
Improve performance and gain room to grow by easily migrating to a modern OpenShift environment on Dell PowerEdge R7615 servers with 4th Generation AMD EPYC processors and high-speed 100GbE Broadcom NICs

We deployed this modern environment, then migrated database VMs from legacy servers and saw performance improvements that support consolidation

Easy to deploy OpenShift and migrate VMs from legacy environment

3 modern servers can replace 4 legacy servers*

44% better database performance**

Reduce OpEx with fewer servers to power, cool, and maintain

*based on database performance improvement over legacy cluster **compared to legacy cluster

May 2024 (Revised)

The science behind the report:

Improve performance and gain room to grow by easily migrating to a modern OpenShift environment on Dell PowerEdge R7615 servers with 4th Generation AMD EPYC processors and high-speed 100GbE Broadcom NICs

This document describes what we tested, how we tested, and what we found. To learn how these facts translate into real-world benefits, read the report Improve performance and gain room to grow by easily migrating to a modern OpenShift environment on Dell PowerEdge R7615 servers with 4th Generation AMD EPYC processors and high-speed 100GbE Broadcom NICs.

We concluded our hands-on testing on March 8, 2024. During testing, we determined the appropriate hardware and software configurations and applied updates as they became available. The results in this report reflect configurations that we finalized on February 7, 2024 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

Our results

To learn more about how we have calculated the wins in this report, go to http://facts.pt/calculating-and-highlighting-wins. Unless we state otherwise, we have followed the rules and principles we outline in that document.

Table 1: Results of our testing.

	Transactions per minute	Percentage improvement with modern solution	Average host CPU utilization
Modern Red Hat® OpenShift® environment on Dell™ PowerEdge™ R7615 servers	9,674,180	44.07%	87.6%
Legacy VMware® vSphere® environment on Dell PowerEdge R7515 servers	6,714,712	N/A	86.6%

CPU utilization

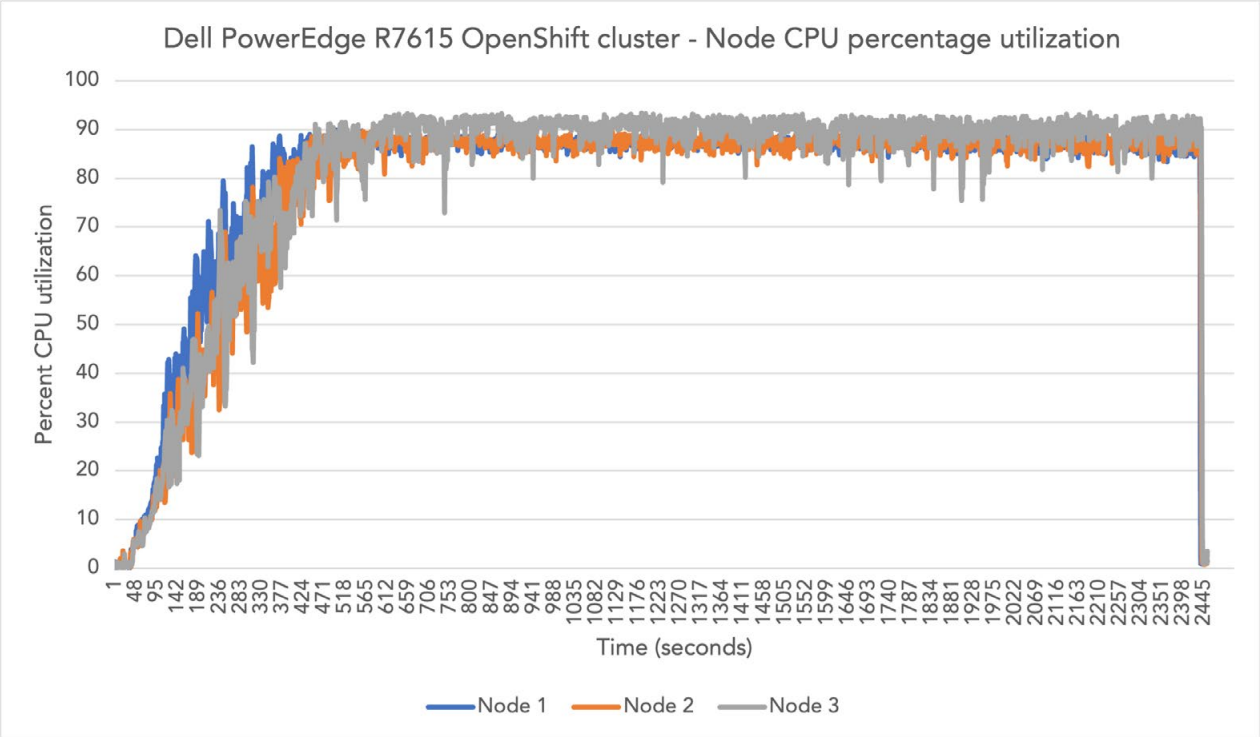

Figure 1: CPU utilization for the modern Dell PowerEdge R7615 solution during testing. Source: Principled Technologies.

