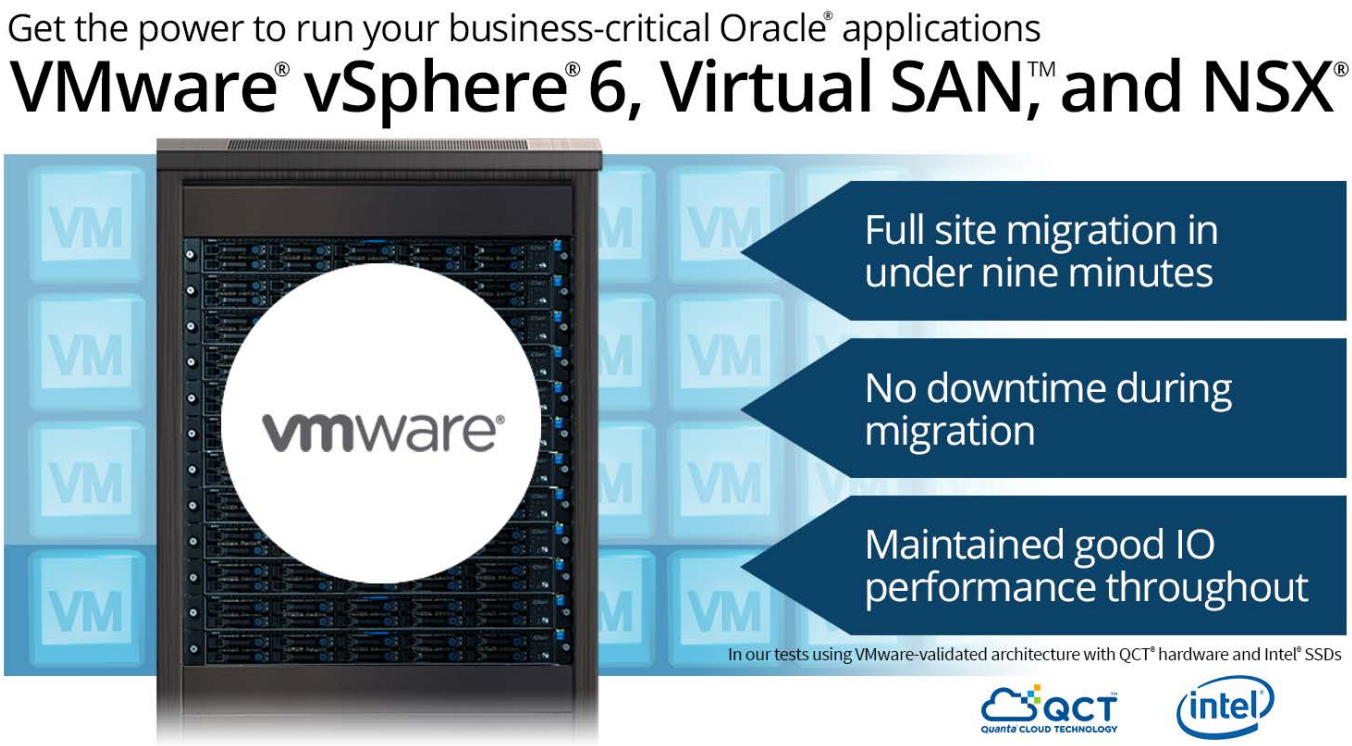
BUSINESS-CRITICAL APPLICATIONS ON VMWARE VSPHERE 6, VMWARE VIRTUAL SAN, AND VMWARE NSX



Business-critical applications need a virtualization platform that's reliable and flexible while providing the high performance that users demand. The software-defined data center (SDDC) with VMware vSphere 6, all-flash VMware Virtual SAN, and VMware NSX can help you deliver strong performance for business-critical apps, and maximum uptime through workload mobility.

Using a VMware Validated Design simulating two hardware sites in different locations, with QCT hardware using Intel® Xeon® processors, Intel Ethernet adapters, and Intel solid-state drives (SSDs), we put the VMware software-defined datacenter solution to the test. First, we tested a single site's performance under a heavy workload running 12 virtualized business-critical applications, specifically Oracle Database 12*c*, using VMware Virtual SAN in an all-flash configuration. We found that during the steady-state period of the test, the primary site delivered 189,170 total IOPS averaged across the test period, and maintained a cluster-wide average of 5ms read latency.

Next, in an active site-to-site evacuation scenario[1] using a stretched Virtual SAN cluster, with NSX providing virtual machine (VM) Layer 2 switching, we ran our heavy business-critical application workload, running it on both the primary and secondary sites. We then performed a complete primary-to-secondary site evacuation. In under 9 minutes, we live-migrated the active VMs from the primary site to the secondary, which doubled the number of active VMs running at the secondary site. All 24 VMs ran their database workloads for the duration of the migration with a combined site average of 427,213 IOPS without interruption in operations. By design, a stretched VSAN cluster eliminates the need to physically migrate data as other solutions do. Migrating data may take hours or even days for intensive workloads and can significantly degrade performance, with the potential for lost business and customers.

[1] Both sites were in the same datacenter but physically and logically separated to simulate a multi-site environment.

JANUARY 2016
A PRINCIPLED TECHNOLOGIES REPORT
Commissioned by VMware

TAKING VIRTUALIZATION FURTHER

For years, companies have seen the benefits of consolidation through virtualization. While virtual machines have mitigated issues such as physical server sprawl, power usage, and resource underutilization, other technologies in the infrastructure, such as shared storage and networking, have remained physical in regards to allocation and management. Traditionally, large and expensive traditional storage arrays and proprietary switching and routing technologies require additional space and special management. By using a VMware software-defined data center, you have the opportunity to take your storage and networking to the next level through storage and network virtualization.

With VMware vSphere 6, VMware builds on its flagship virtualization platform for compute by adding features and virtualization technologies in the areas of storage and networking. With VMware Virtual SAN, you can create virtual storage pools across your server hardware, removing the necessity for traditional, dedicated storage arrays, and allowing for simpler storage provisioning. Additionally, when running on supported switches, VMware NSX enables flexible virtual networking services (e.g., firewall) and connectivity (e.g., spanned Layer 2 networks) implementations and changes without requiring physical switch reconfiguration or re-cabling. NSX lets you relocate VMs across Layer 3 networks without the need for traditional guest OS re-addressing and VM downtime.

Using VMware technologies across compute, storage, and networking to run your software-defined datacenter has the potential to speed up provisioning time and ease the management burden while also reducing the costs associated with underutilized hardware.

In addition to these benefits, it is paramount for business-critical applications to run uninterrupted. Pooling of virtual resources with vSphere at the compute, storage, and networking layers allows you to migrate active critical applications from one set of hardware to another. An updated virtualization strategy can mean less downtime for your critical applications, and help you avoid the associated potential catastrophic costs to your business.

A TALE OF TWO TESTS

Single-site performance configuration

The QCT server rack

To show that VMware vSphere 6 is a fully capable platform that can excel at running business-critical applications, we focused on a single simulated site for our phase 1 testing. This single site, simulating a primary site, consisted of 12 QCT QuantaGrid D51B-1U servers, used to create three pods (clusters) as outlined in the

VMware Validated Design (VVD) reference architecture.[2] Not all of the hardware resources of this infrastructure were committed directly to the tests, as some resources were partially allocated to management and infrastructure to model the VVD design as well as highlight capabilities and future opportunities. The Edge pod contained VMware NSX components and ran on three physical servers. The Management pod contained vCenter Server and other supporting infrastructure components, and ran on three physical servers. The Compute pod containing our host hypervisor nodes and production VMware Virtual SAN ran on the remaining six physical servers. The Oracle 12*c* Database virtual machines ran on the Compute pod. For a visual representation of the primary site, see Figure 1.

The Intel SSDs

We configured an all-flash VMware Virtual SAN on each pod. Each server contained five Intel SSDs: one Intel SSD DC S3710 series SSD for write cache and four Intel SSD DC S3510 series SSDs for the capacity tier, configured as a single disk group. Each server in the Compute pod ran two application VMs (one configured with 64GB of RAM; the other with 96GB of RAM) and each VM contained an Oracle 12*c* Database installation with a 200GB database.

Two-site Live Migration performance configuration

We then identically configured a secondary site, matching the primary site's configuration, with 12 servers broken into three pods, also running business-critical Oracle Database on 12 VMs. For phase 2, we evacuated 12 VMs from the primary site to the secondary site for a total of 24 VMs, using the VMware Virtual SAN stretched cluster feature to stretch our storage resources across both sites. For specifics on VM sizing and Oracle Database configuration, see Appendix C. Figure 1 illustrates the simulated primary and secondary sites.

[2] http://www.vmware.com/software-defined-datacenter/validated-designs.html

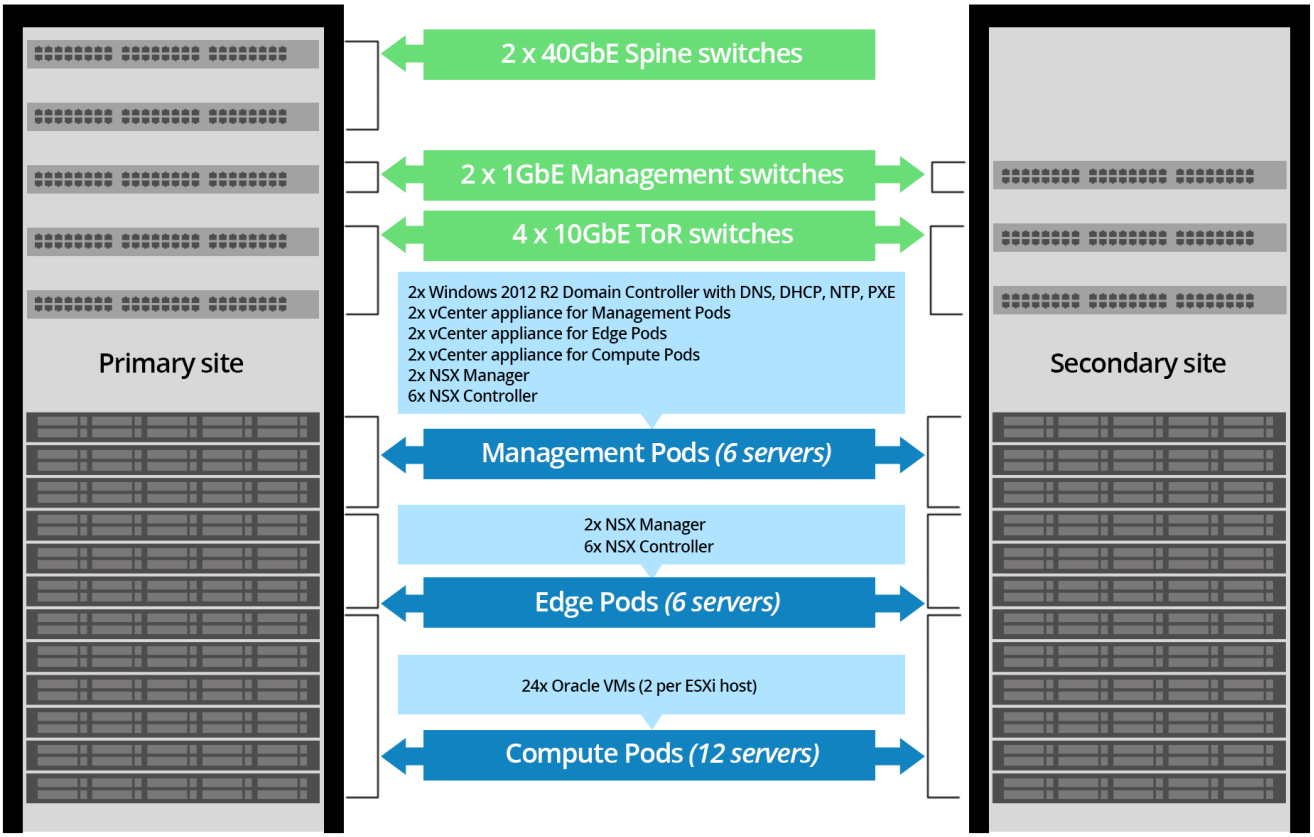


Figure 1: The simulated primary and secondary sites we simulated for testing.

How did the single-site VMware Validated Designs SDDC perform?

Business-critical applications need strong, steady performance to keep business moving. In our tests, we set out to see how well the VMware Validated Designs SDDC solution performed under the most extreme database load. To accomplish this, we employed the Silly Little Oracle Database Benchmark (SLOB) to generate an extremely heavy workload on all twelve Oracle VMs in our simulated primary site. While users can configure SLOB to simulate a more realistic database load, we set it to the maximum number of users (128) with zero think time to hit each database with maximum requests, fully stressing the virtual resources of each VM. We also set the workload to a mix of 75% reads and 25% writes to mimic a transactional database workload. As Figure 2 shows, the VMware Validated Design primary site achieved a stream of high IOPS, approaching 200,000 during the steady state portion of the test. The first 20 minutes were a warm-up period. As stress on the cache tier increased with write activity, the IOPS performance reached a steady state between 150,000-200,000 IOPS, representative of the database performance a business could expect for a similar workload running continuously at peak utilization. This means that our software-defined

datacenter solution provided reliable performance for a business-critical application like Oracle Database, despite the high intensity of the workload.

