
The science behind the report:

# Upgrade to Google Cloud N4 instances featuring 5th Gen Intel Xeon Scalable processors and double Java server-side performance

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 Upgrade to Google Cloud N4 instances featuring 5th Gen Intel Xeon Scalable processors and double Java server-side performance.

We concluded our hands-on testing on May 31, 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 May 12, 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 SPECjbb2015 benchmark testing and price analysis.

| Benchmark results | Small instances (16 vCPUs) | Medium instances (32 vCPUs) | Large instances (64 vCPUs) |
|---|---|---|---|
| Critical-jOPS | | | |
| n1-standard | 9,414.00 | 33,608.00 | 63,015.00 |
| n4-standard | 19,601.00 | 55,512.00 | 127,903.00 |
| Maximum-jOPS | | | |
| n1-standard | 18,580.00 | 45,559.00 | 77,648.00 |
| n4-standard | 38,241.00 | 78,669.00 | 154,575.00 |
| Hourly cost | | | |
| n1-standard | $0.7599960 | $1.5199920 | $3.0399840 |
| n4-standard | $0.7581760 | $1.5163520 | $3.0327040 |
| Critical-jOPs per dollar | | | |
| n1-standard | 4,954.76 | 8,844.25 | 8,291.49 |
| n4-standard | 10,341.13 | 14,643.56 | 16,869.82 |
| Maximum-jOPs per dollar | | | |
| n1-standard | 9,778.99 | 11,989.27 | 10,216.89 |
| n4-standard | 20,175.26 | 20,752.17 | 20,387.74 |

# System configuration information

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

| System configuration information | Small instances (16 vCPUs) | Medium instances (32 vCPUs) | Large instances (64 vCPUs) |
|---|---|---|---|
| Date testing ended | 05/31/2024 | 05/31/2024 | 05/31/2024 |
| Cloud service provider (CSP) | Google Cloud Platform | Google Cloud Platform | Google Cloud Platform |
| Region | us-east1 (South Carolina) | us-east1 (South Carolina) | us-east1 (South Carolina) |
| Workload information | | | |
| Workload name and version | SPECjbb2015 v.1.03 | SPECjbb2015 v.1.03 | SPECjbb2015 v.1.03 |
| Iterations and result choice | 3 runs, median | 3 runs, median | 3 runs, median |
| Cloud instance details | | | |
| Number of VMs | 2 | 2 | 2 |
| Instance size (vCPUs) | 16 | 32 | 64 |
| BIOS name and version | Google BIOS Revision: 1.0 | Google BIOS Revision: 1.0 | Google BIOS Revision: 1.0 |
| vCPUs | 16 | 32 | 64 |
| Core/threads per VM | 8/16 | 16/32 | 32/64 |
| N1-standard VM memory (GB) | 60 | 120 | 240 |
| N4-standard VM memory (GB) | 64 | 128 | 256 |
| N1-standard processor | Intel® Xeon® CPU @ 2.00GHz | Intel Xeon CPU @ 2.00GHz | Intel Xeon CPU @ 2.00GHz |
| N4-standard processor | GENUINE INTEL 0000 | GENUINE INTEL 0000 | GENUINE INTEL 0000 |
| Virtual NUMA information | 1 Numa node | 1 Numa node | 1 Numa node |
| Operating system information | | | |
| Image or template name and UUID | Ubuntu 20.04 LTS x86/64, amd64 focal image built on 2024-05-19 | Ubuntu 20.04 LTS x86/64, amd64 focal image built on 2024-05-19 | Ubuntu 20.04 LTS x86/64, amd64 focal image built on 2024-05-19 |
| Operating system name | Ubuntu 20.04 LTS | Ubuntu 20.04 LTS | Ubuntu 20.04 LTS |
| Operating system build number | Ubuntu 22.04.4 LTS | Ubuntu 22.04.4 LTS | Ubuntu 22.04.4 LTS |
| Kernel version | 6.5.0-1020-gcp | 6.5.0-1020-gcp | 6.5.0-1020-gcp |
| Date patches last applied | 05/12/2024 | 05/12/2024 | 05/12/2024 |
| Changes made from CSP image | No | No | No |
| Instance storage | | | |
| Number of volumes | 1 | 1 | 1 |
| Volume unit | OS drive | OS drive | OS drive |
| N1-standard CSP volume type | Balanced persistent disk | Balanced persistent disk | Balanced persistent disk |
| N4-standard CSP volume type | Hyperdisk Balanced | Hyperdisk Balanced | Hyperdisk Balanced |
| Volume size (GB) | 26 | 26 | 26 |
| N1-standard IOPS requested | N/A | N/A | N/A |
| N4-standard IOPS requested | 3,000 | 3,000 | 3,000 |
| N1-standard throughput requested | N/A | N/A | N/A |
| N4-standard throughput requested | 140 | 140 | 140 |
| Encryption type | Google-managed | Google-managed | Google-managed |