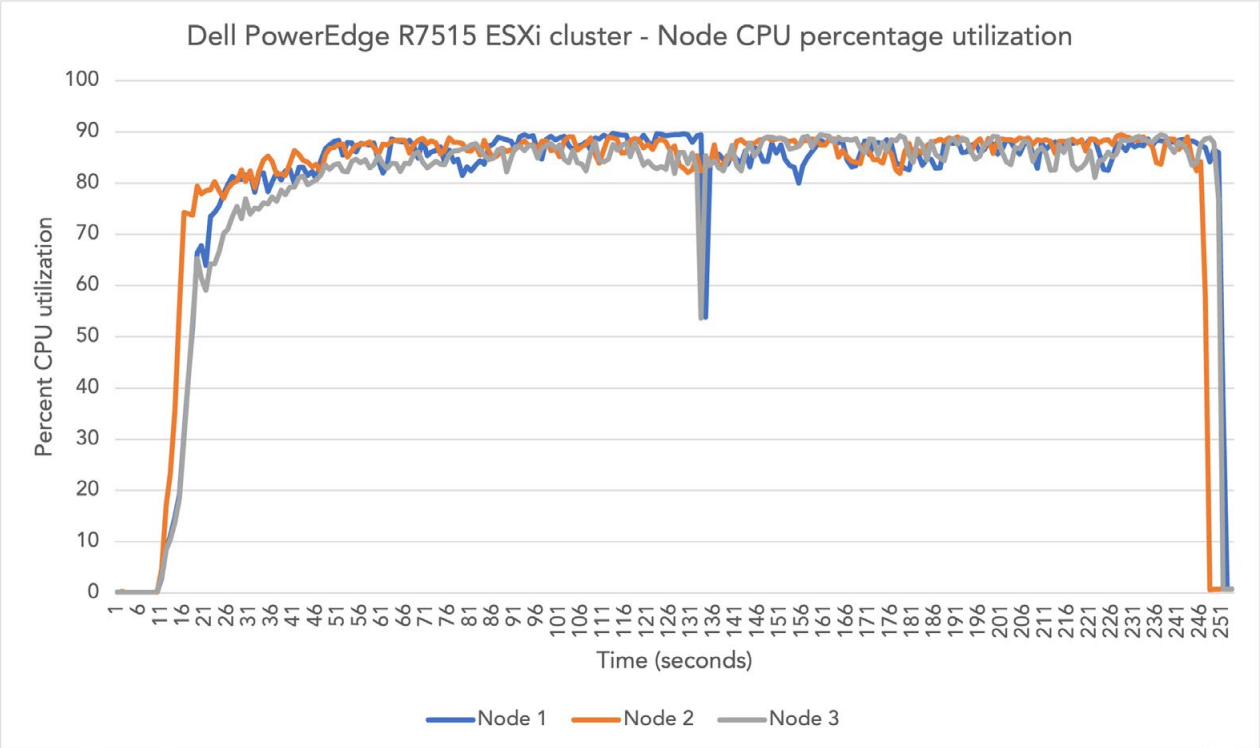

Figure 2: CPU utilization for the 16G Dell PowerEdge R7515 during testing. Source: Principled Technologies.

System configuration information

Table 2: Detailed configuration information for the servers under test.

Server configuration information	Dell PowerEdge R7615	Dell PowerEdge R7515
BIOS name and version	Dell 1.5.8	Dell 2.13.3
Non-default BIOS settings	Performance power profile	Performance power profile
Operating system name and version/ build number	Red Hat Enterprise Linux 9 5.14.0-362.8.1.el9_3.x86_64	VMware ESXi™ 8.0.2 22380479
Date of last OS updates/patches applied	2/9/2024	2/9/2024
Power management policy	Performance	Performance
Processor		
Number of processors	1	1
Vendor and model	AMD™ EPYC™ 9554	AMD EPYC 7663
Core count (per processor)	64	56
Core frequency (GHz)	3.1	2.0
Stepping	1	1
Memory module(s)		
Total memory in system (GB)	1,152	1,024
Number of memory modules	12	16
Vendor and model	Samsung®	Samsung
Size (GB)	M321RYGA0PB0-CWMXH	M393A8G40BB4-CWE
Type	DDR5	DDR4
Speed (MHz)	5,600	3,200
Speed running in the server (MHz)	5,600	2,933
Storage controller		
Vendor and model	Dell PERC H755	Dell HBA330 Mini
Cache size	8 GB	N/A
Firmware version	52.26.0-5179	16.17.01.00
Driver version	7.721.03.00	17.00.13.00
Local storage		
Number of drives	2	1
Drive vendor and model	Kioxia® KPM6XRUG960G	Micron® MTFDDAK480TDN
Drive size (GB)	900	480
Drive information (speed, interface, type)	24Gbps SAS SSD	6Gbps SAS SSD
Network adapter		
Vendor and model	Broadcom® BCM57508	Broadcom Adv. Dual Ethernet
Number and type of ports	2 x 100GbE QSFP	2 x 25GbE SFP
Driver version	22.7	226.0.121.0

