VMware vSphere memory overcommitment delivered greater VM density than Red Hat Virtualization

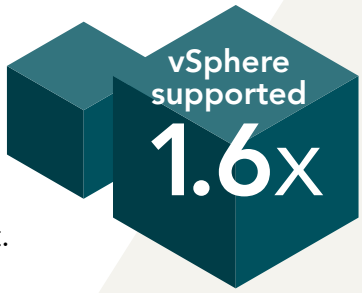
VMware vSphere 6.5 supported more active VMs with out-of-the-box settings than Red Hat Virtualization 4.1

IT admins can increase the number of VMs their servers can support by using memory overcommitment: assigning more virtual memory to VMs than the total physical memory available in the server. IT admins can also rely on temporary memory overcommitment to keep critical applications running when user demand spikes or hardware fails unexpectedly.

Principled Technologies (PT) investigated the virtual machine densities of VMware® vSphere® and Red Hat Virtualization (RHV) 4.1 running Microsoft® SQL Server® 2016 VMs on a Lenovo™ System x3650 M5 rack server. We determined how many VMs each hypervisor could power on and have run an online transaction processing (OLTP) workload. We tested a range of VM counts, memory-overcommitment levels, and hypervisor settings.

Using out-of-the-box settings, the vSphere approach to memory overcommitment allowed it to power on and run 1.60 times as many VMs as it—or RHV—ran without memory overcommitment. vSphere also kept VMs available during a simulated hardware failure without administrator intervention. RHV required an admin to enable Memory Optimization and perform manual tuning to overcommit memory. Even then, RHV could power on and run only 1.07 times its baseline number of VMs and could not deliver high availability.

vSphere
kept VMs available
during a simulated hardware failure



as many VMs
as RHV using out-of-box settings

vSphere handled
MEMORY OVERCOMMITMENT
with no manual tuning

When more is more: Higher VM density through memory overcommitment

When IT admins set up a virtual environment, they carefully evaluate server resources. As they strive to maximize the number of VMs that can run well, they must balance performance with VM density. They want to use their server's resources as efficiently as possible without overburdening resources such as memory.

Memory overcommitment lets admins increase the number of VMs their servers can support by assigning more virtual memory to VMs than the total physical memory available in the server. Temporary memory overcommitment is a way to keep critical applications running in the case of a hardware failure or a spike in demand.

Hypervisors use different approaches to memory commitment. We performed a series of tests to investigate how these differences affected the VM density a server could achieve using VMware vSphere 6.5 and RHV 4.1. We had the two hypervisors run SQL Server 2016 virtual machines on a Lenovo System x3650 M5 rack server under a variety of conditions. The VMs executed an online transaction processing (OLTP) workload using the DVD Store 2 benchmark (DS2). We configured DS2 so that the hypervisors for the two platforms would operate at similar CPU levels. We did three phases of testing:

Phase 1: Determine a standardized baseline workload without memory overcommitment

We used 15 SQL Server VMs and deliberately chose workloads that would deliver the same level of OLTP performance (measured in orders per minute, or OPM) on the two platforms without overcommitting memory or stressing other server resources. We wanted to determine a baseline against which we could compare results from overcommitment scenarios in the second and third phases of testing.

Phase 2: Increase VM density using memory overcommitment

To learn how the two hypervisors would behave when the total virtual memory for the VMs exceeded the physical memory available on the server, we increased the number of VMs beyond the 15 we used in Phase 1. We tested at 16, 20, and 24 VMs per physical host. We first used the hypervisors with their out-of-the-box settings. Then, when necessary, we adjusted settings and performed manual tuning.

Phase 3: Simulate a host power failure in a highly available (HA), multi-node cluster

Our goal was to learn how well the two hypervisors could migrate and reboot VMs after a sudden loss of power to one node of a highly available, multi-node cluster.

The value of VM density

Being able to support more VMs per server has the potential to help your datacenter in many ways:

- Maximizing the number of workloads existing hardware can support
- Saving money by using fewer resources in terms of IT management, datacenter space, power and cooling, and licensing

Phase 1: Determine a standardized baseline workload without memory overcommitment

In this initial phase of testing, our goal was to standardize CPU and I/O across the hypervisors as a baseline. To do so, we tested a non-memory-overcommitted environment with 15 VMs. (After we assigned 16 GB of vRAM to each VM, less than 16 GB of physical RAM remained free on the server host.) Rather than trying to maximize VM density on each platform, as we do in the next phase of testing, we selected DS2 workload levels that would not stress any server resources and deliver roughly the same number of orders per minute (OPM).

As Figure 1 shows, when executing this constrained OLTP workload, the two hypervisors delivered very similar numbers of OPM; they varied by less than one-half of a percent, with RHV being slightly lower.

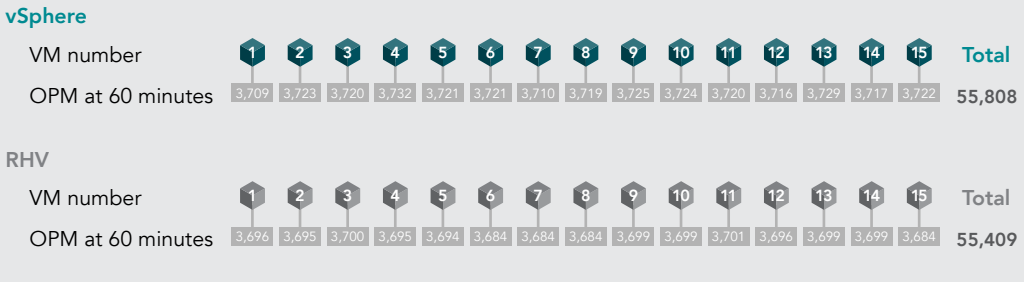


Figure 1: Standardized orders per minute across both hypervisors. We wanted to determine a consistent baseline against which we could compare results from subsequent phases of testing.

Phase 2: Increase VM density using memory overcommitment

In this phase of testing, we powered on additional VMs to increase the SQL Server VM count. As we did so, the total virtual memory assigned to the VMs increased to a level that exceeded the physical memory available on the server.

We started all of our testing using the out-of-the-box settings for the two hypervisors.

Figure 2 shows the default memory settings for vSphere. Memory overcommitment was enabled by default and no further configuration was necessary to take advantage of the memory overcommitment features.

Figure 3 shows the default memory settings for RHV. Note that the memory balloon driver was enabled on the virtual machine, but disabled on the default cluster. Memory ballooning is a process facilitated by a driver installed on the VM operating system. Through the driver, the VM communicated its memory needs and availability to the hypervisor. This let the hypervisor allocate memory across the VMs based on their current demand.

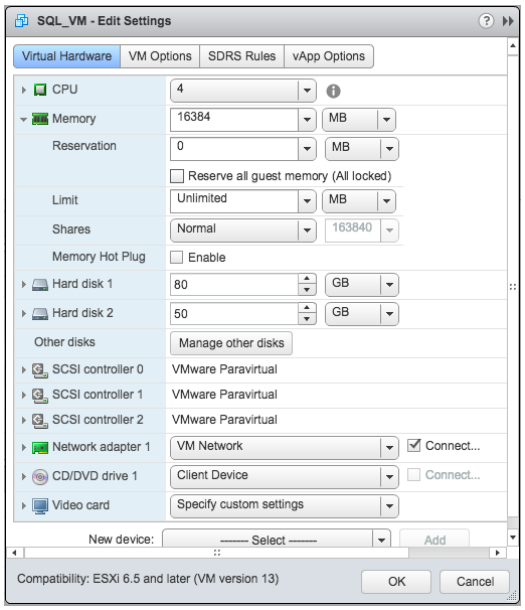


Figure 2: Default memory settings for VMware vSphere.