# How we tested

## Creating VMs on Google Cloud Platform N1 and N4 instances

1. Navigate to Google Cloud.
2. Log in.
3. Click Console.
4. Click Create VM.
5. Add the following information:
   - Name: Name your VM instance
   - Labels: Use any appropriate labels
   - Region: Select your desired region. We used us-east1 (South Carolina).
   - Zone: Select your desired zone. We used us-east1b.
6. Machine Configuration:
   - Machine family: General-purpose
   - Series: N1 or N4
   - Machine type: Select the correct standard instance to match the required vCPU count
   - Keep Turn on display device unchecked
   - Keep Confidential VM Service and Container unchecked
   - Boot Disk, and click Change:
     - Operating System: Ubuntu
     - Version: Ubuntu 22.04 LTS
   - Boot disk type:
     - For N1 VMs: Balanced persistent disk
     - For N4 VMs: Hyperdisk Balanced
       - Size: 26GB.
       - Provisioned IOPs: Leave at 3,060
       - Provisioned Throughput: Leave at 155

   a. Click Select.
7. Identity and API access: Computer Engine default service account:
   - Access scopes, keep Allow default access as is
   - Firewall: Check Allow HTTP traffic, and Allow HTTPs traffic
   - Leave Observability - Ops Agent unchecked
   - Expand Networking, Disks, Security, Management, Sole-Tenancy
8. Click Add New Disk:
   - Name: Name the disk
   - Disk source type: Blank disk
   - Disk type:
     - For N1 VMs: Balanced persistent disk
     - For N4 VMs: Hyperdisk balanced
       - Size: 150GB, 250GB, and 450GB
       - Provisioned IOPS: default is 3,600, maximum is 75,000
       - Labels: Add appropriate label
       - Mode: Read/write
       - Deletion rule: Delete disk
9. Click Save.
10. Click Create.

## Installing and configuring Ubuntu 22.04.4

1. Boot from Ubuntu 22.04.4 media.
2. Click Try or Install Ubuntu Server.
3. At the language menu, leave the defaults, and click Done.
4. Click Update to the new installer.
5. At the keyboard configuration, leave the defaults, and click Done.
6. At the installation type screen, leave the defaults, and click Done.
7. At the network connections screen, leave the defaults, and click Done.
8. At the configure proxy screen, leave the defaults, and click Done.
9. At the configure Ubuntu archive mirror screen, wait for the test to pass, and click Done.
10. At the guided storage configuration screen, leave the defaults, and click Done.
11. At the storage configuration summary screen, leave the defaults, and click Done.
12. To confirm destructive action, click Continue.
13. At the profile setup screen, enter username, server name, and password.
14. Confirm password, and click Done.
15. At the upgrade to Ubuntu Pro screen, leave the defaults, and click Continue.
16. At the SSH setup screen, select Install OpenSSH server, and click Done.
17. At the featured server snaps screen, leave the defaults, and click Done.
18. When the installation is complete, click Reboot now.
19. To log into Ubuntu, use the credentials you created above.

