

---

**azure-cmaq**

**Liz Adams**

**Apr 23, 2024**



## CONTENTS:

<b>1</b>	<b>Overview of CMAQv5.4+ on Azure</b>	<b>3</b>
<b>2</b>	<b>Format of this documentation</b>	<b>5</b>
<b>3</b>	<b>Azure Subscriptions</b>	<b>7</b>
<b>4</b>	<b>Why might I need to use Azure Virtual Machine or CycleCloud?</b>	<b>9</b>
4.1	Introductory Tutorial . . . . .	9
4.2	System Requirements . . . . .	18
4.3	Create Single VM using HB120rs_v3 Tutorial . . . . .	24
4.4	Create CycleCloud HB120rs_v3 Cluster . . . . .	46
4.5	CMAQv5.4+ Benchmark on HBv3_120 compute nodes and beyond . . . . .	71
4.6	Scripts to run combine and post processing . . . . .	88
4.7	Scripts to post-process CMAQ output . . . . .	88
4.8	Install R, Rscript and Packages . . . . .	90
4.9	Install Anaconda on the /shared/build directory . . . . .	93
4.10	QA CMAQ . . . . .	93
4.11	Compare Timing of CMAQ Routines . . . . .	99
4.12	Copy Output to S3 Bucket . . . . .	101
4.13	Logout and Delete CycleCloud . . . . .	103
4.14	Performance Optimization . . . . .	103
4.15	Additional Resources . . . . .	117
4.16	Future Work . . . . .	119
4.17	Contribute to this Tutorial . . . . .	120
4.18	Optional instructions for Creating CycleCloud Cluster using /shared and /lustre . . . . .	120





**Warning:** This documentation is under continuous development. Previous version is available here: [CMAQv5.3.3 on Azure Tutorial](#)



## **OVERVIEW OF CMAQV5.4+ ON AZURE**

This document provides tutorials and information on using Microsoft Azure Online Portal to create either a single Virtual Machine or a Cycle Cloud Cluster to run CMAQ. The tutorials are aimed at users with cloud computing experience that are already familiar with Azure. For those with no cloud computing experience we recommend reviewing the Additional Resources listed in *chapter 15* of this document.



## **FORMAT OF THIS DOCUMENTATION**

This document provides three hands-on tutorials that are designed to be read in order. The Introductory Tutorial will walk you through setting up an Azure Account and logging into the Azure Portal Website. You will learn how to set up your Azure Resource ID, configure and create a demo virtual machine, and exit and delete the virtual machine and all of the resources associated with it by deleting resource group. The Intermediate Tutorial steps you through running a CMAQ test case on a single Virtual Machine with instructions to install CMAQ, libraries, and input data. The Advanced Tutorial explains how to create a CycleCloud (High Performance Cluster) for larger compute jobs and install CMAQ, required libraries and input data. The remaining sections provide instructions on post-processing CMAQ output, comparing output and runtimes from multiple simulations, and copying output from CycleCloud to an Amazon Web Services (AWS) Simple Storage Service (S3) bucket.



## **AZURE SUBSCRIPTIONS**

The ability to use resources available in the Microsoft Azure Cloud is limited by quotas that are set at the subscription level. This tutorial was developed using UNC Chapel Hill's Enterprise account. Additional effort is being made to identify how to use a pay-as-you-go account, but these instructions have not been finalized. There may also be differences in how managed identities and user level permissions are set by the administrator of your enterprise level account that are not covered in this tutorial.





## **WHY MIGHT I NEED TO USE AZURE VIRTUAL MACHINE OR CYCLECLOUD?**

An Azure Virtual Machine may be configured to run code compiled with Message Passing Interface (MPI) on a single high performance compute node. The intermediate tutorial demonstrates how to run CMAQ interactively on a single virtual machine running CMAQ with OpenMPI on multiple cpus.

The Azure CycleCloud may be configured to be the equivalent of a High Performance Computing (HPC) environment, including using job schedulers such as Slurm, running on multiple nodes/virtual machines using code compiled with Message Passing Interface (MPI), and reading and writing output to a high performance, low latency shared disk. The advantage of using the slurm scheduler is that the number of compute nodes that will be provisioned can be adjusted to meet requirements of a given simulation. In addition, the user can reduce costs by using Spot instances rather than On-Demand for the compute nodes. CycleCloud also supports submitting multiple jobs to the job submission queue.

Our goal is make this user guide to running CMAQ on either a single Virtual Machine or the CycleCloud Cluster as helpful and user-friendly as possible. Any feedback is both welcome and appreciated.

Additional information on Azure CycleCloud:

CycleCloud HPC Scalabilty

Azure CycleCloud

### **4.1 Introductory Tutorial**

Introductory Tutorial

#### **4.1.1 Create an Azure Account**

Create an account and configure your azure cyclecloud credentials. [Create Free Azure Account](#)

#### **4.1.2 Sign up for a Developer Azure Support Plan**

New accounts may be restricted to what virtual machines can be created by quota. With a pay-as-you go account or a free account, you need to sign up for the \$29.99 per month support account in order to create a support request to increase the quota limit for the HC44rs or the HBv120 machines that are used in this tutorial. With an enterprise account, the support plan is included.

4.1.3 Use Azure CLI to examine your quota

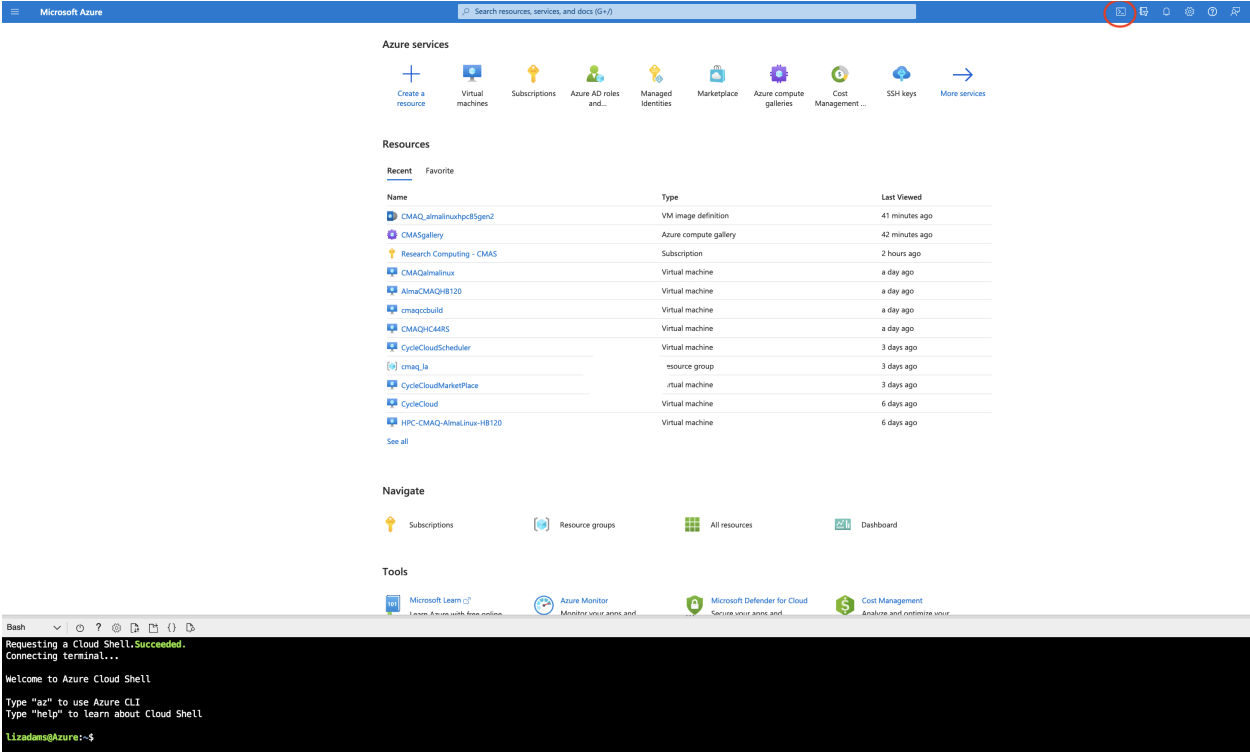
Login to the Azure Portal

In the upper right corner, click on the icon for “Cloud Shell”

You will be asked to create a storage account. According to the command `az help`, Cloud Shell requires an attached Azure file share in order to persist file changes in your \$HOME directory across sessions.

Azure Cloud Shell prompts you to create attached cloud storage

A new shell will be created at the bottom of your portal.



Enter the following at the command prompt to check your quota for the East US Region:

```
az vm list-usage --location "East US" -o table
```

Output:

Name	CurrentValue	Limit
-----	-----	-----
Availability Sets	0	2500
Total Regional vCPUs	0	10
Virtual Machines	0	25000
Virtual Machine Scale Sets	0	2500
Dedicated vCPUs	0	3000
Cloud Services	0	2500
Total Regional Low-priority vCPUs	0	3
Standard LSv3 Family vCPUs	0	0
Standard LASv3 Family vCPUs	0	0
Standard DPLDSv5 Family vCPUs	0	0
Standard DPLSv5 Family vCPUs	0	0

(continues on next page)

(continued from previous page)

Standard DPDSv5 Family vCPUs	0	0
Standard DPSv5 Family vCPUs	0	0
Standard EPDSv5 Family vCPUs	0	0
Standard EPSv5 Family vCPUs	0	0
Standard NCADS_A100_v4 Family vCPUs	0	0
Basic A Family vCPUs	0	10
Standard A0-A7 Family vCPUs	0	10
Standard A8-A11 Family vCPUs	0	10
Standard D Family vCPUs	0	10
Standard Dv2 Family vCPUs	0	10
Standard DS Family vCPUs	0	10
Standard DSv2 Family vCPUs	0	10
Standard G Family vCPUs	0	10
Standard GS Family vCPUs	0	10
Standard F Family vCPUs	0	10
Standard FS Family vCPUs	0	10
Standard NV Family vCPUs	0	12
Standard NC Family vCPUs	0	12
Standard H Family vCPUs	0	8
Standard Av2 Family vCPUs	0	10
Standard LS Family vCPUs	0	10
Standard Dv2 Promo Family vCPUs	0	10
Standard DSv2 Promo Family vCPUs	0	10
Standard MS Family vCPUs	0	0
Standard Dv3 Family vCPUs	0	10
Standard DSv3 Family vCPUs	0	10
Standard Ev3 Family vCPUs	0	10
Standard ESv3 Family vCPUs	0	10
Standard Dv4 Family vCPUs	0	10
Standard DDv4 Family vCPUs	0	10
Standard DSv4 Family vCPUs	0	10
Standard DDSv4 Family vCPUs	0	10
Standard Ev4 Family vCPUs	0	10
Standard EDv4 Family vCPUs	0	0
Standard ESv4 Family vCPUs	0	0
Standard EDSv4 Family vCPUs	0	10
Standard BS Family vCPUs	0	10
Standard FSv2 Family vCPUs	0	10
Standard NDS Family vCPUs	0	0
Standard NCSv2 Family vCPUs	0	0
Standard NCSv3 Family vCPUs	0	0
Standard LSv2 Family vCPUs	0	10
Standard PBS Family vCPUs	0	6
Standard EIV3 Family vCPUs	0	10
Standard EISv3 Family vCPUs	0	10
Standard DCS Family vCPUs	0	8
Standard NVSv2 Family vCPUs	0	0
Standard MSv2 Family vCPUs	0	0
Standard HBS Family vCPUs	0	0
Standard HCS Family vCPUs	0	0
Standard NVSv3 Family vCPUs	0	0
Standard NV Promo Family vCPUs	0	12

(continues on next page)

(continued from previous page)

Standard NC Promo Family vCPUs	0	12
Standard H Promo Family vCPUs	0	8
Standard DAv4 Family vCPUs	0	0
Standard DASv4 Family vCPUs	0	10
Standard EAv4 Family vCPUs	0	0
Standard EASv4 Family vCPUs	0	10
Standard NDSv3 Family vCPUs	0	0
Standard DCSv2 Family vCPUs	0	8
Standard NVSv4 Family vCPUs	0	8
Standard NDSv2 Family vCPUs	0	0
Standard NPS Family vCPUs	0	0
Standard HBrsv2 Family vCPUs	0	0
Standard NCASv3_T4 Family vCPUs	0	0
Standard NDASv4_A100 Family vCPUs	0	0
Standard EIDSv4 Family vCPUs	0	0
Standard XEISv4 Family vCPUs	0	0
Standard EIASv4 Family vCPUs	0	0
Standard HBv3 Family vCPUs	0	0
Standard MDSMediumMemoryv2 Family vCPUs	0	0
Standard MIDSMediumMemoryv2 Family vCPUs	0	0
Standard MSMediumMemoryv2 Family vCPUs	0	0
Standard MISMediumMemoryv2 Family vCPUs	0	0
Standard DASv5 Family vCPUs	0	0
Standard EASv5 Family vCPUs	0	0
Standard Ev5 Family vCPUs	0	0
Standard EIV5 Family vCPUs	0	0
Standard EDv5 Family vCPUs	0	0
Standard EIDv5 Family vCPUs	0	0
Standard ESv5 Family vCPUs	0	0
Standard EISv5 Family vCPUs	0	0
Standard EDSv5 Family vCPUs	0	0
Standard EIDSv5 Family vCPUs	0	0
Standard Dv5 Family vCPUs	0	0
Standard DDv5 Family vCPUs	0	0
Standard DSv5 Family vCPUs	0	0
Standard DDSv5 Family vCPUs	0	0
Standard DCSv3 Family vCPUs	0	0
Standard DDCSv3 Family vCPUs	0	0
Standard DADSv5 Family vCPUs	0	0
Standard EADSv5 Family vCPUs	0	0
Standard FXMDVS Family vCPUs	0	0
Standard NDAMSv4_A100Family vCPUs	0	0
Standard DCASv5 Family vCPUs	0	0
Standard ECASv5 Family vCPUs	0	0
Standard ECIASv5 Family vCPUs	0	0
Standard DCADSv5 Family vCPUs	0	0
Standard ECADSv5 Family vCPUs	0	0
Standard ECIADSv5 Family vCPUs	0	0
Standard NVADSA10v5 Family vCPUs	0	0
Standard EBDsv5 Family vCPUs	0	10
Standard EBSv5 Family vCPUs	0	10
Standard EIASv5 Family vCPUs	0	0

(continues on next page)

(continued from previous page)

Standard EIADSV5 Family vCPUs	0	0
Standard NCADSA10v4 Family vCPUs	0	0
Standard Storage Managed Disks	0	50000
Premium Storage Managed Disks	0	50000
StandardSSDStorageDisks	0	50000
StandardSSDZRSSStorageDisks	0	50000
PremiumZRSSStorageDisks	0	50000
UltraSSDStorageDisks	0	1000
PremiumV2StorageDisks	0	1000
StandardStorageSnapshots	0	75000
StandardSSDStorageSnapshots	0	75000
PremiumStorageSnapshots	0	75000
ZrsStorageSnapshots	0	75000
UltraSSDTotalsizeInGB	0	32768
PremiumV2TotalDiskSizeInGB	0	65536
DiskEncryptionSets	0	1000
DiskAccesses	0	1000
Gallery	0	100
Gallery Image	0	1000
Gallery Image Version	0	10000

### Review list of regions and virtual machines available in each region.

Azure Regions

### Follow the instructions in the link below to increase your vCPU quota to allow you to create a virtual machine and run CMAQ.

With a pay-as-you-go account this request to increase a quota for virtual machines may take 3-5 business days.

Azure Regions

Review the virtual machines available from each region

### Request a quota increase for the HTC Queue - HC Family of vCPUs for a region where they are available.

From the Azure Portal, search for quotas, select the Quotas Service. Click on Compute.

Use the Search Box to search for HC. Select the HC Standard Family vCPUs in the region nearest to your location. (For North Carolina select - East US) by clicking on the check box to the left of that selection.

Click on Request quota increase > Select Enter a new limit. In the sidebar menu on the right hand side, enter 44 in the text box under new limit.

### **Request a quota increase for the HPC Queue - HBv3 Family of vCPUs**

From the Azure Portal, search for quotas, select the Quotas Service. Click on Compute.

Use the Search Box to search for HB Select the HBv3 Family vCPUs in the region nearest to your location. (For North Carolina select - East US) by clicking on the check box to the left of that selection.

Click on Request quota increase > Select Enter a new limit. In the sidebar menu on the right hand side, enter 120 in the text box under new limit.

### **Request a quota increase for the scheduler node D4s\_v3**

From the Azure Portal, search for quotas, select the Quotas Service. Click on Compute.

Use the Search Box to search for Dv3 Select the Standard Dv3 Family vCPUs in the region nearest to your location. (For North Carolina select - East US) by clicking on the check box to the left of that selection.

Click on Request quota increase > Select Enter a new limit. In the sidebar menu on the right hand side, enter 4 in the text box under new limit to request an increase in the quota to 4 vcpu.

### **Request a quota increase for the F2sV2 HTC Compute Node (part of the Fsv2-series instances)**

From the Azure Portal, search for quotas, select the Quotas Service. Click on Compute.

Use the Search Box to search for Select the HBv3 Family vCPUs in the region nearest to your location. (For North Carolina select - East US) by clicking on the check box to the left of that selection.

Click on Request quota increase > Select Enter a new limit. In the sidebar menu on the right hand side, enter 44 in the text box under new limit.

## **4.1.4 Create a virtual machine**

Once your quota limit has been approved, then you will be able to select a virtual machine

From the Azure Portal Click on Create a resource

**Azure services**

Create a resource | Help + support | Quotas | Service Health | Monitor | Log Analytics workspaces | Virtual machines | HDInsight clusters | Subscriptions | More services

**Resources**

Recent | Favorite

Name	Type	Last Viewed
CMAQStandardD8sv4	Virtual machine	4 minutes ago
CMAQStandardD8sv4_group	Resource group	14 minutes ago
CMAQtest_group	Resource group	25 minutes ago
CMAQtest	Virtual machine	29 minutes ago
Azure subscription 1	Subscription	6 days ago
LogAnalyticsWorkspace	Log Analytics workspace	7 days ago
azure_resource_group	Resource group	7 days ago

See all

**Navigate**

Subscriptions | Resource groups | All resources | Dashboard

**Tools**

Microsoft Learn | Azure Monitor | Microsoft Defender for Cloud | Cost Management

**Useful links**

Technical Documentation | Azure Migration Tools | Azure Services | Find an Azure expert | Recent Azure Updates | Quickstart Center

**Azure mobile app**

Download on the App Store | GET IT ON Google Play

**The availability of Images that can be selected depends on the region.**

For high performance computing applications, the recommended operating system is Alma Linux 8 - Gen 2, but not all regions have Gen 2, so you may be limited to Gen 1.

### Enter Name, then Select Region, Select Image and Select Size

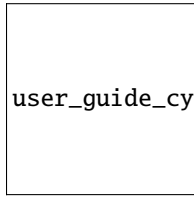
Recommend selecting a machine name that indicates what you will be using it for, the Operating System, and the Machine Size

Virtual Machine Name: CMAQAlmaD8sv3 Region: Central US Image: Alma Linux Gen 1 - first select see all images, then search on HPC, then select Alma Linux. If Gen 2 is available, then select that, if not, select Gen 1. Size: Standard\_D8\_v3 - first select see all sizes, then select an image that is large enough to run CMAQ CONUS Domain

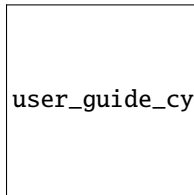




Select Next Disks Click on Create and attach new disk of size 1TB Click on Delete Disk with VM



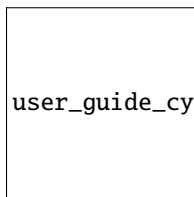
Click Next Networking - accept default settings Click Next Management - Select the System Managed Identity



Click on Review and Create - Click on Create

A pop-up window titled “Generate a new key pair” will appear.

Click on Download private key and create resource.



Once your resource is created in the upper right corner of the new screen, there will be a menu option titled Connect.

Select Connect then SSH

The screen will then provide instructions for you to login to the newly created virtual machine.

### 4.1.5 Login to Virtual Machine

```
ssh -i ./CMAQStandardD8sv4_key.pem azureuser@13.89.128.245
```

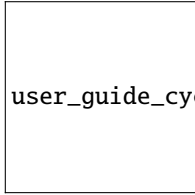
**Verify that the 1024 GiB size disk is not listed as being available**

```
df -h
```

In the intermediate tutorial, instructions are provided to find the disk and mount it as a /shared volume to the virtual machine.

### 4.1.6 Delete the virtual machine and all of the associated resources by deleting the resource group.

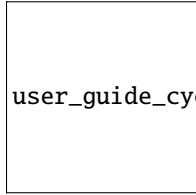
Deleting the resource group will delete the virtual machine and will also delete the associated resources that were



user\_guide\_cyclecloud/cyclecloud-cmaq/docs/azure\_web\_interface\_images/Virtual\_r

created for the virtual machine.

