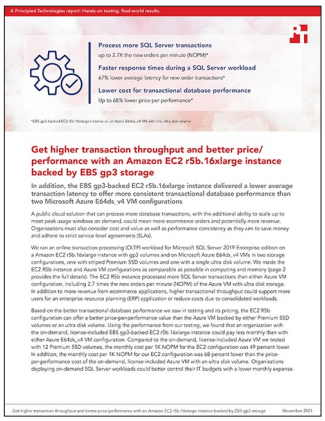

The science behind the report:

Get higher transaction throughput and better price/performance with an Amazon EC2 r5b.16xlarge instance backed by EBS gp3 storage

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 Get higher transaction throughput and better price/performance with an Amazon EC2 r5b.16xlarge instance backed by EBS gp3 storage.

We concluded our hands-on testing on October 6, 2021. 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 September 24, 2021 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.

We used the cost calculators provided by each vendor to estimate the cost for each solution using an on-demand model. To ensure instance costs included Microsoft Windows Server and Microsoft SQL Server 2019 Enterprise edition licensing, in the AWS pricing calculator, we selected "Windows Server with SQL Server Enterprise." In the Azure pricing calculator, we chose "SQL Server" for the Type dropdown and "SQL Enterprise" for the License dropdown.

Table 1: Results of our testing

	Amazon EC2 r5b.16xlarge + EBS gp3 storage	Azure E64ds_v4 VM + P40 storage	Azure E64ds_v4 VM + Ultra Disk storage
NOPM	516,867	288,401	185,272
Latency	6.63	12.71	20.49
Price/1,000	$46.50	$91.93	$148.60

System configuration information

Table 2: Detailed information on the system we tested.

Server configuration information	Amazon EC2 r5b.16xlarge + EBS gp3 storage	Azure E64ds_v4 VM + P40 storage	Azure E64ds_v4 VM + Ultra Disk storage
Tested by	Principled Technologies	Principled Technologies	Principled Technologies
Test date	09/24/2021	09/29/2021	10/06/2021
CSP / region	us-east-1c	US east zone 3	US east zone 3
Workload and version	HammerDB v4.2 TPROC-C	HammerDB v4.2 TPROC-C	HammerDB v4.2 TPROC-C
Workload specific parameters	30,000 warehouses, 144 virtual users	30,000 warehouses, 144 virtual users	30,000 warehouses, 144 virtual users
Iterations and result choice	Three runs, median	Three runs, median	Three runs, median
Server platform	r5b.16xlarge	E64ds_v4	E64ds_v4
BIOS name and version	Amazon EC2 1.0, 10/16/2017	Microsoft Corporation Hyper-V UEFI Release v4.1, 10/27/2020	Microsoft Corporation Hyper-V UEFI Release v4.1, 10/27/2020
Operating system name and version/build number	Microsoft Windows Server 2019 Datacenter Edition Version 10.0.17763 (Build 17763)	Microsoft Windows Server 2019 Datacenter Edition Version 10.0.17763 (Build 17763)	Microsoft Windows Server 2019 Datacenter Edition Version 10.0.17763 (Build 17763)
Date of last OS updates/ patches applied	09/24/2021	09/24/2021	09/24/2021
Processor			
Number of processors	2	2	2
Vendor and model	Intel® Xeon® Platinum 8259CL	Intel Xeon Platinum 8272CL	Intel Xeon Platinum 8272CL
Core count (per processor)	24	24	24
Core frequency (GHz)	2.50	2.60	2.60
Stepping	7	7	7
Hyper-threading	Yes	Yes	Yes
Turbo	Yes	Yes	Yes
CPU vCores per instance or VM type	64	64	64
Memory module(s)			
Total memory in system (GB)	512	504	504
NVMe memory present?	No	No	No
Total memory (DDR + NVMe RAM)	512	504	504