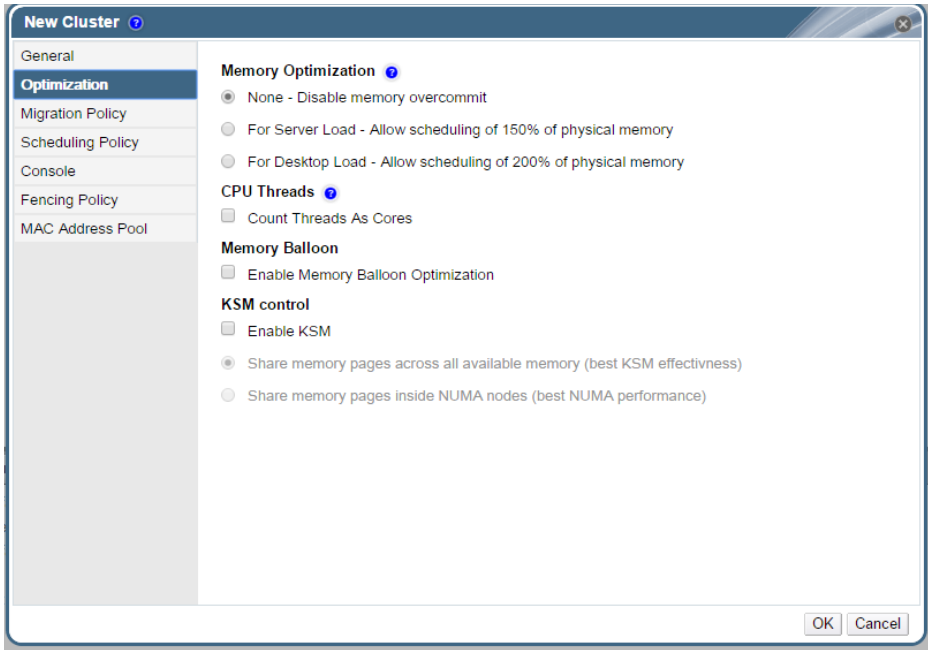


Figure 3: Default memory settings for RHV.

Phase 2a: Increase VM count to 16

Powering on one additional VM brought the total SQL Server VM count from 15 to 16. Doing so increased the total host memory consumed, including hypervisor overhead, to exceed the 256 GB of installed physical memory. This forced a slightly memory-overcommitted environment.

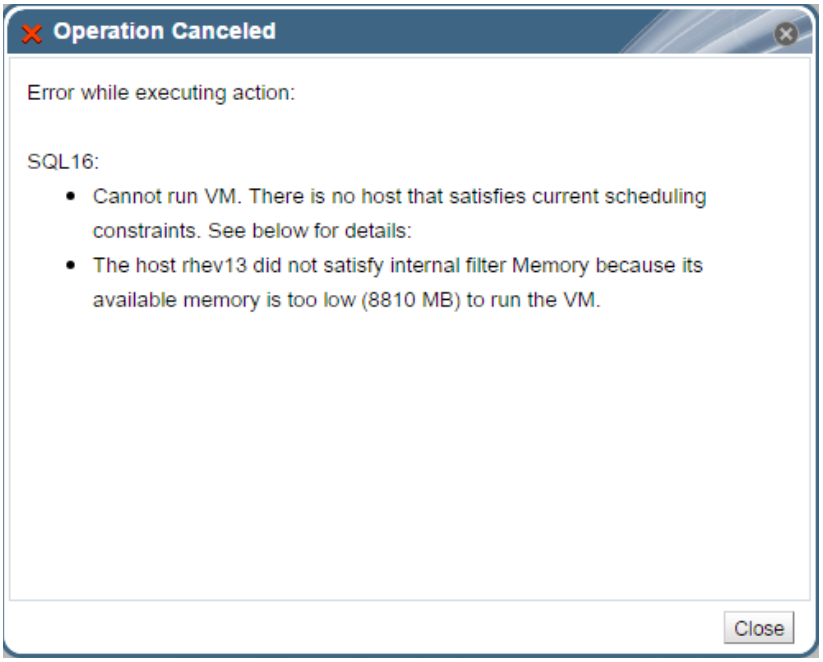
VMware vSphere succeeded right out of the box

We tested vSphere using out-of-the-box settings. With memory overcommit enabled by default, we could add a sixteenth SQL Server 2016 VM while maintaining per-VM performance. With the 16 VMs running, the overall number of OPM on the server increased by 6.4 percent over the baseline non-memory-overcommitted environment.

Figure 4: Total OPM across the server increased from the baseline of 55,808 to 59,445 with the addition of a sixteenth VM.

RHV failed even after we enabled Memory Optimization

As with vSphere, we began our RHV testing using out-of-the-box settings. This means that Memory Optimization was disabled. When we attempted to increase the total VM count from 15 to 16, VM 16 could not power on due to a lack of available server memory (see error at right).

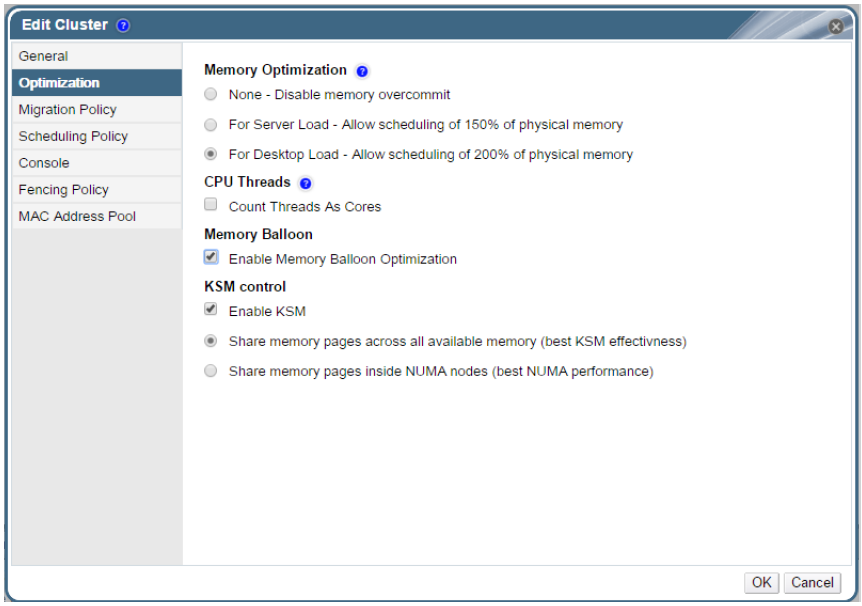


Next, we set out to determine which combination of memory settings at the cluster level would allow RHV to reclaim as much memory from virtual machines as possible on the single node. After some experimentation, we arrived at the settings presented in the table below.

Memory Optimization	We set Memory Optimization to For Desktop Load to allow scheduling of 200 percent of physical memory. This allows the memory page-sharing threshold to be as great as double the amount of system memory on the host.
Memory Balloon	We enabled memory overcommitment on virtual machines by enabling Memory Balloon Optimization, which makes use of the memory balloon driver installed as a part of the RHEL guest tools.[1]
KSM Control	We enabled KSM (Kernel Shared Memory) to allow the sharing of kernel pages between virtual machines to further consolidate memory.[2]

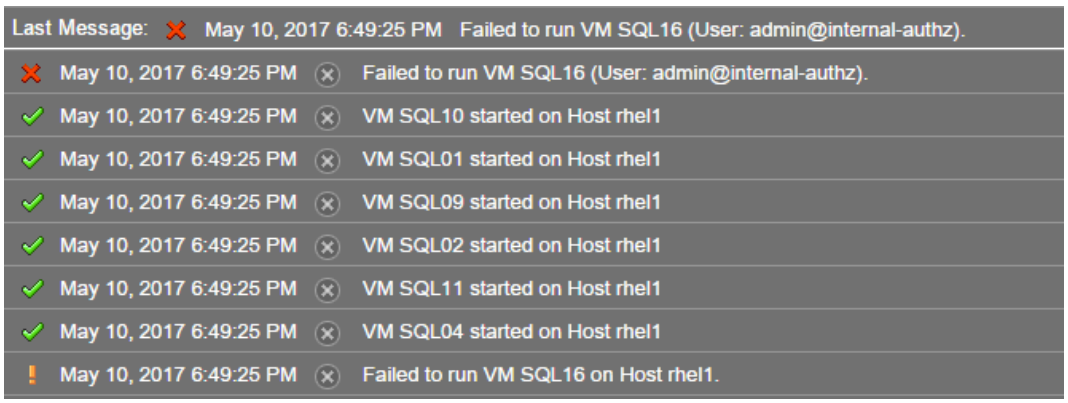
Even though virtual machines created in RHV have the Memory Balloon Driver enabled by default, making use of this setting is disabled by default in the Red Hat Virtualization Manager (RHV-M) cluster settings. Enabling this setting allows RHV-M to reclaim memory from virtual machines, up to the Physical Memory Guaranteed amount. We also enabled Kernel Shared Memory, also disabled by default, which lets RHV-M share certain portions of memory that are identical across virtual machines.