A pop-up window will appear on the right side of the Azure Portal to confirm that you want to delete the resource group.



user\_guide\_cyclecloud/cyclecloud-cmaq/docs/azure\_web\_interface\_images/Virtual\_machine\_click\_on\_resource

This tutorial was developed using UNC's enterprise account. It is unknown if Azure will grant access to these virtual machines on a credit card account.

## 4.2 System Requirements

Description of the compute node and head nodes used for the CycleCloud

### 4.2.1 System Requirements for a Single Virtual Machine or Cycle Cloud Cluster

#### Please set up a alarm on Azure

Set alarm to receive an email alert if you exceed \$100 per month (or what ever monthly spending limit you need). It may be possible to set up daily or weekly spending alarms as well.

#### Azure Documentation on selecting the right VM for your workloads

Description of Azure Virtual Machines

For CMAQ, it is recommended that the user select a High Performance Compute Virtual Machine.

Type	Sizes	Description
General purpose	B, Dsv3, Dv3, Dasv4, Dav4, DSv2, Dv2, Av2, DC, DCv2, Dpdsv5, Dpldsv5, Dpsv5, Dplsv5, Dv4, Dsv4, Ddv4, Ddsv4, Dv5, Dsv5, Ddv5, Ddsv5, Dasv5, Dadsv5, DCasv5, DCadsv5	Balanced CPU-to-memory ratio. Ideal for testing and development, small to medium databases, and low to medium traffic web servers.
Compute optimized	F, Fs, Fsv2, FX	High CPU-to-memory ratio. Good for medium traffic web servers, network appliances, batch processes, and application servers.
Memory optimized	Esv3, Ev3, Easv4, Eav4, Epdsv5, Epsv5, Ev4, Esv4, Edv4, Edsv4, Ev5, Esv5, Edv5, Edsv5, Easv5, Eadsv5, Mv2, M, DSv2, Dv2, ECasv5, ECadsv5	High memory-to-CPU ratio. Great for relational database servers, medium to large caches, and in-memory analytics.
Storage optimized	Lsv2, Lsv3, Lasv3	High disk throughput and IO ideal for Big Data, SQL, NoSQL databases, data warehousing and large transactional databases.
GPU	NC, NCv2, NCv3, NCasT4_v3, NC A100 v4, ND, NDv2, NGads V620, NV, NVv3, NVv4, NDasrA100_v4, NDm_A100_v4	Specialized virtual machines targeted for heavy graphic rendering and video editing, as well as model training and inferencing (ND) with deep learning. Available with single or multiple GPUs.
High performance compute	HB, HBv2, HBv3, HBv4, HC, HX	Our fastest and most powerful CPU virtual machines with optional high-throughput network interfaces (RDMA).

### 4.2.2 Software Requirements for CMAQ on Single VM or CycleCloud Cluster

The software requirements to run CMAQ on Azure are split into three tiers. The first tier includes the software that is provided with the operating system, the second tier includes the libraries required by CMAQ, the third tier includes the CMAQ code and associated pre and post processors, and the third tier includes the R software and packages required by the analysis scripts for verifying output or doing a quality assurance of CMAQ.

Tier 1: Native Operating System (OS) and associated system libraries, compilers for both Single VM or Cycle Cloud Cluster

- Tcsh shell
- Alma Linux Gen. 2
- Git
- Compilers (C, C++, and Fortran) - GNU compilers version gcc (GCC) 9.2.0 (need to use module load gcc-9.2.0)
- MPI (Message Passing Interface) - OpenMPI 4.1.0 (need to use module load mpi/openmpi-4.1.0)

Tier 1: For the Cycle Cloud Cluster

- Slurm Scheduler

Tier 2: additional libraries required for installing CMAQ

- NetCDF (with C, C++, and Fortran support)
- I/O API

Tier 3: Software distributed thru the CMAS Center

- CMAQv5.4+
- CMAQv5.4+ Post Processors

Tier 4: R packages and Scripts

- R QA Scripts

## Hardware Requirements

### Recommended Minimum Requirements

The size of hardware depends on the domain size and resolution for your CMAQ case, and how quickly your turn-around requirements are. Larger hardware and memory configurations are also required for instrumented versions of CMAQ including CMAQ-ISAM and CMAQ-DDM3D.

### Azure Single Virtual Machine

Azure offers generalized, compute, and high performance machines of various sizes. The amount of memory and the number of cpus required to run CMAQ depends on the domain size and resolution of the case that is being run. For this tutorial that uses a two day run of the CONUS2 domain, a minimum size recommended is a HC44rs (44 cpus) or HBv120 (120 cpus) compute node, to allow CMAQ to be run on up to 44 or 120 cpus.

#### HC Series Virtual Machine Overview

Physically, an HC-series server is 2 \* 24-core Intel Xeon Platinum 8168 CPUs for a total of 48 physical cores. Each CPU is a single pNUMA domain, and has unified access to six channels of DRAM. Intel Xeon Platinum CPUs feature a 4x larger L2 cache than in prior generations (256 KB/core -> 1 MB/core), while also reducing the L3 cache compared to prior Intel CPUs (2.5 MB/core -> 1.375 MB/core).

The above topology carries over to the HC-series hypervisor configuration as well. To provide room for the Azure hypervisor to operate without interfering with the VM, we reserve pCores 0-1 and 24-25 (that is, the first 2 pCores on each socket). We then assign pNUMA domains all remaining cores to the VM. Thus, the VM will see:

```
(2 vNUMA domains) * (22 cores/vNUMA) = 44 cores per VM
```

#### HBv3-series Specification

“An HBv3-series server features 2 \* 64-core EPYC 7V73X CPUs for a total of 128 physical “Zen3” cores with AMD 3D V-Cache. Simultaneous Multithreading (SMT) is disabled on HBv3. 448 GB of RAM, and no hyperthreading with 350 GB/sec of memory bandwidth, up to 32 MB of L3 cache per core, up to 7 GB/s of block device SSD performance, and clock frequencies up to 3.675 GHz.” Quote from above link

#### HBv4-series Specification.

“An HBv4-series server features 2 \* 96-core EPYC 9V33X CPUs for a total of 192 physical “Zen4” cores with AMD 3D-V Cache. Simultaneous Multithreading (SMT) is disabled on HBv4. These 192 cores are divided into 24 sections (12 per socket), each section containing 8 processor cores with uniform access to a 96 MB L3 cache.” Quote from above link

## Azure CycleCloud Cluster

Azure CycleCloud Provides the simplest way to manage HPC workloads using any scheduler (like Slurm, Grid Engine, HPC Pack, HTCondor, LSF, PBS Pro, or Symphony).

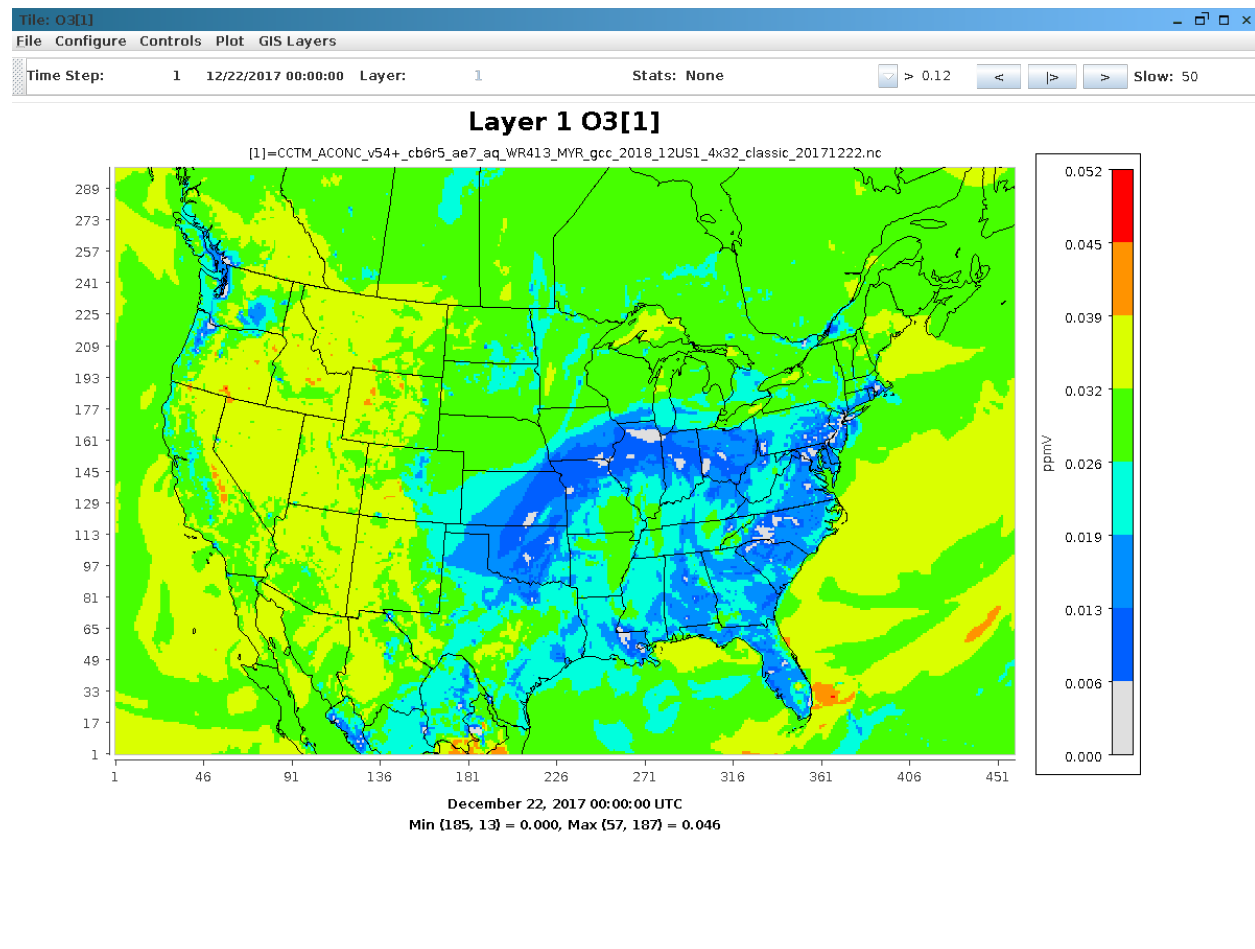
CycleCloud allows you to:

- Deploy full clusters and other resources, including scheduler, compute VMs, storage, networking, and cache
- Orchestrate job, data, and cloud workflows
- Give admins full control over which users can run jobs, as well as where and at what cost
- Customize and optimize clusters through advanced policy and governance features, including cost controls, Active Directory integration, monitoring, and reporting
- Use your current job scheduler and applications without modification
- Take advantage of built-in autoscaling and battle-tested reference architectures for a wide range of HPC workloads and industries

Azure CycleCloud

## 12US1 Benchmark Domain Description

```
GRIDDESC
'12US1'
'LAM_40N97W' -2556000. -1728000. 12000. 12000. 459 299 1
```



### 4.2.3 Storage Options

CMAQ requires low-latency storage, especially if you are running CMAQ on a large domain and using more than 200 processors.

Azure File Storage account for premium file shares is required.

Quote from following link: “Provisioned file shares can be dynamically scaled up or down depending on your storage and IO performance characteristics. The provisioned size of the file share can be increased at any time but can be decreased only after 24 hours since the last increase. After waiting for 24 hours without a quota increase, you can decrease the share quota as many times as you like, until you increase it again. IOPS/throughput scale changes will be effective within a few minutes after the provisioned size change.”

Lustre Managed Server will be available to the public in March 2023. Files can be stored on blob storage and linked to the Lustre server and re-hydrated when needed as input for CMAQ.

Azure Premium File Shares

#### 4.2.4 Recommended Cycle Cloud Configuration for CONUS Domain 12US1

Note, first create a VM using the image: CycleCloud 8.2, and from that VM, the Cycle Cloud is built. VM:

\*F4sV2 (4vcpus, 8 GiB memory) - VM image: CycleCloud 8.2

CycleCloud Configuration:

Scheduler node:

- D4s\_v3

Compute Node for HTC Queue - used for Post-Processing (combine, etc):

- F2sV2 (part of the Fsv2-series instances)

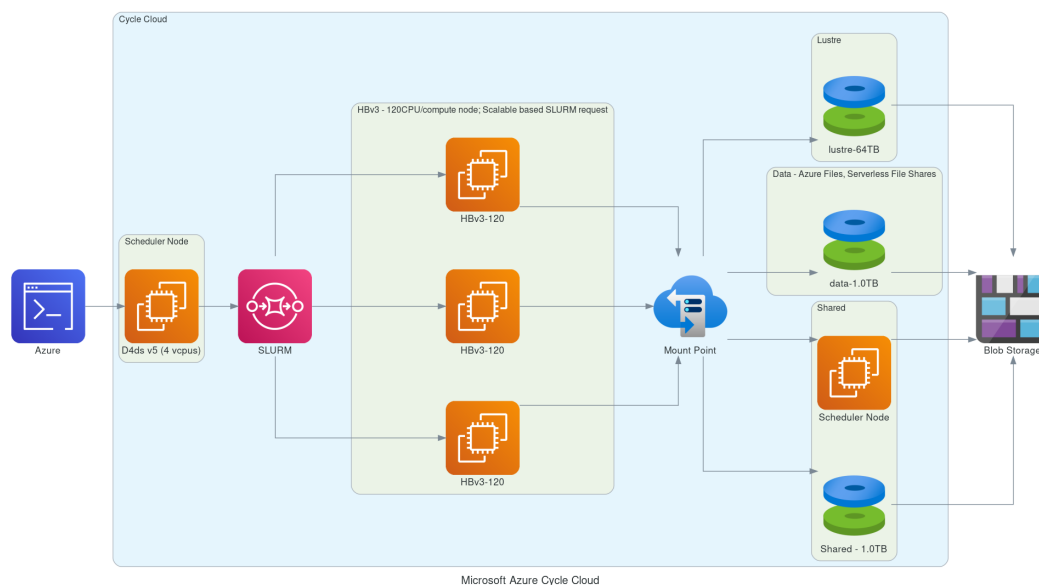
Compute Node for HPC Queue - used to run CMAQ:

- HBv3-120 instance running AlmaLinux

HBv3-series Software Specification

448 GB of RAM, and no hyperthreading with 350 GB/sec of memory bandwidth, up to 32 MB of L3 cache per core, up to 7 GB/s of block device SSD performance, and clock frequencies up to 3.675 GHz.

Figure 1. Cycle Cloud Recommended Cluster Configuration (Number of compute nodes depends on setting for NPCOLxNPROW and #SBATCH -nodes=XX #SBATCH -ntasks-per-node=YY )



### Azure CycleCloud specifies what resource to use for disks, scheduler node, and compute nodes.

Cycle Cloud simply tries to schedule the job according to the slurm scheduler instructions. Slurm controls the launch, terminate, and maintain resources. If you try to allocate more nodes than are available in the Cycle Cloud Configuration, then you will need to edit the HPC config in the cyclecloud web interface to set the CPUs to 480 or more and then run the following on the scheduler node the changes should get picked up:

```
cd /opt/cycle/slurm
sudo ./cyclecloud_slurm.sh scale
```

Number of compute nodes dispatched by the slurm scheduler is specified in the run script using #SBATCH --nodes=XX #SBATCH --ntasks-per-node=YY where the maximum value of tasks per node or YY limited by many CPUs are on the compute node.

For HBv3-120, there are 120 CPUs, so maximum value of YY is 120 or --ntask-per-node=120.

If running a job with 180 processors, this would require the --nodes=XX or XX to be set to 2 compute nodes, as  $90 \times 2 = 180$ .

The setting for NPCOLxNPROW must also be a maximum of 180, ie. 18 x 10 or 10 x 18 to use all of the CPUs in the CycleCloud HPC Node.

HBv3-120 instance

Software:

- Alma Linux
- Spot or OnDemand Pricing
- /shared/build volume install software from git repo
- 1. TB Shared file system
- Slurm Placement Group enabled
- Elastic Fabric Adapter Enabled on HBv3-120

## 4.3 Create Single VM using HB120rs\_v3 Tutorial

Run CMAQ on a single Virtual Machine (VM) using HBv120 and AlmaLinux HPC - Gen2.

Intermediate Tutorial: Run CMAQ from HBv120 Compute Node for CMAQv5.4

Estimated workflow time: 2-3 hours for the complete install input data, build, and run.

Instructions are provided to build and install CMAQ on HBv120 compute node installed from HPC AlmaLinux 8.7 HPC-Gen2 Image that contains modules for git, openmpi and gcc.

The compute node does not have a SLURM scheduler on it, so jobs are run interactively from the command line.

Instructions to install data and CMAQ libraries and model are provided along with sample run scripts to run CMAQ on 96 processors on a single HB120\_v3 (120 cpus) instance.

This will provide users with experience using the Azure Portal to create a Virtual Machine, select AlmaLinux 8.7 HPC - Gen2 as the image, select the size of the VM as HB120rs\_v3 - 120 vcpus, 456 GiB memory, using an SSH private key to login and install and run CMAQ.



**Warning:** Using this method, the user needs to be careful to start and stop the Virtual Machine and only have it run while doing the initial installation, and while running CMAQ. The full HBv120 instance will incur charges as long as it is on, even if a job isn't running on it. This is different than the Azure Cycle-Cloud, where if CMAQ is not running in the queue, then the HBv120 Compute nodes are down, and not incurring costs.

### 4.3.1 Create a HB120rs\_v3 Virtual Machine

1. Login to Azure Portal
2. Select Create a Virtual Machine
3. Click on See all images next to Image and use the search bar to search for HPC. Look for the AlmaLinux 8.7 HPC. Select Gen 2, and click. That option should now pre-populate the form.
4. Select Size - Standard\_HB120rs\_v3 - 120 vcpus, 456 GiB memory (\$2,628.0/monthly)
5. Enter a Virtual Machine Name in the text box
6. Use your username or azureuser
7. Select Authentication type - SSH public key
8. Select SSH public key source - Generate new key pair

Microsoft Azure

Se

[Home](#) > [Virtual machines](#) >

## Create a virtual machine ...

Changing Basic options may reset selections you have made. Review all options prior to creating the virtual machine.

[Basics](#)
[Disks](#)
[Networking](#)
[Management](#)
[Monitoring](#)
[Advanced](#)
[Tags](#)
[Review + create](#)

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. [Learn more](#)

### Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \*

researchComputing-CMAS

Resource group \*

(New) hb120v3\_group

[Create new](#)

### Instance details

Virtual machine name \*

hb120v3

Region \*

(US) East US

Availability options

Availability zone

Availability zone \*

Zones 3

You can now select multiple zones. Selecting multiple zones will create one VM per zone. [Learn more](#)

Security type

Standard

Image \*

AlmaLinux OS 8.7 HPC (x86\_64) - x64 Gen2

[See all images](#) | [Configure VM generation](#)

VM architecture

☐ Arm64
☒ x64

Arm64 is not supported with the selected image.

Run with Azure Spot discount

☐

Size \*

Standard\_HB120rs\_v2 - 120 vcpus, 456 GiB memory (\$2,628.00/month)

[See all sizes](#)

Enable Hibernation (preview)

☐

To enable Hibernation, you must register your subscription. [Learn more](#)

### Administrator account

Authentication type

☒ SSH public key
☐ Password

Azure now automatically generates an SSH key pair for you and allows you to store it for future use. It is a fast, simple, and secure way to connect to your virtual machine.

Review + create

< Previous

Next : Disks >

To find the HB120rs\_v3 size, click on “see all sizes” Then search for hb120 The select “Other Sizes” And then select HB120rs\_v3

For some reason, the instance isn’t available under the “H series” only the HB120rs\_v2 is available there.

Home > Create a virtual machine >

Select a VM size

hb120 vCPUs: All RAM (GiB): All Display cost: Monthly Add filter

Showing 10 of 792 VM sizes | Subscription: researchComputing-CMAS | Region: East US | Current size: Standard\_HB120rs\_v3 | Image: AlmaLinux OS 8.7 HPC (x86\_64) | Learn more about VM sizes

VM Size	Type	vCPUs	RAM (GiB)	Data disks	Max IOPS	Local storage (GiB)	Premium disk	Cost/month
H-Series								
High performance compute VMs								
HB120-16rs_v2	High performance compute	16	456	32	80000	894 (NVMe)	Supported	\$2,628.00
HB120-32rs_v2	High performance compute	32	456	32	80000	894 (NVMe)	Supported	\$2,628.00
HB120-64rs_v2	High performance compute	64	456	32	80000	894 (NVMe)	Supported	\$2,628.00
HB120-96rs_v2	High performance compute	96	456	32	80000	894 (NVMe)	Supported	\$2,628.00
HB120rs_v2	High performance compute	120	456	32	80000	894 (NVMe)	Supported	\$2,628.00
Other sizes								
HB120-16rs_v3	High performance compute	16	456	32	80000	1788 (NVMe)	Supported	\$2,628.00
HB120-32rs_v3	High performance compute	32	456	32	80000	1788 (NVMe)	Supported	\$2,628.00
HB120-64rs_v3	High performance compute	64	456	32	80000	1788 (NVMe)	Supported	\$2,628.00
HB120-96rs_v3	High performance compute	96	456	32	80000	1788 (NVMe)	Supported	\$2,628.00
HB120rs_v3	High performance compute	120	456	32	80000	1788 (NVMe)	Supported	\$2,628.00

Click on Next > Disks

1. Click on Create and attach a new disk - select a 1TB disk
2. Select Checkbox to Delete disk with VM

Microsoft Azure

Search

[Home](#) > [Virtual machines](#) >

Create a virtual machine ...