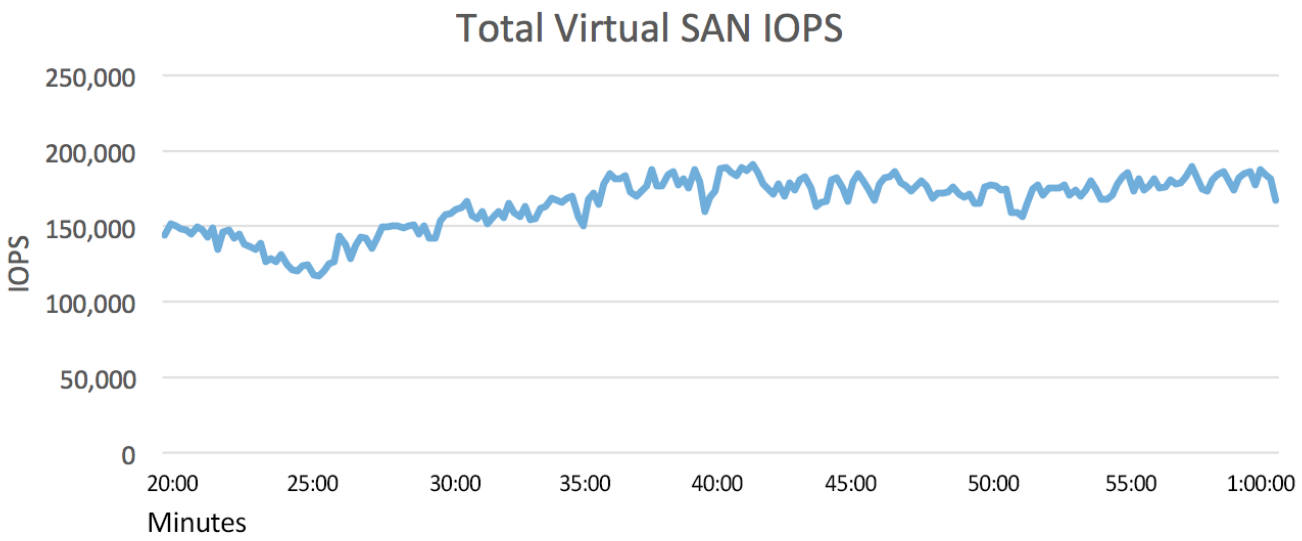


Figure 2: IOPS the software-defined datacenter solution achieved over the steady-state period of our single site performance test.

Latency is a critical measure of how well your workload is running: The longer it takes the server to respond, the longer applications and their respective users must wait. As Figure 3 shows, in the steady state period after a 20-minute warmup period, our VMware vSphere Validated Design SDDC solution ran with relatively low latencies considering it was handling sustained intensive workloads. This shows that this VMware Validated Design SDDC solution can provide quick response times for business-critical applications to help keep customers satisfied even in an extremely heavy I/O scenario. Our purpose was to demonstrate maximum performance during a peak utilization scenario; typical real-world environments should have lower I/O latencies.

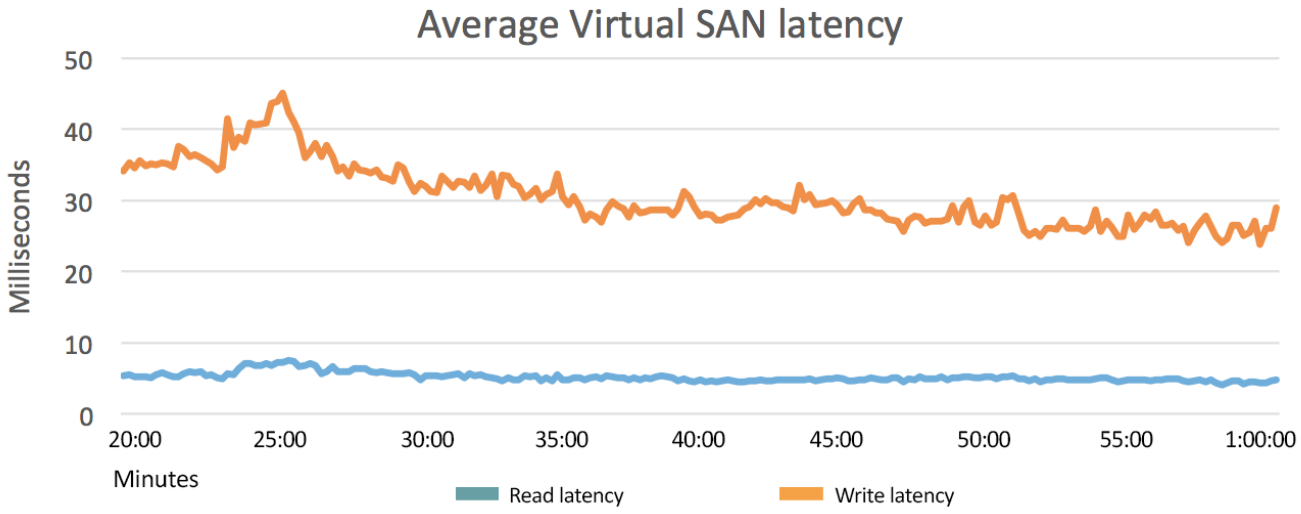


Figure 3: Latency for the software-defined datacenter solution over the steady state period of a single-site performance test.

Figure 4 shows the average host CPU utilization for our single-site performance test. This low average CPU utilization shows that the CPU was not the bottleneck and indicates additional processing resources would be available for future growth.

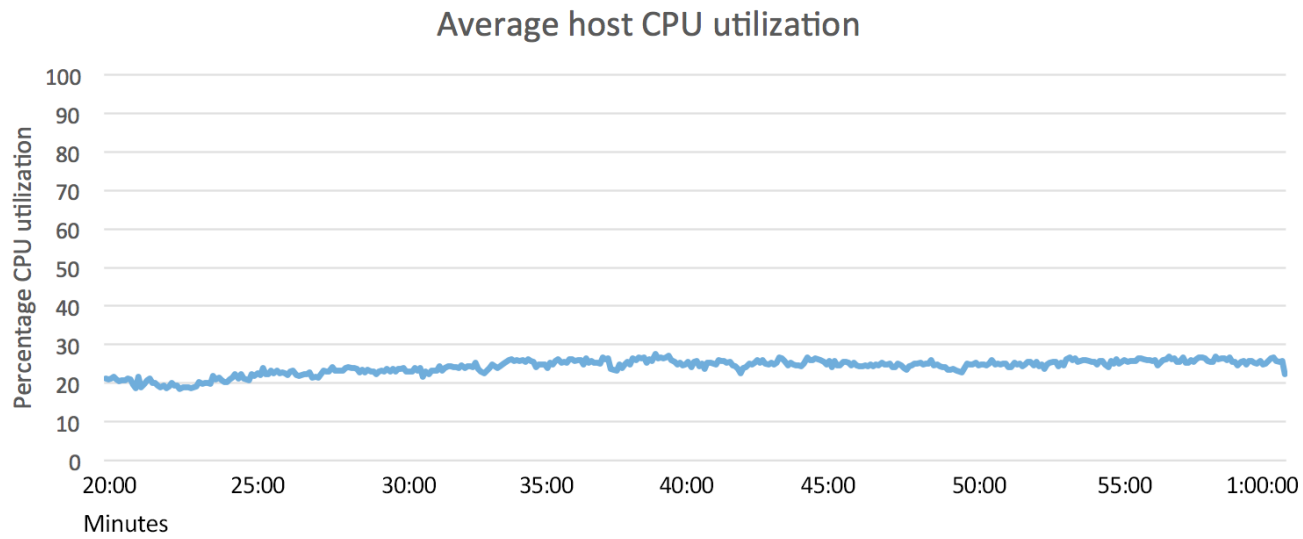


Figure 4: CPU utilization for the software-defined datacenter solution over the steady state period of single-site performance test.

How did the two-site VMware Validated Designs SDDC perform during a site evacuation?

Despite best planning, problems occur or maintenance is required. One of the major benefits of a VMware software-defined datacenter solution is that your virtualized servers, storage, and networking are easy to move to other hardware or sites via a single console to keep your applications up. In this phase of the study, we ran the same heavy workload in two places – on both the primary and secondary site. Then, we executed a primary site evacuation while continuously running the heavy database workloads on both sites.

As Figure 5 shows, we were able to migrate and consolidate our critical workloads at peak utilization from the primary to the secondary site servers in just 8 minutes 14 seconds with no application downtime or relocating of data. For bare metal infrastructures or those with basic virtualization technologies, it could take days to migrate all workloads to another location or to start workloads again from backups. This would result in considerable downtime, something that the VMware Validated Designs SDDC can help prevent. We found that the agility of the VMware Validated Designs SDDC kept every workload running with no downtime, which means that your business doesn't suffer when it's time to move workloads.

VSAN performance during the migration window was 427,213 IOPS, combining the average IOPS of each site, with average write latencies of 16.82ms and read latencies of 5.06ms across both sites. When the workloads were all consolidated to a single site, performance was at a steady average of 275,329 total IOPS. The total workload's output decreased because it lost half the number of physical resources available to it. Most importantly, the secondary site, now hosting all 24 heavy workloads, experienced zero downtime, did not require a physical data migration, and crossed Layer 3 networks without the need for guest OS re-addressing.

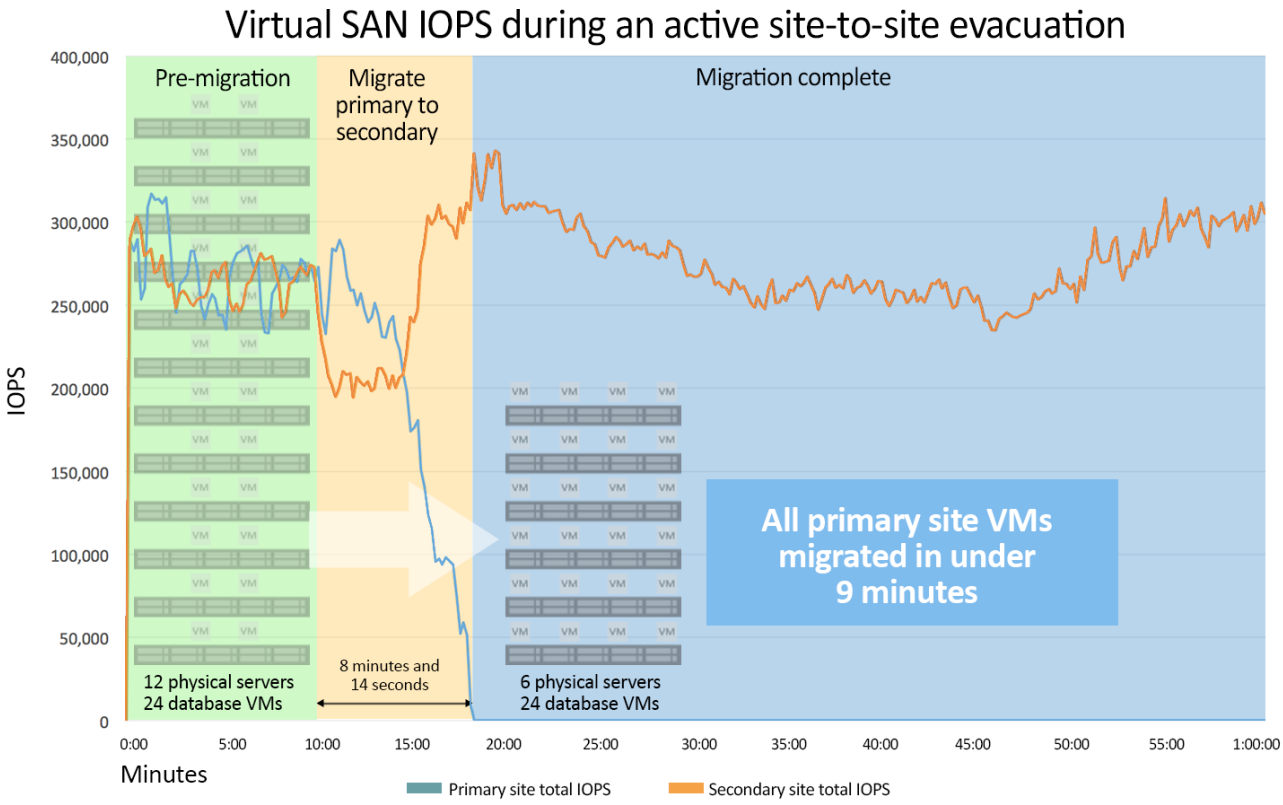


Figure 5: When migrating workloads to the secondary site, the workloads remained up during the 8-minute live migration and continued to run after migration onto a single site. Though physical resources were halved, the secondary site absorbed all work and remained up and available.