Server configuration information	Dell PowerEdge R7615	Dell PowerEdge R7515
Cooling fans		
Vendor and model	Dell Gold	Dell Silver
Number of cooling fans	12	6
Power supplies		
Vendor and model	Dell 06C11WA02	Dell 0726KXA02
Number of power supplies	2	2
Wattage of each (W)	1,600	1,100

Table 3: Detailed configuration information for the storage solution.

Storage configuration information	Dell PowerStore™ 1200T
Controller firmware revision	3.5.0.1
Number of storage controllers	2
Number of storage shelves	1
Number of drives per shelf	12
Drive vendor and model number	Dell 400-BKGZ
Drive size (TB)	7.68
Drive information (speed, interface, type)	PCIe 4.0 x4 NVMe®

Table 4: Detailed configuration information for the network switches we used.

Network switch configuration information	Dell PowerSwitch™ Z9100-ON
Firmware revision	10.5.5.OP1
Number and type of ports	32 x QSFP 100GbE
Number and type of ports used in test	6 x QSFP 100GbE, 3x25GbE 1-to-4 breakout cables
Non-default settings used	Jumbo frames enabled

About our testing

Our testing compared the following three-node solutions:

- 3 x Dell PowerEdge R7615 with AMD EPYC 9554 64-core processors and 1,152 GB of DDR5 memory
- 3 x Dell PowerEdge R7515 with AMD EPYC 7663 56-core processors and 1,024 GB of DDR4 memory

Both clusters used local storage for the hypervisor and network-attached storage for the virtual machines. We used Red Hat OpenShift 4.14 on the Dell PowerEdge R7615s, and VMware vSphere 8.0 on the Dell PowerEdge R7515s. We placed client VMs and infrastructure VMs on separate servers outside of the test clusters. We created four VMs on each server, with 32 vCPUs and 256 GB of memory on the Dell PowerEdge R7615 servers and 28 vCPUs and 256 GB of memory on the R7515 servers. We used Red Hat Enterprise Linux 9 for the guest OS and installed MySQL version 8.2. We ran the HammerDB 4.9 TPROC-C workload and measured the transactions per minute (TPM) each VM achieved.

Using our methodology to aid your own deployments

While the methodology below describes in detail how we accomplished our testing, it is not a deployment guide. However, because we include many basic installation steps for operating systems and testing tools, reading our methodology may help with your own installation.

How we tested

Building the PowerEdge R7615 Red Hat OpenShift environment

Creating the base environment

This section contains the steps we took to install and configure Red Hat OpenShift Virtualization. Prior to this, we set up the prerequisite environment OpenShift requires. We created a VM for Active Directory (AD), Domain Name System (DNS), and Dynamic Host Configuration Protocol (DHCP). We created DNS A records for the Kubernetes API, internal API, and a DNS A Wildcard for the ingress route. We also created a Pfsense gateway VM to route our private network traffic out to the public network.

1. Click the link in the single node installation guide to get to the assisted installer.
2. Click Create Cluster.
3. Give the cluster a cluster name.
4. In the Base Domain field, enter the name for the domain you created in the DNS server.
5. From the drop-down menu, select an OpenShift version, and click Next.
6. Check the box beside Install OpenShift Virtualization, and click Next.
7. Click Add Host.
8. From the Provisioning type drop-down menu, select Full image file - Download a self-contained ISO.
9. Enter the SSH public key of the server running the assisted installer, and click Generate Discovery ISO.
10. Click Download Discovery ISO.
11. Connect the downloaded discovery ISO to the iDRAC virtual media, and power on the server.
12. Boot the server to the ISO.
13. When the server has booted into RHEL Core OS, return to the assisted installer GUI and wait for the installer to discover the node.
14. Once the server status changes to Ready, click Next.
15. On the Storage tab, click Next.
16. On the Networking tab, click Next.
17. On the Review and Create tab, click Install cluster.
18. Once the installation is complete, click Launch OpenShiftConsole, and follow the DNS entry instructions to get to the Web Console URL.
19. Log into the OpenShift Console with the provided username and password.

Configuring ISCSI storage for the VMs

1. Log into the first worker node using the OC CLI:

```
oc debug node/worker1
chroot /host
```

2. Run the following command to discover the PowerStore ISCSI target:

```
iscsiadm -m discovery -t sendtargets -p [PowerStore IP Address]
```

3. Run the following command to connect to the PowerStore:

```
iscsiadm -m node -T [PowerStore Initiator] -p [PowerStore IP Address] -l
```

4. Repeat step 1 for the remaining two worker nodes.
5. Log into the PowerStore UI.
6. Click Storage → Volumes.
7. Click Create.
8. Enter a prefix for the volume names.
9. Set the following values:
   - Category: Relational Databases
   - Application: MySQL
   - Quantity: 8
   - Volume size: 100GB
   - Volume Performance Policy: High
10. Click Next.
11. Check the boxes beside the three worker nodes, and click Next.
12. Review the Summary, and click Create.
13. Repeat steps 5 through 11 two more times for the MySQL data and backup volumes.