Server configuration information	Amazon EC2 r5b.16xlarge + EBS gp3 storage	Azure E64ds_v4 VM + P40 storage	Azure E64ds_v4 VM + Ultra Disk storage
General hardware			
Storage: Network or direct attached / instance	Network attached	Network attached	Network attached
Network bandwidth / instance (Gbps)	20	30	30
Storage bandwidth / instance (Mbps)	40,000	9,600	9,600
OS			
Number of drives	1	1	1
Drive size (GB)	50	128	128
Drive information (speed, interface, type)	gp2, EBS, 100/3000 IOPS	Standard SSD	Standard SSD
Data drive			
Number of drives	10	11	1
Drive size (GB)	500	2,000	4,000
Drive information (speed, interface, type)	gp3, EBS, 16,000 IOPS	Premium, P40, 7,500 IOPS	Ultra, 80,000 IOPS
Temporary drive			
Number of drives	0	1	1
Drive size (GB)	N/A	2,400	2,400
Network adapter			
Vendor and model	Amazon Elastic Network Adapter	Microsoft Hyper-V Network Adapter	Microsoft Hyper-V Network Adapter
Number and type of ports	1 x 20Gb	1 x 50Gb	1 x 50Gb

How we tested

For both the Amazon and Azure solutions, we used different storage configurations on otherwise matching instances or VMs. We first configured an EC2 r5b.16xlarge instance with 10 gp3 volumes attached and configured in a single stripe through the OS. The first Azure solution we tested used an E64ds_v4 VM with 11 P40 drives in a stripe with read caching enabled, as well as a single P60 drive for the log file. We separated these volumes to use Azure's disk read caching and give the Azure solution the best possible performance. We then repeated testing on the Azure VM with a single Ultra Disk, which cannot use disk caching. We compared the online transaction processing performance from the HammerDB 4.2 TPROC-C workload. We used a 30,000-warehouse database size with 144 virtual users driving the tests.

Creating the Microsoft Windows Server 2019 client and Microsoft SQL Server instances on Amazon

Creating the baseline instance

1. Log into AWS, and navigate to the AWS Management Console.
2. Click EC2.
3. Click Launch instance, and to open the Launch Instance in the dropdown wizard, click Launch instance.
4. In the search window, type `Windows Server`, and press enter.
5. Next to Microsoft Windows Server 2019 Base on Quick Start, click Select.
6. On Choose Instance Type, select r5b.16xlarge, and click Next: Configure Instance Details.
7. On Configure Instance, set the following:
    a. Number of instances: 1
    b. Purchasing option: Leave unchecked
    c. Network: Default VPC
    d. Subnet: Choose the region you are working in
    e. Auto-assign public IP: Enable
    f. Placement group: Leave unchecked
    g. Capacity reservation: Open
    h. Domain join directory: No directory
    i. IAM role: None
    j. Shutdown behavior: Stop
8. Click Next: Add storage.
9. On Add Storage, set the following:
    a. Size: 50 GB
    b. Volume type: gp2
    c. Delete on termination: Checked
    d. Encryption: Not encrypted
    e. Add any additional data volumes needed.
10. Click Next: Add Tags.
11. On Add Tags, add any tags you wish to use. Click Next: Configure Security Group.
12. On Configure Security Group, leave defaults, and click Review and Launch.
13. On Review, click Launch.
14. Choose the appropriate option for the key pair, and click Launch Instances.
15. Repeat 6 through 14 to create a second instance, but change the instance type in step 6 to m5n.8xlarge to use as a client instance.