Figure 6 shows the average CPU utilization at each site through the duration of our test. As in the single-site test, CPU resources were not overtaxed, even after the secondary site took on the additional workload of the primary site following migration. The relatively low CPU utilization indicates processing room to grow as you scale out in the future.

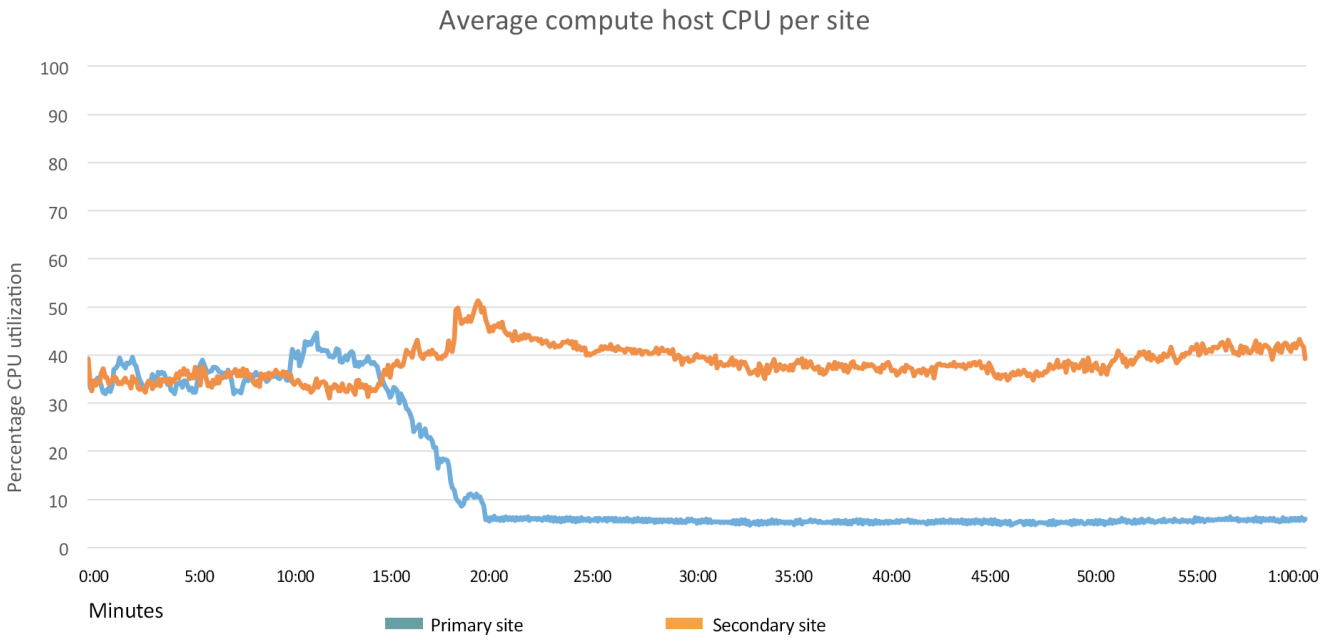


Figure 6: Average CPU utilization at both the primary and secondary sites.

CONCLUSION

Moving to the virtualized, software-defined datacenter can offer real benefits to today's organizations. As our testing showed, virtualizing business-critical applications with VMware vSphere, VMware Virtual SAN, and VMware NSX not only delivered reliable performance in a peak utilization scenario, but also delivered business continuity during and after a simulated site evacuation.

Using this VMware Validated Design with QCT hardware and Intel SSDs, we demonstrated a virtualized critical Oracle Database application environment delivering strong performance, even when under extreme duress.

Recognizing that many organizations have multiple sites, we also proved that our environment performed reliably under a site evacuation scenario, migrating the primary site VMs to the secondary in just over eight minutes with no downtime.

With these features and strengths, the VMware Validated Design SDDC is a proven solution that allows for efficient deployment of components and can help improve the reliability, flexibility, and mobility of your multi-site environment.

APPENDIX A – ABOUT THE COMPONENTS

For detailed hardware specifications and workload parameters, see Appendix B and Appendix C.

About QCT QuantaGrid D51B servers

QCT (Quanta Cloud Technology) is a global datacenter solution provider extending the power of hyperscale datacenter design in standard and open SKUs to datacenter customers. Product lines include servers, storage, network switches, integrated rack systems and cloud solutions, all designed to deliver hyperscale efficiency, scalability, reliability, manageability, serviceability, and optimized performance for each workload.

The QuantaGrid D51B server is a 1U rack server that features the Intel® Xeon® processor E5-2600 v3 product family in a two-socket configuration. QCT designed the server to withstand higher temperatures to assist organizations in saving on cooling costs in the datacenter. The QuantaGrid D51B has 24 memory slots with up to 1.5 TB capacity and can support two PCIe NVMe SSDs for accelerated storage performance.

To learn more about the QuantaGrid D51B, visit www.qct.io/Product/Server/Rackmount-Server/1U/QuantaGrid-D51B-1U-p255c77c70c83c85.

About the Intel SSD Data Center Family

Intel offers a number of drive options to match the demands of your specific workloads. The Intel SSD Data Center family includes SATA, PCIe, and PCIe NVMe SSDs are designed to meet the intense read and write demands of business-critical applications including the Oracle Database workloads we used in our tests.

To learn about the various datacenter class SSD options that Intel offers, visit www.intel.com/content/www/us/en/solid-state-drives/data-center-family.html.

About SLOB 2.2

The Silly Little Oracle Benchmark (SLOB) can assess Oracle random physical I/O capability on a given platform in preparation for potential OLTP/ERP-style workloads by measuring IOPS capacity. The benchmark helps evaluate performance of server hardware and software, storage system hardware and firmware, and storage networking hardware and firmware.

- SLOB contains simple PL/SQL and offers the ability to test the following:
- Oracle logical read (SGA buffer gets) scaling
- Physical random single-block reads (db file sequential read)
- Random single block writes (DBWR flushing capacity)
- Extreme REDO logging I/O

SLOB is free of application contention yet is an SGA-intensive benchmark. According to SLOB's creator Kevin Closson, SLOB can also offer more than testing IOPS capability such as studying host characteristics via NUMA and processor threading. For more information on SLOB, links to information on version 2.2, and links to download the benchmark, visit kevinclosson.net/2012/02/06/introducing-slob-the-silly-little-oracle-benchmark/.

APPENDIX B – SYSTEM CONFIGURATION INFORMATION

Figure 7 provides detailed configuration information for the test systems.

System	QCT QuantaGrid D51B-1U
General	
Number of processor packages	2
Number of cores per processor	12
Number of hardware threads per core	2
System power management policy	Balanced
CPU	
Vendor	Intel
Name	Xeon
Model number	E5-2670 v3
Socket type	FCLGA2011-3
Core frequency (GHz)	2.30
Bus frequency	9.6 GT/s
L1 cache	32 KB + 32 KB
L2 cache	256 KB per core
L3 cache	30 MB
Platform	
Vendor and model number	QCT QuantaGrid D51B-1U
Motherboard model number	Quanta S2B-MB
BIOS name and version	Aptio S2B_3A19
BIOS settings	SATADOMs set to AHCI
Memory modules	
Total RAM in system (GB)	384
Vendor and model number	Samsung® M393A1G40DB0-CPB
Type	PC4-2133P
Speed (MHz)	2,133
Speed running in the system (MHz)	1,600
Size (GB)	16
Number of RAM module(s)	24
Chip organization	Dual-sided
Rank	Dual
Hypervisor	
Name	VMware ESXi™ 6.0.0
Build number	3029758
Language	English
RAID controller	
Vendor and model number	LSI SAS3008
Firmware version	6.00.00.00-IR
Hard drives #1	
Vendor and model number	Intel SSDSC2BA800G4
Number of drives	1
Size (GB)	800

System	QCT QuantaGrid D51B-1U
Type	SSD
Hard drives #2	
Vendor and model number	Intel SSDSC2BB800G6
Number of drives	4
Size (GB)	800
Type	SSD
Hard drives #3	
Vendor and model number	Quanta ADF016GI000 SATADOM
Number of drives	2
Size (GB)	16
Type	MLC SSD
Network adapter #1	
Vendor and model number	Intel Ethernet Converged Network Adapter X520 SFI/SFP+ Dual Port
Type	Discrete
Network adapter #2	
Vendor and model number	Intel Ethernet Gigabit Server Adapter I350 Dual Port
Type	Integrated
USB ports	
Name	2
Type	3.0

Figure 7: Configuration information for our test systems.

APPENDIX C – HOW WE TESTED

About our hardware configuration

Our testbed hardware configuration consisted of 24 QCT QuantaGrid D51B-1U servers split into two racks of 12 servers each to simulate two separate sites. Each server contained an internal SATADOM device on which we installed VMware ESXi 6.0 Update 1. For the Virtual SAN on each server, we utilized one Intel SSD DC S3710 series 800GB SSD and four Intel SSD DCS3510 series 800GB SSDs.

Each host had two Intel Xeon processors E5-2670 v3 and 384 GB of memory. Each host used a dual-port Intel Ethernet Converged Network Adapter X520, each connected to one of two QCT QuantaMesh T3048-LY8 10Gb switches in the rack. Each server also had one port from an onboard Intel Ethernet Gigabit Server Adapter connected to one QCT QuantaMesh T1048-LB9 1Gb switch. The 1Gb ports were used for a temporary management network, which was eventually migrated to the 10Gb network. To bridge the two sites together, we used two QCT QuantaMesh T5032-LY6 40Gb switches. Each switch ran Cumulus Linux 2.5.3a, which we configured with help of VMware to match VMware's Validated Design SDDC. We cabled each switch to be highly fault tolerant with fully redundant cabling from each 1Gb, 10Gb, and 40Gb switch to the other as well as in between network stacks. See Figure 8 for a full network diagram.

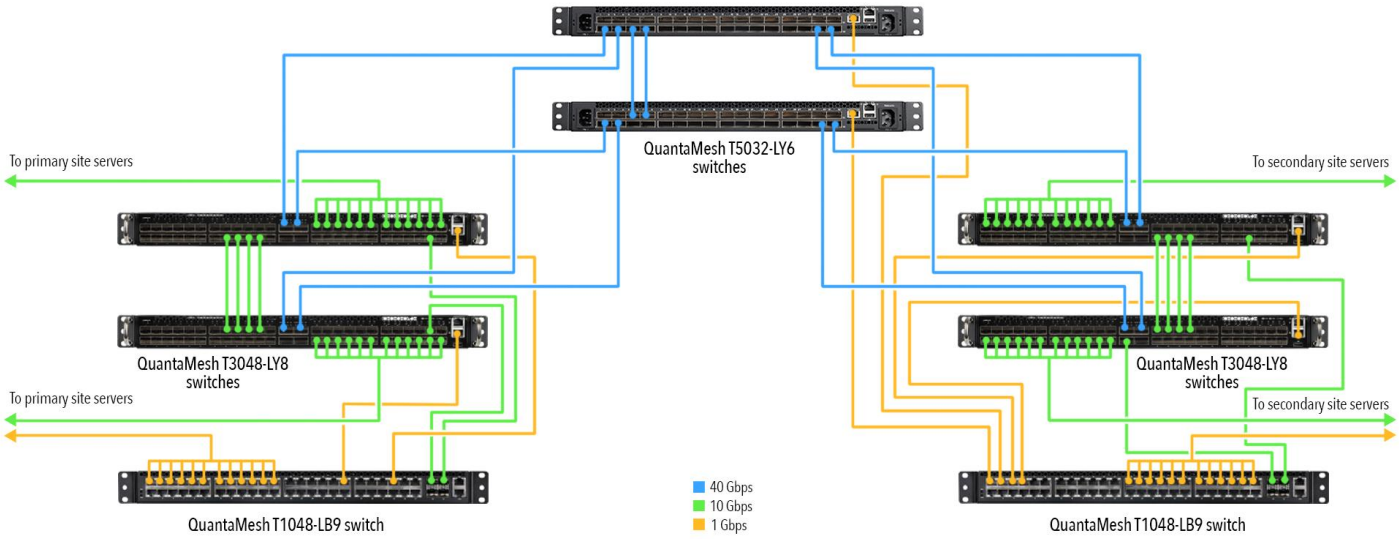


Figure 8: Our networking setup.

For the workload VMs, we placed one medium size VM and one large size VM on each of our compute pod hosts to create a mixed load on each server. Figure 9 shows the vCPUs, memory allocation, and VMDK placement for each size VM.

VM	vCPUs	Memory	OS1	DATA1	DATA2	DATA3	DATA4	LOG1
Medium	4	64 GB	50GB	100GB	100GB	100GB	100GB	30GB
Large	8	96 GB	50GB	100GB	100Gb	100GB	100GB	30GB

Figure 9: VM configurations.