Creating the Persistent Volumes and Persistent Volume Claims for the ISCSI volumes

We created and applied yaml files for each PV and PVC with the oc cli to create persistent volumes and persistent volume claims in OpenShift.

1. Apply the PV and PVC yaml files in the Sections we used for testing:

```
oc apply -f mysql01-os-pv.yaml
oc apply -f mysql01-os-pvc.yaml
oc apply -f mysql01-data-pv.yaml
oc apply -f mysql01-data-pvc.yaml
oc apply -f mysql01-backup-pv.yaml
oc apply -f mysql01-backup-pvc.yaml
```

2. Repeat step 1 for the remaining OS, data, and backup volumes.

Creating the first VM

For each environment, we used a combination of the OC UI and CLI to create and configure the MySQL VMs for the TPROC-C test.

We performed the following steps for each VM environment:

1. Open the Red Hat OpenShift web console.
2. Open a list of virtual machines from the Virtualization drop-down menu by selecting Virtual Machines.
3. Click Create, and select From Template.
4. Select the Red Hat Enterprise Linux 9 VM.
5. Click Customize machine.
6. Change the VM name to mysql1.
7. Click Storage, and set Disk source to Blank, and Disk Size to 30 GiB.
8. Click Customize Virtual Machine parameters.
9. Click the pencil beside CPU | Memory, and set the CPUs to 32 and the Memory to 256GB. Click Save.
10. Click the Disks tab.
11. Click Add disk.
12. Check the box beside Use this disk as a boot source.
13. Give the disk a name.
14. Under the source drop-down menu, select Use an existing PVC.
15. Under the PVC name drop-down menu, select the PVC you uploaded RHEL 9 to, and click Save.
16. Click the three dots beside rootdisk, and select Detach → Detach.

17. Uncheck the box beside Start this Virtual Machine after creation, and click Create Virtual Machine.
18. Once the VM is created, click the Configuration tab.
19. Click Add disk.
20. Name the OS disk.
21. From the Source drop-down menu, select Use an existing PVC.
22. From the PVC name drop-down menu, select the PVC you created for the first VM's OS, and click Save.

Installing and configuring the VM's OS and MySQL

1. From the VirtualMachine dashboard, select mysql1.
2. Click Actions → Start Virtual Machine.
3. Select the Console tab.
4. Boot to the Red Hat Enterprise Linux 9 installation media.
5. Select Install or upgrade an existing system.
6. Choose English, and click Continue.
7. Under Installation Destination, select the desired disk to install the OS.
8. Under Other Storage Options, select I will configure partitioning, and click Done.
9. Select Click here to create them automatically.
10. Remove the /home partition if it exists.
11. Assign all remaining free space to the / partition.
12. Click Done.
13. Click Accept Changes.
14. Select Kdump.
15. Uncheck Enable kdump, and click Done.
16. Click OK.
17. Click Done.
18. Click Software Selection
19. Choose the Base Environment of Minimal Install.
20. Click Done.
21. Click Begin Installation.
22. Select Root Password.
23. Enter a root password, and click Done.
24. When the installation completes, select Reboot to restart the VM.
25. Once the VM reboots into the OS, power it off to add the data and backup disks.
26. Click the Details tab.
27. Click the pencil beside Boot order, and drag the OS disk to the top.
28. Repeat steps 19 through 22 in the previous section to add the data and backup disks.
29. Power on the VM.
30. Register the VM with RedHat:

```
subscription-manager register --auto-attach --username [USERNAME] --password [PASSWORD]
```

31. Configure SSH for passwordless logins:

```
ssh-keygen -t rsa -q
cp id_rsa.pub authorized_keys
printf "%s\n%s\n" "Host *" "StrictHostKeyChecking no" > ~/.ssh/config
```

32. Disable the firewall:

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

33. Disable SELinux:

```
setenforce 0
sed -i 's/^SELINUX=.*/SELINUX=disabled/' /etc/selinux/config
```

34. Update the OS:

```
dnf upgrade -y
```

35. Reboot the VM:

```
reboot
```

36. Create directories for MySQL and backups:

```
mkdir /mysql
mkdir /backup
```

37. Create XFS filesystem on data and backup volumes:

```
mkfs.xfs /dev/vdc
mkfs.xfs /dev/vdd
```

38. Mount the data and backup volumes:

```
mount /dev/vdc /mysql
mount /dev/vdd /backup
```

39. Download and unpack MySQL:

```
wget https://downloads.mysql.com/archives/get/p/23/file/mysql-8.2.0-1.el9.x86_64.rpm-bundle.tar
tar -xf mysql-8.2.0-1.el9.x86_64.rpm-bundle.tar
```