While RHV-M provides options that admins can use to define memory size and maximum memory to define a dynamic memory range, we found that these settings are useful only for manual operations of "hot plugging" memory into a virtual machine.



Unlike ESXi®, which can present a small initial amount of RAM to a virtual machine and allow it to grow with the operating system automatically as needed, in RHV this process is completely manual. The virtual machine OS always sees the value entered for Memory Size; the only way to change this is to have an administrator right-click and edit a virtual machine while it is running. While this strategy is technically possible, it is too labor-intensive to be practical under real-world datacenter conditions.

While RHV-M cannot expand or contract memory automatically in the context of the virtual machine's operating system, it does provide a way of handling memory overcommitment with memory ballooning and kernel shared memory. However, even once we had enabled these settings, we still could not power on all 16 VMs simultaneously. The reason is that over time, RHV-M keeps track of small amounts of virtual machine memory that are not being used, and eventually releases that memory back to the hypervisor to consolidate and re-allocate. When we tried to power on 16 VMs at once, only 15 would power on; the sixteenth VM would not (see screenshot below).



Eventually, after 2 to 5 minutes, RHV-M reclaimed enough memory from the initial 15 virtual machines to allow the sixteenth VM to power on. However, if an admin needed to power on a specific number of virtual machines that incurred even a slight memory overcommitment, even with memory optimization enabled, RHV-M would be unable to complete the task.

What we learned

vSphere default settings allowed us to power on one additional VM, which increased the total number of orders per minute the server could achieve. After we manually changed several settings in RHV, it could power on a sixteenth VM, which it could not do using out-of-the-box settings. However, the VM powered on only after a delay of several minutes, during which the hypervisor waited to reclaim memory to consolidate and reallocate.

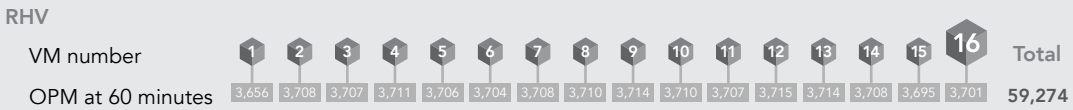


Figure 5: Total OPM across the server increased from the baseline of 55,409 to 59,274 with the addition of a sixteenth VM.

Phase 2b: Increase VM count to 20

To achieve a density target of 20 VMs, we first set up 16 VMs to execute the SQL Server 2016 workload for 60 minutes. Thirty minutes into the run, we powered on four additional VMs. This reflects a typical datacenter scenario where admins respond to increased demand by starting up additional VMs on a server that is already memory constrained.

VMware vSphere succeeded right out of the box

Using out-of-the-box settings, vSphere could handle the 20-VM workload with negligible degradation to per-VM OPM performance. We found that the vSphere environment was much more efficient than RHV in allocating startup memory in overcommitment scenarios. The VMs could power on immediately, with vSphere engaging memory ballooning and compression and eventually swapping RAM from overcommitted VMs to disk. In this phase, total database performance increased by 32.3 percent over the baseline non-memory-overcommitted environment.

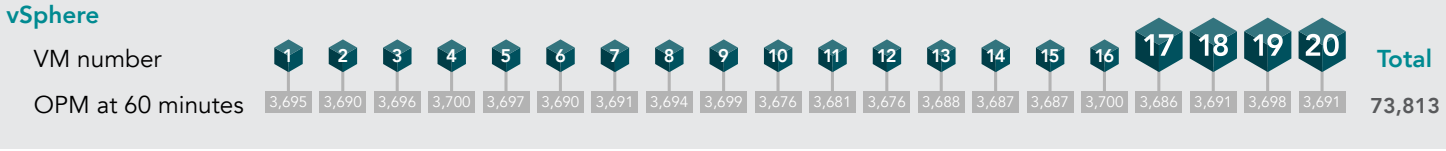


Figure 6: With 20 VMs, total OPM across the server increased from the baseline of 55,808 to 73,813.

RHV did not succeed, even after we enabled Memory Optimization, and manually adjusted settings

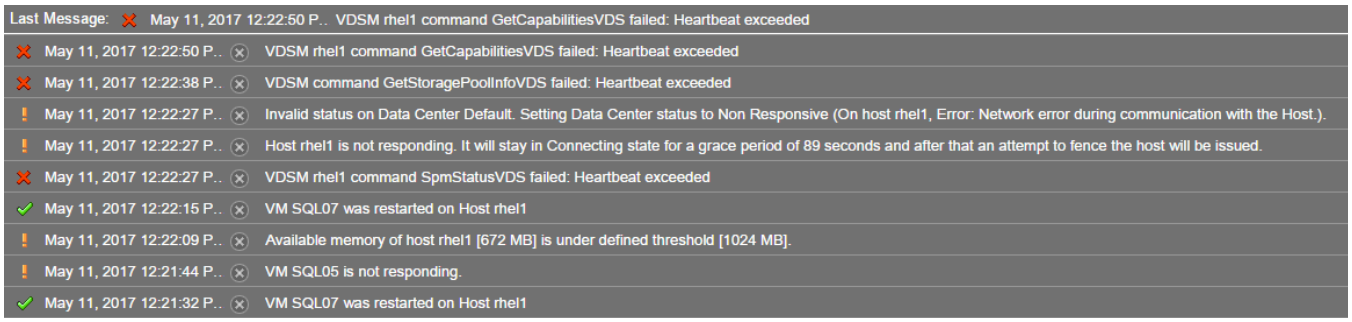
In the RHV environment, powering on 16 VMs to start this test required us to wait for enough memory to be available for the sixteenth VM.

Once we had done so, we attempted to power on all four additional virtual machines at the same time 30 minutes into the run. Not only was RHV unable to successfully power on the third and fourth VMs, the system was so constrained by memory during the VM power-up process that the hypervisor had to restart three of our original 15 VMs (VM 7, VM 12, and VM 14).

Figure 7: RHV powered on only two new VMs and restarted three of the original VMs. A red X represents a failed or restarted VM.

The RHV implementation of memory ballooning comes with few safeguards against memory overcommitment scenarios that could compromise the host. In our case, the host became so memory-constrained that it missed certain heartbeats with the manager, and lost connection to three of our original 15 virtual machines, forcing a restart. An administrator using RHV and wanting to benefit from memory ballooning would have to plan carefully for scenarios where memory overcommitment could prevent the host from communicating effectively with the manager.

The screenshot below shows the errors we saw while attempting to power on the four additional virtual machines.



What we learned

Using default settings and no manual intervention whatsoever, vSphere allowed us to power on four additional VMs for a total of 20 while maintaining the standardized baseline per-VM workload we determined in Phase 1. To achieve this with RHV was not possible, even after we manually changed several settings.

Phase 2c: Increase VM count to 24

VMware vSphere succeeded right out of the box

For our next phase of testing, we attempted to run 24 SQL Server VMs to see how vSphere would handle the heavily overcommitted environment. Even at 24 SQL Server VMs with 16 GB of RAM each, vSphere could power on all of the VMs and complete the workload. As with the lower VM counts, it did so using out-of-the-box settings and required no manual adjustments on the part of an administrator.

Furthermore, vSphere achieved a total of 87,876 orders per minute across the 24 VMs, an increase of 57.5 percent over the baseline total with no memory overcommitment.

Figure 8: With 24 VMs, total OPM across the server increased from the baseline of 55,808 to 87,876.

Figure 9: With 24 VMs, the server achieved more than one and half times the database workload as it did without memory overcommitment.

RHV failed even after we enabled Memory Optimization and manually adjusted a setting

In the RHV environment, when we attempted to power on the additional VMs, we were faced with a similar situation to the 20-VM test. The server became so overcommitted in an attempt to power on the new VMs that seven of the original 16 VMs were lost, and two of the additional VMs were unable to be powered on. The screenshot below illustrates the instability in RHV after we attempted to power on the additional virtual machines to complete the test workload.