Basics

Disks

Networking

Management

Monitoring

Advanced

Tags

Review + create

Azure VMs have one operating system disk and a temporary disk for short-term storage. You can attach additional data disks. The size of the VM determines the type of storage you can use and the number of data disks allowed. [Learn more](#)

**VM disk encryption**

Azure disk storage encryption automatically encrypts your data stored on Azure managed disks (OS and data disks) at rest by default when persisting it to the cloud.

Encryption at host ☐

Encryption at host is not registered for the selected subscription. [Learn more about enabling this feature](#)

**OS disk**

OS disk size

OS disk type \*

Delete with VM ☒

Key management

Enable Ultra Disk compatibility ☐

**Data disks for hb120v3**

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching	Delete with VM
0	hb120v3_DataDisk_0	1024	Premium SSD LRS	Read-only	<input checked="" type="checkbox"/>

[Create and attach a new disk](#)
[Attach an existing disk](#)

Advanced


Review + create

< Previous

Next : Networking >

(note, this will create the disk, but you will need to login and mount the disk as the shared volume following the instructions below.)

### 3. Create new Disk


**Microsoft Azure**

[Home](#) > [Virtual machines](#) > [Create a virtual machine](#) >

## Create a new disk ...

Create a new disk to store applications and data on your VM. Disk pricing varies based on factors including disk size, storage type, and number of transactions. [Learn more](#)

Name *	<input type="text" value="hb120v3_DataDisk_0"/>
Source type * ⓘ	<input type="text" value="None (empty disk)"/>
Size * ⓘ	<div> <b>1024 GiB</b>            Premium SSD LRS  <a href="#">Change size</a> </div>
Key management ⓘ	<input type="text" value="Platform-managed key"/>
Enable shared disk	<input type="radio"/> Yes <input checked="" type="radio"/> No
Delete disk with VM	<input type="checkbox"/>

Click on Next > Management

1. Select check box for Identity > System assigned managed identity

Microsoft Azure

[Home](#) > [Virtual machines](#) >

## Create a virtual machine

Basics

Disks

Networking

Management

Advanced

Tags

Review + create

Configure monitoring and management options for your VM.

### Azure Security Center

Azure Security Center provides unified security management and advanced threat protection across hybrid cloud workloads. [Learn more](#)

✔ Your subscription is protected by Azure Security Center basic plan.

### Monitoring

Boot diagnostics <sup>①</sup> ☒ Enable with managed storage account (recommended)  
☐ Enable with custom storage account  
☐ Disable


Enable OS guest diagnostics <sup>①</sup> ☐

### Identity

System assigned managed identity <sup>①</sup> ☒

### Azure AD

Login with Azure AD <sup>①</sup> ☐

 This image does not support Login with Azure AD.

### Auto-shutdown


Enable auto-shutdown <sup>①</sup> ☐

### Backup

Enable backup <sup>①</sup> ☐

### Guest OS updates

Patch orchestration options <sup>①</sup>

 Some patch orchestration options are not available for this image. [Learn more](#)

Review + create

< Previous

Next : Advanced >

Click on Next > Advanced

don't need to change anything

# Create a virtual machine ...

Basics   Disks   Networking   Management   **Advanced**   Tags   Review + create

Add additional configuration, agents, scripts or applications via virtual machine extensions or cloud-init.

## Extensions

Extensions provide post-deployment configuration and automation.

Extensions ⓘ

[Select an extension to install](#)

## VM applications (preview)

VM applications contain application files that are securely and reliably downloaded on your VM after deployment. In addition to the application files, an install and uninstall script are included in the application. You can easily add or remove applications on your VM after create. [Learn more](#) ↗

[Select a VM application to install](#)

## Custom data and cloud init

Pass a cloud-init script, configuration file, or other data into the virtual machine **while it is being provisioned**. The data will be saved on the VM in a known location. [Learn more about custom data for VMs](#) ↗

Custom data

ⓘ Custom data on the selected image will be processed by cloud-init. [Learn more about custom data for VMs](#) ↗

## User data

Pass a script, configuration file, or other data that will be accessible to your applications **throughout the lifetime of the virtual machine**. Don't use user data for storing your secrets or passwords. [Learn more about user data for VMs](#) ↗

Enable user data

☐

## Host

Azure Dedicated Hosts allow you to provision and manage a physical server within our data centers that are dedicated to your Azure subscription. A dedicated host gives you assurance that only VMs from your subscription are on the host, flexibility to choose VMs from your subscription that will be provisioned on the host, and the control of platform maintenance at the level of the host. [Learn more](#) ↗

Host group ⓘ

No host group found



## Capacity reservations

Capacity reservations allow you to reserve capacity for your virtual machine needs. You get the same SLA as normal virtual machines with the security of reserving the capacity ahead of time. [Learn more](#) ↗

[Review + create](#)

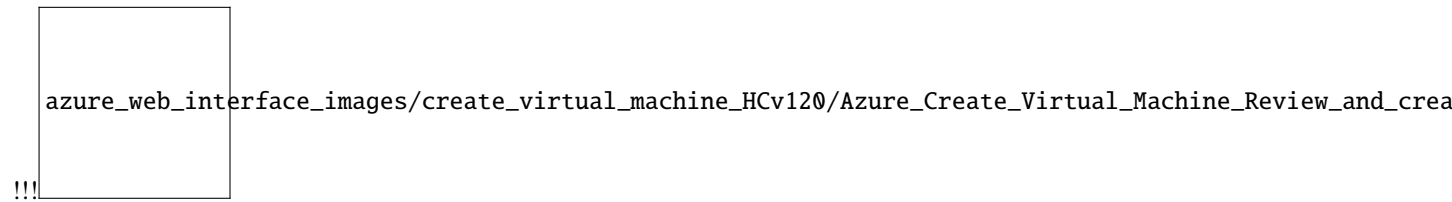
[< Previous](#)

[Next : Tags >](#)

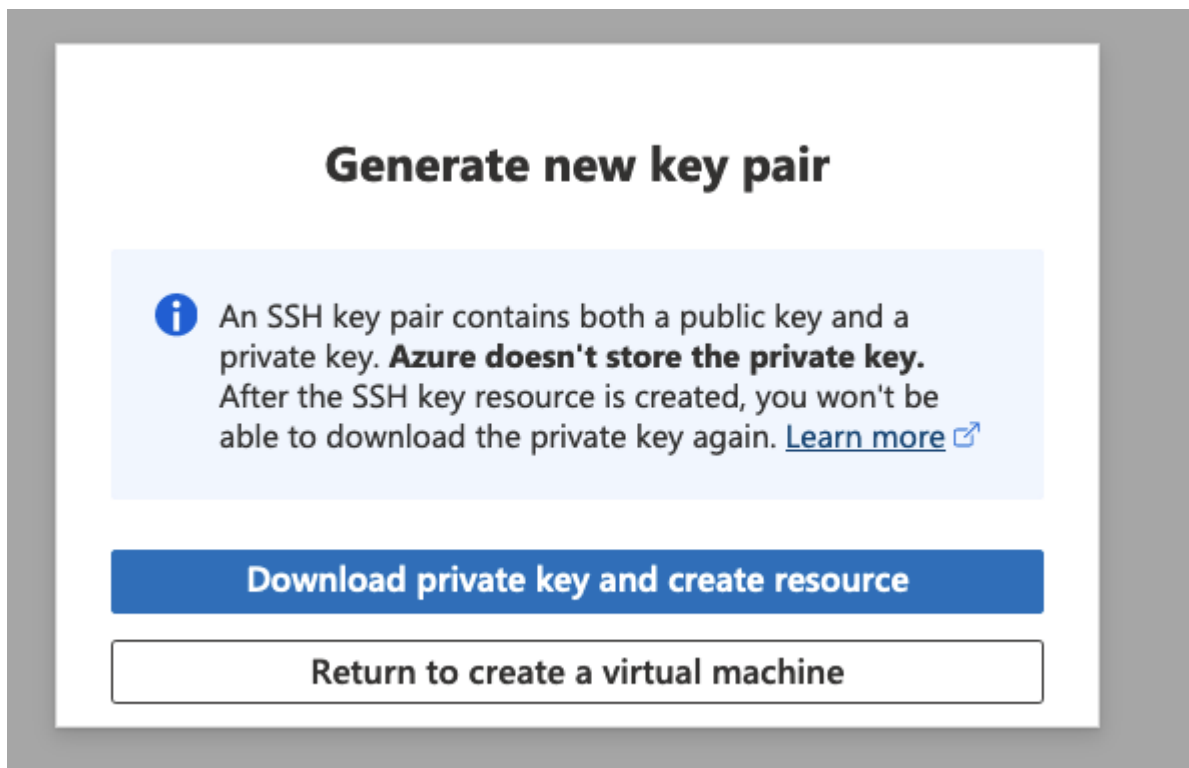
Click on Next > Tags

don't change anything

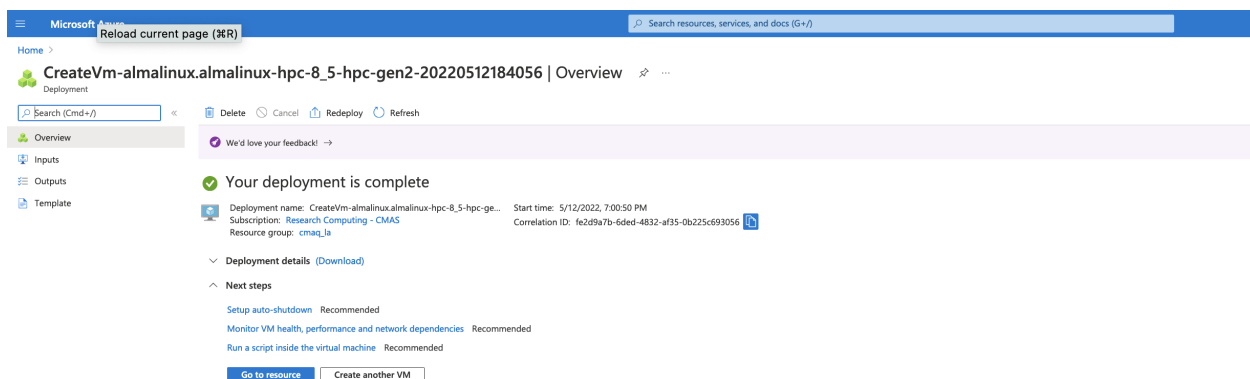
Click on Next > Review and create



Click on download private key and provision resource



Click on Go to Resource once the deployment is completed.





Click on Go to resource to get the IP address.

### 4.3.2 Login to the Virtual Machine

Change the permissions on the public key using command

```
chmod 400 HPC-CMAQ-AlmaLinux-HB120_key.pem
```

Login to the Virtual Machine using ssh to the IP address using the public key.

```
ssh -Y -i ./xxxxxxx_key.pem username@xx.xx.xx.xx
```

### 4.3.3 Mount the disk on the server as /shared using the instructions on the following link:

Mount Disk on Azure Linux Virtual Machine

#### Find the disk

```
lsblk -o NAME,HCTL,SIZE,MOUNTPOINT | grep -i "sd"
```

Output:

```
sda      1:0:0:0      1T
sdb      0:0:0:0      60G
├─sdb1           1000M /boot
├─sdb2           58.5G /
├─sdb14          4M
└─sdb15          495M /boot/efi
sdc      0:0:0:1      480G
└─sdc1           480G /mnt
```

In the above case, the 1 Terrabyte (1T) disk was added as sdc

#### Format the disk

```
sudo parted /dev/sda --script mklabel gpt mkpart xfs xfs 0% 100%
sudo mkfs.xfs /dev/sda1
sudo partprobe /dev/sda1
```

## Mount the disk

```
sudo mkdir /shared
```

## Use mount to mount the filesystem

```
sudo mount /dev/sda1 /shared
```

## Persist the mount

grep sda1 UUID using the command:

```
sudo blkid | grep -i sda1
```

Output

```
/dev/sda1: UUID="97ac20d7-b884-4671-9619-7248b529975c" BLOCK_SIZE="4096" TYPE="xfs"  
↪PARTLABEL="xfspart" PARTUUID="a25cfda9-3d62-42cf-86c2-5fba2f6fa440"
```

## Edit fstab

Next, open the /etc/fstab file in a text editor as follows:

```
sudo nano /etc/fstab
```

In this example, use the UUID value for the /dev/sdc1 device that was created in the previous steps, and the mountpoint of /shared. Add the following line to the end of the /etc/fstab file:

```
UUID=09e461c7-2ac6-4e07-b3c8-6e7f593dfba2 /shared xfs defaults,nofail 1 2
```

## Verify the /shared directory

Change directories and verify that you see the /shared directory with Size of 1T

```
cd /shared  
df -h
```

Output

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	213G	0	213G	0%	/dev
tmpfs	213G	0	213G	0%	/dev/shm
tmpfs	213G	17M	213G	1%	/run
tmpfs	213G	0	213G	0%	/sys/fs/cgroup
/dev/sda2	30G	11G	19G	37%	/
/dev/sda1	495M	193M	302M	39%	/boot
/dev/sda15	495M	5.8M	489M	2%	/boot/efi
/dev/sdb1	472G	73M	448G	1%	/mnt
tmpfs	43G	0	43G	0%	/run/user/1000
/dev/sdc1	1.0T	7.2G	1017G	1%	/shared

### Create subdirectories on /shared

Create a /shared/build, /shared/data and /shared/cyclecloud-cmaq directory and change the permissions from root to your username.

```
cd /
sudo chown azureuser shared
sudo chgrp azureuser shared

cd /shared
mkdir build
mkdir data
mkdir cyclecloud-cmaq
```

### 4.3.4 Alternatively, you can create an nvme stripped disk that has faster performance.

```
mkdir -p /mnt/nvme
mdadm --create /dev/md10 --level 0 --raid-devices 2 /dev/nvme0n1 /dev/nvme1n1
mkfs.xfs /dev/md10
mount /dev/md10 /mnt/nvme
chmod 1777 /mnt/nvme
```

That should create a file system with about 1.8TiB

### 4.3.5 Download the Input data from the S3 Bucket

#### Install aws command line

see Install AWS CLI

```
cd /shared/build
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

#### Install the uncompressed 2018\_12US1 input data using the s3 script

```
cd /shared/cyclecloud-cmaq/s3_scripts/
./s3_copy_nosign_2018_12US1_conus_cmas_opendata_to_shared_20171222_cb6r5_uncompressed.csh
```

Note, you should be getting fast transfer speeds in the 200 MiB/s range, so downloading the files should take 10 minutes.

## Load the openmpi module

```
module load mpi/openmpi-4.1.5
```

## Install Cycle Cloud Repo

```
git clone -b main https://github.com/CMASCenter/cyclecloud-cmaq.git
```

## Install and build netcdf C, netcdf Fortran, I/O API, and CMAQ

```
cd /shared/cyclecloud-cmaq
```

Install netcdf-C and netcdf-Fortran

```
./gcc_install.csh
```

If successful, you will see the following output, that at the bottom shows what versions of the netCDF library were installed.

```
+-----+
| Congratulations! You have successfully installed the netCDF |
| Fortran libraries.                                         |
|                                                           |
| You can use script "nf-config" to find out the relevant   |
| compiler options to build your application. Enter        |
|                                                           |
|     nf-config --help                                       |
|                                                           |
| for additional information.                                |
|                                                           |
| CAUTION:                                                  |
|                                                           |
| If you have not already run "make check", then we strongly |
| recommend you do so. It does not take very long.         |
|                                                           |
| Before using netCDF to store important data, test your    |
| build with "make check".                                   |
|                                                           |
| NetCDF is tested nightly on many platforms at Unidata    |
| but your platform is probably different in some ways.    |
|                                                           |
| If any tests fail, please see the netCDF web site:       |
| https://www.unidata.ucar.edu/software/netcdf/            |
|                                                           |
| NetCDF is developed and maintained at the Unidata Program |
| Center. Unidata provides a broad array of data and software |
| tools for use in geoscience education and research.     |
| https://www.unidata.ucar.edu                             |
+-----+
```

(continues on next page)

(continued from previous page)

```
make[3]: Leaving directory '/shared/build/netcdf-fortran-4.5.4'
make[2]: Leaving directory '/shared/build/netcdf-fortran-4.5.4'
make[1]: Leaving directory '/shared/build/netcdf-fortran-4.5.4'
netCDF 4.8.1
netCDF-Fortran 4.5.4
```

Install I/O API

```
./gcc_ioapi.csh
```

Find what operating system is on the system:

```
cat /etc/os-release
```

Output

```
NAME="AlmaLinux"
VERSION="8.7 (Stone Smilodon)"
ID="almalinux"
ID_LIKE="rhel centos fedora"
VERSION_ID="8.7"
PLATFORM_ID="platform:el8"
PRETTY_NAME="AlmaLinux 8.7 (Stone Smilodon)"
ANSI_COLOR="0;34"
LOGO="fedora-logo-icon"
CPE_NAME="cpe:/o:almalinux:almalinux:8::baseos"
HOME_URL="https://almalinux.org/"
DOCUMENTATION_URL="https://wiki.almalinux.org/"
BUG_REPORT_URL="https://bugs.almalinux.org/"

ALMALINUX_MANTISBT_PROJECT="AlmaLinux-8"
ALMALINUX_MANTISBT_PROJECT_VERSION="8.7"
REDHAT_SUPPORT_PRODUCT="AlmaLinux"
REDHAT_SUPPORT_PRODUCT_VERSION="8.7"
```

### 4.3.6 Change shell to use tcsh

```
sudo usermod -s /bin/tcsh azureuser
```

Log out and then log back in to have the shell take effect.

Copy a file to set paths

```
cd /shared/cyclecloud-cmaq
cp dot.cshrc.vm ~/.cshrc
```

### 4.3.7 Create Environment Module for Libraries

There are two steps required to create your own custom module:

1. write a module file
2. add a line to your ~/.cshrc to update the MODULEPATH

Create a new custom module that will be loaded including any dependencies using the following command:

```
module load ioapi-3.2/netcdf **example only
```

Step 1: Create the module file.

First, create a path to store the module file. The path must contain /Modules/modulefiles/ and should have the general form //Modules/modulefiles// where is typically numerical and is the actual module file.

```
mkdir -p /shared/build/Modules/modulefiles/ioapi-3.2_20200828
```

Next, create the module file and save it in the directory above.

```
cd /shared/build/Modules/modulefiles/ioapi-3.2_20200828
```

```
vim gcc-9.2.0-netcdf
```

Contents of gcc-9.2.0-netcdf:

```
##Module

proc ModulesHelp { } {
    puts stderr "This module adds ioapi-3.2_20200828/gcc-9.2.0 to your path"
}

module-whatis "This module adds ioapi-3.2_20200828/gcc-9.2.0 to your path\n"

set basedir "/shared/build/ioapi-3.2_branch_20200828/"
prepend-path PATH "${basedir}/Linux2_x86_64gfort"
prepend-path LD_LIBRARY_PATH "${basedir}/ioapi/fixed_src"
module load mpi/openmpi-4.1.5
module load gcc-9.2.0
module load netcdf-4.8.1/gcc-9.2.0
```

The example module file above sets two environment variables and loads two system modules and a custom module (that we also need to define).

The modules update the PATH and LD\_LIBRARY\_PATH.

Now create the custom module to define the netCDF libraries that were used to build I/O API.

```
mkdir /shared/build/Modules/modulefiles/netcdf-4.8.1
cd /shared/build/Modules/modulefiles/netcdf-4.8.1
vim gcc-9.2.0
```

Contents of gcc-9.2.0

```
##Module
```

(continues on next page)

(continued from previous page)

```

proc ModulesHelp { } {
    puts stderr "This module adds netcdf-4.8.1/gcc-9.2.0 to your path"
}

module-whatis "This module adds netcdf-4.8.1/gcc-9.2.0 to your path\n"

set basedir "/shared/build/netcdf"
prepend-path PATH "${basedir}/bin"
prepend-path LD_LIBRARY_PATH "${basedir}/lib"
module load mpi/openmpi-4.1.5
module load gcc-9.2.0

```

Step 2: Add the module path to MODULEPATH.

Now that the two custom module files have been created, add the following line to your ~/.cshrc file so that they can be found:

```
vi ~/.cshrc
```

Add the following line to your .cshrc

```
module use --append /shared/build/Modules/modulefiles
```

Source the .cshrc file

```
source ~/.cshrc
```

Step 3: View the modules available after creation of the new module

The module avail command shows the paths to the module files on a given cluster.