40. Install MySQL:

```
yum install mysql-community-{server,client,client-plugins,icu-data-files,common,libs,debuginfo,dev
el}-*
```

41. Create MySQL data and temp volumes:

```
mkdir /mysql/data
mkdir /mysql/tmp
```

42. Add the following lines to /etc/sysctl.conf:

```
vm.swappiness = 1
fs.aio-max-nr = 1048576
vm.nr_hugepages = 98304
```

43. Reboot the VM.
44. Edit /etc/my.cnf to match the my.cnf section in Scripts we used for testing.
45. Change the ownership of the MySQL directory to the mysql user:

```
chown -R mysql:mysql /mysql
```

46. Start MySQL:

```
systemctl start mysqld
```

47. Get the temporary password to log into MySQL for the first time:

```
cat /var/log/mysqld.log | grep temporary
```

48. Log into MySQL using the temporary password:

```
mysql -uroot -p
```

49. Create mysql users, set mysql and root passwords, and grant privileges to all databases:

```
CREATE USER 'root'@'%' IDENTIFIED BY 'PASSWORD';
CREATE USER 'mysql'@'localhost' IDENTIFIED BY 'PASSWORD';
CREATE USER 'mysql'@'%' IDENTIFIED BY 'PASSWORD';
ALTER USER 'root'@'localhost' IDENTIFIED BY 'PASSWORD';
GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost';
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%';
GRANT ALL PRIVILEGES ON *.* TO 'mysql'@'localhost';
GRANT ALL PRIVILEGES ON *.* TO 'mysql'@'%';
q\
```

50. Take a backup of the database:

```
systemctl stop mysqld
cd /mysql/
tar -cf- data/ | pigz -9 -c > /backup/mysql_tprocc_750wh.tar.gz
systectl start mysqld
```

51. Install tmux to automate testing and restores:

```
dnf -y install tmux
```

52. Install nmon:

```
dnf -y install nmon
```

Creating OpenShift services and routes for MySQL, SSH, and SCP

We created a service and route for our HammerDB clients on an external network to connect to MySQL. We also created services and routes for SSH and SCP to automate the test and data collection. We included the yaml files used for creation in the Scripts we used for testing section.

Creating the MySQL service and route

1. Apply the MySQL service yaml file:

```
oc apply -f hammerdb-mysql01-service.yaml
```

2. Create a route for the service:

```
oc expose svc hammerdb-mysql01
```

3. Repeat steps 1 and 2 for the remaining VMs.

Creating the SSH service and route

1. Apply the SSH service yaml file:

```
oc apply -f mysql01-ssh-service.yaml
```

2. Create a route for the service:

```
oc expose svc ssh-mysql01
```

3. Repeat steps 1 & 2 for the remaining VMs.

Creating the SCP service and route

1. Apply the SCP service yaml file:

```
oc apply -f mysql01-scp-service.yaml
```

2. Create a route for the service:

```
oc expose svc scp-mysql01
```

3. Repeat steps 1 and 2 for the remaining VMs.

Building the Dell PowerEdge R7515 VMware vSphere environment

Installing VMware vSphere 8 on the PowerEdge R7515 servers

1. Boot to the VMware vSphere 8 installation media.
2. Press Enter to continue.
3. Press F11 to accept the license agreement.
4. Select the local storage as the installation location.
5. Select a language, and create the root password.
6. Press F11 to install.

Creating the base VM on the Dell PowerEdge R7515 servers

1. Use a web browser to connect and log into the vSphere instance.
2. Right-click the host, and click New VM.
3. Assign the VM the following properties:
   - 28 virtual CPU
   - 256 GB of memory
   - 32GB VMDK on SCSI controller 0 (for operating system)
   - 100GB VMDK on SCSI controller 1 (for MySQL data)
   - VMware Paravirtual controller
   - Thick-provisioned eager-zeroed
   - 100GB VMDK on SCSI controller 2 (for data backup)
   - VMware Paravirtual controller
   - Thick-provisioned eager-zeroed
4. Click Finish.

Installing Red Hat Enterprise Linux 9

1. Boot the VM to the Red Hat Enterprise Linux 9 installation media.
2. Select Install or upgrade an existing system.
3. Choose English and click Continue.