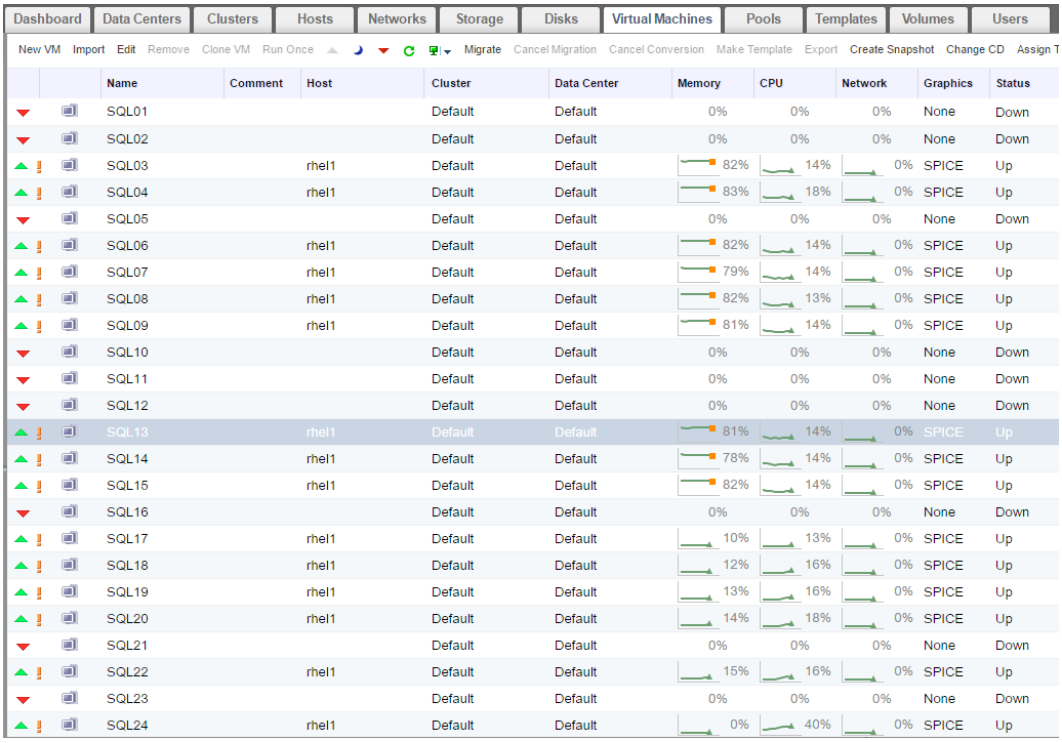


Figure 10: RHV powered on only two new VMs and restarted seven of the original VMs. A red X represents a failed or automatically restarted VM.

What we learned

Using default settings and no manual intervention whatsoever, vSphere allowed us to run 24 VMs while maintaining the standardized baseline per-VM workload we determined in Phase 1. We could tune the RHV environment to accommodate these resource disparities, and could assign VMs different levels of priority to ensure uptime on crucial workloads, but RHV is unable to handle large memory overcommitment scenarios out of the box. It also provides few safeguards against compromising the hypervisor host during large overcommitment scenarios.

Phase 3: Simulate a host power failure in a highly available, multi-node cluster

In our final phase of testing, we introduced a host power failure in a highly available, multi-node environment. We wanted to determine whether the hypervisors could recover the VMs from the node that lost power, reboot them, and resume performance at pre-failure levels.

We returned to the standardized baseline workload of 15 VMs per node. We started with a cluster of three servers for a total of 45 baseline workloads spread across the nodes. We configured each of the three servers with 256 GB of RAM and assigned 16 GB of memory to each SQL Server VM upon startup (the same VM configuration we tested earlier). Each virtual machine began executing our OLTP database workload.

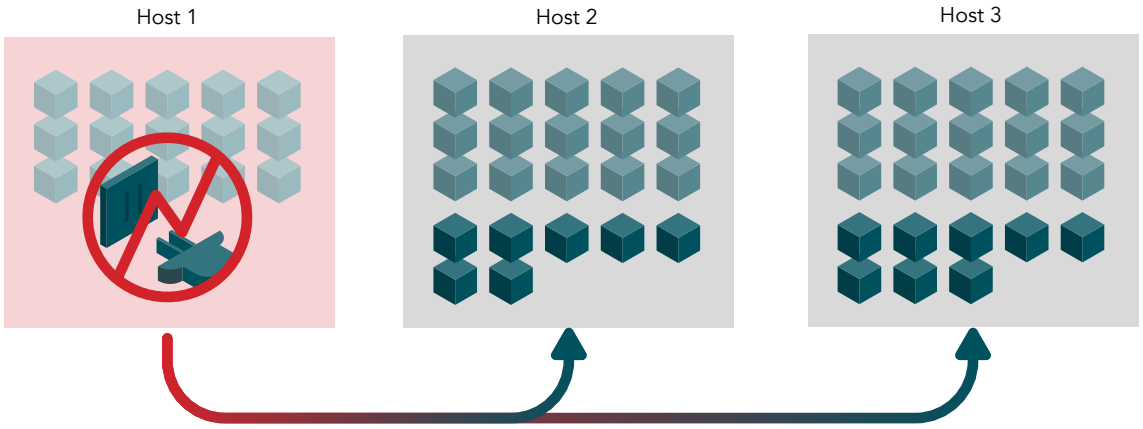
After 10 minutes, we forced a complete loss of power on one of the hosts to simulate an unexpected host failure. We monitored how long it took for the virtual machines on the affected host to migrate, power on, and appear in the hypervisor console, ready and available for administration.

VMware vSphere achieved 100 percent success right out of the box

Using default settings and no manual intervention whatsoever, vSphere migrated all 15 SQL VMs from the failed host onto the remaining two memory-saturated hosts with no issues. After roughly 35 seconds, the failed-over VMs were available to accept a new SQL Server workload, keeping the 45-VM environment accessible.