About our workload configuration

For our SLOB workload VMs, we installed Oracle Enterprise Linux 7 onto one VM and installed Oracle Database 12*c*. We then installed the SLOB benchmark locally on the VM and created a ~200GB database. We cloned out 24 VMs, 12 at each site and two per compute server, and adjusted the vCPU and RAM settings for each VM according to the sizing outlined in Figure 7.

Configuring the hosts

We used the following settings to configure the environment. We installed the hypervisor on one of the internal SATADOMs and left the five SSDs in the front of the server for Virtual SAN. The BIOS in each server was set to defaults with the exception of setting the SATADOM to the boot device.

Installing VMware ESXi 6 Update 1

1. Attach the installation media.
2. Boot the server.
3. At the VMware Installer screen, press Enter.
4. At the EULA screen, press F11 to Accept and Continue.
5. Under Storage Devices, select the appropriate virtual disk, and press Enter.
6. Select US as the keyboard layout, and press Enter.
7. Enter the root password twice, and press Enter.
8. To start installation, press F11.
9. After the server reboots, press F2, and enter root credentials.
10. Select Configure Management Network, and press Enter.
11. Select the appropriate network adapter, and select OK.
12. Select IPv4 settings, and enter the desired IP address, subnet mask, and gateway for the server.
13. Select OK, and restart the management network.
14. Repeat steps 1-13 on all 24 servers.

Deploying the vCenter Server Appliance 6.0

The layout of the environment required two vCenters per site: one to host the management server clusters and one to host the edge and compute server clusters. We deployed all four vCenters to a standalone ESXi server and then migrated them to their final locations once we created the server clusters, virtual networking, and VSAN volumes. To simplify the management of the entire testbed, we connected all four vCenters under the same single sign-on. We used the following steps to deploy each of the four vCenter Server Appliances utilizing a separate controller Windows server to facilitate the deployment process. The vCenter Server Appliance 6.0 ISO is available from the following link https://my.vmware.com/en/web/vmware/info/slug/datacenter_cloud_infrastructure/vmware_vsphere/6_0

1. From a Windows server, mount the vCenter Server Appliance ISO.
2. Install the plugin in the vcsa folder.
3. From the ISO root folder, open the vcsa-setup.html file.
4. Once the VCSA prompt appears, click Install.
5. Accept the license agreement, and click Next.
6. Enter a target management cluster host IP address, and enter the proper root credentials.
7. Click Next, and click Yes to accept the host's certificate.

8. Enter a name for the appliance, and the desired root password. Click Next.
9. Click Install vCenter Server with an Embedded Platform Services Controller, and click Next.
10. For the first vCenter deployment, select Create a new SSO domain, enter the desired administrator password, enter the desired domain name, and enter an SSO site name.
11. For the remaining vCenter deployment, select Join an SSO domain in an existing vCenter 6.0 platform services controller, enter the Platform Services Controller IP address and the vCenter login password. Click Next, and select the existing site.
12. Click Next.
13. Select the desired Appliance size. For our testing, we selected Small. Click Next.
14. Select the host datastore to deploy the appliance to, and click Next.
15. Select Use an embedded database (vPostgres), and click Next.
16. Enter the network information for the new appliance, and click Next.
17. Click Finish.
18. Repeat steps 1-17 for the remaining three vCenters.
19. Once completed, login to the first vCenter's web client at https://*vcenter-ip-address*/vsphere-client/?csp.
20. Add all the necessary ESXi, vCenter, and VSAN licenses to each vCenter.

Creating the clusters and adding the hosts to vCenter

We used the following steps to create each of the three clusters in each site and add the desired servers to the vCenter. The clusters on each site are management, edge, and compute.

1. Once logged into the vCenter, navigate to Hosts and Clusters.
2. Select the primary site management vCenter.
3. Right-click the vCenter object, and select New Datacenter…
4. Enter a name for the new datacenter, and click OK.
5. Right-click the new datacenter, and click New Cluster…
6. Enter a name for the new cluster.
7. Expand DRS, and turn on DRS.
8. Click the drop-menu, and click Partially automated.
9. Expand vSphere HA, and turn on HA.
10. Expand Virtual SAN, and turn on Virtual SAN.
11. Click the drop-down menu, and set it to Manual.
12. Click OK.
13. Once the cluster has been created, right-click the cluster and click Add Host.
14. Enter the IP address for the first management server, and click Next.
15. Enter the root credentials for the server, and click Next.
16. Click Yes to accept the server's certificate.
17. Review the server details, and click Next.
18. Assign the desired license, and click Next.
19. Disable Lockdown mode, and click Next.

20. Click Finish.
21. Repeat steps 13-20 for the remaining management servers.
22. Repeat steps 1-21 on the remaining vCenters for the compute/edge primary and secondary vCenters and the secondary management vCenter, assigning the appropriate servers to each cluster.

Configuring the vSphere Distributed Switch for each cluster

For our networking configuration, we utilized vSphere Distributed Switches for each cluster. We then created three port groups: Management, vMotion, and VSAN. Each port group utilized VLANs: 970 for Management, 980 for vMotion, and 1020 for VSAN. Since the physical switches leverage LACP, we configured the vDS to use a link aggregation group as opposed to traditional NIC teaming.

1. Navigate to the Networking tab.
2. Expand the vCenter for the primary site management cluster.
3. Right-click the datacenter, and click Distributed Switch→New Distributed Switch.
4. Enter a name for the vDS, and click Next.
5. Select Distributed switch: 6.0.0, and click Next.
6. Set the number of uplinks to 2, and enter a name for the new default port group.
7. Click Next.
8. Review the settings, and click Finish.
9. Select the new vDS, and click the manage tab.
10. Under Properties, click Edit…
11. Click Advanced, and enter 9000 for the MTU setting.
12. Click OK.
13. Click LACP.
14. Click the + sign.
15. Enter a name for the new LAG, set the number of port to two.
16. Select Source and destination IP addresses and TCP/UDP port for the load balancing mode, and click OK.
17. On the left-hand side, right click the new port group that was created along with the new vDS.
18. Click Edit Settings.
19. Click VLAN, and enter 970 for the VLAN (for the management port group).
20. Click Teaming and Failover.
21. Set the new LAG as the active uplink, and set all the other uplinks as unused.
22. Click OK.
23. Right-click the vDS, and click Distributed Port Group→New Distributed Port Group.
24. Enter a name for the new port group (e.g. vMotion), and click Next.
25. Under VLAN, set the VLAN to 980 for vMotion.
26. Under Advanced, check the box Customize default policies configuration.
27. Click Next.
28. For Security, click Next.
29. For Traffic Shaping, click Next.

30. For Teaming and Failover, assign the LAG as the active port, and set all other uplinks to unused.
31. Click Next.
32. For Monitoring, click Next.
33. For Miscellaneous, click Next.
34. For Edit additional settings, click Next.
35. Review the port group settings, and click Finish.
36. Repeat steps 23-35 for the VSAN port group using 1020 for the VLAN.
37. Right-click the vDS, and click Add and Manage Hosts…
38. Select Add hosts, and click Next.
39. Click + New hosts…
40. Select all the hosts in the vCenter, and click OK.
41. Click Next.
42. Select Manage physical adapters and Manage VMkernel adapters, and click Next.
43. For each host, select each of the two 10Gb ports, click Assign uplink, and assign them to the two LAG ports.
44. Click Next.
45. On each host, and select the management VMkernel.
46. Click Assign port group.
47. Select the management port group, and click OK.
48. Select the first host, and click +New adapter.
49. Click select an existing network, and click Browse.
50. Select the vMotion port group, and click OK.
51. Click Next.
52. Check the vMotion traffic box, and click Next.
53. Enter the desired network information for the new VMKernel, and click Next.
54. Click OK.
55. Select the new adapter, and click Edit adapter.
56. Click NIC settings.
57. Set the MTU to 9000.
58. Click OK
59. Repeat steps 48-58 for the VSAN port group, selecting VSAN traffic for the service to enable and entering the proper IP settings for the VSAN network.
60. Repeat steps 48-59 for each host.
61. Click Next.
62. Analyze the impact, and click Next.
63. Review the settings and click Next.
64. Repeat steps 1-63 for each cluster.

Configuring Virtual SAN for each cluster

Once the VSAN network was in place, we used the following steps to configure the VSAN cluster, starting with the management clusters.

1. Prior to configuring the VSAN in each cluster, login to each server using an SSH client, and run the following commands to configure larger slabs for the caching of metadata. This was configured due to the randomness of the tested I/O and the large working sets (for additional information on these settings, see http://www.vmware.com/files/pdf/products/vsan/VMware-Virtual-San6-Scalability-Performance-Paper.pdf)

```
# esxcli system settings advanced set -o "/LSOM/blPLOGCacheLines" --int-value
   "131072"
# esxcli system settings advanced set -o "/LSOM/blPLOGLsnCacheLines" --int-
   value "32768"
```

2. Reboot each server after making this change.
3. Login to the vCenter web client, and navigate to Hosts and Clusters.
4. Expand the primary site management vCenter, and select the management cluster.
5. Click the Manage tab, and under Settings→Virtual SAN, click Disk Management.
6. Click the Claim Disks button.
7. vCenter will separate the disks into groups based on type. For the group of Intel S3710 series SSDs, select Cache Tier. For the group of Intel S3510 series SSDs, select Capacity.
8. Click OK.
9. Repeat steps 4-8 for each cluster in the environment.

Once the management clusters were completely configured with the vDS and VSAN, we migrated the vCenter servers to their respective management clusters from the temporary vCenter host, ensure their storage was migrated onto the VSAN and that the network was accessible through vDS.

Configuring the VSAN Stretched Cluster for VSAN (Phase 2 of testing only)

For phase two of our testing, we reconfigured the VSAN to use the stretched cluster feature, expanding the VSAN to incorporate all the compute nodes from each site. To prepare for the stretched cluster, we added all the secondary compute nodes to the primary compute vCenter and compute cluster. We used the following steps to deploy the Virtual SAN Witness ESXi OVA and the VSAN Stretched Cluster.

Deploying and configuring the Virtual SAN Witness ESXi OVA

1. Download the appliance from https://my.vmware.com/web/vmware/details?productId=536&downloadGroup=WITNESS_OVA
2. Open the vCenter web client, and navigate to Hosts and Clusters.
3. Right-click the primary management node datacenter, and click Deploy OVF Template…
4. Select Local file for the source, browse to the witness OVA file, and select it. Click Next.
5. Review the details, and click Next.
6. Accept the license agreement, and click Next.
7. Give the VM a name, and select the Discovered virtual machine folder. Click Next.
8. For the configuration, select Normal (up to 500 VMs), and click Next.

9. For the storage, select the management VSAN volume, and click Next.
10. Select the management VM network, and click Next.
11. Enter a password for the root user, and click Next.
12. Click Finish.
13. Once the witness is deployed, right-click it and select Edit Settings…
14. Assign the second NIC to the VSAN port group, and click OK.
15. Open a console to the witness VM, and assign an IP to the management network as you would on a typical ESXi host.

Adding the witness host to vCenter and configuring the VSAN network

1. Once the VM is connected to the management network, create a new datacenter for the witness VM on the primary compute vCenter.
2. Right-click the datacenter, and click Add Host…
3. Enter the IP of the witness VM, and click Next.
4. Enter the credentials of the witness VM, and click Next.
5. Click Yes to accept the certificate.
6. Click Next.
7. Click Next to accept the default license.
8. Click Next to disable Lockdown Mode.
9. Click Next.
10. Click Finish.
11. Once the host is added to the datacenter, select it, and navigate to Manage→Networking.
12. Select the witnessPG portgroup, and edit the VMkernel.
13. Enable the Virtual SAN traffic services, and click NIC Settings.
14. Enter 9000 for the MTU setting, and click IPv4 Settings.
15. Enter the desired IP settings for the VSAN network using the same subnet as the compute hosts VSAN network.
16. Click OK.

Configuring the Stretched VSAN cluster

1. Navigate to the primary compute cluster, selecting Manage→Virtual SAN – General.
2. Ensure that the VSAN is healthy and that all the SSDs are configured in their proper disk groups.
3. Select Fault Domains, and click the Configure VSAN Stretched Cluster icon.
4. Assign all the primary site hosts to the preferred fault domain and all the secondary site hosts to the secondary fault domain. Click Next.
5. Select the witness appliance host that was deployed in the previous steps, and click Next.
6. Select the witness appliance's virtual flash disk and HDD, and click Next.
7. Verify the settings, and click Next.
8. Once the stretched cluster is created, navigate to Monitor→Virtual SAN→Health, and ensure that the VSAN is healthy.

To ensure that the proper VMs power up on the right sites, we create host groups for each host and VM groups for each pair of VMs. We then setup host-affinity rules, stating that one pair of VMs (one medium, one large) should startup on each host. This helped place VMs on the stretched VSAN, 12 VMs per site, and allowed easy repeatability of our test.