```
module avail
```

Step 4: Load the new module

```
module load ioapi-3.2_20200828/gcc-9.2.0-netcdf
```

Output:

```

Loading ioapi-3.2_20200828/gcc-9.2.0-netcdf
  Loading requirement: gcc-9.2.1 mpi/openmpi-4.1.1 netcdf-4.8.1/gcc-9.2.0

```

Verify that the libraries required for netCDF and I/O API have been added to the \$LD\_LIBRARY\_PATH environment variable

```
echo $LD_LIBRARY_PATH
```

Output:

```

/shared/build/ioapi-3.2_branch_20200828//ioapi/fixed_src:/opt/openmpi-4.1.5/lib:/opt/gcc-
↪9.2.0/lib64:/shared/build/netcdf/lib

```

Verify that the I/O API bin directory and netCDF bin directory that you specified in the custom module has been added to the \$PATH environment variable

```
echo $PATH
```

Output

```
/shared/build/ioapi-3.2_branch_20200828//Linux2_x86_64gfort:/opt/openmpi-4.1.5/bin:/opt/
↪gcc-9.2.0/bin:/usr/share/Modules/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:
↪/shared/build/netcdf/bin:/shared/build/ioapi-3.2/Linux2_x86_64gfort:/opt/slurm/bin:/
↪usr/local/bin
```

see Custom-Modules from Princeton Research Computing

### 4.3.8 Install and Build CMAQ

```
./gcc_cmaq_v54+.csh
```

Verify that the executable was successfully built.

```
ls /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/*.exe
```

Output

```
/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
```

### 4.3.9 Copy the run scripts from the repo to the run directory

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
cp /shared/cyclecloud-cmaq/run_scripts/run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.96.
↪ncclassic.csh .
```

Note, this Virtual Machine does not have Slurm installed or configured. Also note, the first few timings reported here were using the HB120\_v2, instead of HB120\_v3. See the last timing report for the HB120\_v3, which is much faster.

### 4.3.10 Run CMAQ interactively using the following command:

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
./run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.96.ncclassic.csh |& tee ./run_cctm_2018_
↪12US1_v54_cb6r5_ae6.20171222.96.ncclassic.log
```

When the run has completed, record the timing of the two day benchmark.

```
tail -n 30 run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.96.ncclassic.log
```

Output:

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
```

(continues on next page)



(continued from previous page)

```

Domain Name:          12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers:     35
Number of Processes:  96
  All times are in seconds.

```

```

Num  Day      Wall Time
01   2017-12-22  3175.9
02   2017-12-23  3484.4
    Total Time = 6660.30
    Avg. Time = 3330.15

```

If runs are submitted immediately after a successful completion of a run, then you may skey the scaling results. It would be ideal to wait 30 minutes before running a second job.

### Run second job interactively using the following command:

```

./run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x120.ncclassic.csh |& tee ./run_cctm_2018_
↪12US1_v54_cb6r5_ae6.20171222.1x120.ncclassic.log

```

### Output

```
tail -n 18 ./run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x120.ncclassic.log
```

```

=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day:   2017-12-23
Number of Simulation Days: 2
Domain Name:          12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers:     35
Number of Processes:  96
  All times are in seconds.

Num  Day      Wall Time
01   2017-12-22  3075.2
02   2017-12-23  3477.9
    Total Time = 6553.10
    Avg. Time = 3276.55

```

### 4.3.11 Created another single VM using HBv120\_v2 and ran again

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
./run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x96.ncclassic.csh |& tee ./run_cctm_2018_
↪12US1_v54_cb6r5_ae6.20171222.1x96.ncclassic.log
```

When it finished, examined the log file:

```
tail -n 18 run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x96.ncclassic.log
```

Output:

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 96
    All times are in seconds.

Num  Day      Wall Time
01   2017-12-22  3069.9
02   2017-12-23  3445.2
    Total Time = 6515.10
    Avg. Time = 3257.55
```

### 4.3.12 Created another VM using the HB120v3 cpus

### 4.3.13 Verify that the correct number of cpus are installed using lscpu

```
lscpu
```

Output:

```
[azureuser@hb120v3manish output_v54_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_2x64_classic]$ ↪
↪lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                120
On-line CPU(s) list:   0-119
Thread(s) per core:    1
Core(s) per socket:    60
Socket(s):             2
NUMA node(s):         4
Vendor ID:             AuthenticAMD
CPU family:            25
Model:                 1
```

(continues on next page)

(continued from previous page)

```

Model name:      AMD EPYC 7V73X 64-Core Processor
Stepping:        2
CPU MHz:         3094.426
BogoMIPS:        3693.10
Hypervisor vendor: Microsoft
Virtualization type: full
L1d cache:       32K
L1i cache:       32K
L2 cache:        512K
L3 cache:        98304K
NUMA node0 CPU(s): 0-29
NUMA node1 CPU(s): 30-59
NUMA node2 CPU(s): 60-89
NUMA node3 CPU(s): 90-119

```

has context menu

### 4.3.14 Timing information

```

=====
***** CMAQ TIMING REPORT *****
=====

Start Day: 2017-12-22
End Day:   2017-12-23
Number of Simulation Days: 2
Domain Name:      12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers:  35
Number of Processes: 96
    All times are in seconds.

Num  Day      Wall Time
01   2017-12-22  2818.3
02   2017-12-23  3205.8
    Total Time = 6024.10
    Avg. Time = 3012.05

```

The HB120\_v3 has much faster performance than the HB120\_v2.

### 4.3.15 Review performance metrics in the Azure portal

<https://portal.azure.com>

Click on Virtual Machines

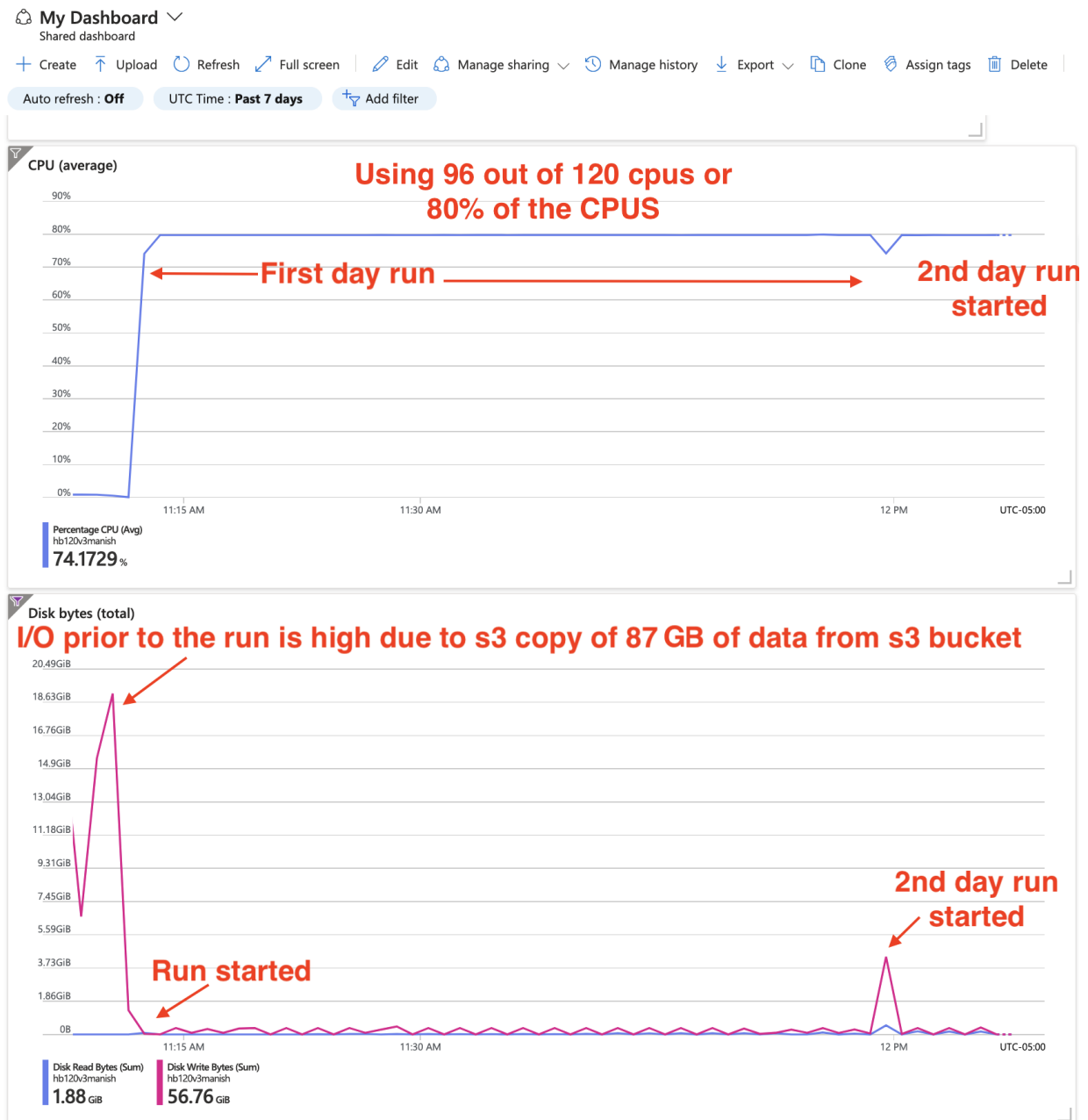
Select the Virtual Machine that you created

Under Essentials, click on Monitoring, and then pin the CPU Average and Disk bytes total to a shared dashboard.

Then under the Azure Portal search for shared dashboards

Follow that link to Go to Dashboard.

You can click in the upper right of each dashboard on the ... and select Configure Dashboard to specify the time period (ie. past hour, or past 4 hours. In this example, the past hour time period was used.



Second image saved after the initial s3 copy data transfer had been completed, giving a scale that allows you to see the transfer rate.



Full time to install input data, build code, and run CMAQv5.4+ for the 12US1 domain on HB120rs\_v3 using 96 of 120 cpus.

azure\_web\_interface\_images/Full\_build\_and\_run\_withing\_2\_hrs.png

### 4.3.16 IF your performance is much slower than this, then we recommend that you terminate the resource group and re-build the VM

## 4.4 Create CycleCloud HB120rs\_v3 Cluster

Use [portal.azure.com](https://portal.azure.com) to create CycleCloud Cluster

### 4.4.1 Create Cyclecloud CMAQ Cluster

Documentation for Azure CycleCloud [Documentation](#)

#### Configure the Cycle Cloud Application Host using the Azure Portal

Log into the [Azure Portal](#)

In the search bar, enter “Marketplace”, Click on Marketplace Icon.

In the Marketplace search bar, enter “CycleCloud”.

Click on the heart in the Azure CycleCloud box to add this as a favorite resource.

Use the Create pulldown menu to select Azure CycleCloud 8.6

#### Customize your Host Virtual Machine for the CycleCloud Application

1. Choose your Subscription
2. Select or create a new Resource Group that your CycleCloud instance will run in: note, leave this blank initially, as it will be named after the instance name below by appending \_group to the instance name
3. Name your CycleCloud instance using Virtual Machine name : example name: CycleCloudHost
4. Select Region: example name: US East
5. Verify Image is Azure CycleCloud 8.6 - x64 Gen2
6. Select Size, click on see all sizes, enter D4s into the search button and select Standard\_D4s\_v3- 4cpus, 16GiB memory (\$140.16/month)
7. Select Authentication Type SSH public key
8. Create the Username that you will use to log into the instance: example name: azureuser
9. SSH public key source - select Generate new key pair
10. Select the Management tab and enable System assigned managed identity
11. Click on the Review button and then the Create button

When a pop-up menu is displayed: click on option to Download private key and create resource.

You will see a message ... Deployment is in progress

Wait until the resource has been deployed before proceeding to the next step.

Figure 1. Create a virtual Machine - Customize Host Virtual Machine Note: this virtual machine will be used to host the CycleCloud Application that is used to create the Cycle Cloud Cluster from it's Web located at: UI <https://IP-address/home>

#### 4.4. Create CycleCloud HB120rs\_v3 Cluster

Microsoft Azure

Search resources, services, and docs (G+/)

[Home](#) > [Create a resource](#) > [Azure CycleCloud](#) >

## Create a virtual machine ...

Basics

**Disks**

Networking

Management

Advanced

Tags

Review + create

Azure VMs have one operating system disk and a temporary disk for short-term storage. You can attach additional data disks. The size of the VM determines the type of storage you can use and the number of data disks allowed. [Learn more](#)

### Disk options

OS disk type \*

Delete with VM ☒

Encryption at host ☐

**i** Encryption at host is not registered for the selected subscription. [Learn more about enabling this feature](#)

Encryption type \*

Enable Ultra Disk compatibility ☐  
Ultra disk is supported in Availability Zone(s) 1,2,3 for the selected VM size Standard\_F4s\_v2.

### Data disks for 'cyclecloud-ea'

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching	Delete with VM
0	Pre-defined by the ...			<input type="text" value="Read-only"/>	<input type="checkbox"/>

[Create and attach a new disk](#) [Attach an existing disk](#)

Advanced

Review + create

< Previous

Next : Networking >

Figure 3. Select Network Interface for the Azure Virtual Machine - use default options



Microsoft Azure

Home > Create a resource > Azure CycleCloud >

## Create a virtual machine ...

Basics
Disks
**Networking**
Management
Advanced
Tags
Review + create

Define network connectivity for your virtual machine by configuring network interface card (NIC) settings. You can control ports, inbound and outbound connectivity with security group rules, or place behind an existing load balancing solution. [Learn more](#)

### Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network \* ⓘ

(new) cycle\_cloud\_resource\_group-vnet

Create new

Subnet \* ⓘ

(new) default (10.11.0.0/24)

Public IP ⓘ

(new) CycleCloudHost-ip

Create new

NIC network security group ⓘ

☐ None
☐ Basic
☒ Advanced

*i* This VM image has preconfigured NSG rules

Configure network security group \*

(new) CycleCloudHost-nsg

Create new

Delete public IP and NIC when VM is deleted ⓘ
☒

Accelerated networking ⓘ
☐

The selected image does not support accelerated networking.

### Load balancing

You can place this virtual machine in the backend pool of an existing Azure load balancing solution. [Learn more](#)

Place this virtual machine behind an existing load balancing solution?
☐

Review + create
< Previous
Next : Management >

Figure 4. Select System assigned Managed Identity

Microsoft Azure

Search

[Home](#) > [Marketplace](#) >

## Create a virtual machine

Basics

Disks

Networking

**Management**

Advanced

Tags

Review + create

Configure monitoring and management options for your VM.

### Microsoft Defender for Cloud

Microsoft Defender for Cloud provides unified security management and advanced threat protection across hybrid cloud workloads. [Learn more](#)

✓ Your subscription is protected by Microsoft Defender for Cloud basic plan.

### Monitoring

Boot diagnostics ⓘ ☒ Enable with managed storage account (recommended)  
☐ Enable with custom storage account  
☐ Disable

Enable OS guest diagnostics ⓘ ☐

### Identity

System assigned managed identity ⓘ ☒

### Azure AD

Login with Azure AD ⓘ ☐

⚠ This image does not support Login with Azure AD.

### Auto-shutdown

Enable auto-shutdown ⓘ ☐

### Guest OS updates

Patch orchestration options ⓘ 

Image default

Some patch orchestration options are not available for this image. [Learn more](#)

Review + create

< Previous

Next : Advanced >

Figure 5. Create a Virtual Machine - Deployment is in Progress

The screenshot shows the Azure portal interface. The top navigation bar includes the Microsoft Azure logo and a search bar. The main header displays the deployment name and an 'Overview' link. On the left, a sidebar lists navigation options: Overview, Inputs, Outputs, and Template. The main content area shows the deployment status as 'Deployment is in progress'. It includes a feedback prompt, deployment details (name, subscription, resource group, start time, and correlation ID), and a table of resources being deployed.

Resource	Type	Status
cyclecloud-ea	Microsoft.Compute/virtualMachines	Created
cyclecloud-ea986	Microsoft.Network/networkInterfaces	Created
cmaq_la-vnet	Microsoft.Network/virtualNetworks	OK
cyclecloud-ea-nsg	Microsoft.Network/networkSecurityGroups	OK
cyclecloud-ea-ip	Microsoft.Network/publicIpAddresses	OK

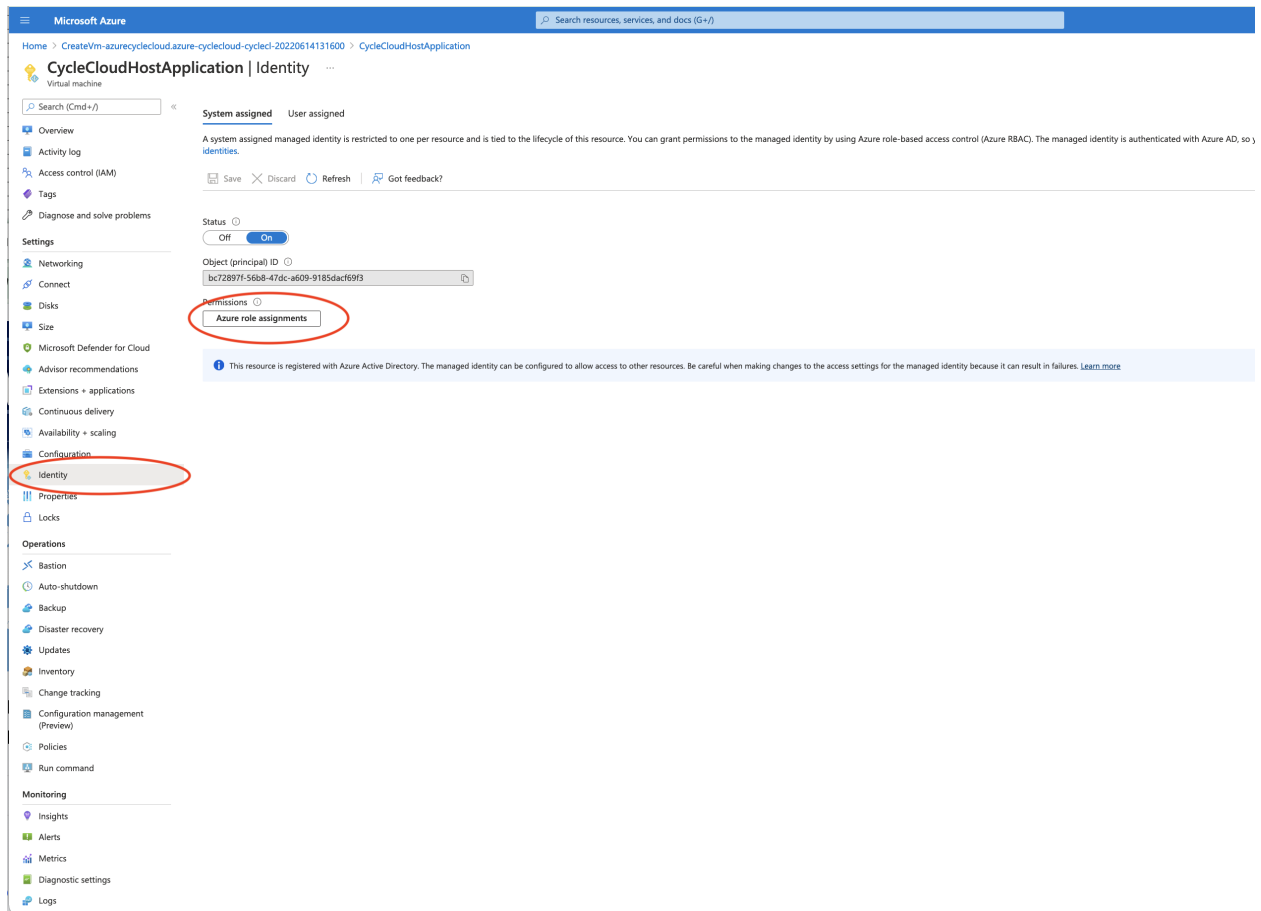
Figure 6. Your Deployment is complete - click on blue button Go to resource

The screenshot shows the Azure portal interface for a completed deployment. The main header displays the deployment name and an 'Overview' link. The left sidebar lists navigation options: Overview, Inputs, Outputs, and Template. The main content area shows the deployment status as 'Your deployment is complete'. It includes a feedback prompt, deployment details (name, subscription, resource group, start time, and correlation ID), and a section for 'Next steps' with recommended actions. At the bottom, there are two buttons: 'Go to resource' and 'Create another VM'.