4. Under Installation Destination, select the OS VMDK to install the OS.
5. Click Done.
6. Click Accept Changes.
7. Select Kdump.
8. Uncheck Enable kdump, and click Done.
9. Select Network & Hostname.
10. Enter the desired hostname for the system.
11. Turn on the desired network ports, and click Configure.
12. On the General tab, select Automatically connect to this network when it is available.
13. On the IPv4 Settings tab, choose the Method drop-down menu, and select Manual.
14. Under Addresses, click Add, and enter the desired static IP information for the server.
15. Enter the desired DNS information.
16. Click save, and click Done.
17. Select Date & Time, and ensure the correct date, time, and time zone are set.
18. Click the cog beside the Network Time On/Off switch to add your NTP server.
19. Add the IP address of your NTP server and click +.
20. Uncheck all other NTP servers.
21. Click OK.
22. Click Done.
23. Click Software Selection.
24. Choose the Base Environment of Minimal Install.
25. Click Done.
26. Click Begin Installation.
27. Select Root Password.
28. Enter the desired root password, and click Done.
29. When the installation completes, select Reboot to restart the server.

Configuring Red Hat Enterprise Linux 9

1. Log onto the server as root.
2. Disable the firewall:

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

3. Disable SELinux:

```
vi /etc/selinux/config
SELINUX=disabled
```

4. Register your RHEL subscription:

```
subscription-manager register --username <RHEL_USERNAME> --password <RHEL_PASSWORD> --auto-attach
```

5. Install wget, sysstat, and gdisk:

```
yum install -y sysstat wget gdisk
```

6. Update RHEL 9:

```
yum update -y
```

7. Disable auditd:

```
systemctl disable auditd
```

8. Install MySQL using the steps from the PowerEdge R7615 section.
9. Repeat the steps from creating the base VM to create the client VM on a separate server outside the test cluster. The client VM does not need the additional 100GB data and backup VMDKs.

Running the tests

Installing HammerDB

1. Log on to the client VM as root.
2. Download the MySQL 8.0 bundle for x86 on RHEL from https://downloads.mysql.com/archives/get/p/23/file/mysql-8.0.35-1.el9.x86_64.rpm-bundle.tar
3. Extract the MySQL bundle:

```
tar -xf mysql-8.0.35-1.el9.x86_64.rpm-bundle.tar
```

4. Install MySQL Community Server 8.0 and all dependencies.

```
sudo yum --disablerepo=* localinstall *.rpm
```

5. Download the tarball of the latest version of HammerDB from https://github.com/TPC-Council/HammerDB/releases/download/v4.9/HammerDB-4.9-Linux.tar.gz
6. Install HammerDB:

```
tar -xfv HammerDB-4.9-Linux.tar.gz
```

Building the TPC-C schema

1. Log onto the HammerDB client.
2. Change directories into the HammerDB directory:

```
cd HammerDB-4.9
```

3. Create the file `build.tcl` from the Scripts section.
4. Run the build script:

```
./hammerdbcli auto build.tcl
```

Running the TPROC-C test

We created scripts to automate the testing and restore process to help ensure consistency. In the Scripts we used for testing section, we include the my.cnf, build.tcl, run.sh script, restore.sh script, and run.tcl scripts. We configured the run.tcl script and copied them to each HammerDB client VM for our TPROC-C run parameters. Next, we created and copied the restore.sh script to each MySQL VM. Finally, we created a run.sh script that calls on run.tcl and restore.sh to run the TPROC-C test and then restore the database to a clean state once the run completes.

1. Run the test from the HammerDB controller VM:

```
./run.sh
```

Migrating from VMware vSphere to Red Hat OpenShift

We performed the following steps to migrate a VM from the VMware vSphere environment to the Red Hat OpenShift environment.

Installing the MTV Operator

1. Log into the Red Hat OpenShift console.
2. Navigate to Operators → OperatorHub.
3. Search for MTV.
4. Click the Migration Toolkit for Virtualization Operator.
5. Click Install.
6. Leave the defaults selected, and click Install.

Adding the VMware environment as a provider

1. Navigate to Migration → Providers for virtualization.
2. Click Create Provider.
3. Select vSphere, and click Create provider.
4. Provide the credentials and certificate thumbprint, and click Create provider.

Creating the network map

1. Navigate to Migration → NetworkMaps for virtualization.
2. Click Create NetworkMap.
3. Select the VMware provider for the source provider and the OpenShift provider as the target provider.
4. Click Add.
5. Select the private network as the source network and the Pod network for the target network.
6. Click Create.

Creating the storage map

1. Navigate to Migration → StorageMaps for virtualization.
2. Click Create StorageMap.
3. Select the VMware provider for the source provider and the OpenShift provider as the target provider.
4. Click Add.
5. Select the VM datastore for the source datastore and local-disks for the target storage class.
6. Click Create.

Creating and running the migration plan

1. Navigate to Migration → Plans for virtualization.
2. Click Create Plan.
3. Select the VMware provider for the source provider and the OpenShift provider as the target provider.
4. Select the MySQL namespace, and click Next.
5. Select the desired source hosts, and click Next.
6. Select the desired VMs, and click Next.
7. Select the previously created network map, and click Next.
8. Select the previously created storage map, and click Next.
9. Select cold migration, and click Next.
10. Click Next.
11. Click Finish.
12. Click Start.