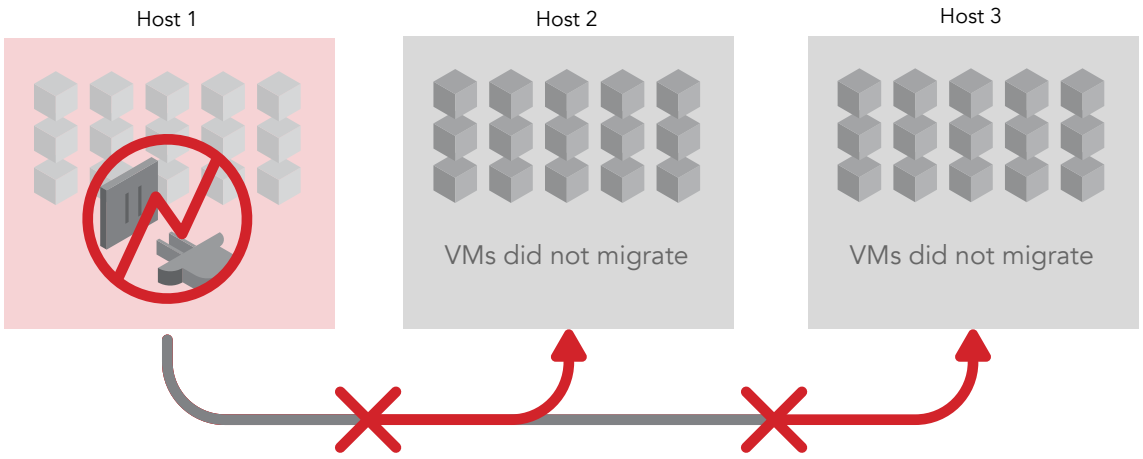
vSphere 45-VM cluster

With out-of-the-box settings: 100% recovery in 35 seconds

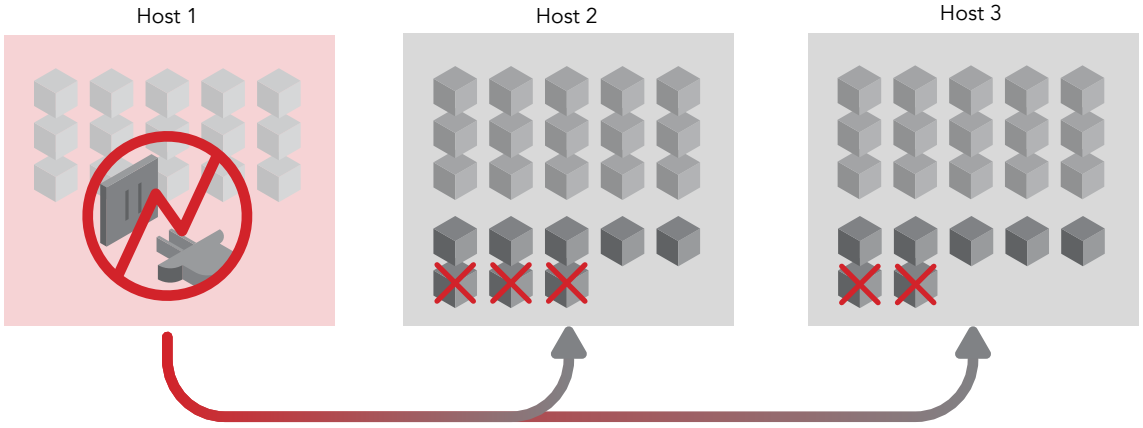


RHV 45-VM cluster

With out-of-the-box settings: 0% recovery



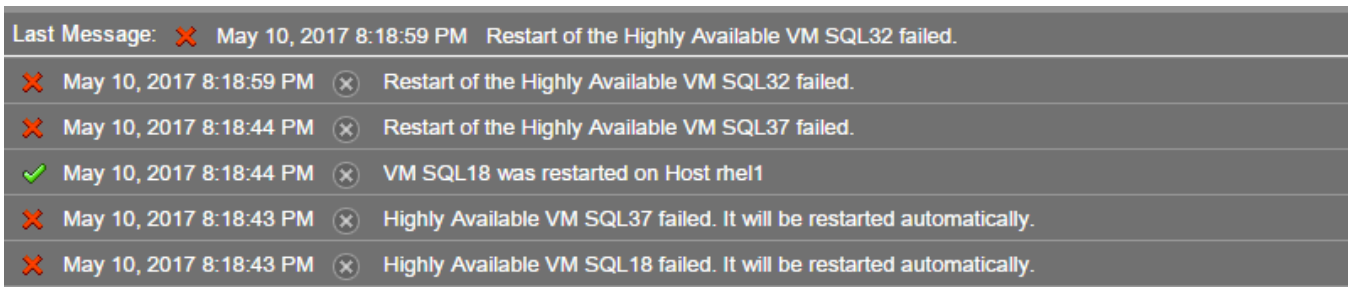
With optimized settings: 67% recovery in >16 minutes



RHV failed out of the box and was only partially successful after we performed manual tuning

After we configured manual power management for each host, enabled High Availability on each virtual machine in our RHV environment, and secured a High Availability lease for each virtual machine on the storage, we found that when using default cluster settings, if one of our hosts failed, RHV would not check the host's resource capacity before trying to migrate virtual machines.

This meant that as soon as RHV-M detected a server was down, it would attempt to migrate the VMs onto one of the other servers, both of whose memory was already fully utilized. This continued, ad infinitum, to the point of failure. Only intervention on the part of the administrator would stop the virtual machines from churning among the host servers. The screenshot below illustrates what happened when we attempted to restart highly available VMs in the RHV environment.



Next, we enabled High Availability Reservation on the cluster, and tried again. After starting the database workload and waiting 10 minutes to simulate a power failure, we pulled the power on one of our hosts to monitor how RHV handled the fail over. RHV-M could fail over some virtual machines in the memory-over provisioned environment, but the process took more than 16 minutes, and a total of five VMs were left in a Down Status. RHV-M constantly attempted to boot virtual machines – some would succeed, but most would fail while the hypervisor waited to reclaim memory from the freshly migrated and restarted virtual machines.

In our testing, the virtual machines were not configured to resume their workload after a restart. RHV was able to power on and restart as many VMs as it did in large part because the restarted virtual machines were idling, not running the SQL Server workload, and using very little overall memory. In a production environment, administrators would have the VMs immediately resume their workloads after restarting rather than allowing them to idle. This means that less memory would be available for failing over and restarting virtual machines than it was in our test environment, and it is likely that fewer VMs would successfully migrate.

What we learned

Using default settings and no manual intervention whatsoever, vSphere migrated all 15 VMs on the server with the simulated failure in 35 seconds. Even with manual tuning on the RHV cluster, RHV was unable to fail over all virtual machines in the memory overcommitted environment, and needed more than 16 minutes to migrate and restart only two-thirds of the 15 VMs.

Conclusion: Boost VM density with VMware vSphere

Figure 11 illustrates how total database performance increased along with VM density in our test environments. The VMware vSphere server comfortably ran 24 VMs, 1.60 times as many as it ran without memory overcommitment, and delivered 1.57 times the total number of orders per minute. As we have emphasized throughout this report, this dramatic improvement occurred using the vSphere out-of-the-box settings and required no manual tuning. The RHV could not run more than 16 VMs.

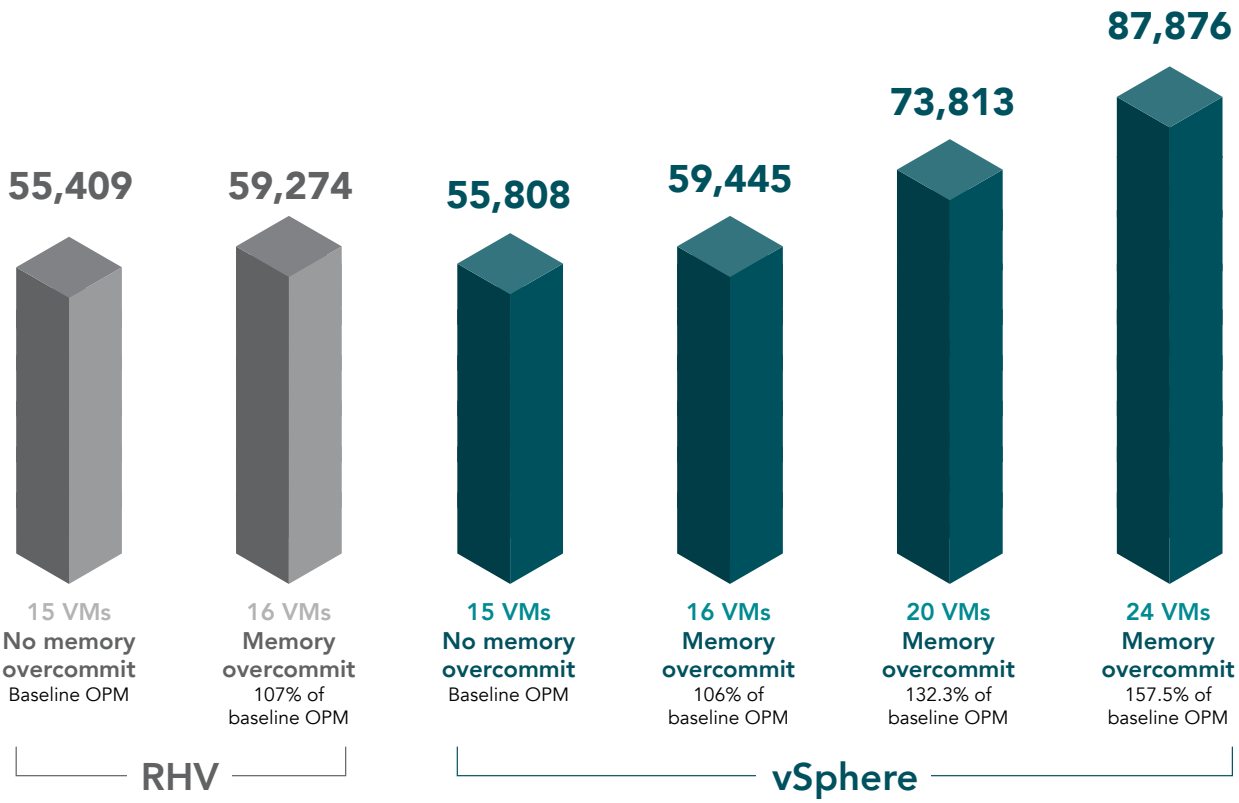


Figure 11: Total database performance as VM density increased in our test environments.

vSphere employs multiple memory management techniques, including ballooning and compression to keep VMs running and highly available. In extreme cases when additional memory is necessary to keep VMs running, vSphere engages memory swapping as long as sufficient disk space is available to swap the overcommitted memory. This comes at a small cost to per-VM performance, but ensures that VMs can power on in almost any configuration of RAM allocation, as long as other server resources, such as CPU and storage, are not constrained. Note that admins can also enable Transparent Page Sharing to consolidate memory even further; because we wanted to represent the out-of-the-box experience with vSphere, we did not enable Transparent Page Sharing or use it in any of our tests.