After the CycleCloud Host Machine has been deployed click on Go to resource

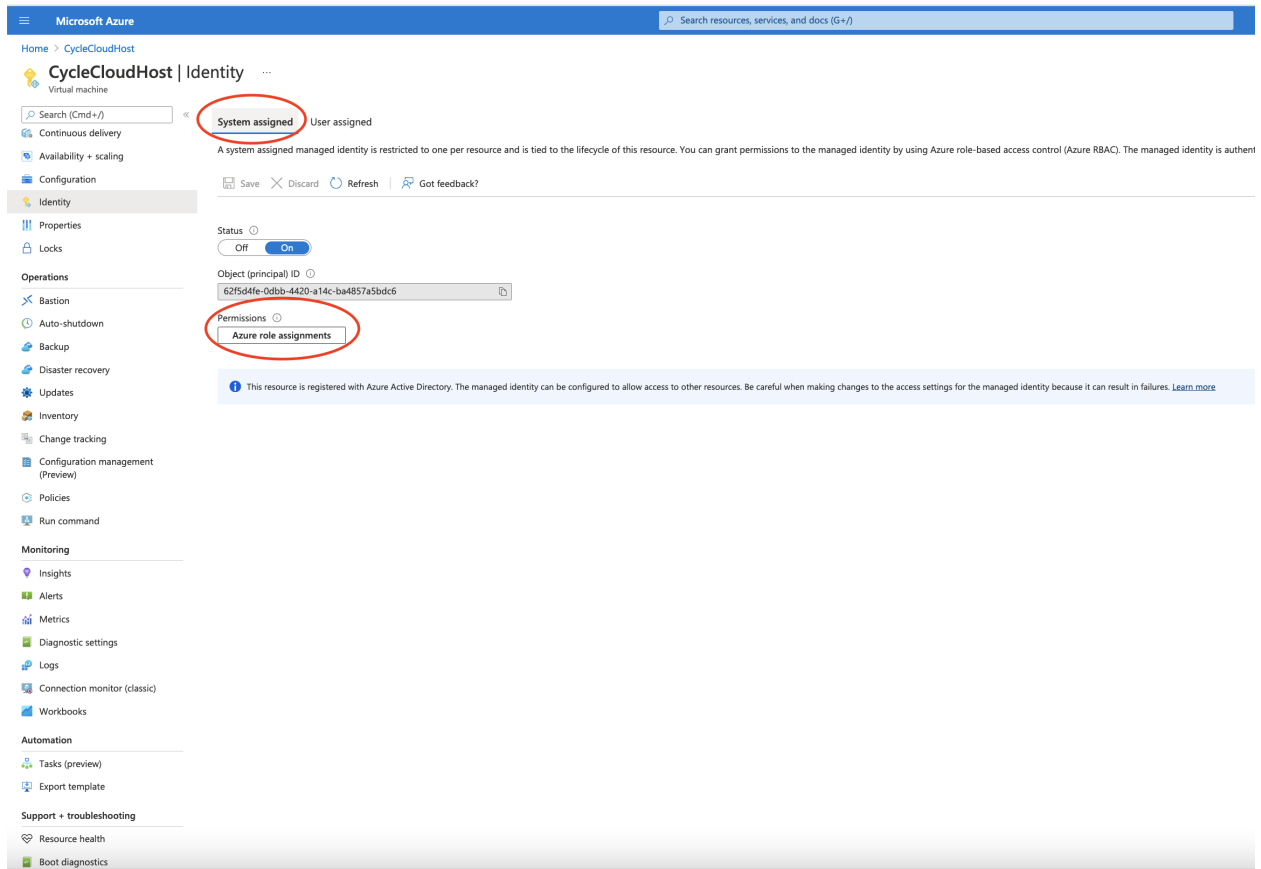
Add Contributor Role to Virtual Machine -

Figure 7. Click on Identity Icon on left side of CycleCloudHost Application Virtual Machine



Click on the Identity Menu on the left side of the newly created virtual machine. Make sure you select the System Assigned Tab at the top of the window. Click on the button Azure Role Assignments

Figure 8. Make sure you select the System Assigned Tab at the top of the window.



On the Azure role assignments window click on the + Add role assignment(Preview)

Figure 9. Add System Assigned Role Assignment - Management Identity

Click on Azure role assignments Search for Managed Identity Operator

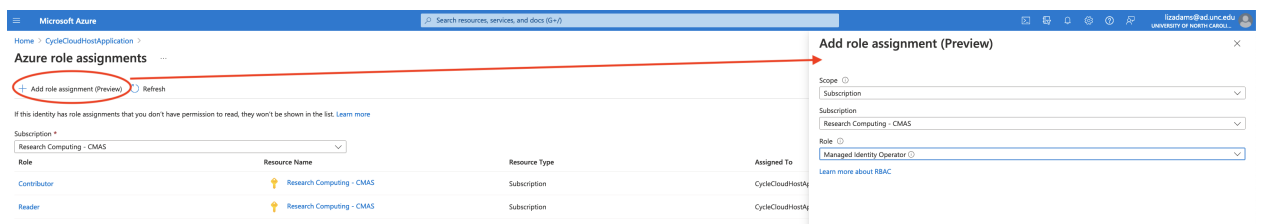


Figure 10. Add Role Assignment

1. Click Identity Icon under Settings on the left side menu
2. Click Azure role assignments
3. Click Add role assignment
4. Search for Contributor

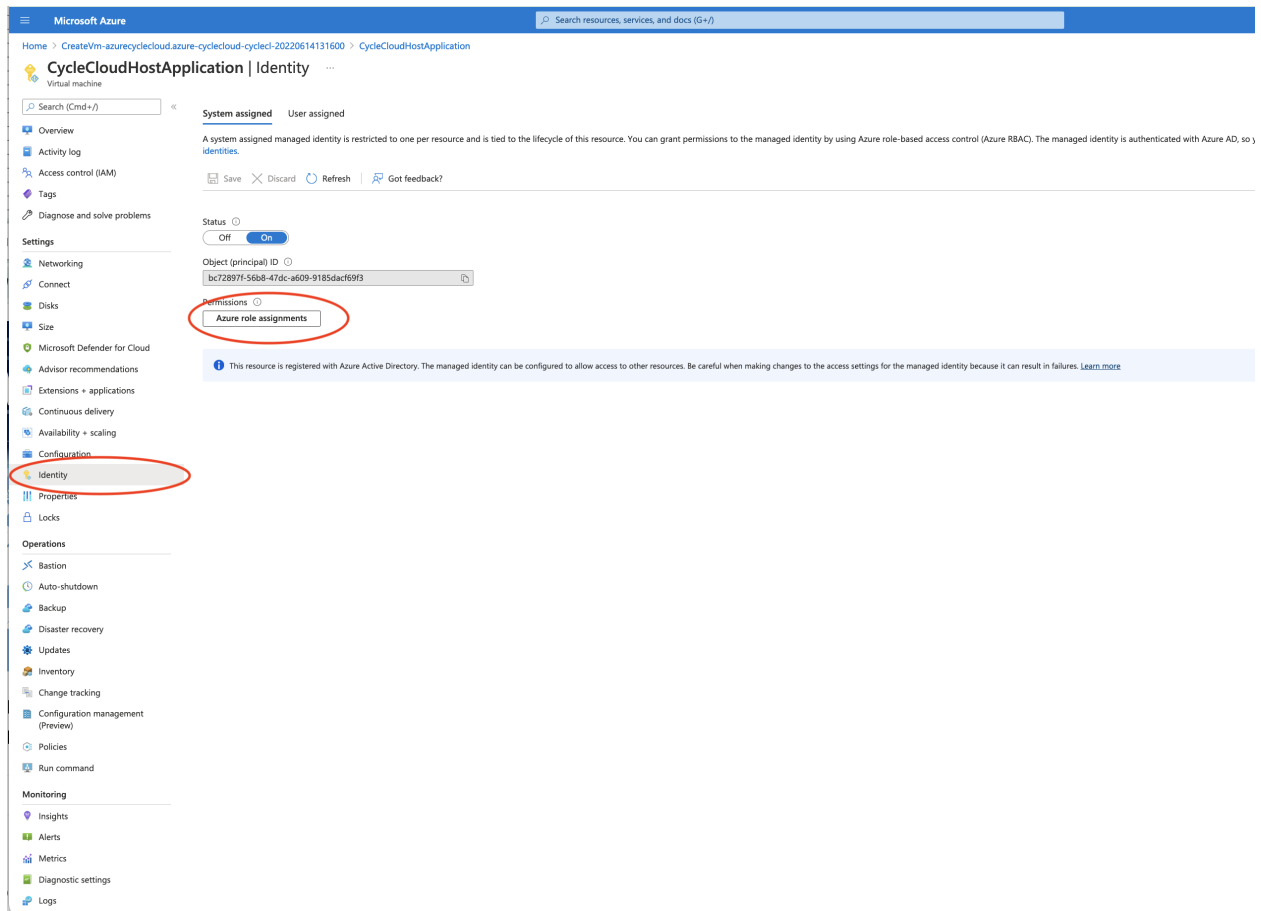
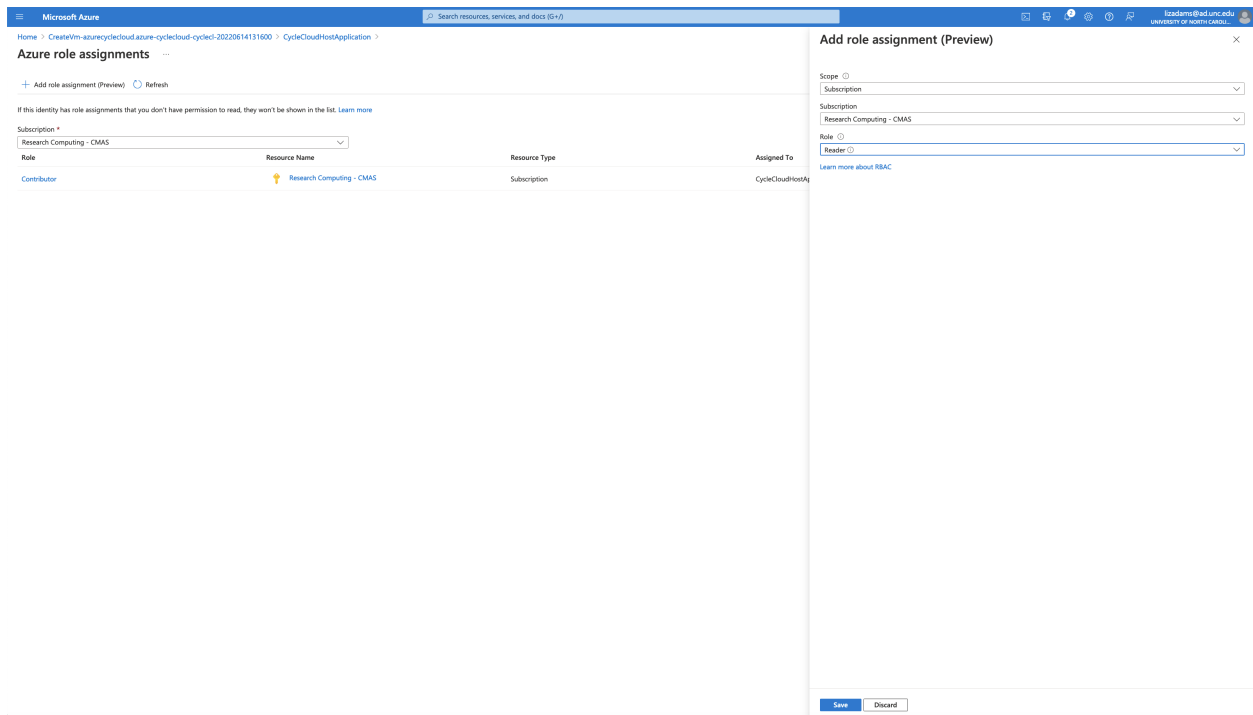


Figure 11. Add Reader Role to Virtual Machine



## Create Storage Account

Question: Do I need to create a new storage account for each CycleCloud Virtual Machine Host?

In the search bar, enter Storage Account, then select + Create Storage Account

Select the resource group associated with the CycleCloudHost that you created CycleCloudHost\_group. Select a lowercase name. Then switch from the Basics tab to the Advanced Tab. Uncheck the box next to Enable blob public access. Click Review and Create. After the verification passed message is received, click Create.

Figure 12. Azure Create Storage Account Details



Microsoft Azure

Search resources, set

Home > Storage accounts >

## Create a storage account

Basics

Advanced

Networking

Data protection

Encryption

Tags

Review + create

Azure Storage is a Microsoft-managed service providing cloud storage that is highly available, secure, durable, scalable, and redundant. Azure Storage includes Azure Blobs (objects), Azure Data Lake Storage Gen2, Azure Files, Azure Queues, and Azure Tables. The cost of your storage account depends on the usage and the options you choose below. [Learn more about Azure storage accounts](#)

### Project details

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription \*

Research Computing - CMAS

Resource group \*

CycleCloudHostApplication\_group

[Create new](#)

### Instance details

If you need to create a legacy storage account type, please click [here](#).

Storage account name ⓘ \*

cyclecloudhostappstorage

Region ⓘ \*

(US) East US

Performance ⓘ \*

☒ Standard: Recommended for most scenarios (general-purpose v2 account)

☐ Premium: Recommended for scenarios that require low latency.

Redundancy ⓘ \*

Geo-redundant storage (GRS)

☒ Make read access to data available in the event of regional unavailability.

Review + create

< Previous

Next : Advanced >

For Redundancy choose Local Redundant Storage instead of Geo-Redundant Storage to reduce costs.

Figure 13. Azure Storage Account disable Public Blob Access

Disable the Public Blob Access by unclicking the box next to `Enable blob public access`

Microsoft Azure

Search resources, ...

Home > Storage accounts >

Create a storage account ...

Basics

Advanced

Networking

Data protection

Encryption

Tags

Review + create

Certain options have been disabled by default due to the combination of storage account performance, redundancy, and region.

Security

Configure security settings that impact your storage account.

Require secure transfer for REST API operations ⓘ

☒

Enable blob public access ⓘ

☐

Enable storage account key access ⓘ

☒

Default to Azure Active Directory authorization in the Azure portal ⓘ

☐

Minimum TLS version ⓘ

Version 1.2

Data Lake Storage Gen2

The Data Lake Storage Gen2 hierarchical namespace accelerates big data analytics workloads and enables file-level access control lists (ACLs). [Learn more](#)

Enable hierarchical namespace

☐

Blob storage

Enable SFTP (preview) ⓘ

☐

**i** To enable SFTP, 'hierarchical namespace' must be enabled.

Enable network file system v3 ⓘ

☐

**i** To enable NFS v3 'hierarchical namespace' must be enabled. [Learn more about NFS v3](#)

Allow cross-tenant replication ⓘ

☒

Access tier ⓘ

☒ Hot: Frequently accessed data and day-to-day usage scenarios

☐ Cool: Infrequently accessed data and backup scenarios

Azure Files

Enable large file shares ⓘ

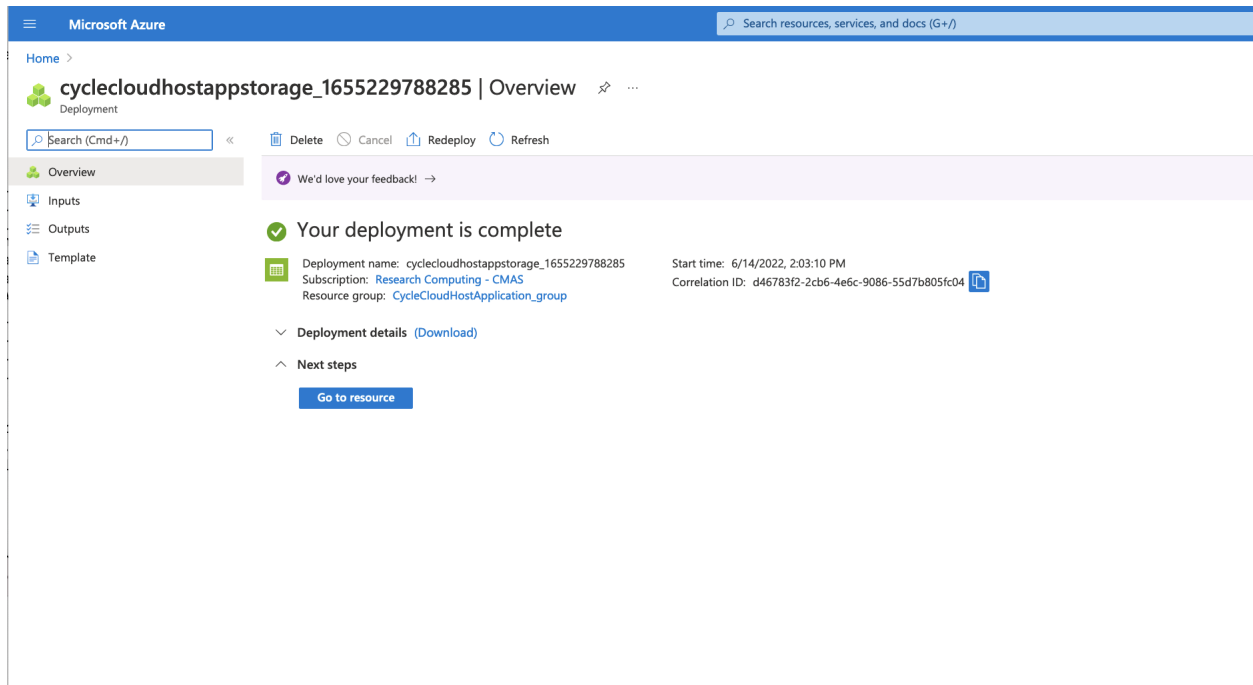
☐

Review + create

< Previous

Next : Networking >

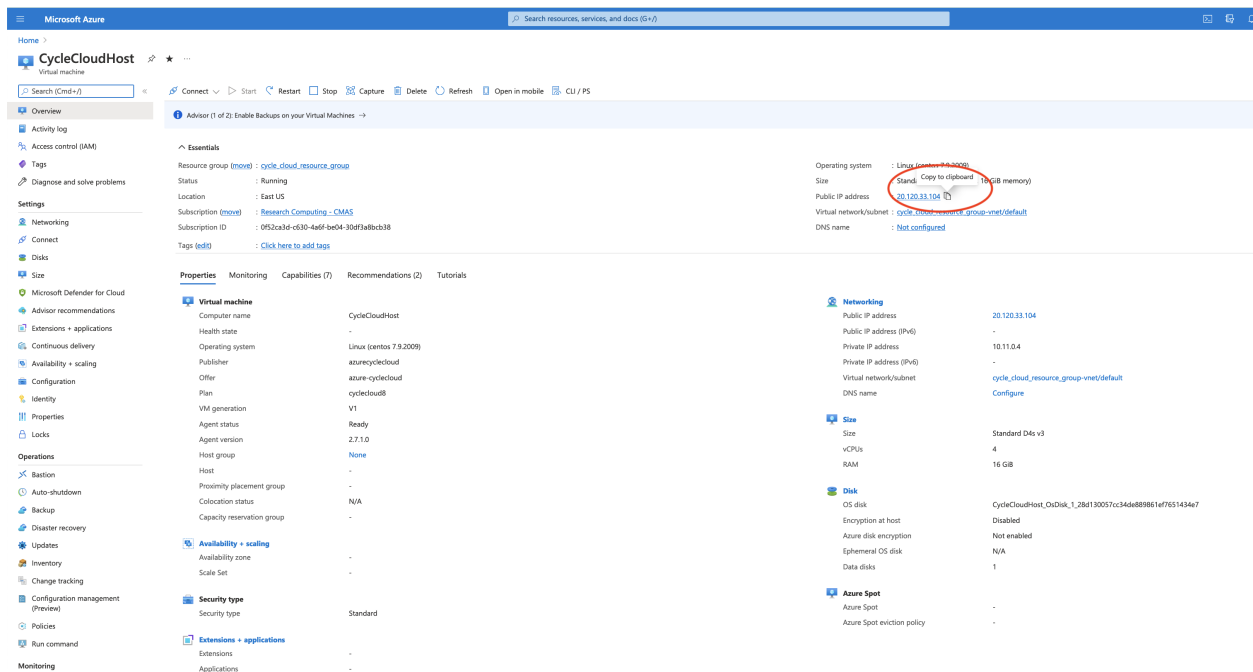
Figure 14. Storage Account Deployment is complete



Click on Home to return to the Azure Portal and then Click on the CycleCloudHostApplication Virtual Machine

Click on Copy next to the Public IP address to copy it.

Figure 15. Azure Cycle Cloud Host Machine IP address



## Connect to Cyclecloud Web Interface

In your web browser, create a new tab, and enter the IP address that you copied from the step above.

`https://-IP-ADDRESS/welcome`

If you get a warning, potential security risk ahead, click on Advanced, then accept risk and continue.