Scripts we used for testing

In this section, we print the scripts used in the methodology steps above.

my.cnf

```
[mysqld]
datadir=/mnt/mysqldata/mysql
default_authentication_plugin=mysql_native_password
socket=/mnt/mysqldata/mysql/mysqld.sock
log-error=/var/log/mysql/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
port=3306
bind_address=0.0.0.0
# general
max_connections=4000
table_open_cache=8000
table_open_cache_instances=16
back_log=1500
default_password_lifetime=0
ssl=0
performance_schema=OFF
max_prepared_stmt_count=128000
skip_log_bin=1
character_set_server=latin1
collation_server=latin1_swedish_ci
transaction_isolation=REPEATABLE-READ
# files
innodb_file_per_table
innodb_log_file_size=1024M
innodb_log_files_in_group=16
innodb_open_files=4000
# buffers
innodb_buffer_pool_size=168720M
innodb_buffer_pool_instances=16
innodb_log_buffer_size=64M
# tune
innodb_doublewrite=0
innodb_thread_concurrency=0
innodb_flush_log_at_trx_commit=0
innodb_max_dirty_pages_pct=90
innodb_max_dirty_pages_pct_lwm=10
join_buffer_size=32K
sort_buffer_size=32K
innodb_use_native_aio=1
innodb_stats_persistent=1
innodb_spin_wait_delay=6
innodb_max_purge_lag_delay=300000
innodb_max_purge_lag=0
innodb_flush_method=O_DIRECT_NO_FSYNC
innodb_checksum_algorithm=none
innodb_io_capacity=2000
innodb_io_capacity_max=4000
innodb_lru_scan_depth=9000
innodb_change_buffering=none
innodb_read_only=0
innodb_page_cleaners=4
innodb_undo_log_truncate=off
# perf special
innodb_adaptive_flushing=1
innodb_flush_neighbors=0
innodb_read_io_threads=16
innodb_write_io_threads=16
innodb_purge_threads=4
innodb_adaptive_hash_index=0
# monitoring
innodb_monitor_enable='%'
[client]
socket=/mnt/mysqldata/mysql/mysqld.sock
```

build.tcl

```
dbset db mysql
diset connection mysql_host <IP_ADDRESS>
diset tpcc mysql_user mysql
diset tpcc mysql_pass <PASSWORD>
diset tpcc mysql_count_ware 750
diset tpcc mysql_partition true
diset tpcc mysql_num_vu 16
diset tpcc mysql_storage_engine innodb
buildschema
```

run.tcl

```
dbset db mysql
diset connection mysql_host <IP_ADDRESS>
diset tpcc mysql_user mysql
diset tpcc mysql_pass <PASSWORD>
diset tpcc mysql_count_ware 750
diset tpcc mysql_partition true
diset tpcc mysql_allwarehouse false
diset tpcc mysql_num_vu 48
diset tpcc mysql_storage_engine innodb
diset tpcc rampup 10
diset tpcc duration 30
vucreate
vurun
```

restore.sh

We used this file to automate the restore process after a run on each of the MySQL VMs.

```
#!/bin/bash
systemctl stop mysqld
rm -rf /mysql/data
pigz -d -c /backup/mysql_tprocc_750wh.tar.gz | tar -C /mysql/ -xf-
chown -R mysql:mysql /mysql/
systemctl start mysqld
```

