem -o StrictHostKeyChecking=no ${VM} "dd if=/dev/zero of=/dev/vdb
bs=1M count=$BLOCKS ; sync"
    ./run_all_storage.sh "sync"
    sleep $DELAY
  else
    nova volume-attach ${VM} `cinder list --display-name ${VM}_${VOLTYPE}vol | awk
'/available/{print $2}'` /dev/vdb
  fi
done
wait
```

Installing and configuring the base VM

Our VM base image used a 4GB qcow2 virtual disk, 4,096 MB of RAM, and two vCPUs. We installed Red Hat Enterprise Linux 6.4, chose to connect the virtual NIC automatically, and performed a custom disk layout. The disk layout configuration included one standard partition where we used the whole disk with the ext4 filesystem, mountpoint=/, and no swap. We chose the minimal package selection on installation. Below we show the additional steps we followed for the base VM, as well as installed and updated packages.

```
#### Disable selinux.
sed -i 's/SELINUX=.*/SELINUX=disabled/' /etc/selinux/config

#### Disable firewall.
iptables -F
/etc/init.d/iptables save
chkconfig iptables off

#### Subscribe system to RHN:
subscription-manager register
subscription-manager refresh

#### Install updates:
yum update -y
```

```
Installed:
```

```
  kernel.x86_64 0:2.6.32-358.18.1.el6

Updated:
  bash.x86_64 0:4.1.2-15.el6_4
  chkconfig.x86_64 0:1.3.49.3-2.el6_4.1
  coreutils.x86_64 0:8.4-19.el6_4.2
  coreutils-libs.x86_64 0:8.4-19.el6_4.2
  curl.x86_64 0:7.19.7-37.el6_4
  db4.x86_64 0:4.7.25-18.el6_4
  db4-utils.x86_64 0:4.7.25-18.el6_4
  dbus-glib.x86_64 0:0.86-6.el6_4
  dhclient.x86_64 12:4.1.1-34.P1.el6_4.1
  dhcp-common.x86_64 12:4.1.1-34.P1.el6_4.1
  dmidecode.x86_64 1:2.11-2.el6_1
  e2fsprogs.x86_64 0:1.41.12-14.el6_4.2
  e2fsprogs-libs.x86_64 0:1.41.12-14.el6_4.2
  glibc.x86_64 0:2.12-1.107.el6_4.4
  glibc-common.x86_64 0:2.12-1.107.el6_4.4
  gzip.x86_64 0:1.3.12-19.el6_4
  initscripts.x86_64 0:9.03.38-1.el6_4.2
  iputils.x86_64 0:20071127-17.el6_4.2
  kernel-firmware.noarch 0:2.6.32-358.18.1.el6
  krb5-libs.x86_64 0:1.10.3-10.el6_4.6
  libblkid.x86_64 0:2.17.2-12.9.el6_4.3
  libcom_err.x86_64 0:1.41.12-14.el6_4.2
  libcurl.x86_64 0:7.19.7-37.el6_4
  libnl.x86_64 0:1.1.4-1.el6_4
  libselinux.x86_64 0:2.0.94-5.3.el6_4.1
  libselinux-utils.x86_64 0:2.0.94-5.3.el6_4.1
  libss.x86_64 0:1.41.12-14.el6_4.2
  libuuid.x86_64 0:2.17.2-12.9.el6_4.3
  libxml2.x86_64 0:2.7.6-12.el6_4.1
  libxml2-python.x86_64 0:2.7.6-12.el6_4.1
  module-init-tools.x86_64 0:3.9-21.el6_4
  mysql-libs.x86_64 0:5.1.69-1.el6_4
  nspr.x86_64 0:4.9.5-2.el6_4
  nss.x86_64 0:3.14.3-4.el6_4
  nss-softokn.x86_64 0:3.14.3-3.el6_4
  nss-softokn-freebl.x86_64 0:3.14.3-3.el6_4
  nss-sysinit.x86_64 0:3.14.3-4.el6_4
  nss-tools.x86_64 0:3.14.3-4.el6_4
  nss-util.x86_64 0:3.14.3-3.el6_4
  openldap.x86_64 0:2.4.23-32.el6_4.1
  openssl.x86_64 0:1.0.0-27.el6_4.2
  python.x86_64 0:2.6.6-37.el6_4
  python-dmidecode.x86_64 0:3.10.13-3.el6_4
  python-libs.x86_64 0:2.6.6-37.el6_4
  python-rhsm.x86_64 0:1.8.17-1.el6_4
  rhn-check.noarch 0:1.0.0.1-8.el6
  rhn-client-tools.noarch 0:1.0.0.1-8.el6
  rhn-setup.noarch 0:1.0.0.1-8.el6
  rhnlib.noarch 0:2.5.22-15.el6
  rsyslog.x86_64 0:5.8.10-7.el6_4
  selinux-policy.noarch 0:3.7.19-195.el6_4.12
  selinux-policy-targeted.noarch 0:3.7.19-195.el6_4.12
  setup.noarch 0:2.8.14-20.el6_4.1
  subscription-manager.x86_64 0:1.8.22-1.el6_4
```