1. Enter a Site Name - a unique name for the CycleCloud. example CycleCloudHostApplicationManager
2. Read and click that you agree to the CycleCloud Software License Agreement
3. Create your CycleCloud Administrator Account. This requires a public rsa key. Instructions for creating this are available [here](#)

Figure 16. Web Interface to CycleCloud - connect using the ip address for the Scheduler Node above `http://-IP-ADDRESS/welcome`

Step 1 of 3

### Welcome to Azure CycleCloud!

This setup wizard will lead you through the steps required to configure Azure CycleCloud. Consult the [documentation](#) for details on advanced configuration.

Please fill out the details below and click "Next" to proceed with Azure CycleCloud setup.

#### Site Name

A unique name for this installation. This will be used when tagging remote resources.  
**Examples:** *production, my-organization*

Site Name  A label for this installation

#### Usage Data

CycleCloud collects anonymized usage data to support and improve the product. Please view our [data policy](#) to learn more about this data and our privacy policy.

Next

Figure 17. Azure CycleCloud Add Subscription ID

The Subscriptions page will show if the cluster subscription was created. You may need to pull the State window to enlarge it.

When it says created, with nothing under the Failed column, then it was successful, click Back to Clusters.

Figure 18. Check Cluster Creation Status in Subscriptions Table

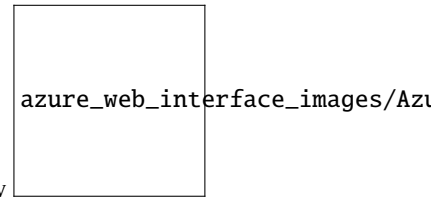
Name	Provider	Provider Id	Master Credentials	Default	State	Status	Failed	Disabled
Research Computing - CMAS	azure	0f52ca3d-c630-4a6f-be04-30df3a8bcb...	Research Computing - CM...	true	Created	---		

Figure 19. Azure CycleCloud Create a New Cluster - Select SLURM workload Manager

Figure 20. Azure CycleCloud New Slurm Cluster - add a Cluster Name

Example name: CMAQSlurmHC44rsAlmaLinux

Figure 21. Azure CycleCloud HPC Queue Select Machine



In the Compute Type, select High Performance Compute Select HB120rs\_v3, then select Apply

Figure 22. Select Max HPC Cores

Select Auto-Scaling Max HPC Cores to be a multiple of the number of cpus available on the compute node. For HB120rs\_v3 for a maximum of 4 nodes, it would be  $120 \times 4 = 480$  Max HPC Cores Choose the Networking SubnetID that was created for the CycleCloud.

Edit beondtest2-copy

About

Required Settings

Network Attached Storage

Advanced Settings

Security

Cloud-init

Virtual Machines

The cluster, in this case, has two roles: the scheduler node with shared file and the execute hosts. Configure which VM types to use based on the requirements of your application.

Region

East US

Scheduler VM Type

Standard\_D4ads\_v5

Choose

Login node

The VM type for scheduler node

Choose

HPC VM Type

Standard\_HB120rs\_v3

Choose

HTC VM Type

Standard\_F2s\_v2

Choose

Dyn VM Type

Standard\_F2s\_v2

Choose

Auto-Scaling

The cluster can autoscale to the workload, adding execute hosts as jobs are queued. To enable this check the box below and choose the initial and maximum core counts for the cluster.

Autoscale

☒

Start and stop execute instances automatically

Max HPC Cores

480

Max HTC Cores

4

Max Dyn Cores

100

Max VMs per VMSS

100

HTC Spot

☐

Use Spot VMs for HTC execute hosts

DynSpot

☐

Use Spot VMs for Dynamic execute hosts

Num Login Nodes

0

Networking

Subnet ID\*

cyclecloud8.5host\_group: cyclecloud8.5host-vnet-c

High Availability

Slurm can be setup in HA mode - where slurmctld is running on two nodes with failover. Note that checking this box will require an external NFS, so any reference to the 'builtin' NFS will be hidden.

Slurm HA Node

☐

Previous

Next

Save

Cancel

Figure 23. Azure CycleCloud Network Attached Storage

Change the size from 100 GB of network attached storage to 1000 GB of network attached storage for the /shared

directory, where CMAQ and the input data will be installed.

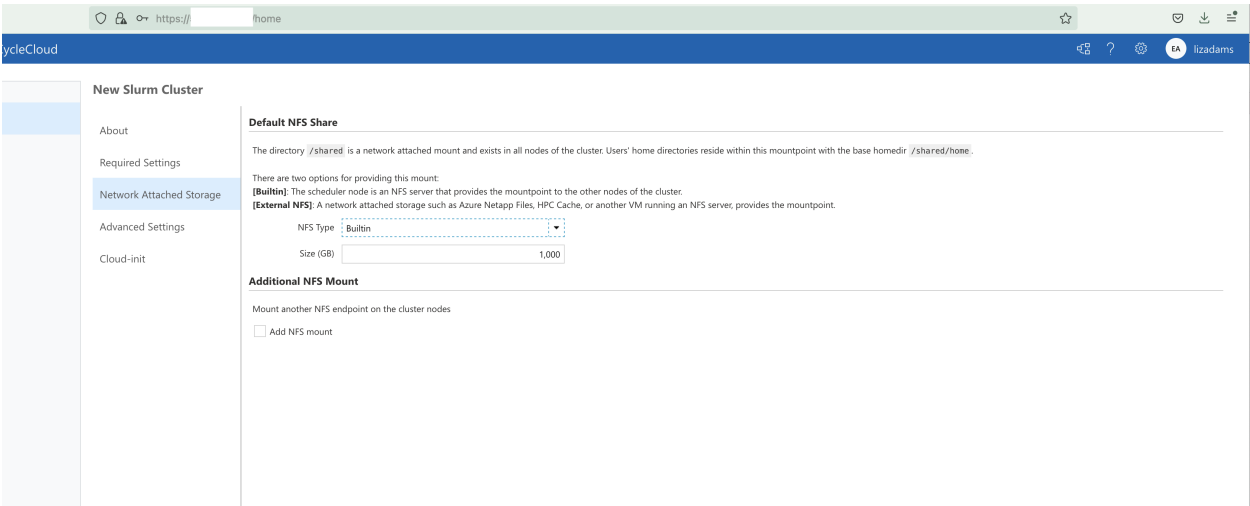


Figure 24. Azure CycleCloud Select OS and Uncheck Name as HostName

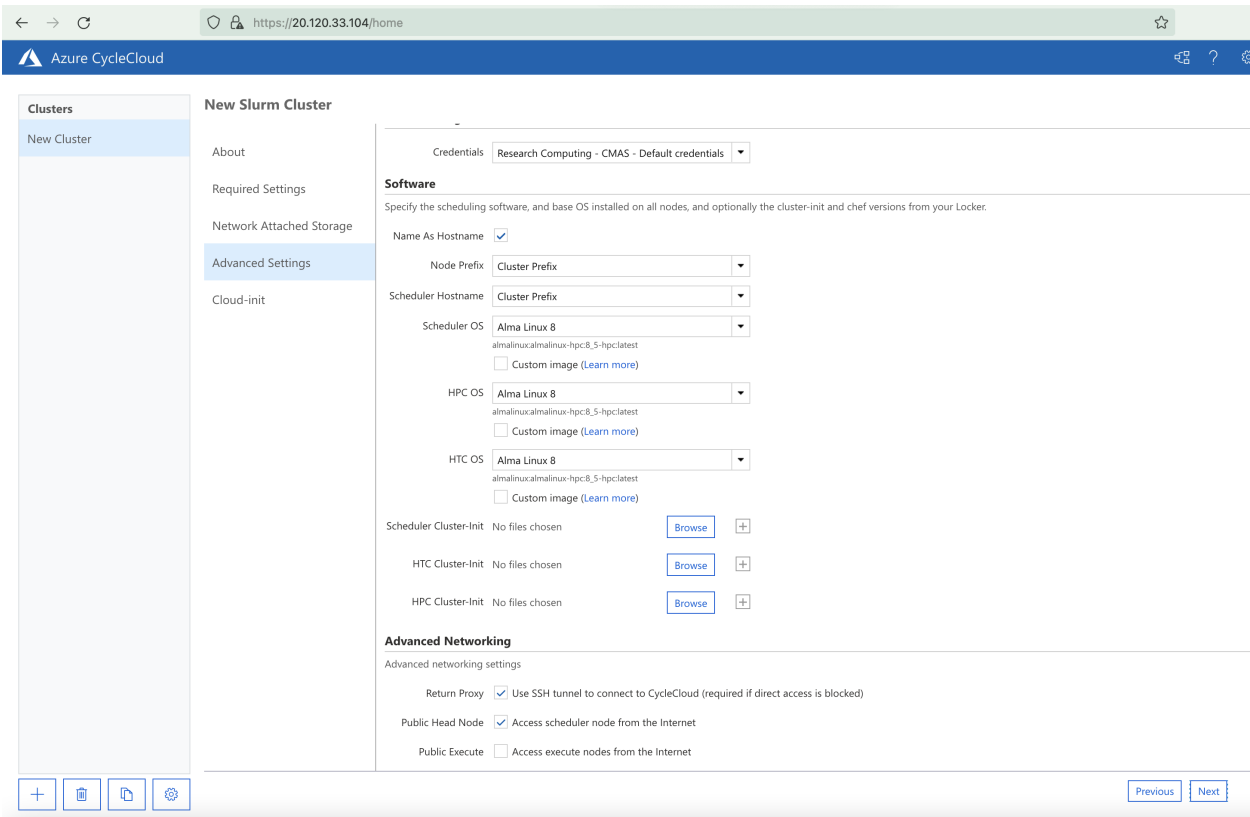


Figure 25. Azure CycleCloud Select Machine Type for HPC Node



#### 4.4. Create CycleCloud HB120rs\_v3 Cluster

Edit beendtest2-copy

×

About

Required Settings

Network Attached Storage

Advanced Settings

Security

Cloud-init

Virtual Machines

The cluster, in this case, has two roles: the scheduler node with shared file and the execute hosts. Configure which VM types to use based on the requirements of your application.

Region

East US

▼

Scheduler VM Type

Standard\_D4ads\_v5

Choose

Login node VM Type

Standard\_D8as\_v4

Choose

HPC VM Type

Standard\_HB120rs\_v3

Choose

HTC VM Type

Standard\_F2s\_v2

Choose

Dyn VM Type

Standard\_F2s\_v2

Choose

Auto-Scaling

The cluster can autoscale to the workload, adding execute hosts as jobs are queued. To enable this check the box below and choose the initial and maximum core counts for the cluster.

Autoscale

☒

Start and stop execute instances automatically

Max HPC Cores

480

Max HTC Cores

4

Max Dyn Cores

100

Max VMs per VMSS

100

HTC Spot

☐

Use Spot VMs for HTC execute hosts

DynSpot

☐

Use Spot VMs for Dynamic execute hosts

Num Login Nodes

0

Networking

Subnet ID\*

cyclecloud8.5host\_group: cyclecloud8.5host-vnet-c

▼

High Availability

Slurm can be setup in HA mode - where slurmctld is running on two nodes with failover. Note that checking this box will require an external NFS, so any reference to the 'builtin' NFS will be hidden.

Slurm HA Node

☐

Previous

Next

Save

Cancel

Note: the maximum number of CPUs specified for the HPC Compute node can be changed after the cluster has been created. See section 4.1.4 for the command line commands.

Figure 27. Azure Cycle Cloud Subscriptions Registering Service Providers

Azure CycleCloud

⚙️ ? ⚙️ L lizadams

← Back to clusters

Subscriptions

⌵ Show Detail Create Edit Delete Import Retry

🔍 Search

Name	Provider	Provider Id	Master Credentials	Default	State	Status	Failed	Disabled
Research Computing - CMAS	azure	0f52ca3d-c630-4a6f-be04-30df3a8bcb...	Research Computing - CM...	true	Configuration	Registering service providers	...	...

Figure 28. Azure cycle Cloud Subscription Created Successfully

66

Chapter 4. Why might I need to use Azure Virtual Machine or CycleCloud?

beeondtest2-copy

- [Terminate](#)
- ✎ [Edit](#)
- 🔑 [Access](#)
- 🔄 [Refresh](#)

---

- 🔍 [Support](#)

State **Started** at 10:45 AM (up 1m 57s) - [View in Portal](#)

Nodes **1** acquiring

Users **2** admins | [Show](#)

Size **1** instance, **4** cores (**\$0.21** per hour)

Usage **412.7 core-hours (~\$18)** in the last 24 hours

Alerts [🔔 Create new alert](#)

Issues No issues found

[illegible]

#### 4.4. Create CycleCloud HB120rs\_v3 Cluster

## beeondtest2-copy

[□ Terminate](#)[✎ Edit](#)[🔗 Access](#)[🔄 Refresh](#)[? Support](#)State **Started** at 10:45 AM (up 56s) - [View in Portal](#)Nodes **1** acquiringUsers **2** admins | [Show](#)Size **1** instance, **4** cores (**\$0.21** per hour)Usage **412.6 core-hours (~\$18)** in the last 24 hoursAlerts [🔔 Create new alert](#)

Issues No issues found

Nodes	Arrays	Activity	Monitoring			
View: Array			Show Detail	Edit	Search	
Name	Type	Creds	Target	Core Target	Status	
dynamic	Standard_F2s_v2	researchComputing-CMAS	....	....	Activated	
hpc	Standard_HB120rs_v3	researchComputing-CMAS	....	....	Activated	
htc	Standard_F2s_v2	researchComputing-CMAS	....	....	Activated	
login	Standard_D8as_v4	researchComputing-CMAS	....	....	Activated	
scheduler-ha	Standard_D4ads_v5	researchComputing-CMAS	....	....	Activated	

Azure Cycle Cloud Start Cluster In the Nodes table, it will say scheduler 1 node, 4 cores, Status Message: Staging Resources

Login to Azure Cycle Cloud and verify that the following command works.

Click on the Scheduler node, and obtain the IP address, then login using

```
ssh -Y $USER@IP-ADDRESS
```

Run a bash script for 1 minute by submitting to the hpc node using `srn`.

```
srn -t 1:00 -n 2 --pty /bin/bash
```

You should see the hpc acquiring a single node.

Figure 31. Azure CycleCloud Acquiring Compute Node after running `srn` command.



(continued from previous page)

```
job_state = R
queue = hpc
qtime = Thu Mar 7 16:14:22 2024
mtime = Thu Mar 7 16:14:22 2024
ctime = Thu Mar 7 16:15:22 2024
exec_host = beeondtest2-copy-hpc-1/2
Priority = 4294901758
euser = lizadams(20001)
egroup = lizadams(20001)
Resource_List.walltime = 00:01:00
Resource_List.nodect = 1
Resource_List.ncpus = 2
```

Figure 32. Azure Cycle Cloud Showing usage of Scheduler Node and Compute Nodes for Srun command

beeondtest2-copy

Terminate

Edit

Access

Refresh

Support

State **Started** at 10:45 AM (up 7m 30s) - [View in Portal](#)

Nodes **2** ready

Users **2** admins | [Show](#)

Scalesets **1** created

Size **2** instances, **124** cores (**\$3.81** per hour)

Usage **415.1 core-hours (~\$19)** in the last 24 hours

Alerts [Create new alert](#)

Issues No issues found

Nodes

Arrays

Activity

Monitoring

Scalesets

View: Template

Actions

Search

Template	Nodes	Cores	Status	Last Status Message
hpc	1	120		....
scheduler	1	4		....

View: Details

Show Detail

Edit

Connect

Actions

Search

Name	Status	Cores	Host/IP	Placement Group	Keep Alive	Status Message
beeondtest2-copy-hpc-1	Ready	120	40.117.62.1...	Standard_HB120rs_v3_p...	....	....

## Instructions to upgrade the number of processors available to the Cycle Cloud Cluster (only needed if you want to modify the number of nodes in the HPC queue)

Edit the HPC config in the cyclecloud web interface to set the CPUs to 600 Run the following on the scheduler node the changes should get picked up:

```
cd /opt/cycle/slurm
sudo ./cyclecloud_slurm.sh scale
```

## 4.5 CMAQv5.4+ Benchmark on HBv3\_120 compute nodes and beeond

Run CMAQv5.4+ on a using pre-loaded software and input data on Beeond using HBv3\_120 Cycle Cloud. Running using the Beeond filesystem, which uses the /nvme or fast file systems on each compute node is free, and gives performance that is almost as good as running on lustre. In addition, the beeond filesystem is created and destroyed as part of the run script. For the lustre managed filesystem, we did not have a way to create the filesystem when the job was started and stop it when it stopped, so we incurred costs for leaving it on for months.

Note about Lustre managed filesystem, the cost varies by performance and has a minimum size that varies by performance type, for the benchmarks run, we used a 250MB/s Lustre filesystems that was 20 TB in size?.

Lustre 250 MB/s for provisioned for 20 TiB our account was charged \$3,386 / month

Due to these significant costs, we do not recommend using Lustre, but instead recommend the Beeond filesystem, that is free.

CMAQv4+ CONUS Benchmark Tutorial using 12US1 Domain

### 4.5.1 Use Cycle Cloud with CMAQv5.4+ software and 12US1 Benchmark data.

Step by step instructions for running the CMAQ 12US1 Benchmark for 2 days on a Cycle Cloud using beeond parallel filesystem for input data. Note, cpus are required to create the beeond shared filesystem, to copy the data in, and copy the data out. Therefore, it is necessary to leave some cpus available for this work, and to not use all of the cpus in the CMAQ domain decomposition (NPCOLxNPROW)

Input files are \*.nc (uncompressed netCDF)

### Use the files under the following directory to set up the CycleCloud Cluster to use beeond.

Using a modified version of the instructions available on this Blog Post, updated for the new CycleCloud version 8.5 and slurm 22.05.10.

Full Beeond: BeeGFS on Demand User Manual

Edit the cluster and use the Cloud-init option for your CycleCloud to install the code in the file: beeond-cloud-init-almalinux8.5-HBv3

Do not use the Apply to all option. Select Scheduler and copy and paste the contents of scheduler-cloud-init.sh obtained from github as follows:

```
wget https://raw.githubusercontent.com/CMASCenter/cyclecloud-cmaq/main/install_scripts/
↪beeond/scheduler-cloud-init.sh
```

Select hpc and copy and paste the contents of hpc-cloud-init.sh into the shell

```
wget https://raw.githubusercontent.com/CMASCenter/cyclecloud-cmaq/main/install_scripts/
↪ becond/hpc-cloud-init.sh
```

Save this setting, and then terminate and then restart the cluster.

### 4.5.2 Log into the new cluster

To find this IP address you need to go to the webpage for your Azure CycleCloud Clusters: <https://IP-address/home>

Double Click Scheduler, Under view details double, click scheduler, and a pop-up window will appear Click on the connect button in the upper right corner. Copy and past the login command that is provided. It will have the following syntax:

```
ssh -Y $USER@IP-address
```

Make the /shared/build directory and change ownership from root to your account.

```
sudo mkdir /shared/build
sudo chown $USER /shared/build
```

Make the /shared/cyclecloud directory and change ownership from root to your account.

```
sudo mkdir /shared/cyclecloud-cmaq
sudo chown $USER /shared/cyclecloud-cmaq
```

Install the cyclecloud-cmaq repo

```
cd /shared
git clone -b main https://github.com/CMASCenter/cyclecloud-cmaq.git cyclecloud-cmaq
```

Make the /shared/data directory and change ownership to your account

```
sudo mkdir /shared/data
sudo chown $USER /shared/data
```

Create the output directory

```
mkdir -p /shared/data/output
```

The becond filesystem will be created using the 1.8 T nvme disks that are on the compute nodes when the run script is submitted. If you use two nodes, the shared becond filesysetm will be a size of 3.5 T.

```
beegfs_ondemand 3.5T 103M 3.5T 1% /mnt/becond
```

### 4.5.3 Download the input data from the AWS Open Data CMAS Data Warehouse using the aws copy command.

Install AWS CLI to obtain data from AWS S3 Bucket

```
cd /shared/build
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```



Download the data

```
cd /shared/cyclecloud-cmaq/s3_scripts/
./s3_copy_nosign_2018_12US1_conus_cmas_opendata_to_shared_20171222_cb6r5_uncompressed.csh
```

#### 4.5.4 Verify Input Data

```
cd /shared/data/2018_12US1
du -h
```

Output

```
40K      ./CMAQ_v54+_cb6r5_scripts
44K      ./CMAQ_v54+_cracmm_scripts
1.5G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/cmv_c1c2_12
2.3G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/cmv_c3_12
3.3G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/merged_nobeis_norwc
1.1G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/othpt
990M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/pt_oilgas
4.5M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptagfire
206M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptegu
14M      ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire
1004K    ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire_grass
944K     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire_othna
4.7G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptnonipm
14G      ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready
2.9G     ./emis/cb6r3_ae6_20200131_MYR/premerged/rwc
2.9G     ./emis/cb6r3_ae6_20200131_MYR/premerged
17G      ./emis/cb6r3_ae6_20200131_MYR
60K      ./emis/emis_dates
17G      ./emis
2.2G     ./epic
13G      ./icbc/CMAQv54_2018_108NHEMI_M3DRY
17G      ./icbc
26G      ./met/WRFv4.3.3_LTNG_MCIP5.3.3_compressed
26G      ./met
3.9G     ./misc
697M     ./surface
66G      .
```

#### 4.5.5 Install CMAQv5.4+

Change directories to install and build the libraries and CMAQ

Install netCDF C and Fortran Libraries

```
cd /shared/cyclecloud-cmaq
./gcc_netcdf_cluster.csh
cp dot.cshrc ~/.cshrc
```

Execute the .cshrc shell

```
csh
env
```

Verify the LD\_LIBRARY\_PATH environment variable

```
echo $LD_LIBRARY_PATH
```

Output

```
/opt/openmpi-4.1.5/lib:/opt/gcc-9.2.0/lib64:/shared/build/netcdf/lib
```

Install I/O API Library

```
cd /shared/cyclecloud-cmaq
./gcc_ioapi_cluster.csh
```

Build CMAQ

```
./gcc_cmaq_v54+.csh
```

## 4.5.6 Copy and Examine CMAQ Run Scripts

Obtain a copy of the CMAQ run script that has been edited to use the /mnt/beeond shared filesystem.