run.sh

We used this script which calls run.tcl and restore.sh to automate the TPROC-C run and restore process for each test.

```
#!/bin/bash
TIMESTAMP=$(date '+%Y%m%d_%H%M%S')
HDB_SCRIPT=run.tcl
RESULTS_DIR=/root/results
HDB_DIR=/root/HammerDB-4.9
SCRIPT_DIR=/root/scripts
HDB_RUN=${HDB_DIR}/${HDB_SCRIPT}
HDB_LOG=/tmp/hammerdb.log
#HammerDB run parameters
WAREHOUSE_COUNT=750
RAMPUP=10 #  minutes
DURATION=30   #  minutes
VIRTUAL_USERS=48
TOTAL_ITERATIONS=1000000000  #Run length
WARMUP=$((RAMPUP*60))
RUNTIME=$((DURATION*60))
RUN_LENGTH=$(((WARMUP+RUNTIME)+60))
#nmon parameters
STEP=2  # seconds
SAMPLES_TOTAL=$(((WARMUP+RUNTIME)/STEP+5))
for i in {01..12};
```

```
do
      ssh ocp-client${i} "sed -i 's/_host.*/_host hammerdb-mysql${i}-mysql.apps.pt.grandy.
com/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/_port.*/_port 300${i}/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/_total_iterations.*/_total_iterations ${TOTAL_
ITERATIONS}/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/_count_ware.*/_count_ware ${WAREHOUSE_COUNT}/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/_allwarehouse.*/_allwarehouse false/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/_rampup.*/_rampup ${RAMPUP}/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/_duration.*/_duration ${DURATION}/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/_num_vu.*/_num_vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
      ssh ocp-client${i} "sed -i 's/vuset vu .*/vuset vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
done

#Make results folder for run and copy hammerdb config files
mkdir -p ${RESULTS_DIR}/${TIMESTAMP}/client_configs
RUN_DIR=${RESULTS_DIR}/${TIMESTAMP}
CLIENT_CFG_DIR=${RESULTS_DIR}/${TIMESTAMP}/client_configs
mkdir -p ${CLIENT_CFG_DIR}/hammerdb
cp ${HDB_RUN} ${CLIENT_CFG_DIR}/hammerdb/ocp-client${i}.run.tcl

for i in {01..12};
do
      scp ocp-client${i}:${HDB_RUN} ${CLIENT_CFG_DIR}/hammerdb/ocp-client${i}.run.tcl
done

#Start performance monitoring on mysql vms
for i in {01..12};
do
      ssh -p 310${i} ssh-mysql${i}-mysql.apps.pt.grandy.com "killall -q -w nmon; sync; rm -f /tmp/
mysql${i}.nmon"
done

for i in {01..12};
do
      ssh -p 310${i} ssh-mysql${i}-mysql.apps.pt.grandy.com "nmon -F /tmp/mysql${i}.nmon -s${STEP}
-c$((SAMPLES_TOTAL)) -J -t"
done

#Run Test
echo -e "\nRunning test for $((RAMPUP+DURATION)) minutes!"

for i in {01..12};
do
      ssh ocp-client${i} "rm -f ${HDB_LOG}"
done

sleep 5

for i in {01..12};
do
      ssh ocp-client${i} ${SCRIPT_DIR}/run_hammerdb.sh &
done

sleep ${RUN_LENGTH}

#Stop performance monitoring and copy files
mkdir -p ${RUN_DIR}/nmon
NMON_DIR=${RUN_DIR}/nmon


for i in {01..12};
do
      ssh -p 310${i} ssh-mysql${i}-mysql.apps.pt.grandy.com "killall -q -w nmon; sync"
done

for i in {01..12};
do
      scp -P 320${i} scp-mysql${i}-mysql.apps.pt.grandy.com:/tmp/mysql${i}.nmon ${NMON_DIR}
done

#Copy HammerDB results
```

```
for i in {01..12};
do
      scp ocp-client${i}:${HDB_LOG} ${RUN_DIR}/ocp-client${i}-hammerdb.log
done

sleep 60

#Restore database after the run finishes
./restore.sh

#Parse results and output run summary
for i in {01..12};
do
      cat ${RUN_DIR}/ocp-client${i}-hammerdb.log | grep TPM | awk '{ print $10 }' >> ${RUN_DIR}/TPM.txt
      cat ${RUN_DIR}/ocp-client${i}-hammerdb.log | grep NOPM | awk '{ print $7 }' >> ${RUN_DIR}/NOPM.txt
done

touch ${RUN_DIR}/run_summary.txt
SUMMARY=${RUN_DIR}/run_summary.txt

echo "HammerDB run $(date)" >> ${SUMMARY}
echo "" >> ${SUMMARY}
echo "" >> ${SUMMARY}

echo "Number of virtual users: ${VIRTUAL_USERS}" >> ${SUMMARY}
echo "Rampup (min): ${RAMPUP}" >> ${SUMMARY}
echo "Duration (min): ${DURATION}" >> ${SUMMARY}
echo "" >> ${SUMMARY}

TPM_TOTAL=0
NOPM_TOTAL=0

for i in $(cat ${RUN_DIR}/TPM.txt);
do
      TPM_TOTAL=$((${TPM_TOTAL}+${i}))
done

for i in $(cat ${RUN_DIR}/NOPM.txt);
do
      NOPM_TOTAL=$((${NOPM_TOTAL}+${i}))
done

for i in {01..12};
do
      echo "ocp-client${i} TPM: $(cat ${RUN_DIR}/ocp-client${i}-hammerdb.log | grep TPM | awk '{ print
$10 }')" >> ${SUMMARY}
done

echo "" >> ${SUMMARY}
echo "TPM total: ${TPM_TOTAL}" >> ${SUMMARY}
echo "" >> ${SUMMARY}

for i in {01..12};
do
      echo "ocp-client${i} NOPM: $(cat ${RUN_DIR}/ocp-client${i}-hammerdb.log | grep NOPM | awk '{ print
$7 }')" >> ${SUMMARY}
done

echo "" >> ${SUMMARY}
echo "NOPM total: ${NOPM_TOTAL}" >> ${SUMMARY}

for i in {01..12};
do
      ~/nmonchart ${NMON_DIR}/mysql${i}.nmon ${NMON_DIR}/mysql${i}.html
done
```

Read the report at https://facts.pt/2V6p3FG

This project was commissioned by Dell Technologies.

Principled Technologies®

Facts matter.®