## Configuring Ubuntu for the SPECjbb2015 workload

1. Type `sudo apt-get update -y`, and click enter.
2. Type `sudo apt-get upgrade -y`, and click enter.
3. Type `sudo reboot`, and click enter.
4. Type `Sudo apt install nmon`, and click enter.
5. Install OpenJDK:
    a. Type `sudo apt-get install openjdk-17-jdk`, and click enter.
6. Type `sudo apt update -y && sudo apt install -y numactl tuned linux-tools-common linux-tools-$(uname -r) psmisc font-manager`, and click enter.
7. For sudo passwd, type `Pass for root`, and click enter.

## Installing the SPECjbb2015 workload

1. Navigate to https://pro.spec.org/private/osg/incoming/
2. Login with the correct credentials.
3. Download the most up to date version of SPECjbb2015, at the time of testing this was SPECjbb2015 v1.3.
4. Copy the downloaded file to the test environment.
5. Run `sudo apt-get install` unzip.
6. In the current directory, unzip the SPECjbb2015 v1.3 zip file.
7. Configure SPECjbb2015.
8. Prior to running SPECjbb2015, configure multi.sh and specjbb2015.props.

## Changing the Run_multi.sh config:

```
# Number of Groups (TxInjectors mapped to Backend) to expect
GROUP_COUNT=`numactl --hardware | awk '/available/{print $2;exit}'`

# Number of TxInjector JVMs to expect in each Group
TI_JVM_COUNT=1
# Number of vCPUs
VCPU_COUNT=$(nproc)
# Size of memory (in GB) for each NUMA node
MEM_SIZE=`numactl --hardware | awk '/node 0 size/{printf("%i",$4/1000);exit}'`
BE_XMS=$((MEM_SIZE*85/100-4/GROUP_COUNT))
BE_XMN=$((BE_XMS-3))
JAVA_OPTS_C="-server -Xms2g -Xmx2g -Xmn1536m -XX:+UseLargePages -XX:LargePageSizeInBytes=2m
-XX:+UseParallelGC -XX:ParallelGCThreads=2"
JAVA_OPTS_TI="-server -Xms2g -Xmx2g -Xmn1536m -XX:+UseLargePages -XX:LargePageSizeInBytes=2m
-XX:+UseParallelGC -XX:ParallelGCThreads=2"
JAVA_OPTS_BE="-server -Xms${BE_XMS}G -Xmx${BE_XMS}G -Xmn${BE_XMN}G -XX:+UseLargePages
-XX:LargePageSizeInBytes=2m -XX:+UseParallelGC -XX:ParallelGCThreads=${VCPU_COUNT} -XX:+AlwaysPreTouch
-XX:-UseAdaptiveSizePolicy -XX:MaxTenuringThreshold=15 -XX:TargetSurvivorRatio=95 -XX:-UsePerfData
-Xnoclassgc -XX:SurvivorRatio=64 -XX:InitialCodeCacheSize=25m -XX:InlineSmallCode=10k"
numactl --interleave=all $JAVA $JAVA_OPTS_C $SPEC_OPTS_C -jar ../specjbb2015.jar -m MULTICONTROLLER
$MODE_ARGS_C 2>controller.log > controller.out &
numactl --cpunodebind=$((gnum-1)) --localalloc $JAVA $JAVA_OPTS_TI $SPEC_OPTS_TI -jar ../specjbb2015.jar
-m TXINJECTOR -G=$GROUPID -J=$JVMID $MODE_ARGS_TI > $TI_NAME.log 2>&1 &
numactl --cpunodebind=$((gnum-1)) --localalloc $JAVA $JAVA_OPTS_BE $SPEC_OPTS_BE -jar ../specjbb2015.jar
-m BACKEND -G=$GROUPID -J=$JVMID $MODE_ARGS_BE > $BE_NAME.log 2>&1 &
```