Installing and configuring VMware NSX

Our NSX setup was built in an environment with four active vDS port groups:

- Management
- vMotion
- VSAN

Deploying and configuring the NSX manager template

1. Log into the vCenter web client.
2. Right-click on the management cluster and select Deploy OVF Template.
3. In the Deploy OVF Template window, choose Local file and Browse.
4. Select your NSX Manager ova and click Open.
5. Click Next.
6. In Review Details, check Accept extra configuration options and click Next.
7. In Accept License Agreements, click Accept, and click Next.
8. In Select name and folder, name your NSX manager, and click Next.
9. In Select storage, choose the storage for your VM and click Next.
10. In Setup networks, select the management network and click Next.
11. In Customize template, fill in your NSX manager's passwords, hostname, and IP address, and click Next.
12. In Ready to complete, check the Power on after deployment checkbox and click Finish.
13. After the NSX manager has booted, open up a new web browser and navigate to the IP address you set in step 11
14. In NSX Manager Appliance Management, click Manage vCenter Registration.
15. In the NSX Management Service window, go to Lookup Service and click Edit.
16. In the Lookup Service window, type in the hostname of the vCenter you are attaching the NSX manager to, the port (default 443), and the credentials, then click OK.
17. Back in the NSX Management Service window, go to vCenter Server and click Edit.
18. In the vCenter Server window, type the hostname of your vCenter server and the credentials to log in and click OK.
19. Repeat steps 1 through 18 for each vCenter installation.

Creating NSX controllers

1. Log into vCenter
2. In the home screen, navigate to Networking & Security
3. On the left pane, click Installation.
4. In the Management tab, click the Add Controller button.
5. In the Add Controller button, configure your controller so that it's connected to the management network of the NSX manager and an IP pool that is in the same subnet as the NSX manager, and click OK.
6. Wait for the first NSX controller node to deploy, then create two more controllers.
7. When the controllers have deployed, click on the Host Preparation tab.
8. Click on each host cluster, then click the Actions button and select Install.

9. When the Installation Status column shows a green checkmark, click on Not Configured on the VXLAN column.
10. In the Configure VXLAN Networking window, select your datacenter's DVS, then put in the appropriate VLAN and MTU information (we used VLAN 3000), use the IP pool you made earlier, and click OK.
11. Click on the Logical Network Preparation tab.
12. In the Logical Network Preparation tab, click on Segment ID.
13. In Segment ID, click Edit.
14. Enter a range for your Segment ID (we wrote 5000-5200) and click OK.
15. Click back to the Management tab.
16. In the Management tab, click on the first NSX Manager and select Actions → Assign Primary Role.
17. Click on the each of the other NSX Managers and select Actions → Assign Secondary Role.

Configuring the NSX universal networking

1. Log into vCenter.
2. In the home screen, navigate to Networking & Security
3. On the left pane, click Installation.
4. In the Installation window, click Segment ID.
5. In the Segment ID space, click Edit.
6. Add in the Universal Segment ID pool (we used 900000-909999) and click OK.
7. Click Transport Zones.
8. In the Transport Zones window, click New Transport Zone.
9. In the New Transport Zone window, name your transport zone, connect it to your clusters, and make sure that Mark this object for universal synchronization is checked, then click OK.
10. In the left pane, click Logical Switches.
11. Click the New Logical Switch button.
12. In the New Logical Switch window, name your switch, add your universal transport zone, and click OK.

Creating the workload virtual machines – SLOB

Creating the VM

1. In VMware vCenter, navigate to Virtual Machines.
2. Click the icon to create a new VM.
3. Leave Create a new virtual machine selected and click Next.
4. Enter a name for the virtual machine and click Next.
5. Place the VM on the desired host with available CPUs and click Next.
6. Select the VSAN datastore for the 50GB OS VMDK and click next.
7. Click Next.
8. Select the guest OS as Oracle Enterprise Linux 7 and click Next.
9. In the Customize Hardware section, make the following changes:
10. Increase the vCPUs to 4 or 8 (depending on VM sizing).
11. Increase the memory to 64GB or 96 (depending on VM sizing).
12. Add four 100GB VMDKs for Oracle data and one 30GB VMDK for Oracle logs. Place the VMDK in the VSAN datastore and ensure that the VSAN storage policy is enable for the disk.
13. Create 3 additional VMware Paravirtual SCSI controllers, and place two data VMDKs on one, two data VMDKs on another, and the one log VMDK on the last one.
14. Connect the VM to the NSX network.

15. Click Next.
16. Click Finish.

We then installed Oracle Enterprise Linux on the VM, using the basic install. We all adjusted the partition layout to include a 20GB swap for Oracle Database. We used the steps below to configure the OS, install Oracle Database 12c, and configure the SLOB database.

Initial configuration tasks

Complete the following steps to provide the functionality that Oracle Database requires. We performed all of these tasks as root.

1. Install VMware tools

```
# yum install open-vm-tools
```

2. Disable firewall services. In the command line (as root), type:

```
# systemctl stop firewalld
# systemctl disable firewalld
```

1. Edit `/etc/selinux/config`:

```
SELINUX=disabled
```

2. Modify `/etc/hosts` to include the IP address of the internal IP and the hostname.
3. Install 12*c* RPM packages, resolve package dependencies, and modify kernel parameters:

```
# yum install oracle-rdbms-server-12cR1-preinstall –y
```

4. Using yum, install the following prerequisite packages for the oracleasm tool:

```
# yum install oracleasm-support oracleasmlib oracleasm
```

5. Create the `oracle` user account and groups and password:

```
# groupadd -g 1003 oper
# groupadd -g 1004 asmadmin
# groupadd -g 1005 asmdba
# groupadd -g 1006 asmoper
# usermod -G dba,oper,asmadmin,asmdba,asmoper oracle
```

6. Create a password for the oracle user.

```
# passwd oracle
```

7. Create the `/u01` directory for Oracle inventory and software and give it to the oracle user:

```
# mkdir -p /u01/app/oracle/product/12.1.0/grid
# mkdir -p /u01/app/oracle/product/12.1.0/dbhome_1
# chown -R oracle:oinstall /u01
# chmod -R 775 /u01
```

8. Edit bash profiles to set up user environments:

```
# vim /home/oracle/.bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs
```

```
PATH=$PATH:$HOME/.local/bin:$HOME/bin

export PATH

# Oracle settings for the database environment
export TMP=/tmp
export TMPDIR=$TMP
export ORACLE_BASE=/u01/app/oracle
export GRID_HOME=$ORACLE_BASE/product/12.1.0/grid
export DATA_HOME=$ORACLE_BASE/product/12.1.0/dbhome_1
export ORACLE_HOME=$DATA_HOME
export ORACLE_SID=orcl
export BASE_PATH=/usr/sbin:$PATH
export PATH=$ORACLE_HOME/bin:$BASE_PATH
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib
# Two shortcuts for switching between database and grid environments.
alias grid_env='. /home/oracle/grid_env'
alias data_env='. /home/oracle/data_env'

# vim /home/oracle/grid_env

export ORACLE_SID=+ASM
export ORACLE_HOME=$GRID_HOME
export PATH=$ORACLE_HOME/bin:$BASE_PATH
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib

# vim /home/oracle/db_env

export ORACLE_SID=orcl
export ORACLE_HOME=$DATA_HOME
export PATH=$ORACLE_HOME/bin:$BASE_PATH
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib
```

Configuring disks for ASM

1. For each of the five shared disks, create a GPT label, and create one partition. For example, see the following shell script:

```
for disk in b c d e f;
      do parted /dev/sd${disk} mklabel gpt;
      parted /dev/sd${disk} mkpart primary "1 -1";
done
```

2. Initialize Oracle ASM on each server by executing the following commands as root on each node.

```
# oracleasm configure -e -s y -u oracle -g asmadmin
# oracleasm init
```

3. Label each shared disk-partition with an appropriate ASM name. For example, following the OS partition names created above, execute the following commands on one system:

```
# oracleasm createdisk DATA1 /dev/sdb1
```

```
# oracleasm createdisk DATA2/dev/sdc1
# oracleasm createdisk DATA3 /dev/sdd1
# oracleasm createdisk DATA4 /dev/sde1
# oracleasm createdisk LOGS /dev/sdf1
```

4. On each server, scan the disks to make the disks immediately available to Oracle ASM.

```
# oracleasm scandisks
# oracleasm listdisks
```

Adding hugepages

Edit `/etc/sysctl.conf` by adding the following lines:

For the large VM configuration, use:

```
vm.nr_hugepages = 40448
vm.hugetlb_shm_group = 54321
```

For the medium VM configuration, use:

```
vm.nr_hugepages = 28672
vm.hugetlb_shm_group = 54321
```

Installing Oracle Grid Infrastructure 12*c*

1. Log in as the `oracle` user and enter `grid_env`.
2. Unzip `linuxamd64_12c_grid_1of2.zip` and `linuxamd64_12c_grid_2of2.zip`
3. Open a terminal to the unzipped database directory.
4. Type `grid_env` to set the Oracle grid environment.
5. To start the installer, type `./runInstaller`
6. At the Updates screen, select Skip updates.
7. In the Select Installation Option screen, select Install and Configure Grid Infrastructure for a Standalone Server, and click Next.
8. Choose the language, and click Next.
9. In the Create ASM Disk Group screen, choose the Disk Group Name, and change redundancy to External.
10. Select the four disks that you are planning to use for the database, and click Next.
11. In the Specify ASM Password screen, choose Use same password for these accounts, write the passwords for the ASM users, and click Next.
12. Leave the default Operating System Groups, and click Next.
13. Leave the default installation, and click Next.
14. Leave the default inventory location, and click Next.
15. Under Root script execution, select Automatically run configuration scripts and enter root credentials.
16. In the Prerequisite Checks screen, make sure that there are no errors.
17. In the Summary screen, verify that everything is correct, and click Finish to install Oracle Grid Infrastructure.
18. At one point during the installation, the installation prompts you to execute two configuration scripts as root. Follow the instructions to run the scripts.
19. At the Finish screen, click Close.
20. To run the ASM Configuration Assistant, type `asmca`.
21. In the ASM Configuration Assistant, click Create.

22. In the Create Disk Group window, name the new disk group `log`, choose redundancy External (None), and select the log disk for redo logs.
23. Click Advanced Options, and type 12.1.0.0.0 in ASM Compatibility and Database Compatibility. Click OK.
24. Right-click the DATA drive, and choose Edit Attributes. Make sure both ASM and Database Compatibility fields list 12.1.0.0.0, and click OK.
25. Exit the ASM Configuration Assistant.

Installing Oracle Database 12*c*

1. Unzip `linuxamd64_12c_database_1_of_2.zip` and `linuxamd64_12c_database_2_of_2.zip`.
2. Open a terminal to the unzipped database directory.
3. Type `db_env` to set the Oracle database environment.
4. Run `./runInstaller.sh`.
5. Wait until the GUI installer loads.
6. On the Configure Security Updates screen, enter the credentials for My Oracle Support. If you do not have an account, uncheck the box I wish to receive security updates via My Oracle Support, and click Next.
7. At the warning, click Yes.
8. On the Download Software Updates screen, enter the desired update option, and click Next.
9. On the Select Installation Option screen, select Install database software only, and click Next.
10. On the Grid Installation Options screen, select Single instance database installation, and click Next.
11. On the Select Product Languages screen, leave the default setting of English, and click Next.
12. On the Select Database Edition screen, select Enterprise Edition, and click Next.
13. On the Specify Installation Location, leave the defaults, and click Next.
14. On the Create Inventory screen, leave the default settings, and click Next.
15. On the Privileged Operating System groups screen, keep the defaults, and click Next.
16. Allow the prerequisite checker to complete.
17. On the Summary screen, click Install.
18. Once the Execute Configuration scripts prompt appears, ssh into the server as `root`, and run the following command:

```
/home/oracle/app/oracle/product/12.1.0/dbhome_1/root.sh
```

19. Return to the prompt, and click OK.
20. Once the installer completes, click Close.

Creating and configuring the database

1. Using Putty with X11 forwarding enabled, SSH to the VM.
2. Type `dbca`, and press Enter to open the Database configuration assistant.
3. At the Database Operation screen, select Create Database, and click Next.
4. Under Creation Mode, select Advanced Mode, and click Next.
5. At the Select Template screen, select General Purpose or Transaction Processing. Click Next.
6. Enter a Global database name and the appropriate SID.