In contrast, the RHV approach to managing memory overcommitment requires manual tuning to increase VM density. Moreover, unless an individual VM's Maximum RAM is available on the host RHV cannot power on the VM. This means that even with Memory Optimization enabled, RHV cannot overcommit memory indefinitely.

While vSphere comfortably ran 24 VMs per server using out-of-the-box settings, RHV with a reasonable amount of adjusting settings maxed out at 16 VMs (see Figure 12). Based on these findings, you'd be able to run 72 VMs on only three VMware vSphere servers, more than the 64 VMs that **four** RHV servers would support.

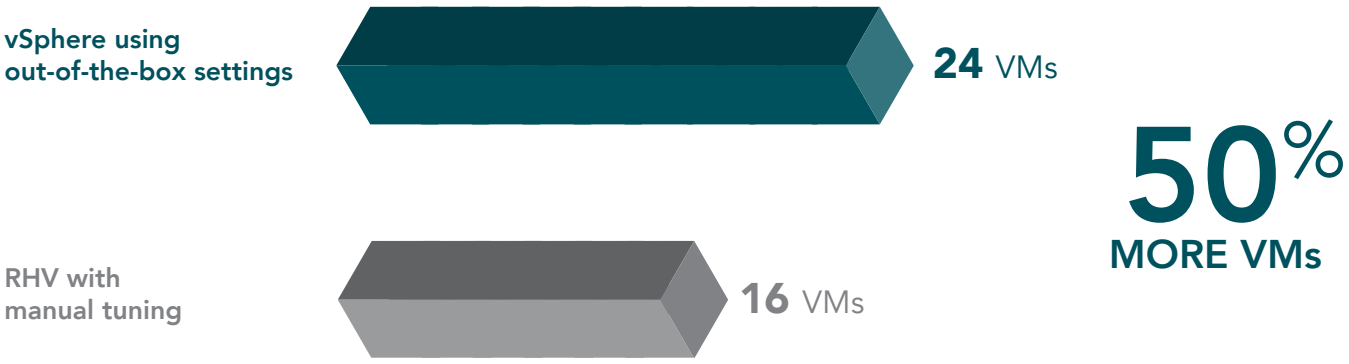


Figure 12: Maximum VM density our Lenovo System X 3650 M5 supported in our testing.

The greater VM density that the vSphere approach to memory overcommitment makes possible has enormous potential to optimize your server usage and boost datacenter efficiency. Every server in your datacenter comes with a hefty price tag—for space, power and cooling, software licensing, and IT management. Being able to do more with fewer servers has a direct benefit on your company's bottom line.

In addition to the greater VM density potential with vSphere, its fast and effective recovery during a hardware failure is a great asset to your data integrity and your ability to provide consistently high service. What would make you feel more confident—being able to rely on your hypervisor to migrate 100 percent of VMs from a failed node in 35 seconds using out-of-the-box settings or knowing that even after you had manually tuned settings, only two-thirds of the VMs successfully migrated. And it took more than 16 minutes!

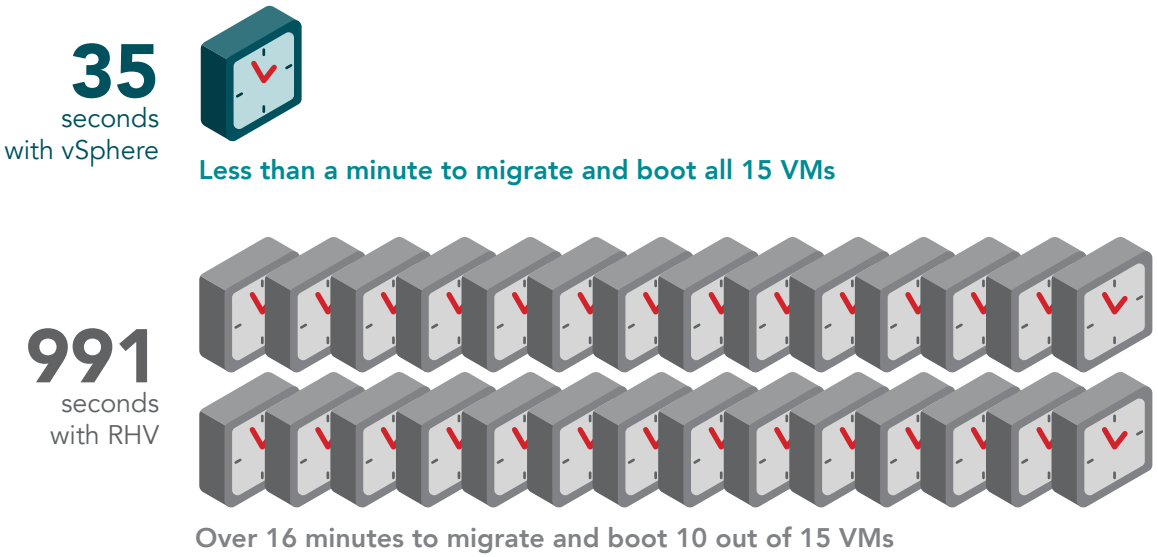


Figure 13: Time the two hypervisors needed to try to migrate and boot 15 VMs after a node failure. vSphere had a 100 percent success rate while RHV, which took more than 28 times as long, had a success rate of only 67 percent.

1 According to Red Hat documentation, "Additionally, in some scenarios ballooning may cause sub-optimal performance for a virtual machine. Administrators are advised to use ballooning optimization with caution." https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.1/html-single/administration_guide/

2 While it was necessary to enable KSM control to support greater VM density, doing so has the potential to introduce security issues. See the following for more information: https://access.redhat.com/blogs/766093/posts/1976303, https://www.usenix.org/system/files/conference/woot15/woot15-paper-barresi.pdf, https://staff.aist.go.jp/k.suzaki/EuroSec2011-suzaki.pdf.

Appendix A: Test components and benchmark

About VMware vSphere 6.5

According to VMware, VMware vSphere 6.5 is the next-generation infrastructure for next-generation applications. It provides a powerful, flexible, and secure foundation for business agility that accelerates the digital transformation to cloud computing and promotes success in the digital economy. vSphere 6.5 supports both existing and next-generation apps through its 1) simplified customer experience for automation and management at scale; 2) comprehensive built-in security for protecting data, infrastructure, and access; and 3) universal application platform for running any app anywhere. With vSphere 6.5, customers can now run, manage, connect, and secure their applications in a common operating environment, across clouds and devices. http://www.vmware.com/products/vsphere.html

About Red Hat Virtualization 4.1

According to Red Hat, Red Hat Virtualization 4.1 is the latest release of the company's Kernel-based Virtual Machine (KVM)-powered enterprise virtualization platform that provides an open source infrastructure and centralized management solution for virtualized servers and workstations and built on the enterprise-grade backbone of Red Hat Enterprise Linux. Learn more at https://www.redhat.com/en/about/press-releases/red-hat-offers-launchpad-it-transformation-latest-version-red-hat-virtualization.

About Lenovo System x3650 M5 rack servers

According to Lenovo, with the powerful, versatile 2U two-socket System x3650 M5 rack server, you can run even more enterprise workloads, 24/7, and gain faster business insights. Integrated with up to two Intel® Xeon® processors E5-2600 v4 series (up to 44 cores per system), fast TruDDR4 2400MHz Memory, and massive storage capacity, the x3650 M5 fast forwards your business. You can select from an impressive array of storage configurations (up to 28 drive bays) that optimize diverse workloads from Cloud to Big Data. http://www3.lenovo.com/us/en/data-center/servers/racks/x3650-M5/p/77XS7HV7V64

About Microsoft SQL Server 2016

According to Microsoft, "SQL Server 2016 is the biggest leap forward in Microsoft data platform history. Gain real time insights across your transactional and analytical data with a scalable database platform that has everything built in, from unparalleled in-memory performance, new security innovations and high availability, to advanced analytics that make mission-critical applications intelligent." Learn more about Microsoft SQL Server 2016 at https://www.microsoft.com/en-us/server-cloud/products/sql-server/.