```
  tzdata.noarch 0:2013c-2.el6
  upstart.x86_64 0:0.6.5-12.el6_4.1
  util-linux-ng.x86_64 0:2.17.2-12.9.el6_4.3
  yum-rhn-plugin.noarch 0:0.9.1-49.el6

#### after updates reboot
reboot

#### Edit /boot/grub/grub.conf
vi /boot/grub/grub.conf

# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You do not have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /, eg.
#          root (hd0,0)
#          kernel /boot/vmlinuz-version ro root=/dev/vda1
#          initrd /boot/initrd-[generic-]version.img
#boot=/dev/vda
default=0
timeout=3
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Server (2.6.32-358.18.1.el6.x86_64)
        root (hd0,0)
        kernel /boot/vmlinuz-2.6.32-358.18.1.el6.x86_64 ro root=/dev/vda1 console=tty0
console=ttyS0
        initrd /boot/initramfs-2.6.32-358.18.1.el6.x86_64.img
title Red Hat Enterprise Linux (2.6.32-358.el6.x86_64)
        root (hd0,0)
        kernel /boot/vmlinuz-2.6.32-358.el6.x86_64 ro root=/dev/vda1 console=tty0
console=ttyS0
        initrd /boot/initramfs-2.6.32-358.el6.x86_64.img

#### Reboot after grub edit
reboot

#### Install additional packages:
yum install -y acpid openssh-clients sysstat tuned vim wget

Installed:
  acpid.x86_64 0:1.0.10-2.1.el6
  openssh-clients.x86_64 0:5.3p1-84.1.el6
  sysstat.x86_64 0:9.0.4-20.el6
  tuned.noarch 0:0.2.19-11.el6.1
  vim-enhanced.x86_64 2:7.2.411-1.8.el6
  wget.x86_64 0:1.12-1.8.el6

Dependency Installed:
  gpm-libs.x86_64 0:1.20.6-12.el6
  libedit.x86_64 0:2.11-4.20080712cvs.1.el6
  perl.x86_64 4:5.10.1-131.el6_4
  perl-Module-Pluggable.x86_64 1:3.90-131.el6_4
  perl-Pod-Escapes.x86_64 1:1.04-131.el6_4
  perl-Pod-Simple.x86_64 1:3.13-131.el6_4
  perl-libs.x86_64 4:5.10.1-131.el6_4
```