7. At the Management Options screen, select Configure Enterprise Manager (EM) Database Express. Click Next.
8. At the Database Credentials screen, select Use the Same Administrative Password for All Accounts. Enter a password, and click Next.
9. At the Network Configuration screen, click Next.
10. At the Storage Configuration screen, select Automatic Storage Management, and select `+DATA` as the Database.
11. At the Database Options screen, click Next.
12. At the Initialization Parameters screen, click use Automatic Memory Management.
13. At the Creation Options screen, select Create Database, and click Next.
14. At the summary screen, click Finish.
15. Close the Database Configuration Assistant.
16. In a Web browser, browse to `https://vm.ip.address:5500/em` to open the database manager.
17. Log in as system with the password you specified.
18. Go to Storage→Tablespaces.
19. Click Create.
20. Enter `SLOB` as the Name, and check the Set As Default box. Click OK.
21. Go to Storage→Redo Log Groups.
22. Click Actions→Switch file… until you get one of the groups to go inactive.
23. Highlight the inactive group, and click Actions→Drop group.
24. In Putty, enter sqlplus and type `ALTER DATABASE ADD LOGFILE GROUP 3 ('+logs/orcl/onlinelog/redox.log') SIZE 14G`; where x is the number of the log file (1, 2, etc.).
25. Repeat steps 21 – 24 until all default logs are dropped, and two new ones exist.
26. See our Oracle spfile configuration files for each VM size below.

Installing SLOB and populating the database

1. Download the SLOB kit from www.kevinclosson.net/slob/.
2. Copy and untar the files to `/home/oracle/SLOB/`.
3. Edit the slob.conf file prior to running setup. See below for our slob.conf file.
4. Type `./setup.sh SLOB 128` to start the data population to the SLOB tablespace we created earlier.
5. When the setup is complete, the database is populated.

Running the performance test

To run our test scenarios, we first started up all VMs and allowed them to sit idle for at least 20 minutes to ensure that all Oracle processes had started. Prior to starting the SLOB workload, we started gathering esxtop data on all the compute hosts and sar data collection on the VMs. To start the SLOB workload, we ran./runit.sh 128. We configured the workload for 0ms think time and a run time of 1 hour. Once the workload completed, we gathered all the statistics output from SLOB, esxtop, and sar. Due to the design of SLOB, no restore was needed in between runs.

For phase one, we only ran the workload on the primary site with twelve VMs split evenly between to six compute nodes. For the second phase, we incorporated the secondary and ran the workload on twenty-four VMs spread

evenly across twelve compute nodes. We allowed the workload to run for ten minutes before we initiated a mass evacuation of the primary site VMs to the secondary site, measuring the time it took to complete the vMotion. We used PowerCLI and a script to start the migration asynchronously to ensure accuracy and repeatability.

ORACLE PARAMETERS

For our Oracle instances we used the following parameter spfiles with different RAM settings for our medium and large VM sizes.

Medium VM – 64GB

```
orcl.__data_transfer_cache_size=0
orcl.__db_cache_size=54895050752
orcl.__java_pool_size=134217728
orcl.__large_pool_size=268435456
orcl.__oracle_base='/u01/app/oracle'#ORACLE_BASE set from environment
orcl.__pga_aggregate_target=1073741824
orcl.__sga_target=56908316672
orcl.__shared_io_pool_size=536870912
orcl.__shared_pool_size=939524096
orcl.__streams_pool_size=0
*.audit_file_dest='/u01/app/oracle/admin/orcl/adump'
*.audit_trail='db'
*.compatible='12.1.0.2.0'
*.control_files='+DATA/ORCL/CONTROLFILE/current.268.893606719','/u01/app/oracle/f
ast_recovery_area/ORCL/controlfile/o1_mf_c2f7myfq_.ctl'
*.db_block_size=8192
*.db_create_file_dest='+DATA'
*.db_domain=''
*.db_name='orcl'
*.db_recovery_file_dest='/u01/app/oracle/fast_recovery_area'
*.db_recovery_file_dest_size=4560m
*.db_writer_processes=4
*.diagnostic_dest='/u01/app/oracle'
*.dispatchers='(PROTOCOL=TCP) (SERVICE=orclXDB)'
*.filesystemio_options='setall'
*.local_listener='LISTENER_ORCL'
*.open_cursors=300
*.pga_aggregate_limit=2147483648
*.pga_aggregate_target=1073741824
*.processes=300
*.remote_login_passwordfile='EXCLUSIVE'
*.sga_max_size=56908316672#internally adjusted
*.sga_target=56908316672
*.undo_tablespace='UNDO'
*.use_large_pages='ONLY'
```

Large VM – 96GB

```
orcl.__data_transfer_cache_size=0
orcl.__db_cache_size=71672266752
orcl.__java_pool_size=536870912
orcl.__large_pool_size=536870912
```

```
orcl.__oracle_base='/u01/app/oracle'#ORACLE_BASE set from environment
orcl.__pga_aggregate_target=1073741824
orcl.__sga_target=82678120448
orcl.__shared_io_pool_size=536870912
orcl.__shared_pool_size=9126805504
orcl.__streams_pool_size=0
*.audit_file_dest='/u01/app/oracle/admin/orcl/adump'
*.audit_trail='db'
*.compatible='12.1.0.2.0'
*.control_files='+DATA/ORCL/CONTROLFILE/current.268.893606719','/u01/app/oracle/f
ast_recovery_area/ORCL/controlfile/o1_mf_c2f7myfq_.ctl'
*.db_block_size=8192
*.db_create_file_dest='+DATA'
*.db_domain=''
*.db_name='orcl'
*.db_recovery_file_dest='/u01/app/oracle/fast_recovery_area'
*.db_recovery_file_dest_size=4560m
*.db_writer_processes=8
*.diagnostic_dest='/u01/app/oracle'
*.dispatchers='(PROTOCOL=TCP) (SERVICE=orclXDB)'
*.filesystemio_options='setall'
*.local_listener='LISTENER_ORCL'
*.open_cursors=300
*.pga_aggregate_limit=2147483648
*.pga_aggregate_target=1073741824
*.processes=300
*.remote_login_passwordfile='EXCLUSIVE'
*.sga_max_size=82678120448#internally adjusted
*.sga_target=82678120448
*.undo_tablespace='UNDO'
*.use_large_pages='ONLY'
```

CONFIGURING SLOB

The following file is the SLOB configuration file we used to create the database and run the test.

```
UPDATE_PCT=25
RUN_TIME=3600
WORK_LOOP=0
SCALE=210000
WORK_UNIT=64
REDO_STRESS=LITE
LOAD_PARALLEL_DEGREE=4

THREADS_PER_SCHEMA=1

# Settings for SQL*Net connectivity:
#ADMIN_SQLNET_SERVICE=slob
#SQLNET_SERVICE_BASE=slob
#SQLNET_SERVICE_MAX=2
#SYSDBA_PASSWD=change_on_install

#########################
#### Advanced settings:
#
# The following are Hot Spot related parameters.
# By default Hot Spot functionality is disabled (DO_HOTSPOT=FALSE).
#
DO_HOTSPOT=FALSE
HOTSPOT_MB=8
HOTSPOT_OFFSET_MB=16
HOTSPOT_FREQUENCY=3

#
# The following controls operations on Hot Schema
# Default Value: 0. Default setting disables Hot Schema
#
HOT_SCHEMA_FREQUENCY=0

# The following parameters control think time between SLOB
# operations (SQL Executions).
# Setting the frequency to 0 disables think time.
#
THINK_TM_FREQUENCY=0
THINK_TM_MIN=.1
THINK_TM_MAX=.5
#########################

export UPDATE_PCT RUN_TIME WORK_LOOP SCALE WORK_UNIT LOAD_PARALLEL_DEGREE
REDO_STRESS
export DO_HOTSPOT HOTSPOT_MB HOTSPOT_OFFSET_MB HOTSPOT_FREQUENCY
HOT_SCHEMA_FREQUENCY THINK_TM_FREQUENCY THINK_TM_MIN THINK_TM_MAX
```

CUMULUS OS SWITCH CONFIGURATION FILES

With the assistance of VMware, we configured the interfaces file for each switch to configure the switch ports and create the bonds and LAGs. This configuration follows the VMware validated architecture design guidelines.

40GB spine switch 1

```
auto lo
iface lo inet loopback

auto eth0
iface eth0
    gateway 172.0.0.2
        address 172.0.1.3
        netmask 255.255.0.0


auto bd-rk1
iface bd-rk1
    bond-slaves swp1 swp2
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 1

auto bd-rk2
iface bd-rk2
    bond-slaves swp3 swp4
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 2

auto bd-rk3
iface bd-rk3
    bond-slaves swp5 swp6
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 3
```

```
auto bd-rk4
iface bd-rk4
    bond-slaves swp7 swp8
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 4

auto bd-rk5
iface bd-rk5
    bond-slaves swp9 swp10
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 5

auto bd-rk6
iface bd-rk6
    bond-slaves swp11 swp12
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 6

auto bd-rk7
iface bd-rk7
    bond-slaves swp13 swp14
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 7

auto bd-rk8
iface bd-rk8
    bond-slaves swp15 swp16
    bond-mode 802.3ad
```

```
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 8

auto bd-rk9
iface bd-rk9
    bond-slaves swp17 swp18
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 9

auto bd-rk10
iface bd-rk10
    bond-slaves swp19 swp20
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 10

auto bd-rk11
iface bd-rk11
    bond-slaves swp21 swp22
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 11

auto bd-rk12
iface bd-rk12
    bond-slaves swp23 swp24
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
```

```
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 12

auto bd-rk13
iface bd-rk13
    bond-slaves swp25 swp26
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 13

auto bd-rk14
iface bd-rk14
    bond-slaves swp27 swp28
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 14

auto bd-rk15
iface bd-rk15
    bond-slaves swp29 swp30
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 15

auto peerlink
iface peerlink
    bond-slaves swp31 swp32
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4

auto peerlink.4093
iface peerlink.4093
```

```
    address 169.254.255.3/24
    clagd-priority 4096
    clagd-peer-ip 169.254.255.4
    clagd-sys-mac 44:38:39:ff:4f:fa

auto br-data
iface br-data
    bridge-vlan-aware yes
    bridge-stp on
    bridge-ports peerlink bd-rk1 bd-rk2 bd-rk3 bd-rk4 bd-rk5 bd-rk6 bd-rk7 bd-rk8
bd-rk9 bd-rk10 bd-rk11 bd-rk12 bd-rk13 bd-rk14 bd-rk15
    bridge-pvid 1
    bridge-vids 3-3299 970 980 1020 3000
    bridge_ageing 300
#source /etc/network/interfaces.d/hms-clag
```

40GB spine switch 2

```
auto lo
iface lo inet loopback

auto eth0
iface eth0
    gateway 172.0.0.2
        address 172.0.1.4
        netmask 255.255.0.0


auto bd-rk1
iface bd-rk1
    bond-slaves swp1 swp2
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 1

auto bd-rk2
iface bd-rk2
    bond-slaves swp3 swp4
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 2
```

```
auto bd-rk3
iface bd-rk3
    bond-slaves swp5 swp6
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 3

auto bd-rk4
iface bd-rk4
    bond-slaves swp7 swp8
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 4

auto bd-rk5
iface bd-rk5
    bond-slaves swp9 swp10
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 5

auto bd-rk6
iface bd-rk6
    bond-slaves swp11 swp12
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 6

auto bd-rk7
iface bd-rk7
    bond-slaves swp13 swp14
    bond-mode 802.3ad
```

```
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 7

auto bd-rk8
iface bd-rk8
    bond-slaves swp15 swp16
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 8

auto bd-rk9
iface bd-rk9
    bond-slaves swp17 swp18
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 9

auto bd-rk10
iface bd-rk10
    bond-slaves swp19 swp20
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 10

auto bd-rk11
iface bd-rk11
    bond-slaves swp21 swp22
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
```

```
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 11

auto bd-rk12
iface bd-rk12
    bond-slaves swp23 swp24
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 12

auto bd-rk13
iface bd-rk13
    bond-slaves swp25 swp26
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 13

auto bd-rk14
iface bd-rk14
    bond-slaves swp27 swp28
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 14

auto bd-rk15
iface bd-rk15
    bond-slaves swp29 swp30
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 15
```

```
auto peerlink
iface peerlink
    bond-slaves swp31 swp32
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4

auto peerlink.4093
iface peerlink.4093
    address 169.254.255.4/24
    clagd-priority 4096
    clagd-peer-ip 169.254.255.3
    clagd-sys-mac 44:38:39:ff:4f:fa

auto br-data
iface br-data
    bridge-vlan-aware yes
    bridge-stp on
    bridge-ports peerlink bd-rk1 bd-rk2 bd-rk3 bd-rk4 bd-rk5 bd-rk6 bd-rk7 bd-rk8
bd-rk9 bd-rk10 bd-rk11 bd-rk12 bd-rk13 bd-rk14 bd-rk15
    bridge-pvid 1
    bridge-vids 3-3299 970 980 1020 3000
    bridge_ageing 300
```