Creating the Windows Server 2019 client and SQL Server instances on Azure

Create the baseline Image VM

1. Log into the Azure Portal, and navigate to the Virtual Machines service.
2. To open the Add VM wizard, click Add.
3. On Basics, set the following:
    a. From the dropdown menu, choose your Subscription.
    b. From the dropdown menu, choose your Resource group.
    c. Name the Virtual Machine.
    d. From the dropdown menu, choose your Region.
    e. Leave the Availability options set to No infrastructure redundancy required.
    f. From the Image dropdown menu, choose Windows Server 2019 Datacenter.
    g. Leave Azure Spot instance set to No.
    h. Select the instance size you wish to use; we used Standard E64ds_v4.
    i. Enter a new Username and Password for the Administrator account.
    j. Leave public inbound ports set to Allow selected ports.
    k. For Select inbound ports, choose SSH (22).
4. On Disks, set the following:
    a. From the dropdown menu, choose Standard SSD for the OS disk type.
    b. Add any additional data volumes needed.
    c. Leave the default Encryption type.
5. On Networking, set the following:
    a. From the dropdown menu, choose your Virtual network.
    b. To create a new public IP, choose Create new.
    c. Leave the rest of the settings at defaults.
6. On Management, set the following:
    a. From the dropdown menu, choose your Diagnostics storage account.
    b. Leave the rest set to defaults.
7. On Advanced, leave all defaults
8. On Tags, add any tags you wish to use.
9. On Review + create, review your settings, and click Create.
10. Repeat steps 2 through 9 steps to create a second instance, but change the instance type in step 3.8 to E32ds_v4 to use as a client instance.

Configuring Windows Server 2019

1. Open Server Manager, and click Local Server.
2. Disable IE Enhanced Security Configuration.
3. Change the time zone to your local time zone.
4. Change the name of your server, and when prompted, reboot.
5. Open Server Manager again, and click Local Server.
6. Click to run updates.
7. Run updates, rebooting when prompted, until the server shows no new updates to install.

Configuring Microsoft SQL Server 2019 on the system under test (AWS and Azure solutions)

Partitioning and formatting the data drives (AWS and Azure multi-disk solutions)

1. Log into the SQL VM or server under test.
2. Open a PowerShell instance with Administrator privileges.
3. To pool, partition, and format the disks as a single striped volume, run the following command:

```
$PhysicalDisks = Get-PhysicalDisk | Where-Object {$_.CanPool -eq $true}
New-StoragePool -FriendlyName "DataFiles" -StorageSubsystemFriendlyName "Windows Storage*" '
-PhysicalDisks $PhysicalDisks | New-VirtualDisk -FriendlyName "Datafiles" '
-Interleave 65536 -NumberOfColumns $PhysicalDisks.Count -ResiliencySettingName simple '
-UseMaximumSize |Initialize-Disk -PartitionStyle GPT -PassThru |New-Partition -AssignDriveLetter '
-UseMaximumSize |Format-Volume -FileSystem NTFS -NewFileSystemLabel "DataDisks" '
-AllocationUnitSize 65536 -Confirm:$false
```

Installing Microsoft SQL Server 2019

1. Attach the SQL Server 2019 installation media .ISO to the VM.
2. Click Run SETUP.EXE.
3. In the left pane, click Installation.
4. Click New SQL Server stand-alone installation or add features to an existing installation.
5. To accept the license terms, click the checkbox, and click Next.
6. Click Use Microsoft Update to check for updates, and click Next.
7. To install the setup support files, click Install.
8. If the installer doesn't display any failures, click Next.
9. At the Setup Role screen, choose SQL Server Feature Installation, and click Next.
10. At the Feature Selection screen, select Database Engine Services, Full-Text and Semantic Extractions for Search, Client Tools Connectivity, and Client Tools Backwards Compatibility.
11. Click Next.
12. At the Instance Configuration screen, leave the instance default selection, and click Next.
13. At the Database Engine Configuration screen, select the authentication method you prefer. For our testing purposes, we selected Mixed Mode.
14. For the system administrator account, enter and confirm a password.
15. Click Add Current user. This may take several seconds.
16. Click Data Directories.
17. Change the database directory, the database log directory, and the backup directory storage locations to the data, log, and backup volumes.
18. Click TempDB.
19. Change the number of files to eight.
20. Change the initial file size to 1,024 MB.
21. Change the initial size of the log to 1,024 MB.
22. Click Next.
23. At the Error and usage reporting screen, click Next.
24. At the Installation Configuration Rules screen, check that there are no failures or relevant warnings, and click Next.
25. At the Ready to Install screen, click Install.
26. Once the installation finishes, navigate to the Installation tab in the Installation Center, and click Install SQL Server Management Tools.
27. In the browser that opens, click Download SQL Server Management Studio 17.X.
28. To open the installer, click the download.
29. Click Run.
30. Click Install.
31. Once the installation completes, close the installation window.

