The science behind the report:

Google Cloud N2 instances featuring 3rd Gen Intel Xeon Scalable processors executed more MongoDB operations per second

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 Google Cloud N2 instances featuring 3rd Gen Intel Xeon Scalable processors executed more MongoDB operations per second.

We concluded our hands-on testing on October 27, 2022. 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 October 25, 2022 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

Instance type and processor	YCSB operations per second	Advantage with 3rd Gen Intel Xeon Platinum 8373C processor
8vCPU N2 instance with 3rd Gen Intel® Xeon® Platinum 8373C processor	48,414	1.28
8vCPU N2 instance with 2nd Gen Intel Xeon Gold 6268CL processor	37,811	N/A
16vCPU N2 instance with 3rd Gen Intel Xeon Platinum 8373C processor	92,294	1.26
16vCPU N2 instance 2nd Gen Intel Xeon Gold 6268CL processor	73,506	N/A
64vCPU N2 instance with 3rd Gen Intel Xeon Platinum 8373C processor	151,799	1.35
64vCPU N2 instance 2nd Gen Intel Xeon Gold 6268CL processor	112,704	N/A

System configuration information

Table 2: Detailed information on the Google Cloud N2 instances we tested.

System configuration information	8/16/64 3rd Gen Intel Xeon Scalable instance	8/16/64 2nd Gen Intel Xeon Scalable instance
Tested by	Principled Technologies	Principled Technologies
Test date	10/27/2022	10/27/2022
CSP / Region	Google Cloud Platform – Central US	Google Cloud Platform – Central US
Workload & version	YCSB 0.17.0 MongoDB 6.0	YCSB 0.17.0 MongoDB 6.0
WL specific parameters	80% read, 20% write 20/40/80 million records Three clients, 64 threads each	80% read, 20% write 20/40/80 million records Three clients, 64 threads each
Iterations and result choice	3 runs, median	3 runs, median
Server platform	With CPU set to Ice Lake: n2-standard-8 n2-standard-16 n2-standard-64	With CPU set to Cascade Lake: n2-standard-8 n2-standard-16 n2-standard-64
BIOS name and version	Google 1.0 10/11/2022	Google 1.0 10/11/2022
Operating system name and version/ build number	RHEL 9.0 5.14.0-70.26.1.el9_0.x86_64	RHEL 9.0 5.14.0-70.26.1.el9_0.x86_64
Date of last OS updates/ patches applied	10/25/2022	10/25/2022
Processor		
Number of processors	1-2	1-2
Vendor and model	Intel Xeon Platinum 8373C CPU @ 2.60GHz	Intel Xeon Gold 6268CL CPU @ 2.80GHz
Core count (per processor)	28	24
Core frequency (GHz)	2.6	2.8
Family, Model, Stepping	6, 106, 6	6, 85, 7
SMT	Yes	Yes
Turbo	Yes (3.4 GHz)	Yes (3.4 GHz)
Number of vCPU per instance	8/16/64	8/16/64
Memory module(s)		
Total memory in system (GB)	32/64/256	32/64/256
NVMe memory present?	No	No
Total memory (DDR+NVMe RAM)	32/64/256	32/64/256
General HW		
Storage: NW or Direct Att / Instance	Direct Att / Instance	Direct Att / Instance

System configuration information	8/16/64 3rd Gen Intel Xeon Scalable instance	8/16/64 2nd Gen Intel Xeon Scalable instance
Local storage		
OS		
Number of drives	1	1
Drive size (GB)	20	20
Drive information (speed, interface, type)	Balanced persistent disk	Balanced persistent disk
Data drive		
Number of drives	1	1
Drive size (GB)	200	200
Drive information (speed, interface, type)	SSD persistent disk	SSD persistent disk
Network adapter		
Vendor and model	Google Cloud Platform premium tier	Google Cloud Platform premium tier
Number and type of ports	1x 10Gb	1x 10Gb

How we tested

Testing overview

For this project, we tested Google Cloud instances featuring 3rd Gen Intel Xeon Scalable processors vs. instances featuring 2nd Gen Intel Xeon Scalable processors. We ran the YCSB MongoDB workload to measure performance in terms of total throughput. We created three database instances, a single config instance, and three client instances. Our results reflect what customers can expect to see using the current-generation series vs. the previous-generation series on Google Cloud.

Using our methodology to aid your own deployments

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

Creating the test environment

Creating the base instance

1. Log into the Google Cloud Platform and navigate to VM Instances.
2. Click Create Instance.
3. Give the instance a name.
4. Choose the geographical region for the instance. We used Central US 1a.
5. Select the N2 series and standard-8/16/64 machine type.
6. Under CPU platform, select Ice Lake or Cascade Lake.
7. Under Boot disk, click Change.
8. Set the OS and version to RHEL 9.0, and click Select.
9. Click Create.
10. Repeat steps 2 through 9 for a total of three database instances, one config instance, and three client instances.
11. When each database instance is ready, go to the instance page, and click Edit.
12. Click Add New Disk.
13. Give the disk a name.
14. Under Disk settings, select SSD persistent disk.
15. Under Size, enter 200GB.
16. Click Save.