## Changing specjbb2015.props for 16vCPU instance size

```
specjbb.controller.type=HBIR_RT
specjbb.forkjoin.workers.Tier1=31
specjbb.forkjoin.workers.Tier2=1
specjbb.forkjoin.workers.Tier3=2
specjbb.group.count=1
specjbb.comm.connect.client.pool.size=232
specjbb.customerDriver.threads=75
specjbb.customerDriver.threads.probe=69
specjbb.customerDriver.threads.saturate=8
specjbb.mapreducer.pool.size=16
specjbb.comm.connect.selector.runner.count=4
specjbb.comm.connect.worker.pool.max=16
specjbb.comm.connect.worker.pool.min=1
specjbb.comm.connect.timeouts.connect=600000
specjbb.comm.connect.timeouts.read=600000
specjbb.comm.connect.timeouts.write=600000
```

## Changing specjbb2015.props for 32vCPU instance size

```
specjbb.controller.type=HBIR_RT
specjbb.forkjoin.workers.Tier1=63
specjbb.forkjoin.workers.Tier2=1
specjbb.forkjoin.workers.Tier3=4
specjbb.group.count=1
specjbb.comm.connect.client.pool.size=232
specjbb.customerDriver.threads=75
specjbb.customerDriver.threads.probe=69
specjbb.customerDriver.threads.saturate=8
specjbb.mapreducer.pool.size=32
specjbb.comm.connect.selector.runner.count=8
specjbb.comm.connect.worker.pool.max=32
specjbb.comm.connect.worker.pool.min=1
specjbb.comm.connect.timeouts.connect=600000
specjbb.comm.connect.timeouts.read=600000
specjbb.comm.connect.timeouts.write=60000
```

## Changing specjbb2015.props for 64vCPU instance size

```
specjbb.controller.type=HBIR_RT
specjbb.forkjoin.workers.Tier1=124
specjbb.forkjoin.workers.Tier2=1
specjbb.forkjoin.workers.Tier3=8
specjbb.group.count=1
specjbb.comm.connect.client.pool.size=232
specjbb.customerDriver.threads=75
specjbb.customerDriver.threads.probe=69
specjbb.customerDriver.threads.saturate=8
specjbb.mapreducer.pool.size=64
specjbb.comm.connect.selector.runner.count=16
specjbb.comm.connect.worker.pool.max=64
specjbb.comm.connect.worker.pool.min=1
specjbb.comm.connect.timeouts.connect=600000
specjbb.comm.connect.timeouts.read=600000
specjbb.comm.connect.timeouts.write=60000
```

## Running the SPECjbb2015 workload

1. To elevate user privileges to super user, choose `sudo su`.
2. Enter the following:

```
tuned-adm profile throughput-performance
systemctl stop systemd-update-utmp-runlevel.service
echo always → /sys/kernel/mm/transparent_hugepage/defrag
echo always → /sys/kernel/mm/transparent_hugepage/enabled
echo 56000000 → /sys/kernel/debug/sched/latency_ns
echo 180000 → /sys/kernel/debug/sched/min_granularity_ns
echo 750 → /sys/kernel/debug/sched/migration_cost_ns
echo 1000000 → /sys/kernel/debug/sched/wakeup_granularity_ns
echo 12 → /sys/kernel/debug/sched/nr_migrate
ulimit -n 1024000
ulimit -v 800000000
ulimit -m 800000000
ulimit -l 800000000
```

3. Run the adjusted `run_multi.sh` with `./run_multi.sh` and wait for the run to finish.
4. Once the run finishes, reboot the system, and repeat step 3 two more times, for three total runs.
5. Record median results.

**Read the report at https://facts.pt/tfNsf4Z** ▶

This project was commissioned by Intel.

Principled Technologies®

Facts matter.®