About Windows Server 2016

According to Microsoft, Windows Server 2016 is a cloud-ready operating system that supports your current workloads while introducing new technologies that make it easy to transition to cloud computing. https://www.microsoft.com/en-us/cloud-platform/windows-server

About our test tool, DVD Store 2

DVD Store 2 (DS2) models an online DVD store, where customers log in, search for movies, and make purchases. DS2 reports these actions in OPM that the system could handle, to show what kind of performance you could expect for your customers. The DS2 workload also performs other actions, such as adding new customers, to exercise the wide range of database functions you would need to run your ecommerce environment. For more information about the DS2 tool, see www.delltechcenter.com/page/DVD+Store.

Appendix B: System configuration information

On April 21, 2017, we finalized the hardware and software configurations we tested. We concluded hands-on testing on May 12, 2017.

Server configuration information	
Server model	Lenovo System X 3650 M5
Number of servers	6
BIOS name and version	10.2 build DSALA8N
Non-default BIOS settings	N/A
Operating system name and version for Red Hat solution	Red Hat Virtualization 4.1
Operating system name and version for VMware solution	VMware ESXi 6.5.0
Date of last OS updates/patches applied	11-03-2016
Power management policy	Performance
Processor	
Number of processors	2
Vendor and model	Intel® Xeon® E5-2670 v3
Core count (per processor)	12
Core frequency (GHz)	2.30
Stepping	M1
Memory module(s)	
Total memory in system (GB)	256
Number of memory modules	16
Vendor and model	Hynix HMA42GR7MFR4N-TF
Size (GB)	16
Type	DDR4
Speed (MHz)	2,133
Speed running in the server (MHz)	2,133
Storage controller	
Vendor and model	IBM® MegaRAID M5210
Cache size (GB)	2
Firmware version	24.12.0-0024
Driver version	N/A

Server configuration information	
Local storage	
Number of drives	2
Drive vendor and model	Seagate® ST9500620NS
Drive size (GB)	500
Drive information (speed, interface, type)	SATA 7200RPM 6.0Gb/s
Network adapter	
Vendor and model	IBM Broadcom® NetXtreme® Gigabit Ethernet Controller
Number and type of ports	4
Driver version	17.4.4.2a
Cooling fans	
Vendor and model	N/A
Number of cooling fans	6
Power supplies	
Vendor and model	Delta 94Y8143
Number of power supplies	2
Wattage of each (W)	750

Appendix C: Detailed test procedure

Overview

We configured six identical Lenovo System x3650 M5 servers, three for testing vSphere and three for testing RHV. We set up the servers using BIOS default settings that disabled Power Capping and set the power performance setting to Maximum Performance. We installed the hypervisors on local storage using a two-disk RAID1 configuration. For single-node testing, VM storage was provided by an SSD-based storage array connected to the servers via redundant 10GbE iSCSI connections. For multi-node testing, VM storage was provided by a higher-capacity all-flash Fibre Channel storage array with four 16GbE Fibre Channel ports and connected to the servers via dual port 16GbE Fibre Channel cards. We configured an additional Lenovo System x3650 M5 server as a management server for each testbed.

We configured and deployed similarly configured SQL Server VMs on each platform and used the same benchmark settings to perform tests across multiple scenarios. Each VM had four vCPUs and 16 GB of memory and ran the latest version of Window Server 2016 and SQL Server 2016. We deployed a total of 45 SQL Server VMs on each platform. Our DVD Store 2 tests consisted of a 15-minute warm-up period followed by a 60-minute test period.

In all phases of testing, we assigned 16 GB of vRAM to each VM upon startup to cache database files as necessary, ensure optimal SQL Server performance, and reflect real-world conditions. The VMs all ran Windows Server® 2016 and SQL Server 2016 and had four vCPUs and a 20GB database. In all phases of testing, the host servers had 256 GB of physical RAM.

Setting up VMware vSphere

Installing VMware ESXi 6.5.0

We used the following steps to configure each server on our VMware environment:

1. Attach the installation media.
2. Boot the Lenovo System x3650 M5 server.
3. At the VMware Installer screen, press Enter.
4. At the EULA screen, to Accept and Continue, press F11.
5. Under Storage Devices, select the appropriate virtual disk, and press Enter.
6. For the keyboard layout, select US, 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 through 13 on the remaining servers.

Deploying and configuring vCenter Server Appliance 6.5

We used the following steps to deploy vCenter server appliance on the management server:

1. Mount installation ISO and launch the vcsa-setup.html file.
2. In the wizard that appears, click Next.
3. Accept the terms of the license agreement, and click Next.
4. Leave the default installation directory, and click Next.
5. Click Install.
6. Click Finish.
7. In the mounted image's top directory, open vcsa-setup.html.
8. When prompted, click Allow.
9. Click Install.
10. Accept the terms of the license agreement, and click Next.
11. Enter the FQDN or IP address of the host onto which the vCenter Server Appliance will be deployed.
12. Provide a username and password for the system in question, and click Next.
13. Accept the certificate of the host you chose to connect to by clicking Yes.
14. Select the appropriate datacenter, and click Next.
15. Select the appropriate resource, and click Next.

16. Provide a name and password for the virtual machine, and click Next.
17. Select Install vCenter Server with an Embedded Platform Services Controller, and click Next.
18. Select Create a new SSO domain.
19. Provide a password, and confirm it.
20. Provide an SSO Domain name and SSO Site name, and click Next.
21. Set an appropriate Appliance Size, and click Next.
22. Select an appropriate datastore, and click Next.
23. Select Use an embedded database (PostgreSQL), and click Next.
24. At the Network Settings page, configure the network settings as appropriate for your environment, and click Next.
25. Review your settings, and click Finish.
26. When installation completes, click Close.
27. Log into the vCenter's web client at https://vcenter-ip-address/vsphere-client/?csp.
28. Add all necessary ESXi and vCenter licenses to the vCenter.

Creating a cluster and adding hosts to vCenter Server

We used the following steps to add servers to vCenter and create a highly available server cluster:

1. Once you have logged into 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. Under vSphere HA select Turn On.
8. Click OK.
9. Once the cluster has been created, right-click the cluster, and click Add Host.
10. Enter the IP address for the first server, and click Next.
11. Enter the root credentials for the server, and click Next.
12. To accept the server's certificate, click Yes.
13. Review the server details, and click Next.
14. Assign the desired license, and click Next.
15. Disable Lockdown mode, and click Next.
16. Click Finish.
17. Repeat steps 8 through 15 for the remaining servers.
18. Once you have added all of the servers, create dedicated VM, storage and vMotion networks and mount the appropriate datastores for testing.

Creating the SQL Server VMs

We used the following steps to create 15 SQL Server 2016 VMs per server:

1. In VMware vCenter, navigate to Virtual Machines.
2. To create a new VM, click the icon.
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 appropriate datastore to host the VM, and click Next.
7. Select the guest OS as Windows Server 2016, and click Next.
8. In the Customize Hardware section, use the following settings:
   - Set the vCPU count to 4.
   - Set the Memory to 16GB.
   - Add 1 x 80GB VMDK for OS, 1 x 50GB VMDKs for database files and 1 x 28GB VMDK for database logs. Set all VMDKs to thick provision eager zeroed.
   - Create three additional VMware Paravirtual SCSI controllers, and assign the VMDKs to the new controllers.
   - Attach a Windows Server 2016 ISO to the CD/DVD drive.
9. Click Next.
10. Click Finish.