```
  perl-version.x86_64 3:0.77-131.el6_4
  vim-common.x86_64 2:7.2.411-1.8.el6

#### Add Common channel:
rhn-channel --add --channel=rhel-x86_64-server-rh-common-6

#### Install and configure cloud-init packages:
yum install -y cloud-init

Installed:
  cloud-init.noarch 0:0.7.1-2.el6

Dependency Installed:
  PyYAML.x86_64 0:3.10-3.1.el6
  audit-libs-python.x86_64 0:2.2-2.el6
  libcgroup.x86_64 0:0.37-7.2.el6_4
  libselinux-python.x86_64 0:2.0.94-5.3.el6_4.1
  libsemanage-python.x86_64 0:2.0.43-4.2.el6
  libyaml.x86_64 0:0.1.3-1.1.el6
  make.x86_64 1:3.81-20.el6
  policycoreutils-python.x86_64 0:2.0.83-19.30.el6
  python-argparse.noarch 0:1.2.1-2.1.el6
  python-boto.noarch 0:2.5.2-1.1.el6
  python-cheetah.x86_64 0:2.4.1-1.el6
  python-configobj.noarch 0:4.6.0-3.el6
  python-markdown.noarch 0:2.0.1-3.1.el6
  python-prettytable.noarch 0:0.6.1-1.el6
  python-pygments.noarch 0:1.1.1-1.el6
  python-setuptools.noarch 0:0.6.10-3.el6
  setools-libs.x86_64 0:3.3.7-4.el6
  setools-libs-python.x86_64 0:3.3.7-4.el6

#### Edit datasources file
vim /etc/cloud/cloud.cfg.d/10_datasources.cfg

datasource:
  Ec2:
    timeout : 10
    max_wait : 30
    metadata_urls:
     - http://10.0.0.1:9697

  MAAS:
    timeout : 10
    max_wait : 30

  NoCloud:
    seedfrom: None


#### Edit cloud.cfg file
vim /etc/cloud/cloud.cfg

#### Change...
ssh_pwauth:   0

#### To...
```

```
ssh_pwauth:   1
datasource_list: ["ConfigDrive", "Ec2", "NoCloud"]


#### Install IOZONE:
yum install -y http://apt.sw.be/redhat/el6/en/x86_64/rpmforge/RPMS/iozone-3.408-
1.el6.rf.x86_64.rpm

Installed:
  iozone.x86_64 0:3.408-1.el6.rf


#### Disable NIC udev rename:
ln -s /dev/null /etc/udev/rules.d/75-persistent-net-generator.rules
rm -f /etc/udev/rules.d/70-persistent-net.rules
vim /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

Tuning the guest

```
vi /etc/tune-profile/virtual-guest/ktune.sh

#### Change this line:
multiply_disk_readahead 4

#### To:
multiply_disk_readahead 16

#### Apply profile:
tuned-adm profile virtual-guest

reboot
```

Cleaning up and preparing for qcow2 compact

```
yum clean all
rm -rf /var/tmp/*
rm -rf /tmp/*
dd if=/dev/zero of=/zerofile.tmp bs=64k ; sync ; rm -f /zerofile.tmp ; sync
```

Running the tests

Each VM used its own cinder volume, backed by either Red Hat Storage or Ceph Storage, for each test. We formatted the virtual disk using ext4. The disk was reformatted before running the tests for every VM/node count combination.

```
mkfs.ext4 /dev/vdb
```

Example IOzone command:

```
iozone -+m iozone_ceph_4HV_16VM_write_64k_32g/vms.ioz -+h 192.168.43.11 -w -c -C -e -i 0 -+n -r 64k -s 32g -t 16 -+u
```

We used the following two scripts to run IOzone tests on the VMs from a remote system.