Primary Site - 10GB leaf switch 1

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
      address 172.0.1.5
      netmask 255.255.0.0
      gateway 172.0.0.2

auto swp1
iface swp1
    mtu 9194
    bridge-access 1

auto swp2
iface swp2
    mtu 9194
    bridge-access 1

auto swp3
iface swp3
    mtu 9194
    bridge-access 1
```

```
auto swp4
iface swp4
    mtu 9194
    bridge-access 1

auto swp5
iface swp5
    mtu 9194
    bridge-access 1

auto swp6
iface swp6
    mtu 9194
    bridge-access 1

auto swp7
iface swp7
    mtu 9194
    bridge-access 1

auto swp8
iface swp8
    mtu 9194
    bridge-access 1

auto swp9
iface swp9
    mtu 9194
    bridge-access 1

auto swp10
iface swp10
    mtu 9194
    bridge-access 1

auto swp11
iface swp11
    mtu 9194
    bridge-access 1

auto swp12
iface swp12
    mtu 9194
    bridge-access 1

auto swp13
iface swp13
    mtu 9194
    bridge-access 1

#bondmgmt
```

```
auto bondmgmt
iface bondmgmt
    bond-slaves swp48
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    bridge-access 1

auto bondspine
iface bondspine
    bond-slaves swp49  swp50
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 2

auto peerlink
iface peerlink
    bond-slaves swp39 swp40 swp41 swp42
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194

auto peerlink.4092
iface peerlink.4092
    mtu 9194
    address 169.254.255.5/24
    clagd-priority 8192
    clagd-peer-ip 169.254.255.6
    clagd-sys-mac 44:38:39:ff:36:3a


auto bondh1
iface bondh1
    bond-slaves swp1
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
```

```
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 3

auto bondh2
iface bondh2
    bond-slaves swp2
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 4

auto bondh3
iface bondh3
    bond-slaves swp3
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 5

auto bondh4
iface bondh4
    bond-slaves swp4
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 6

auto bondh5
iface bondh5
    bond-slaves swp5
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 7
```

```
auto bondh6
iface bondh6
    bond-slaves swp6
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 8

auto bondh7
iface bondh7
    bond-slaves swp7
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 9

auto bondh8
iface bondh8
    bond-slaves swp8
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 10

auto bondh9
iface bondh9
    bond-slaves swp9
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 11

auto bondh10
iface bondh10
    bond-slaves swp10
    bond-mode 802.3ad
```

```
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 12

auto bondh11
iface bondh11
    bond-slaves swp11
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 13

auto bondh12
iface bondh12
    bond-slaves swp12
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 14

auto bondh13
iface bondh13
    bond-slaves swp13
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 15

auto brvrack-data
iface brvrack-data
    bridge-vlan-aware yes
    bridge-stp on
    bridge-ports peerlink bondspine bondmgmt bondh1 bondh2 bondh3 bondh4 bondh5
bondh6 bondh7 bondh8 bondh9 bondh10 bondh11 bondh12 bondh13
    bridge-pvid 1
    bridge-vids 970 980 1020 3000
```

```
    mstpctl-portautoedge bondmgmt=yes
    mstpctl-portbpdufilter bondmgmt=yes
```

Primary Site - 10GB leaf switch 2

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
      address 172.0.1.6
      netmask 255.255.0.0
      gateway 172.0.0.2

auto swp1
iface swp1
    mtu 9194
    bridge-access 1

auto swp2
iface swp2
    mtu 9194
    bridge-access 1

auto swp3
iface swp3
    mtu 9194
    bridge-access 1

auto swp4
iface swp4
    mtu 9194
    bridge-access 1

auto swp5
iface swp5
    mtu 9194
    bridge-access 1

auto swp6
iface swp6
    mtu 9194
    bridge-access 1

auto swp7
iface swp7
    mtu 9194
    bridge-access 1

auto swp8
iface swp8
    mtu 9194
```

```
    bridge-access 1

auto swp9
iface swp9
    mtu 9194
    bridge-access 1

auto swp10
iface swp10
    mtu 9194
    bridge-access 1

auto swp11
iface swp11
    mtu 9194
    bridge-access 1

auto swp12
iface swp12
    mtu 9194
    bridge-access 1

auto swp13
iface swp13
    mtu 9194
    bridge-access 1

#bondmgmt
auto bondmgmt
iface bondmgmt
    bond-slaves swp48
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    bridge-access 1

auto bondspine
iface bondspine
    bond-slaves swp49  swp50
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 2
```

```
auto peerlink
iface peerlink
    bond-slaves swp39 swp40 swp41 swp42
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194

auto peerlink.4092
iface peerlink.4092
    mtu 9194
    address 169.254.255.6/24
    clagd-priority 8192
    clagd-peer-ip 169.254.255.5
    clagd-sys-mac 44:38:39:ff:36:3a


auto bondh1
iface bondh1
    bond-slaves swp1
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 3

auto bondh2
iface bondh2
    bond-slaves swp2
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 4

auto bondh3
iface bondh3
    bond-slaves swp3
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
```

```
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 5

auto bondh4
iface bondh4
    bond-slaves swp4
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 6

auto bondh5
iface bondh5
    bond-slaves swp5
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 7

auto bondh6
iface bondh6
    bond-slaves swp6
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 8

auto bondh7
iface bondh7
    bond-slaves swp7
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 9
```

```
auto bondh8
iface bondh8
    bond-slaves swp8
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 10

auto bondh9
iface bondh9
    bond-slaves swp9
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 11

auto bondh10
iface bondh10
    bond-slaves swp10
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 12

auto bondh11
iface bondh11
    bond-slaves swp11
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 13

auto bondh12
iface bondh12
    bond-slaves swp12
    bond-mode 802.3ad
```

```
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 14

auto bondh13
iface bondh13
    bond-slaves swp13
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 15

auto brvrack-data
iface brvrack-data
    bridge-vlan-aware yes
    bridge-stp on
    bridge-ports peerlink bondspine bondmgmt bondh1 bondh2 bondh3 bondh4 bondh5
bondh6 bondh7 bondh8 bondh9 bondh10 bondh11 bondh12 bondh13
    bridge-pvid 1
    bridge-vids 970 980 1020 3000
    mstpctl-portautoedge bondmgmt=yes
    mstpctl-portbpdufilter bondmgmt=yes
```

Secondary Site - 10GB leaf switch 1

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
      address 172.0.1.7
      netmask 255.255.0.0
      gateway 172.0.0.2

auto swp1
iface swp1
    mtu 9194
    bridge-access 1

auto swp2
iface swp2
    mtu 9194
    bridge-access 1
```

```
auto swp3
iface swp3
    mtu 9194
    bridge-access 1

auto swp4
iface swp4
    mtu 9194
    bridge-access 1

auto swp5
iface swp5
    mtu 9194
    bridge-access 1

auto swp6
iface swp6
    mtu 9194
    bridge-access 1

auto swp7
iface swp7
    mtu 9194
    bridge-access 1

auto swp8
iface swp8
    mtu 9194
    bridge-access 1

auto swp9
iface swp9
    mtu 9194
    bridge-access 1

auto swp10
iface swp10
    mtu 9194
    bridge-access 1

auto swp11
iface swp11
    mtu 9194
    bridge-access 1

auto swp12
iface swp12
    mtu 9194
    bridge-access 1

auto swp13
iface swp13
```

```
    mtu 9194
    bridge-access 1

#bondmgmt
auto bondmgmt
iface bondmgmt
    bond-slaves swp48
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    bridge-access 1

auto bondspine
iface bondspine
    bond-slaves swp49  swp50
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 2

auto peerlink
iface peerlink
    bond-slaves swp39 swp40 swp41 swp42
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194

auto peerlink.4094
iface peerlink.4094
    mtu 9194
    address 169.254.255.7/24
    clagd-priority 8192
    clagd-peer-ip 169.254.255.8
    clagd-sys-mac 44:38:39:ff:36:3e


auto bondh1
iface bondh1
    bond-slaves swp1
    bond-mode 802.3ad
```

```
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 3

auto bondh2
iface bondh2
    bond-slaves swp2
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 4

auto bondh3
iface bondh3
    bond-slaves swp3
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 5

auto bondh4
iface bondh4
    bond-slaves swp4
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 6

auto bondh5
iface bondh5
    bond-slaves swp5
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
```

```
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 7

auto bondh6
iface bondh6
    bond-slaves swp6
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 8

auto bondh7
iface bondh7
    bond-slaves swp7
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 9

auto bondh8
iface bondh8
    bond-slaves swp8
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 10

auto bondh9
iface bondh9
    bond-slaves swp9
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 11
```

```
auto bondh10
iface bondh10
    bond-slaves swp10
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 12

auto bondh11
iface bondh11
    bond-slaves swp11
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 13

auto bondh12
iface bondh12
    bond-slaves swp12
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 14

auto bondh13
iface bondh13
    bond-slaves swp13
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 15

auto brvrack-data
iface brvrack-data
    bridge-vlan-aware yes
    bridge-stp on
```

```
    bridge-ports peerlink bondspine bondmgmt bondh1 bondh2 bondh3 bondh4 bondh5
bondh6 bondh7 bondh8 bondh9 bondh10 bondh11 bondh12 bondh13
    bridge-pvid 1
    bridge-vids 970 980 1020 3000
    mstpctl-portautoedge bondmgmt=yes
    mstpctl-portbpdufilter bondmgmt=yes
```

Secondary Site - 10GB leaf switch 2

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
      address 172.0.1.8
      netmask 255.255.0.0
      gateway 172.0.0.2

auto swp1
iface swp1
    mtu 9194
    bridge-access 1

auto swp2
iface swp2
    mtu 9194
    bridge-access 1

auto swp3
iface swp3
    mtu 9194
    bridge-access 1

auto swp4
iface swp4
    mtu 9194
    bridge-access 1

auto swp5
iface swp5
    mtu 9194
    bridge-access 1

auto swp6
iface swp6
    mtu 9194
    bridge-access 1

auto swp7
iface swp7
    mtu 9194
    bridge-access 1
```

```
auto swp8
iface swp8
    mtu 9194
    bridge-access 1

auto swp9
iface swp9
    mtu 9194
    bridge-access 1

auto swp10
iface swp10
    mtu 9194
    bridge-access 1

auto swp11
iface swp11
    mtu 9194
    bridge-access 1

auto swp12
iface swp12
    mtu 9194
    bridge-access 1

auto swp13
iface swp13
    mtu 9194
    bridge-access 1

#bondmgmt
auto bondmgmt
iface bondmgmt
    bond-slaves swp48
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    bridge-access 1
    clag-id 1

auto bondspine
iface bondspine
    bond-slaves swp49  swp50
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
```

```
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 2

auto peerlink
iface peerlink
    bond-slaves swp39 swp40 swp41 swp42
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 1
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194

auto peerlink.4094
iface peerlink.4094
    mtu 9194
    address 169.254.255.8/24
    clagd-priority 8192
    clagd-peer-ip 169.254.255.7
    clagd-sys-mac 44:38:39:ff:36:3e


auto bondh1
iface bondh1
    bond-slaves swp1
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 3

auto bondh2
iface bondh2
    bond-slaves swp2
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 4

auto bondh3
iface bondh3
    bond-slaves swp3
```

```
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 5

auto bondh4
iface bondh4
    bond-slaves swp4
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 6

auto bondh5
iface bondh5
    bond-slaves swp5
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 7

auto bondh6
iface bondh6
    bond-slaves swp6
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 8

auto bondh7
iface bondh7
    bond-slaves swp7
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
```

```
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 9

auto bondh8
iface bondh8
    bond-slaves swp8
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 10

auto bondh9
iface bondh9
    bond-slaves swp9
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 11

auto bondh10
iface bondh10
    bond-slaves swp10
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 12

auto bondh11
iface bondh11
    bond-slaves swp11
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 13
```

```
auto bondh12
iface bondh12
    bond-slaves swp12
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 14

auto bondh13
iface bondh13
    bond-slaves swp13
    bond-mode 802.3ad
    bond-miimon 100
    bond-use-carrier 1
    bond-lacp-rate 0
    bond-min-links 1
    bond-xmit-hash-policy layer3+4
    mtu 9194
    clag-id 15

auto brvrack-data
iface brvrack-data
    bridge-vlan-aware yes
    bridge-stp on
    bridge-ports peerlink bondspine bondmgmt bondh1 bondh2 bondh3 bondh4 bondh5
bondh6 bondh7 bondh8 bondh9 bondh10 bondh11 bondh12 bondh13
    bridge-pvid 1
    bridge-vids 970 980 1020 3000
    mstpctl-portautoedge bondmgmt=yes
    mstpctl-portbpdufilter bondmgmt=yes
```

APPENDIX D – ORACLE AWR PERFORMANCE DATA

Figures 10 and 11 detail Oracle AWR report data for OS statistics, by instance, for both phases of testing.