```
cp /shared/cyclecloud-cmaq/run_scripts/CMAQ_v54+_beeond/run_cctm_2018_12US1_v54_cb6r5_
↪ae6.20171222.*.ncclassic.csh /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/
```

---

**Note:** The time that it takes the 2 day CONUS benchmark to run will vary based on the number of CPUs used, and the compute node that is being used, and what disks are used for the I/O (shared, beeond or lustre). The timings reported below are from the beeond filesystem on HB120v3 compute nodes.

---

Examine how the run script is configured

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/
head -n 30 /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/run_cctm_2018_12US1_v54_cb6r5_
↪ae6.20171222.2x96.ncclassic.csh
```

```
#!/bin/csh -f

## For CycleCloud 120pe
## data on /lustre data directory
## https://dataverse.unc.edu/dataset.xhtml?persistentId=doi:10.15139/S3/LDTWKH
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=96
#SBATCH --exclusive
#SBATCH -J CMAQ
#SBATCH -o /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/run_cctm5.4+_Bench_2018_12US1_
↪cb6r5_ae6_20200131_MYR.192.16x12pe.2days.20171222start.2x96.log
#SBATCH -e /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/run_cctm5.4+_Bench_2018_12US1_
↪cb6r5_ae6_20200131_MYR.192.16x12pe.2days.20171222start.2x96.log
#SBATCH -p hpc
```

(continues on next page)

(continued from previous page)

```
##SBATCH --constraint=BEEOND
###SBATCH --beond
```

The “beond start” command was added to the top of the job script, and “beond stop” to the end. Note, the -m 2 option is specific to using two nodes, this should be modified to match the number of nodes specified in the `SBATCH --nodes=` command above:

```
# =====
#> Start Beeond filesystem
# =====

beond start -P -m 2 -n /shared/home/$SLURM_JOB_USER/nodefile-$SLURM_JOB_ID -d /mnt/
↪nvme -c /mnt/beond -f /etc/beegfs

## Copy files to /mnt/beond, note, it may take 5 minutes to prepare the /mnt/beond
↪filesystem and to copy the data

beond-cp stagein -n ~/nodefile-$SLURM_JOB_ID -g /shared/data/2018_12US1 -l /mnt/beond/
↪data/2018_12US1
```

**Note:** In this run script, slurm or SBATCH requests 2 nodes, each node with 96 pes, or  $2 \times 96 = 192$  pes

Verify that the NPCOL and NPROW settings in the script are configured to match what is being requested in the SBATCH commands that tell slurm how many compute nodes to provision. In this case, to run CMAQ using on 192 cpus (SBATCH --nodes=2 and --ntasks-per-node=96), use NPCOL=16 and NPROW=12.

```
grep NPCOL run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic.csh
```

Output:

```
#> Horizontal domain decomposition
  setenv NPCOL_NPROW "1 1"; set NPROCS    = 1 # single processor setting
  @ NPCOL  = 16; @ NPROW = 12
  @ NPROCS = $NPCOL * $NPROW
  setenv NPCOL_NPROW "$NPCOL $NPROW";
```

Verify that the modules are loaded

```
module list
```

Currently Loaded Modulefiles:

```
1) gcc-9.2.0  2) mpi/openmpi-4.1.5
```

## 4.5.7 Submit Job to Slurm Queue to run CMAQ on beond

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic.csh
```

### Check status of run

```
squeue
```

Output:

```
[lizadams@ip-0A0A0000A scripts]$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	ODELIST(Reason)
1	hpc	CMAQ	lizadams	CF	0:01	2	beondtest2-hpc-[1-2]

It takes about 5-8 minutes for the compute nodes to spin up, after the nodes are available, the status will change from CF to R.

### Successfully started run

```
squeue
```

Output:

```
↩ hpc- [3-4]
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	ODELIST(Reason)
25	hpc	CMAQ	lizadams	R	56:26	2	CycleCloud8-5Beond-

### Check that the /mnt/beeond filesystem has been created on the compute nodes

Login to the compute node by getting the IP address of the compute node. To find this IP address you need to go to the webpage where you configured the Azure CycleCloud Clusters: <https://IP-address/home> Double click on hpc to show the view details panel. Double click on one of the hpc compute nodes, and a pop-up window will appear, click on connect to obtain the IP address of the compute node.

```
ssh $USER@IP-address-compute-node
```

If you are running on 2 compute nodes, then there are two 1.8 TB /nvme drives. Beeond will create a shared 3.5 TB shared drive on /mnt/beeond

```
df -h
```

Output:

```
[lizadams@ip-0A0A0000B ~]$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	225G	0	225G	0%	/dev
tmpfs	225G	0	225G	0%	/dev/shm
tmpfs	225G	18M	225G	1%	/run

(continues on next page)

(continued from previous page)

```

tmpfs                225G      0 225G   0% /sys/fs/cgroup
/dev/sda2             59G    27G  33G  45% /
/dev/sda1             994M   209M  786M  21% /boot
/dev/sda15            495M    5.9M  489M   2% /boot/efi
/dev/sdb1             472G   216K  448G   1% /mnt
/dev/md10             1.8T    16G  1.8T   1% /mnt/nvme
10.10.0.10:/sched     30G   247M   30G   1% /sched
10.10.0.10:/shared  1000G    95G  906G  10% /shared
tmpfs                 45G      0  45G   0% /run/user/20001
beegfs_ondemand       3.5T    31G  3.5T   1% /mnt/beeond

```

### Check on the log file status

```

grep -i 'Processing completed.' run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.192.
↪16x12pe.2days.20171222start.2x96.log

```

Output:

```

Processing completed... 5.8654 seconds
Processing completed... 5.8665 seconds
Processing completed... 5.8610 seconds
Processing completed... 5.8422 seconds
Processing completed... 5.8657 seconds
Processing completed... 5.8616 seconds
Processing completed... 5.8824 seconds
Processing completed... 5.8581 seconds
Processing completed... 5.8653 seconds
Processing completed... 5.8961 seconds
Processing completed... 7.9473 seconds
Processing completed... 5.4089 seconds
Processing completed... 5.8996 seconds
Processing completed... 5.9659 seconds
Processing completed... 5.9462 seconds
Processing completed... 5.8966 seconds
Processing completed... 5.9326 seconds

```

Once the job has completed running the two day benchmark check the log file for the timings.

```

tail -n 30 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.192.16x12pe.2days.
↪20171222start.2x96.log

```

Output:

```

=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day:   2017-12-23
Number of Simulation Days: 2

```

(continues on next page)

(continued from previous page)

```

Domain Name:          12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers:     35
Number of Processes:  192
  All times are in seconds.

```

```

Num  Day      Wall Time
01   2017-12-22  1966.4
02   2017-12-23  2170.2
    Total Time = 4136.60
    Avg. Time = 2068.30
INFO: Using status information file: /tmp/beeond.tmp
INFO: Checking reachability of host 10.10.0.5
INFO: Checking reachability of host 10.10.0.7
INFO: Unmounting file system on host: 10.10.0.5
sudo: do_stoplocal: command not found
INFO: Unmounting file system on host: 10.10.0.7
sudo: do_stoplocal: command not found
INFO: Stopping remaining processes on host: 10.10.0.5
INFO: Stopping remaining processes on host: 10.10.0.7
INFO: Deleting status file on host: 10.10.0.5
INFO: Deleting status file on host: 10.10.0.7

```

## 4.5.8 submit job to run on 1 node x 96 processors

```
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x96.ncclassic.csh
```

Check result after job has finished

```

tail -n 30 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.96.8x12pe.2days.
→20171222start.1x96.log

```

Output

```

=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day:   2017-12-23
Number of Simulation Days: 2
Domain Name:          12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers:     35
Number of Processes:  96
  All times are in seconds.

Num  Day      Wall Time
01   2017-12-22  3278.9
02   2017-12-23  3800.7

```

(continues on next page)

(continued from previous page)

```
Total Time = 7079.60
Avg. Time = 3539.80
```

#### 4.5.9 Submit job to run on 3 nodes

```
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.3x96.ncclassic.csh
```

Verify the size of the becond filesystem when using 3 nodes is 5.3 T.

```
ssh $USER@IP-address-compute-node
```

```
df -h
```

Output:

```
df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        225G   0    225G   0% /dev
tmpfs           225G  789M   224G   1% /dev/shm
tmpfs           225G   18M   225G   1% /run
tmpfs           225G   0    225G   0% /sys/fs/cgroup
/dev/sda2        59G   27G   33G   45% /
/dev/sda1        994M  209M  786M  21% /boot
/dev/sda15       495M   5.9M  489M   2% /boot/efi
/dev/sdb1        472G  216K  448G   1% /mnt
/dev/md10        1.8T   39G   1.8T   3% /mnt/nvme
10.10.0.10:/sched 30G  247M   30G   1% /sched
10.10.0.10:/shared 1000G 223G  778G  23% /shared
tmpfs           45G   0    45G   0% /run/user/20001
beegfs_ondemand  5.3T  116G   5.2T   3% /mnt/beeond
```

#### 4.5.10 Check how quickly the processing is being completed

```
grep -i 'Processing completed' run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.288.
↪16x18pe.2days.20171222start.3x96.log
```

Output

```
Processing completed... 7.4152 seconds
Processing completed... 5.7284 seconds
Processing completed... 5.6439 seconds
Processing completed... 5.5742 seconds
Processing completed... 5.6011 seconds
Processing completed... 5.5687 seconds
Processing completed... 5.5505 seconds
Processing completed... 5.5686 seconds
Processing completed... 5.5193 seconds
Processing completed... 5.5192 seconds
```

(continues on next page)

(continued from previous page)

Processing completed...	5.4985 seconds
Processing completed...	6.7259 seconds
Processing completed...	6.3606 seconds
Processing completed...	5.5312 seconds

4.5.11 Check results when job has completed successfully

```
tail -n 30 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.288.16x18pe.2days.  
↪20171222start.3x96.log
```

Output

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 288
All times are in seconds.

Num Day Wall Time
01 2017-12-22 1588.4
02 2017-12-23 1721.8
Total Time = 3310.20
Avg. Time = 1655.10
```

4.5.12 Check to see if spot VMs are available

Spot Pricing Search for HB120rs\_v3

4.5.13 Unsuccessful slurm status messages

The NODELIST reason “Nodes required for the job are DOWN...” Will be generated if a batch is submitted prior to the previous job successfully terminating the nodes Wait 5 -10 minutes and see if the status changes from PD (pending) to CF (configuring).

```
squeue
```

Output

squeue									
	JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)	

(continues on next page)



(continued from previous page)

```

3 hpc CMAQ lizadams PD 0:00 3 (Nodes required for
↪ job are DOWN, DRAINED or reserved for jobs in higher priority partitions)

```

The NODELIST reason “launch failed requested held” requires that the job be canceled. Note, if you get this message, it may result in the HPC compute nodes staying up and charging, without running the job, so is important to cancel the job using scancel.

```

squeue
      JOBID PARTITION   NAME     USER ST       TIME  NODES NODELIST(REASON)
      3      hpc      CMAQ lizadams PD      0:00      3 (launch failed,
↪ requested held)

```

```
scancel 3
```

Confirm that the HPC VMs are deleted by viewing the CycleCloud webpage.

#### 4.5.14 Change to HB176\_v4 compute node

Terminate the cluster

Edit the cluster configuration

Select HB174\_v4 for the HPC compute nodes

Start the cluster

Submit following run script

```

cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x176.ncclassic.beeond.csh

```

Login to the compute node to verify beeond was created

```
df -h
```

output

```

df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        378G   0    378G   0% /dev
tmpfs           378G   0    378G   0% /dev/shm
tmpfs           378G  18M   378G   1% /run
tmpfs           378G   0    378G   0% /sys/fs/cgroup
/dev/sda2        59G   27G   33G   45% /
/dev/sda1       994M  209M  786M  21% /boot
/dev/sda15       495M   5.9M  489M   2% /boot/efi
/dev/sdb1       472G  216K  448G   1% /mnt
/dev/md10        3.5T   28G   3.5T   1% /mnt/nvme
10.10.0.10:/sched 30G  247M   30G   1% /sched
10.10.0.10:/shared 1000G 421G  580G  43% /shared
tmpfs           76G   0    76G   0% /run/user/20001
beegfs_ondemand 3.5T   28G   3.5T   1% /mnt/beeond

```

Note, some of these instructions do not work, as azslurm is not found on the AlmaLinux8 OS. Additional instructions are available here: [Azure CycleCloud 8 help for Slurm](#)

### 4.5.15 To recover from failure use the terminate cluster option

If the job does not begin to configure, then you may need to terminate and then restart the cluster.

The terminate option does not delete the software, it only shuts down the scheduler and compute nodes. The terminate option is equivalent to stopping the cluster. Once it has been stopped, the cluster can be restarted using the Start button.

### 4.5.16 If SLURM jobs are in a bad state

When the job fails the compute nodes are put into an unusable Slurm state. You can try to reset them with scontrol like so:

```
sudo scontrol update nodename=hpc-[1-2] state=resume
```

If that doesn't reset them you can try the CycleCloud command to shutdown the nodes (suspend):

```
sudo -i azslurm suspend --node-list hpc-[1-2]
```

Likewise you can start nodes that are in a "bad" Slurm state like this:

```
sudo -i azslurm resume --node-list hpc-[1-2]
```

In all the cases above replace hpc-[1-2] with your specific node list.

### 4.5.17 Run DESID CMAQ on hbv3\_120 using the becond filesystem

#### Run CMAQ for DESID

#### Edit the DESID Namelist

##### 1. Edit the CMAQ DESID Chemical Species Control File

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc
cp CMAQ_Control_DESID_cb6r5_ae7_aq.nml CMAQ_Control_DESID_cb6r5_ae7_aq_RED_EGU_POINT_NY.
↪nml
vi CMAQ_Control_DESID_cb6r5_ae7_aq_RED_EGU_POINT_NY.nml
```

##### 2. Add the following lines to the bottom of the file according to the DESID Tutorial Instructions

[https://github.com/USEPA/CMAQ/blob/main/DOCS/Users\\_Guide/Tutorials/CMAQ\\_UG\\_tutorial\\_emissions.md#scale\\_stream](https://github.com/USEPA/CMAQ/blob/main/DOCS/Users_Guide/Tutorials/CMAQ_UG_tutorial_emissions.md#scale_stream)  
(place the line before the / file marker)

```
! PT_EGU Emissions Scaling reduce PT_EGU emissions in New York by 25%. Note, to_
↪reduce the emissions by 25% we use DESID to multiply what had been 100% emissions by .
↪75, so that the resulting emissions is reduced by 25%.
'NY' , 'PT_EGU' , 'All' , 'All' , 'All' , .75 , 'UNIT', 'o',
```

##### 3. Activate DESID Diagnostics

Create a DESID Control File and edit it to define NY as a region, and activate DESID emissions diagnostics Define NY as a region in the DESID Region Definitions

```
cp CMAQ_Control_DESID.nml CMAQ_Control_DESID_RED_EGU_POINT_NY.nml
vi CMAQ_Control_DESID_RED_EGU_POINT_NY.nml &
```

Modify the following section to use the NY region that is specified in the CMAQ\_MASKS file, note the CMAQ\_MASKS file is defined in the DESID Run script.

```
&Desid_RegionDef
Desid_Reg_nml =
!           Region Label   | File_Label   | Variable on File
!           'EVERYWHERE'   | 'N/A'        | 'N/A',
!           'NY'           | 'CMAQ_MASKS' | 'NY',
!<Example> 'ALL'         | 'ISAM_REGIONS' | 'ALL',
/
```

#### 4. Create two stream family definitions, one that includes all point source emissions, and the second that only contains PT\_EGU

```
!-----!
! Emissions Scaling Family Definitions                                     !
!   This component includes definitions for families of emission streams and !
!   region combinations.                                                !
!-----!
&Desid_StreamFamVars
Desid_N_Stream_Fams = 2           ! Exact number of stream family definitions
Desid_Max_Stream_Fam_Members = 20 ! Larger than the number of streams on all
                                   ! family definitions
/

&Desid_StreamFam
! For emission streams available in several run scripts under CCTM/scripts

StreamFamilyName(1)      = 'PT_SOURCES'
StreamFamilyMembers(1,1:8)= 'PT_NONEGU', 'PT_OTHER', 'PT_AGFIRES', 'PT_FIRES', 'PT_
→RXFIRES', 'PT_OTHFIRES', 'PT_OILGAS', 'PT_CMV_C1C2'

StreamFamilyName(2)      = 'PT_EGUS'
StreamFamilyMembers(2,1:1)= 'PT_EGU'
```

#### 5. activate DESID diagnostics to report the reduction in PT\_EGU emissions.

Note, if you define only one diagnostic rule, you must comment out all other rules.

```
&Desid_DiagVars
Desid_N_Diag_Rules = 1           ! Exact Number of Diagnostic Rules Below
Desid_Max_Diag_Streams=20        ! Maximum number of species variables on all rules
                                   ! below (do not count expansions)
Desid_Max_Diag_Spec = 80         ! Maximum number of species variables on all rules
                                   ! below (do not count expansions)
/
```

```
! Create a diagnostic of the sum of the components of the PT_SOURCES
! family (defined in the stream family section). This file will be column sums
! and will include all the emitted species as long as they appear on at least
```

(continues on next page)

(continued from previous page)

```
! one of the streams within PT_SOURCES.
```

```
Desid_Diag_Streams_Nml(1,:)= 'PT_EGUS'
Desid_Diag_Fmt_Nml(1)      = 'COLSUM'
Desid_Diag_Spec_Nml(1,:)   = 'ALL'
```

#### 6. Verify that the settings are correct by comparing to the version in the github repo directory

```
diff CMAQ_Control_DESID_RED_EGU_POINT_NY.nml /shared/pcluster-cmaq/qa_scripts/workshop/
↪CMAQ_Control_DESID_RED_EGU_POINT_NY.nml
```

### Edit runscrip to use DESID Namelist

#### 1. Copy the Run script and edit it to use the DESID namelist files

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/
cp run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic.csh run_cctm_2018_12US1_v54_
↪cb6r5_ae6.20171222.2x96.ncclassic_DESID_RED_NY.csh
```

#### 2. Change APPL to a new name

```
vi run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic_DESID_RED_NY.csh
```

Change APPL

```
set APPL      = 2018_12US1_DESID_REDUCE      #> Application Name (e.g. Gridname)
```

#### 3. Verify the following emission stream names match the names used in the DESID namelist.

```
grep STK_EMIS_LAB_00 run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic_DESID_RED_
↪NY.csh
```

Output

```
setenv STK_EMIS_LAB_001 PT_NONEGU
setenv STK_EMIS_LAB_002 PT_EGU
setenv STK_EMIS_LAB_003 PT_OTHER
setenv STK_EMIS_LAB_004 PT_AGFIRE
setenv STK_EMIS_LAB_005 PT_FIRE
setenv STK_EMIS_LAB_006 PT_RXFIRE
setenv STK_EMIS_LAB_007 PT_OTHFIRE
setenv STK_EMIS_LAB_008 PT_OILGAS
setenv STK_EMIS_LAB_009 PT_CMV_C1C2
```

#### 4. Compare the above settings to those used in the Emission Stream Family defined in the DESID Namelist.