iozone_test.sh :

```
#!/bin/sh
export RSH="ssh -i GUEST_KEY.pem"
VM_COUNT=${5}
COMPUTE_COUNT=${6}
STORAGE_COUNT=4
VM_DEV=vdb
STORAGE_DEV=sda
INTERVAL=5
CEPH_MON=0

IOZ_RUNPATH="/usr/bin/iozone"
IOZ_TESTDIR="/mnt/test"

IOTYPE=read
if [ $2 -eq 0 ]; then
  IOTYPE=write
fi

if [ "${1}" = "ceph" ]; then
  STORAGE_DEV="sda sdb sdc sdd sde sdf sdg sdh"
  CEPH_MON=1
fi

# Prepare results folder and filename
IOZ_RESULT="iozone_${1}_${COMPUTE_COUNT}HV_${VM_COUNT}VM_${IOTYPE}_${3}_${4}"
RESULT_DIR=${IOZ_RESULT}
mkdir ${RESULT_DIR}

# Cleanup and drop caches
for i in `seq 1 ${VM_COUNT}`; do
  if [ $2 -eq 0 ]; then
    ssh -i GUEST_KEY.pem vm${i} "rm -fv /mnt/test/iozone*"
  fi
  ssh -i GUEST_KEY.pem vm${i} "sync ; echo 1 > /proc/sys/vm/drop_caches ; sync"
  echo "vm${i} ${IOZ_TESTDIR} ${IOZ_RUNPATH}" >> ${RESULT_DIR}/vms.ioz
done
for i in `seq 1 ${COMPUTE_COUNT}`; do
  ssh compute${i} "sync ; echo 1 > /proc/sys/vm/drop_caches ; sync"
done
for i in `seq 1 ${STORAGE_COUNT}`; do
  ssh storage${i} "sync ; echo 1 > /proc/sys/vm/drop_caches ; sync"
  #ssh storage${i} "sync"
done
sleep 1

# Start statistics collection
echo Starting stat collection...
for i in `seq 1 ${VM_COUNT}`; do
  ssh -i GUEST_KEY.pem vm${i} "pkill vmstat ; vmstat -n ${INTERVAL} | sed -e 's/^[
\t]*//' -e '3d'" > ${RESULT_DIR}/vmstat_vm${i}.log &
  ssh -i GUEST_KEY.pem vm${i} "pkill iostat ; iostat -xdm | grep Device ; iostat -xdm
${INTERVAL} ${VM_DEV} | awk '/${VM_DEV}/{i++; if (i>1) print}'" >
${RESULT_DIR}/iostat_vm${i}.log &
done

for i in `seq 1 ${COMPUTE_COUNT}`; do
```

```
  ssh compute${i} "pkill vmstat ; vmstat -n ${INTERVAL} | sed -e 's/^[ \t]*//' -e '3d'" >
${RESULT_DIR}/vmstat_compute${i}.log &
done

for i in `seq 1 ${STORAGE_COUNT}`; do
  ssh storage${i} "pkill vmstat ; vmstat -n ${INTERVAL} | sed -e 's/^[ \t]*//' -e '3d'" >
${RESULT_DIR}/vmstat_storage${i}.log &
  for STOR_DEV in ${STORAGE_DEV}; do
    ssh storage${i} "pkill iostat ; iostat -xdm | grep Device ; iostat -xdm ${INTERVAL}
${STOR_DEV} | awk '/${STOR_DEV}/{i++; if (i>1) print}'" >
${RESULT_DIR}/iostat_storage${i}_${STOR_DEV}.log &
  done
done

# Run iozone
sleep `expr $INTERVAL \* 2`

iozone -+m ${RESULT_DIR}/vms.ioz -+h 192.168.43.11 -w -c -C -e -i ${2} -+n -r ${3} -s
${4} -t ${VM_COUNT} -+u | tee ${RESULT_DIR}/${IOZ_RESULT}.txt

sleep `expr $INTERVAL \* 2`

# Stop statistics collection
for i in `seq 1 ${VM_COUNT}`; do
  ssh -i GUEST_KEY.pem vm${i} "pkill vmstat ; pkill iostat"
done
for i in `seq 1 ${COMPUTE_COUNT}`; do
  ssh compute${i} "pkill vmstat ; pkill iostat"
done
for i in `seq 1 ${STORAGE_COUNT}`; do
  ssh storage${i} "pkill vmstat ; pkill iostat"
done

wait

echo Test $IOZ_RESULT complete!
sleep $INTERVAL
```

run_tests.sh :

```
#!/bin/bash
VM_COUNT=$2
COMPUTE_COUNT=$1
#STORAGE_TYPE=gluster
STORAGE_TYPE=ceph
./prepare_vms.sh $VM_COUNT
sleep 1
./iozone_test.sh $STORAGE_TYPE 0 64k 32g $VM_COUNT $COMPUTE_COUNT
./iozone_test.sh $STORAGE_TYPE 1 64k 32g $VM_COUNT $COMPUTE_COUNT
```