Configuring RHEL 9

1. Via ssh, log into the instance.
2. Switch to the root user:

```
sudo su
```

3. Add the database, config, and client instance private IP addresses to the hosts file.
4. Turn off and disable the firewall:

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

5. Disable SELinux:

```
setenforce 0
vi /etc/selinux/config (modify "enforcing" to "disabled" in the file)
```

6. Update the OS:

```
yum update -y
yum upgrade -y
```

7. Set the tuned profile:

```
tuned-adm profile virtual-guest
```

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

```
fs.file-max=98000
kernel.pid_max=64000
kernel.threads-max=64000
vm.max_map_count=128000
```

9. Add the following lines to /etc/security/limits.conf:

```
* - nofile 64000
* - cpu unlimited
* - nproc 64000
* - fsize unlimited
* - memlock unlimited
* - as unlimited
* - rss unlimited
```

10. Create the file /etc/systemd/disable-transparent-huge-pages.service:

```
[Unit]
Description=Disable Transparent Huge Pages (THP)
DefaultDependencies=no
After=sysinit.target local-fs.target
Before=mongod.service
[Service]
Type=oneshot
ExecStart=/bin/sh -c 'echo never | tee /sys/kernel/mm/transparent_hugepage/enabled > /dev/null'
[Install]
WantedBy=basic.target
```

11. Enable the disable-transparent-huge-pages service:

```
systemctl start disable-transparent-huge-pages
systemctl enable disable-transparent-huge-pages
```

12. On the database instances, format and mount the data drive:

```
mkfs.xfs /dev/sdb
mkdir /data
mount /dev/sdb /data
mkdir /data/db
chown mongod:mongod /data/db
```

Installing MongoDB

1. Create the file /etc/yum.repos.d/mongodb-org.repo:

```
[mongodb-org-6.0]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/redhat/8/mongodb-org/6.0/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-6.0.asc
```

2. Set the crypto policy to legacy:

```
update-crypto-policies --set LEGACY
```

3. Install MongoDB:

```
yum install -y mongodb-org
```

4. Edit /etc/mongod.conf:

```
#Database servers:
storage:
  dbPath: /data/db
net:
  bindIp: hostname
replication:
  replSetName: <rs1, rs2, or rs3, one for each VM>
sharding:
  clusterRole: shardsvr

#Config server:

net:
  bindIp: hostname
replication:
  replSetName: cfg1
sharding:
  clusterRole: configsvr
```

5. Start MongoDB:

```
systemctl start mongodb
systemctl enable mongodb
```

Configuring and starting the cluster

1. On the config server, enable the config replication set:

```
mongosh
rs.initiate()
```

2. On each database server, enable the single-node replication set:

```
mongosh
rs.initiate()
```

3. On each database server, create the file ~/mongos.conf:

```
sharding:
  configDB: cfg1/<config server hostname>:27017
net:
  port: 27018
  bindIp: 127.0.0.1, <config server hostname>
```

4. On each database server, start the mongos process:

```
mongos --config ~/mongos.conf --fork --syslog
```

5. From a client instance, connect to a mongos:

```
mongosh --host <database VM hostname> --port 27018
```

6. Create the shards and index, and shard the database:

```
sh.addShard( "rs1/<database1 VM hostname>" )
sh.addShard( "rs2/<database2 VM hostname>" )
sh.addShard( "rs3/<database3 VM hostname>" )
use ycsb
db.usertable.createIndex( { _id: "hashed" } )
sh.shardCollection("ycsb.usertable", { _id : "hashed" } )
```

Running the tests

Installing and running YCSB

1. On the client servers, clone the YCSB repository:

```
git clone https://github.com/brianfrankcooper/YCSB
```

2. Create the file <YCSB directory>/workloads/my8020workload:

```
recordcount=<20, 40, or 80 million>
operationcount=10000000
workload=site.ycsb.workloads.CoreWorkload

readallfields=true

readproportion=.80
updateproportion=.20
scanproportion=0
insertproportion=0

requestdistribution=zipfian
```

3. Run the following command with 20, 40, or 80 million records to build the test database:

```
/home/ycsb-0.17.0/bin/ycsb load mongodb -s -P /home/ycsb-0.17.0/workloads/my8020workload -threads 64
-p mongodb.url=mongodb://<database1 VM hostname>:27018,<database2 VM hostname>:27018,<database3 VM
hostname>:27018/ycsb?w=1
 -p recordcount=<20, 40, or 80 million>
```

4. Once the database has finished building, reboot all instances, let them idle for 10 minutes, and run the following command on each YCSB client simultaneously, substituting for the target database instance:

```
/home/ycsb-0.17.0/bin/ycsb run mongodb -s -P /home/ycsb-0.17.0/workloads/my8020workload -threads 64
-p mongodb.url=mongodb://<target database VM hostname>:27018/ycsb?w=1
```

5. Repeat steps 3 and 4 three times and record the median score.

Read the report at https://facts.pt/C51W4Ai ▶

This project was commissioned by Intel.

Principled Technologies®

Facts matter.®