Enabling the Windows option for locking pages in memory

1. In the Start menu, click Run. In the Open box, type `gpedit.msc`.
2. On the Local Group Policy Editor console, expand Computer Configuration, and expand Windows Settings.
3. Expand Security Settings, and expand Local Policies.
4. Select the User Rights Assignment folder.
5. In the details pane, double-click Lock pages in memory.
6. In the dialog box for Local Security Setting - Lock pages in memory, click Add User or Group.
7. In the dialog box for Select Users, Service Accounts, or Groups, select the SQL Server Service account.
8. For this setting to take effect, restart the SQL Server Service.

Setting the maximum memory in SQL Server Management Studio (AWS and on-premises solutions)

1. In SQL Server Management Studio, connect to the desired SQL Server database engine, right-click the desired instance, and click Properties.
2. From the left-side list in the Server Properties dialog box, select the Memory item.
3. In the Maximum server memory (in MB) option, type `707788`.
4. Click ok.

Configuring the test database

Creating the database

1. Open SQL Server Management Studio.
2. Right-click Databases→New Database.
3. Name the database. We named ours tpcc.
4. Navigate to the Filegroups tab, and create a new primary filegroup.
5. Navigate to the Files tab, and to add a data file, click Add.
6. Pregrow the data file to 3 TB.
7. Pregrow the log file to 96 GB by steps of 8 GB.
8. Name the database files, and click OK.

Installing HammerDB on the client system

1. Download the latest version of HammerDB from www.hammerdb.com/download.html. We used 4.2.
2. Double-click the .EXE file, choose English, and click OK.
3. Click Yes.
4. Click Next.
5. Choose a destination location, and click Next.
6. Click Next.
7. Click Finish.

Populating the database

1. Open HammerDB, and click Options→Benchmark.
2. Choose MSSQL Server and TPROC-C.
3. Expand SQL Server→TPROC-C→Schema Build.
4. Double-click Options.
5. In the SQL Server field, input the IP of the SUT.
6. Select 30,000 warehouses and 16 virtual users.
7. Click OK.
8. Double-click Build. This build could take a few days.

Backing up the database

1. Open SQL Server Management Studio.
2. Right-click the TPROC-C database, and click Tasks→Back up….
3. Choose a location to store the backup, and click OK.

Performing the test on Windows Server 2019

1. On the client system, start HammerDB.
2. Set the database server to SQL Server, and set the workload to TPROC-C.
3. Open the Options panel for the Driver Script: SQL Server→TPROC-C→Driver Script→Options.
4. In the SQL Server field, enter the IP of the system under test.
5. Change the ODBC Driver to ODBC Driver 17 for SQL Server.
6. Choose SQL Server Authentication, and change the SQL Server User Password to the password you chose during SQL Server setup.
7. Choose Timed Driver Script.
8. Change the Total Transactions per User to 1000000000.
9. Type `10` for the Ramp-up Time, and `20` for the Test Duration.
10. Select Use All Warehouses.
11. Check the box for Time Profile.
12. Click OK.
13. Open the Options panel for the Virtual Users: SQL Server→TPROC-C→Virtual User→Options.
14. Use 144 Virtual Users.
15. Select the following: Show Output, Log Output to Temp, and Use Unique Log Name.
16. Click OK.
17. To capture CPU, RAM, and disk performance counters, start a custom Perfmon data collector set.
18. Click the green arrow.
19. Note the name of the log file. To start the run, click OK.
20. When the run finishes, stop Perfmon.
21. Save the HammerDB results text file and Perfmon output.
22. Drop and restore the database from the system under test, reboot the system.
23. Complete steps 1 through 22 three times, and identify the median run.

Read the report at http://facts.pt/izcNe5y ▶

This project was commissioned by AWS.

Principled Technologies®

Facts matter.®