Example output from an iozone test:

```
      Iozone: Performance Test of File I/O
              Version $Revision: 3.408 $
```

```
            Compiled for 64 bit mode.
            Build: linux

      Contributors:William Norcott, Don Capps, Isom Crawford, Kirby Collins
                   Al Slater, Scott Rhine, Mike Wisner, Ken Goss
                   Steve Landherr, Brad Smith, Mark Kelly, Dr. Alain CYR,
                   Randy Dunlap, Mark Montague, Dan Million, Gavin Brebner,
                   Jean-Marc Zucconi, Jeff Blomberg, Benny Halevy, Dave Boone,
                   Erik Habbinga, Kris Strecker, Walter Wong, Joshua Root,
                   Fabrice Bacchella, Zhenghua Xue, Qin Li, Darren Sawyer.
                   Ben England.

      Run began: Wed Oct 23 17:20:15 2013

      Network distribution mode enabled.
      Hostname = 192.168.43.11
      Setting no_unlink
      Include close in write timing
      Include fsync in write timing
      No retest option selected
      Record Size 64 KB
      File size set to 33554432 KB
      CPU utilization Resolution = 0.001 seconds.
      CPU utilization Excel chart enabled
      Command line used: iozone -+m iozone_ceph_4HV_16VM_write_64k_32g/vms.ioz -+h
192.168.43.11 -w -c -C -e -i 0 -+n -r 64k -s 32g -t 16 -+u
      Output is in Kbytes/sec
      Time Resolution = 0.000001 seconds.
      Processor cache size set to 1024 Kbytes.
      Processor cache line size set to 32 bytes.
      File stride size set to 17 * record size.
      Throughput test with 16 processes
      Each process writes a 33554432 Kbyte file in 64 Kbyte records

      Test running:
      Children see throughput for 16 initial writers  =  659177.61 KB/sec
      Min throughput per process                      =   34639.73 KB/sec
      Max throughput per process                      =   45219.37 KB/sec
      Avg throughput per process                      =   41198.60 KB/sec
      Min xfer                                = 25659584.00 KB
      CPU Utilization: Wall time  821.603    CPU time  326.315    CPU utilization  39.72
%

      Child[0] xfer count = 28329152.00 KB, Throughput =   38267.25 KB/sec, wall=816.876,
cpu=20.045, %=  2.45
      Child[1] xfer count = 31997504.00 KB, Throughput =   43012.71 KB/sec, wall=780.945,
cpu=20.956, %=  2.68
      Child[2] xfer count = 30749248.00 KB, Throughput =   41581.59 KB/sec, wall=791.910,
cpu=20.230, %=  2.55
      Child[3] xfer count = 33554432.00 KB, Throughput =   45219.37 KB/sec, wall=742.037,
cpu=20.109, %=  2.71
      Child[4] xfer count = 33336896.00 KB, Throughput =   44960.52 KB/sec, wall=750.150,
cpu=21.691, %=  2.89
      Child[5] xfer count = 30831360.00 KB, Throughput =   41581.80 KB/sec, wall=789.213,
cpu=20.562, %=  2.61
      Child[6] xfer count = 25659584.00 KB, Throughput =   34639.73 KB/sec, wall=821.603,
cpu=20.341, %=  2.48
```

```
      Child[7] xfer count = 30675136.00 KB, Throughput =   40858.56 KB/sec, wall=802.442,
cpu=20.050, %=  2.50
      Child[8] xfer count = 30791744.00 KB, Throughput =   41599.13 KB/sec, wall=790.786,
cpu=20.049, %=  2.54
      Child[9] xfer count = 30474752.00 KB, Throughput =   40741.44 KB/sec, wall=805.724,
cpu=19.748, %=  2.45
      Child[10] xfer count = 30769472.00 KB, Throughput =   42462.61 KB/sec,
wall=780.233, cpu=20.737, %=  2.66
      Child[11] xfer count = 28304960.00 KB, Throughput =   38084.23 KB/sec,
wall=813.566, cpu=19.939, %=  2.45
      Child[12] xfer count = 32530752.00 KB, Throughput =   44900.33 KB/sec,
wall=753.461, cpu=20.932, %=  2.78
      Child[13] xfer count = 29965504.00 KB, Throughput =   40478.07 KB/sec,
wall=792.468, cpu=20.133, %=  2.54
      Child[14] xfer count = 31770816.00 KB, Throughput =   42964.90 KB/sec,
wall=773.909, cpu=19.952, %=  2.58
      Child[15] xfer count = 28077248.00 KB, Throughput =   37825.36 KB/sec,
wall=811.191, cpu=20.843, %=  2.57


iozone test complete.
```

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.