Installing and configuring the Red Hat Virtualization environment

1. Download, mount, and install the RHEL 7.3 installation media to begin the RHEVM installation.
2. When the RHEL installer launches, select the installer language, and click Continue.
3. From the installation summary screen, set the Date & Time, choose an installation destination, and choose the software selection. We chose the infrastructure server installation type.
4. Click Network & Host Name, enter a host name for the server, and enable the management interface. Click Done to save, and return to the installer summary page.
5. Click Begin Installation. Set a root password while the media installs.
6. When installation completes, reboot the server and remove the installation media.
7. Log into the Red Hat Enterprise Linux (RHEL) server, subscribe to the required entitlements for RHEV-Manager, and install updates with the following command: `yum update`
8. Install RHEV-Manager with the following command: `yum install rhevm`
9. Run the configuration script to install RHEV-Manager with the following command: `engine-setup`
   Use the default configuration options.
10. Download, mount, and install the Red Hat Virtualization Host hypervisor installation media on each host.
11. When the RHV installer launches, select the installer language, and click Continue.
12. From the installation summary screen, set the Date & Time, and choose an installation destination.
13. Click Network & Host Name, enter a host name for the server, and enable the management interface. Click Done to save, and return to the installer summary page.
14. Click Begin Installation. Set a root password while the media installs.
15. When installation completes, reboot the server and remove the installation media.
16. Once RHEV-M and the hypervisor hosts are installed, use a web browser to navigate to the hostname of the RHEV-M server, and log into the Red Hat Virtualization administration portal.
17. From the System menu, click the Hosts tab, and click New to add the first host.
18. Enter the host IP and Password, and click OK.
19. Enable power management, and enter the IMM credentials.
20. Add remaining hosts.
21. Click the Storage tab, import the ISO storage domain to host the Windows Server image, and create a new storage domain for the networked storage, ensuring each host has access to the storage.
22. Select the Default Cluster, and click Edit. We enabled Memory Optimization for Desktop Load (200%) and enabled Memory Balloon Optimization.
23. Navigate to the virtual machine tab, and create a new virtual machine using the following settings:
    - Set the vCPU count to 4.
    - Set the Memory to 16384 MB.
    - Enable High Availability.
    - Select network storage for a VM Lease.
    - Set the Priority to High.
24. Attach the Windows Server installation media using the ISO storage domain, and install the operating system along with the Red Hat Virtualization guest tools.
25. Once the virtual machine is set up for testing, right-click the VM, and convert to template.
26. Clone the desired number of virtual machines.

Installing and configuring SQL Server VMs

Installing Windows Server 2016

We used the following steps to install and configure SQL Server VMs:

1. Attach the Windows Server 2016 ISO to the virtual machine.
2. Open the VM console, and start the VM.
3. When prompted to boot from DVD, press any key.
4. When the installation screen appears, leave language, time/currency format, and input method as default, and click Next.
5. Click Install now.
6. When the installation prompts you, enter the product key.
7. Select Windows Server 2016 Datacenter Edition (Server with a GUI), and click Next.
8. Check I accept the license terms, and click Next.
9. Click Custom: Install Windows only (advanced).
10. Select Drive 0 Unallocated Space, and click Next. This starts Windows automatically, and Windows will restart automatically after completing.
11. When the Settings page appears, fill in the Password and Reenter Password fields with the same password. Log in with the password you set up previously.
12. Install VMware Tools in the VMs hosted on the ESXi servers, or install RHV Guest Agent on the VMs hosted in the RHV servers.

Installing SQL Server 2016

1. Attach the installation media ISO for SQL Server 2016 to the VM.
2. Click Run SETUP.EXE. If Autoplay does not begin the installation, navigate to the SQL Server 2016 DVD, and double-click it.
3. In the left pane, click Installation.
4. Click New SQL Server stand-alone installation or add features to an existing installation.
5. Specify Evaluation as the edition you are installing, and click Next.
6. To accept the license terms, click the checkbox, and click Next.
7. Click Use Microsoft Update to check for updates, and click Next.
8. At the Feature Selection screen, select Database Engine Services, Full-Text and Semantic Extractions for Search, Client Tools Connectivity, and Client Tools Backwards Compatibility.
9. Click Next.
10. At the Instance configuration screen, leave the default selection of default instance, and click Next.
11. At the Server Configuration screen, accept defaults, and click Next.
12. At the Database Engine Configuration screen, select the authentication method you prefer. For our testing purposes, we selected Mixed Mode.
13. Enter and confirm a password for the system administrator account.
14. Click Add Current user. This may take several seconds.
15. Click Next.
16. At the Ready to Install screen, click Install.
17. Close the installation window.
18. In the SQL Server Installation Center, click on Install SQL Server Management Tools.
19. Click Download SQL Server Management Studio.
20. Click Run.
21. When the Microsoft SQL Server Management Studio screen appears, click Install.
22. When the installation completes, click Close.

Configuring and running the DVD Store 2 benchmark

Data generation overview

We generated the data using the Install.pl script included with DVD Store version 2.1 (DS2), providing the parameters for our 20GB database size and the database platform we used. We ran the Install.pl script on a utility system running Linux® to generate the database schema.

After processing the data generation, we transferred the data files and schema creation files to a Windows- based system running SQL Server 2016. We built the 20GB database in SQL Server, then performed a full backup, storing the backup file remotely for quick access. We used that backup file to restore the database when necessary.

The only modification we made to the schema creation scripts were the specified file sizes for our database. We explicitly set the file sizes higher than necessary to ensure that no file-growth activity would affect the outputs of the test. Other than this file size modification, we created and loaded the database in accordance to the DVD Store documentation. Specifically, we followed these steps:

1. Generate the data, and create the database and file structure using database creation scripts in the DS2 download. Make size modifications specific to our 20GB database, and make the appropriate changes to drive letters.
2. Transfer the files from our Linux data generation system to a Windows system running SQL Server.
3. Create database tables, stored procedures, and objects using the provided DVD Store scripts.
4. Set the database recovery model to bulk-logged to prevent excess logging.
5. Load the data we generated into the database. For data loading, use the import wizard in SQL Server Management Studio. Where necessary, retain options from the original scripts, such as Enable Identity Insert.
6. Create indices, full-text catalogs, primary keys, and foreign keys using the database-creation scripts.
7. Update statistics on each table according to database-creation scripts, which sample 18 percent of the table data.
8. On the SQL Server instance, create a ds2user SQL Server login using the following Transact SQL (TSQL) script:

```
USE [master]
GO
CREATE LOGIN [ds2user] WITH PASSWORD=N'',
    DEFAULT_DATABASE=[master],
    DEFAULT_LANGUAGE=[us_english],
    CHECK_EXPIRATION=OFF,
    CHECK_POLICY=OFF
GO
```

9. Set the database recovery model back to full.
10. Create the necessary full text index using SQL Server Management Studio.
11. Create a database user, and map this user to the SQL Server login.
12. Perform a full backup of the database. This backup allows you to restore the databases to a pristine state.

Running the DVD Store tests

We created a series of batch files, SQL scripts, and shell scripts to automate the complete test cycle. DVD Store outputs an orders-per-minute metric, which is a running average calculated through the test. In this report, we report the last OPM that each target reported.

Each complete test cycle consisted of general steps:

1. Clean up prior outputs from the target system.
2. Drop the database from the target.
3. Restore the database on the target.
4. Shut down the target.
5. Reboot the target host.
6. Wait for a ping response from the server under test and the client system.
7. Let the test server idle for 10 minutes.
8. Start the DVD Store driver on the four clients.

We used the following DVD Store 2 parameters for testing:

```
ds2sqlserverdriver.exe --target=<target _ IP> --ramp_rate=10 --run_time=60 --n_threads=32 --db_size=20GB
--think_time=0.05 --detailed_view=Y --warmup_time=15 --report_rate=1 --pct_newcustomers=20 --csv_
output=<drivepath>
```

This project was commissioned by VMware.

Principled Technologies®

Facts matter.®

Principled Technologies is a registered trademark of Principled Technologies, Inc.
All other product names are the trademarks of their respective owners.

DISCLAIMER OF WARRANTIES; LIMITATION OF LIABILITY:
Principled Technologies, Inc. has made reasonable efforts to ensure the accuracy and validity of its testing, however, Principled Technologies, Inc. specifically disclaims any warranty, expressed or implied, relating to the test results and analysis, their accuracy, completeness or quality, including any implied warranty of fitness for any particular purpose. All persons or entities relying on the results of any testing do so at their own risk, and agree that Principled Technologies, Inc., its employees and its subcontractors shall have no liability whatsoever from any claim of loss or damage on account of any alleged error or defect in any testing procedure or result.

In no event shall Principled Technologies, Inc. be liable for indirect, special, incidental, or consequential damages in connection with its testing, even if advised of the possibility of such damages. In no event shall Principled Technologies, Inc.'s liability, including for direct damages, exceed the amounts paid in connection with Principled Technologies, Inc.'s testing. Customer's sole and exclusive remedies are as set forth herein.