Phase 1 OS statistics, by instance												
VM #	CPUs	Load begin	Load end	% Busy	% Usr	% Sys	% WIO	% Idl	Busy time (s)	Idle time (s)	Total time (s)	Memory (M)
1	4	0.61	88.42	15.55	11.3	3.21	84.41	84.45	2,171.47	11,794.47	13,965.94	64,433.80
2	8	0.3	126.27	8.79	6.33	2.09	91.16	91.21	2,462.58	25,542.72	28,005.30	96,689.37
3	4	0.02	91.03	20.04	14.29	4.11	79.93	79.96	2,776.10	11,079.91	13,856.01	64,433.80
4	8	0.08	115.79	10.28	7.22	2.49	89.67	89.72	2,858.26	24,947.57	27,805.83	96,689.37
5	4	1.48	101.48	17.9	12.75	3.73	82.07	82.1	2,486.39	11,402.38	13,888.77	64,433.80
6	8	0	113.86	10.5	7.34	2.53	89.44	89.5	2,932.11	24,998.90	27,931.01	96,689.37
7	4	0.11	103.3	15.42	11.19	3.27	84.55	84.58	2,149.90	11,794.57	13,944.47	64,433.80
8	8	0	114.64	9.71	7.09	2.19	90.22	90.29	2,725.08	25,329.76	28,054.84	96,689.37
9	4	0	108.94	21.58	15.49	4.43	78.39	78.42	2,984.03	10,845.63	13,829.66	64,433.80
10	8	0.11	118.2	10.13	7.24	2.39	89.8	89.87	2,833.16	25,139.07	27,972.23	96,689.37
11	4	0.15	88.2	13.44	9.84	2.85	86.53	86.56	1,882.75	12,129.98	14,012.73	64,433.80
12	8	0.1	128.44	7	5.17	1.64	92.97	93	1,973.58	26,221.79	28,195.37	96,689.37

Figure 10: OS statistics by instance for phase 1 testing.

Phase 2 OS statistics, by instance												
VM #	CPUs	Load begin	Load end	% Busy	% Usr	% Sys	% WIO	% Idl	Busy time (s)	Idle time (s)	Total time (s)	Memory (M)
1	4	0.24	107.12	14.74	10.91	2.84	85.23	85.26	2,054.12	11,881.56	13,935.68	64,433.80
2	8	0.16	128.1	9.06	6.73	1.83	90.87	90.94	2,534.88	25,444.98	27,979.86	96,689.37
3	4	0.17	109.98	16.97	12.49	3.16	82.99	83.03	2,354.44	11,516.47	13,870.91	64,433.80
4	8	0.25	125.63	9.26	6.73	1.95	90.69	90.74	2,513.87	24,646.65	27,160.52	96,689.37
5	4	1.63	109.59	19.52	14.4	3.7	80.44	80.48	2,700.99	11,133.65	13,834.64	64,433.80
6	8	0.02	127.88	9.13	6.84	1.88	90.81	90.87	2,544.94	25,320.43	27,865.37	96,689.37
7	4	0.14	104.55	17.51	12.93	3.39	82.45	82.49	2,430.83	11,451.07	13,881.90	64,433.80
8	8	0.09	117.78	9.59	7.07	1.97	90.35	90.41	2,663.71	25,116.49	27,780.20	96,689.37
9	4	0	111.53	14.36	10.66	2.74	85.61	85.64	2,000.49	11,933.57	13,934.06	64,433.80
10	8	0	119	9.33	6.83	1.95	90.57	90.67	2,607.26	25,337.74	27,945.00	96,689.37
11	4	0	104.28	17.73	13.01	3.3	82.23	82.27	2,463.56	11,428.11	13,891.67	64,433.80
12	8	0	128.76	9.14	6.72	1.91	90.68	90.86	2,536.73	25,215.63	27,752.36	96,689.37
13	4	0.67	108.47	21.85	16.19	4.08	78.13	78.15	3,021.98	10,809.52	13,831.50	64,433.80
14	8	0.13	128.7	7.61	5.56	1.71	92.26	92.39	2,139.27	25,988.24	28,127.51	96,689.37
15	4	0.04	101.37	12.68	9.43	2.55	87.3	87.32	1,778.02	12,245.93	14,023.95	64,433.80
16	8	0	119.44	6.74	5.04	1.47	93.21	93.26	1,902.68	26,330.36	28,233.04	96,689.37
17	4	0.08	105.92	12.94	9.56	2.65	86.97	87.06	1,812.82	12,195.81	14,008.63	64,433.80
18	8	0.05	118.51	5.63	4.27	1.21	94.31	94.37	1,595.18	26,723.11	28,318.29	96,689.37
19	4	1.33	102.22	25.72	19.16	4.61	74.24	74.28	3,562.01	10,286.96	13,848.97	64,433.80
20	8	0	117.63	8.31	6.06	1.83	91.64	91.69	2,331.65	25,712.08	28,043.73	96,689.37
21	4	0	107.86	14.31	10.62	2.85	85.66	85.69	1,993.81	11,938.80	13,932.61	64,433.80
22	8	0.01	111.61	15.36	11.1	3.14	84.58	84.64	4,264.52	23,497.83	27,762.35	96,689.37

Phase 2 OS statistics, by instance												
VM #	CPUs	Load begin	Load end	% Busy	% Usr	% Sys	% WIO	% Idl	Busy time (s)	Idle time (s)	Total time (s)	Memory (M)
23	4	1.33	108.12	14.54	10.82	2.91	85.32	85.46	2,034.03	11,950.98	13,985.01	64,433.80
24	8	0.06	128.96	6.3	4.78	1.32	93.65	93.7	1,778.26	26,436.44	28,214.70	96,689.37

Figure 11: OS statistics by instance for phase 2 testing.

Figures 12 and 13 present the top Oracle Automatic Database Diagnostic Monitor (ADDM) findings for both phases of testing.

Top ADDM findings for phase 1 testing			
VM #	Finding name	Avg active sessions of the task	Percent active sessions of finding
VM #1	Top SQL Statements	127.89	99.98
	"User I/O" wait Class	127.89	93.27
	Undersized SGA	127.89	26.88
	Free Buffer Waits	127.89	6.42
VM #2	Top SQL Statements	127.94	99.97
	"User I/O" wait Class	127.94	99.75
	Undersized SGA	127.94	49.89
VM #3	Top SQL Statements	127.95	99.97
	"User I/O" wait Class	127.95	93.43
	Undersized SGA	127.95	26.98
	Free Buffer Waits	127.95	6.21
VM #4	Top SQL Statements	127.94	99.97
	"User I/O" wait Class	127.94	99.71
	Undersized SGA	127.94	50.3
VM #5	Top SQL Statements	127.94	99.97
	"User I/O" wait Class	127.94	92.22
	Undersized SGA	127.94	26.6
	Free Buffer Waits	127.94	7.47
VM #6	Top SQL Statements	127.91	99.97
	"User I/O" wait Class	127.91	99.38
	Undersized SGA	127.91	50.41
VM #7	Top SQL Statements	127.92	99.98
	"User I/O" wait Class	127.92	93.73
	Undersized SGA	127.92	27
	Free Buffer Waits	127.92	6
VM #8	Top SQL Statements	127.88	99.96
	"User I/O" wait Class	127.88	99.72
	Undersized SGA	127.88	50.15
VM #9	Top SQL Statements	127.88	99.96
	"User I/O" wait Class	127.88	94.15

Top ADDM findings for phase 1 testing			
VM #	**Finding name**	**Avg active sessions of the task**	**Percent active sessions of finding**
	Undersized SGA	127.88	27.12
	Free Buffer Waits	127.88	5.48
VM #10	Top SQL Statements	127.91	99.97
	"User I/O" wait Class	127.91	99.7
	Undersized SGA	127.91	49.28
VM #11	Top SQL Statements	127.91	99.99
	"User I/O" wait Class	127.91	94.97
	Undersized SGA	127.91	27.61
	I/O Throughput	127.91	5.29
	Free Buffer Waits	127.91	4.79
VM #12	Top SQL Statements	127.94	99.98
	"User I/O" wait Class	127.94	99.79
	Undersized SGA	127.94	50.22

Figure 12: Top ADDM findings by average active session for phase 1 testing.

Top ADDM findings for phase 2 testing			
VM #	**Finding name**	**Avg active sessions of the task**	**Percent active sessions of finding**
VM #1	Top SQL Statements	128.83	99.25
	"User I/O" wait Class	128.83	95.73
	Undersized SGA	128.83	21.18
	I/O Throughput	128.83	16.51
	Free Buffer Waits	128.83	3.52
VM #2	Top SQL Statements	128.91	99.3
	"User I/O" wait Class	128.91	99.08
	Undersized SGA	128.91	27.29
	I/O Throughput	128.91	15.17
VM #3	Top SQL Statements	128.91	99.28
	"User I/O" wait Class	128.91	93.14
	Undersized SGA	128.91	21.56
	Free Buffer Waits	128.91	5.68
	I/O Throughput	128.91	3.26
VM #4	Top SQL Statements	128.81	99.27
	"User I/O" wait Class	128.81	96.9
	Undersized SGA	128.81	27.15
	I/O Throughput	128.81	12.38
VM #5	Top SQL Statements	128.73	99.35
	"User I/O" wait Class	128.73	94.05
	Undersized SGA	128.73	22.71
	Free Buffer Waits	128.73	5.19
VM #6	Top SQL Statements	128.86	99.27

Top ADDM findings for phase 2 testing			
VM #	**Finding name**	**Avg active sessions of the task**	**Percent active sessions of finding**
	"User I/O" wait Class	128.86	98.67
	Undersized SGA	128.86	30.19
VM #7	Top SQL Statements	128.68	99.33
	"User I/O" wait Class	128.68	95.22
	Undersized SGA	128.68	22.15
	Free Buffer Waits	128.68	4.1
VM #8	Top SQL Statements	128.85	99.17
	"User I/O" wait Class	128.85	97.97
	Undersized SGA	128.85	28.62
	I/O Throughput	128.85	7.48
VM #9	Top SQL Statements	128.84	99.28
	"User I/O" wait Class	128.84	96.08
	Undersized SGA	128.84	21.4
	I/O Throughput	128.84	15.59
	Free Buffer Waits	128.84	3.12
VM #10	Top SQL Statements	128.82	99.25
	"User I/O" wait Class	128.82	98.24
	Undersized SGA	128.82	29.42
	I/O Throughput	128.82	2.08
VM #11	Top SQL Statements	128.67	99.34
	"User I/O" wait Class	128.67	93.62
	Undersized SGA	128.67	21.43
	I/O Throughput	128.67	5.92
	Free Buffer Waits	128.67	5.61
VM #12	Top SQL Statements	128.63	99.22
	"User I/O" wait Class	128.63	98.09
	Undersized SGA	128.63	27.79
	I/O Throughput	128.63	10.34
VM #13	Top SQL Statements	128.83	99.38
	"User I/O" wait Class	128.83	93.07
	Undersized SGA	128.83	23.06
	Free Buffer Waits	128.83	6.43
VM #14	"User I/O" wait Class	128.73	99.56
	Top SQL Statements	128.73	99.13
	Undersized SGA	128.73	26.9
	I/O Throughput	128.73	18.62
VM #15	Top SQL Statements	128.76	99.35
	"User I/O" wait Class	128.76	95.33
	Undersized SGA	128.76	21.2
	I/O Throughput	128.76	13.28
	Free Buffer Waits	128.76	4.33
VM #16	"User I/O" wait Class	128.81	99.63

Top ADDM findings for phase 2 testing			
VM #	**Finding name**	**Avg active sessions of the task**	**Percent active sessions of finding**
	Top SQL Statements	128.81	99.39
	Undersized SGA	128.81	26.27
	I/O Throughput	128.81	22.39
VM #17	Top SQL Statements	128.7	99.35
	"User I/O" wait Class	128.7	95.23
	Undersized SGA	128.7	20.83
	I/O Throughput	128.7	17.54
	Free Buffer Waits	128.7	4.41
VM #18	"User I/O" wait Class	128.89	99.58
	Top SQL Statements	128.89	99.34
	I/O Throughput	128.89	28.07
	Undersized SGA	128.89	25.08
VM #19	Top SQL Statements	128.79	99.36
	"User I/O" wait Class	128.79	96.02
	Undersized SGA	128.79	24.55
	Free Buffer Waits	128.79	3.47
VM #20	"User I/O" wait Class	128.93	99.55
	Top SQL Statements	128.93	99.35
	Undersized SGA	128.93	29.43
	I/O Throughput	128.93	4.9
VM #21	Top SQL Statements	128.87	99.33
	"User I/O" wait Class	128.87	97.94
	Undersized SGA	128.87	21.24
	I/O Throughput	128.87	20.81
VM #22	Top SQL Statements	128.72	99.37
	"User I/O" wait Class	128.72	98.98
	Undersized SGA	128.72	34.25
VM #23	Top SQL Statements	128.62	99.37
	"User I/O" wait Class	128.62	95.98
	Undersized SGA	128.62	21.68
	I/O Throughput	128.62	11.3
	Free Buffer Waits	128.62	3.56
VM #24	"User I/O" wait Class	128.95	99.6
	Top SQL Statements	128.95	99.33
	Undersized SGA	128.95	26.63
	I/O Throughput	128.95	20.95

Figure 13: Top ADDM findings by average active session for phase 2 testing.

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.