```
grep -A 2 -B 2 StreamFamilyMembers ./BLD_CCTM_v54_gcc/CMAQ_Control_DESID_RED_EGU_POINT_
↪NY.nml
```

Output

```
StreamFamilyName(1)      = 'PT_SOURCES'
StreamFamilyMembers(1,1:4)= 'PT_NONEGU', 'PT_OTHER', 'PT_AGFIRES', 'PT_FIRES', 'PT_
↪RXFIRES', 'PT_OTHFIRES', 'PT_OILGAS', 'PT_CMV_C1C2'

StreamFamilyName(2)      = 'PT_EGUS'
StreamFamilyMembers(2,1:1)= 'PT_EGU'
```

**Note:** CMAQ won't crash if the stream name in CMAQ\_Control\_DESID\_\_RED\_EGU\_POINT\_NY.nml was set incorrectly. CMAQ just ignores the incorrect stream name and won't apply scaling.

#### 5. Update the DESID namelist file names in the run script to use the Reduced PT\_EGU and diagnostic instructions.

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
vi run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic_DESID_RED_NY.csh
```

Modify the namelist setting to use the DESID namelist:

```
setenv DESID_CTRL_NML ${BLD}/CMAQ_Control_DESID_RED_EGU_POINT_NY.nml
setenv DESID_CHEM_CTRL_NML ${BLD}/CMAQ_Control_DESID_${MECH}_RED_EGU_POINT_NY.nml
```

#### 6. Update the Spatial Masks for Emissions Scaling to use a file that contains state definitions for New York.

```
#> Spatial Masks For Emissions Scaling
setenv CMAQ_MASKS $SZpath/GRIDMASK_STATES_12US1.nc
```

#### 7. Verify that the file contains New York

```
ncdump /shared/data/2018_12US1/surface/GRIDMASK_STATES_12US1.nc | grep NY
```

Output

```
float NY(TSTEP, LAY, ROW, COL) ;
    NY:long_name = "NY          " ;
    NY:units = "fraction      " ;
    NY:var_desc = "NY fractional area per grid cell"
```

#### Edit the output directory name in the

```
# need to make the output directory prior to the beeond-cp
mkdir -p /shared/data/output/output_v54_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_DESID_
↪REDUCE/LOGS

beeond-cp stagein -n ~/nodefile-$SLURM_JOB_ID -g /shared/data/output/output_v54_cb6r5_
↪ae7_aq_WR413_MYR_gcc_2018_12US1_DESID_REDUCE -l /mnt/beeond/data/output/output_v54_
↪cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_DESID_REDUCE
```

## Run CMAQ using DESID

**Note:** The CMAQ run script has been configured to run on 192 cores (2 compute nodes of hb120v3 with 96 cores/node)

### 1. Change directories to the run script location

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
```

### 2. Submit the Run script to the SLURM queue

```
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic_DESID_RED_NY.csh
```

### 3. Check the status of the job

```
squeue
```

Output

```
1 hpc CMAQ lizadams CF 0:11 2 becondtest2-hpc-[1-2]
```

Wait for the status to change from CF to R

### 4. Login to the compute node, install and run htop

```
ssh -Y IP-address
sudo yum install -y htop
htop
```

```
0[100.] 11[100.] 22[100.] 33[0.0%] 44[100.] 55[100.] 66[100.] 77[0.6%] 88[100.] 99[100.] 110[100.] 121[0.0%] 132[100.] 143[100.] 154[100.] 165[0.0%]
1[100.] 12[100.] 23[100.] 34[0.0%] 45[100.] 56[100.] 67[100.] 78[0.0%] 89[100.] 100[100.] 111[100.] 122[0.0%] 133[100.] 144[100.] 155[100.] 166[0.0%]
2[100.] 13[100.] 24[0.0%] 35[0.0%] 46[100.] 57[100.] 68[0.0%] 79[0.0%] 90[100.] 101[100.] 112[0.0%] 123[0.0%] 134[100.] 145[100.] 156[0.0%] 167[0.0%]
3[100.] 14[100.] 25[0.0%] 36[0.0%] 47[100.] 58[100.] 69[0.0%] 80[0.0%] 91[100.] 102[100.] 113[0.0%] 124[0.0%] 135[100.] 146[100.] 157[0.0%] 168[0.0%]
4[100.] 15[100.] 26[0.0%] 37[0.0%] 48[100.] 59[100.] 70[0.0%] 81[0.0%] 92[100.] 103[100.] 114[0.0%] 125[0.0%] 136[100.] 147[100.] 158[0.0%] 169[0.0%]
5[100.] 16[100.] 27[0.0%] 38[0.0%] 49[100.] 60[100.] 71[0.0%] 82[0.0%] 93[100.] 104[100.] 115[6.2%] 126[0.0%] 137[100.] 148[100.] 159[0.0%] 170[0.0%]
6[100.] 17[100.] 28[0.0%] 39[0.0%] 50[100.] 61[100.] 72[0.0%] 83[0.0%] 94[100.] 105[100.] 116[0.0%] 127[0.0%] 138[100.] 149[100.] 160[0.0%] 171[0.0%]
7[100.] 18[100.] 29[0.0%] 40[0.0%] 51[100.] 62[100.] 73[1.2%] 84[0.0%] 95[100.] 106[100.] 117[0.0%] 128[0.0%] 139[100.] 150[100.] 161[0.0%] 172[0.0%]
8[100.] 19[100.] 30[0.0%] 41[0.0%] 52[100.] 63[100.] 74[0.0%] 85[0.0%] 96[100.] 107[100.] 118[0.0%] 129[0.0%] 140[100.] 151[100.] 162[0.0%] 173[0.0%]
9[100.] 20[100.] 31[0.0%] 42[0.0%] 53[100.] 64[100.] 75[0.0%] 86[0.0%] 97[100.] 108[100.] 119[0.0%] 130[0.0%] 141[100.] 152[100.] 163[0.0%] 174[0.0%]
10[100.] 21[100.] 32[0.0%] 43[0.0%] 54[100.] 65[100.] 76[0.0%] 87[0.0%] 98[100.] 109[100.] 120[0.0%] 131[0.0%] 142[100.] 153[100.] 164[0.0%] 175[0.0%]
Mem[|||||] 98.8G/756G Tasks: 156, 1612 thr, 1746 kthr; 98 running
Swp[|] 0K/0K Load average: 67.40 21.92 8.39
Uptime: 00:09:46
```

Main		PID USER		PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
55883	lizadams	20	0	2387M	939M	59720	R	99.5	0.1	0:54.03			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55884	lizadams	20	0	2418M	1033M	59860	R	99.5	0.1	0:53.94			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55887	lizadams	20	0	2335M	946M	59732	R	99.5	0.1	0:54.16			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55889	lizadams	20	0	2365M	978M	59540	R	99.5	0.1	0:55.00			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55892	lizadams	20	0	2306M	922M	58792	R	99.5	0.1	0:54.75			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55893	lizadams	20	0	2380M	987M	59596	R	99.5	0.1	0:54.38			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55897	lizadams	20	0	2307M	923M	59796	R	99.5	0.1	0:55.01			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55910	lizadams	20	0	2331M	931M	60696	R	99.5	0.1	0:53.82			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55916	lizadams	20	0	2384M	984M	60644	R	99.5	0.1	0:49.83			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55917	lizadams	20	0	2337M	935M	60404	R	99.5	0.1	0:53.72			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55920	lizadams	20	0	2362M	959M	60260	R	99.5	0.1	0:54.01			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55923	lizadams	20	0	2398M	998M	60768	R	99.5	0.1	0:48.59			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55927	lizadams	20	0	2635M	1235M	60580	R	99.5	0.2	0:53.98			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55928	lizadams	20	0	2407M	1015M	60568	R	99.5	0.1	0:54.43			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55929	lizadams	20	0	2319M	916M	60480	R	99.5	0.1	0:53.99			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55931	lizadams	20	0	2356M	956M	60496	R	99.5	0.1	0:54.11			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55932	lizadams	20	0	2463M	1071M	60528	R	99.5	0.1	0:53.83			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55935	lizadams	20	0	2633M	1244M	60856	R	99.5	0.2	0:50.19			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55880	lizadams	20	0	2379M	942M	59572	R	98.9	0.1	0:53.92			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55881	lizadams	20	0	2387M	940M	60228	R	98.9	0.1	0:53.98			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55882	lizadams	20	0	2391M	942M	60268	R	98.9	0.1	0:53.98			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55885	lizadams	20	0	2321M	934M	59760	R	98.9	0.1	0:47.21			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55886	lizadams	20	0	2322M	929M	59596	R	98.9	0.1	0:54.04			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55888	lizadams	20	0	2372M	980M	59732	R	98.9	0.1	0:53.71			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55890	lizadams	20	0	2360M	967M	59544	R	98.9	0.1	0:53.91			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55891	lizadams	20	0	2360M	968M	59516	R	98.9	0.1	0:53.89			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55894	lizadams	20	0	2324M	935M	59648	R	98.9	0.1	0:55.12			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55895	lizadams	20	0	2341M	935M	60636	R	98.9	0.1	0:54.13			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
55896	lizadams	20	0	2312M	923M	59656	R	98.9	0.1	0:54.84			/shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe

After the becond copy copies the input data from /shared/data to the /mnt/becond/data then CMAQ should start running. Htop should show that 96 processes are running and that 96G out of 756 G of memory is being used. ~

## Review Log file from DESID run

**Note:** The CMAQ run script has been configured to run on 192 cores (2 compute nodes of hb120v3 with 96 cores/node)

1. Review the Emissions Scaling Report Section in the CTM\_LOG File to verify that for the NY region, the EGU emissions were scaled by 75%

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
grep -A 20 'Stream Type: "Point Emissions File 2' CTM_LOG_001*
```

Output:

Stream Type: "Point Emissions File 2" | Sector Label: PT\_EGU (04)

Table of Aerosol Size Distributions Available for Use Sector-Wide.  
Note that Mode 1 is reserved for gas-phase species and emission variable.

Number Em. Var. Mode Reference Mode (see AERO\_DATA.F)

2	FINE	FINE_REF
3	COARSE	COARSE_REF

	CMAQ Species	Phase/Mode	Em. Var.	Region	Op	ScaleFac	
↪Basis	FinalFac						
↪UNIT	NO2	GAS	NO2	EVERYWHERE	a	1.000	↪
↪UNIT	1.000			NY	o	0.750	↪
↪UNIT	0.750						
↪UNIT	NO	GAS	NO	EVERYWHERE	a	1.000	↪
↪UNIT	1.000			NY	o	0.750	↪
↪UNIT	0.750						
↪UNIT	HONO	GAS	HONO	EVERYWHERE	a	1.000	↪
↪UNIT	1.000			NY	o	0.750	↪
↪UNIT	0.750						
↪UNIT	SO2	GAS	SO2	EVERYWHERE	a	1.000	↪
↪UNIT	1.000			NY	o	0.750	↪
↪UNIT	0.750						
↪UNIT	SULF	GAS	SULF	EVERYWHERE	a	0.000	↪
↪UNIT	0.000			NY	o	0.750	↪
↪UNIT	0.750						

### #Post-process and QA

Post-processing CMAQ Run, Install R and packages Instructions to install R and packages for QA of CMAQ difference in output between two runs.



## 4.6 Scripts to run combine and post processing

### 4.6.1 Build the POST processing routines

Run the bldit script for combine.

```
cd /shared/build/openmpi_gcc/CMAQ_v54/POST/combine/scripts
./bldit_combine.csh gcc |& tee ./bldit_combine.gcc.log
```

Run the bldit script for calc\_tmetric

```
cd /shared/build/openmpi_gcc/CMAQ_v54/POST/calc_tmetric/scripts
./bldit_calc_tmetric.csh gcc |& tee ./bldit_calc_tmetric.gcc.log
```

Run the bldit script for hr2day

```
cd /shared/build/openmpi_gcc/CMAQ_v54/POST/hr2day/scripts
./bldit_hr2day.csh gcc |& tee ./bldit_hr2day.gcc.log
```

## 4.7 Scripts to post-process CMAQ output

### 4.7.1 Note, the post-processing analysis should be done on the head node

Ideally the post-processing would be done on the HTC queue, but the R packages were installed to the head node system library and were not accessible to the compute nodes. Installing the R software and packages to the /shared volume will be investigated in future work.

Verify that the compute nodes are no longer running if you have completed all of the benchmark runs

squeue

You should see that no jobs are running.

Show compute nodes

scontrol show nodes

### 4.7.2 Edit, Build and Run the POST processing routines

You need to run the post processing scripts for every benchmark case.

```
cd /shared/build/openmpi_gcc/CMAQ_v54/POST/combine/scripts
cp /shared/cyclecloud-cmaq/run_scripts/run_combine_conus.csh .
```

Examine the run script

```
cat run_combine_conus.csh
```

The post processing scripts are set up for a specific case, example:

```
setenv APPL Bench_2016_12SE1
```

note, you will need to change the sed command to a different configuration if you ran another case, example:

```
setenv APPL 2018_12US1_3x96
```



Modify the scripts using the instructions below.

```
setenv DIR /shared/build/openmpi_gcc/CMAQ_v54/

cd $DIR/POST/combine/scripts
cp run_combine.csh run_combine_conus.csh
sed -i 's/Bench_2016_12SE1/2018_12US1_3x96/g' run_combine_conus.csh
sed -i 's/intel/gcc/g' run_combine_conus.csh
sed -i 's/2016-07-01/2017-12-22/g' run_combine_conus.csh
sed -i 's/2016-07-14/2017-12-23/g' run_combine_conus.csh
sed -i 's/cb6r3_ae7_aq/cb6r5_ae7_aq/g' run_combine_conus.csh
sed -i 's/METCR03D_${YY$MM$DD}.nc/METCR03D_${YYYY$MM$DD}.nc/g' run_combine_conus.csh
sed -i 's/METCR02D_${YY$MM$DD}.nc/METCR02D_${YYYY$MM$DD}.nc/g' run_combine_conus.csh
sed -i 's/${VRSN}_${compilerString}_${APPL}/${VRSN}_${MECH}_WR413_MYR_${compilerString}_${APPL}/g' run_combine_conus.csh
setenv METDIR /shared/data/2018_12US1/met/WRFv4.3.3_LTNG_MCIP5.3.3_compressed/
setenv CMAQ_DATA /shared/data/output
./run_combine_conus.csh |& tee ./run_combine_conus.log

cd $DIR/POST/calc_tmetric/scripts
cp run_calc_tmetric.csh run_calc_tmetric_conus.csh
sed -i 's/Bench_2016_12SE1/2018_12US1_3x96/g' run_calc_tmetric_conus.csh
sed -i 's/intel/gcc/g' run_calc_tmetric_conus.csh
sed -i 's/201607/201712/g' run_calc_tmetric_conus.csh
sed -i 's/cb6r3_ae7_aq/cb6r5_ae7_aq/g' run_calc_tmetric_conus.csh
sed -i 's/${VRSN}_${compilerString}_${APPL}/${VRSN}_${MECH}_WR413_MYR_${compilerString}_${APPL}/g' run_calc_tmetric_conus.csh
setenv CMAQ_DATA /shared/data/output
./run_calc_tmetric_conus.csh |& tee ./run_calc_tmetric_conus.log

cd $DIR/POST/hr2day/scripts
cp run_hr2day.csh run_hr2day_conus.csh
sed -i 's/Bench_2016_12SE1/2018_12US1_3x96/g' run_hr2day_conus.csh
sed -i 's/intel/gcc/g' run_hr2day_conus.csh
sed -i 's/2016182/2017356/g' run_hr2day_conus.csh
sed -i 's/2016195/2017357/g' run_hr2day_conus.csh
sed -i 's/201607/201712/g' run_hr2day_conus.csh
sed -i 's/cb6r3_ae7_aq/cb6r5_ae7_aq/g' run_hr2day_conus.csh
sed -i 's/${VRSN}_${compilerString}_${APPL}/${VRSN}_${MECH}_WR413_MYR_${compilerString}_${APPL}/g' run_hr2day_conus.csh
sed -i 's/bldoverlay/hr2day/g' run_hr2day_conus.csh
setenv CMAQ_DATA /shared/data/output
./run_hr2day_conus.csh |& tee ./run_hr2day_conus.log
```

## 4.8 Install R, Rscript and Packages

Install R from source on the scheduler node into a local mylibs directory (/shared/build/R-4.3.3) If R is installed using yum install, then need to install R packages as sudo, otherwise the packages are missing dependencies. If R is installed under /usr/bin, then when you terminate the cluster, anything in the root or default install directory is deleted. Only files on the /shared and /home directory are retained.

Using R on HPC Clusters

The instruction to install using a package manager were obtained from the following reference: How to install R using package manager (not recommended for this application)

Use the following commands, and also install packages - note, see website above for full details:

Install R from source

cd /shared/build/

```
setenv R_VERSION 4.3.3
curl -O https://cran.rstudio.com/src/base/R-4/R-${R_VERSION}.tar.gz
tar -xzf R-${R_VERSION}.tar.gz
cd R-${R_VERSION}
```

Installing R from source

Install packages required.

```
sudo yum update -y
sudo yum install readline-devel -y
sudo yum install libX11-devel libXt-devel -y
sudo yum install bzip2-devel -y
sudo yum install pcre2-devel -y
sudo yum install libcurl-devel -y
sudo yum install java-1.8.0-openjdk -y
sudo yum install libjpeg libjpeg-devel -y
sudo yum install xorg-x11-server-Xvfb -y
```

Install ImageMagick to allow display back to your local computer.

```
sudo yum groupinstall "Development Tools" -y
sudo yum install ImageMagick -y
sudo yum install ImageMagick-devel -y
sudo yum install xauth -y
sudo yum install firefox -y
sudo yum install mesa-libGL -y
sudo yum install mesa-libGL-devel -y
```

Logout and then log back in using the ssh -option -Y

Install R in /shared/build directory

```
./configure --prefix=/shared/build/R/${R_VERSION} --enable-R-shlib --enable-memory-
↪profiling
make
make install
```

Verify that R is working

```
/shared/build/R/${R_VERSION}/bin/R --version
```

Output:

```
/shared/build/R/${R_VERSION}/bin/R --version
R version 4.3.3 (2024-02-29) -- "Angel Food Cake"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)
```

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under the terms of the
GNU General Public License versions 2 or 3.
For more information about these matters see
https://www.gnu.org/licenses/.
```

Install packages

```
sudo yum install epel-release -y
sudo yum config-manager --set-enabled powertools -y
```

Alternatively, you can install as root, to the /bin directory, but then if you terminate the cluster, you will need to reinstall R

```
sudo yum install R
R --version
```

Install packages as root - to make them available to all users (if installed R as root) Otherwise, install without sudo

```
sudo -i R
#or
R

install.packages("stringr")
install.packages("patchwork")
install.packages("dplyr")
install.packages("jpeg")
install.packages("ggrepel")
```

Had an issue installing ncdf4 see: <https://cirrus.ucsd.edu/~pierce/ncdf/> install ncdf4

ncdf4 REQUIRES the netcdf library be version 4 or above, AND installed with HDF-5 support (i.e., the netcdf library must be compiled with the `--enable-netcdf-4` flag). If you do not want to install the full version of netcdf-4 with HDF-5 support, then please install the old, deprecated ncdf package instead.

ERROR: configuration failed for package 'ncdf4'

- removing '/usr/lib64/R/library/ncdf4'

building netcdf with HDF5 support requires curl.

```
sudo yum install curl -y
sudo yum install libcurl-devel -y
```

Load the gcc and openmpi module before building the hdf5 enabled netcdf libraries.

```
module avail
module load mpi/openmpi-4.1.5
module load gcc-9.2.0
```

```
cd /shared/cyclecloud-cmaq
./gcc_install_hdf5.cyclecloud.csh
```

Need to specify the location of nc-config in your .cshrc

```
set path = ($path /shared/build/install/bin /shared/build/ioapi-3.2/Linux2_x86_64gfort /shared/build-hdf5/install/bin )
```

Run command to install ncdf4 package

```
cd /shared/cyclecloud-cmaq/qa_scripts/R_packages
```

```
R CMD INSTALL ncdf4_1.13.tar.gz --configure-args="--with-nc-config=/shared/build-hdf5/
install/bin/nc-config"
```

or to install to local directory

```
R CMD INSTALL ncdf4_1.13.tar.gz --configure-args="--with-nc-config=/shared/build-hdf5/
install/bin/nc-config" -l "/shared/build/R_libs"
```

Install additional packages as root so that all users will have access.

```
R
install.packages("fields")
install.packages("mapdata")
```

M3 package requires gdal

```
sudo yum install gdal -y
sudo yum install gdal-devel proj-devel -y
```

```
R
install.packages("sp")
```

```
cd /shared/cyclecloud-cmaq/qa_scripts/R_packages
R CMD INSTALL rgdal_1.6-7.tar.gz
```

```
R
install.packages("ggplot2")
```

```
cd /shared/cyclecloud-cmaq/qa_scripts/R_packages
R CMD INSTALL M3_0.3.tar.gz
```

If you are having X11 display issues:

[how-to-fix-x11-forwarding-request-failed-on-channel-0](#)

Make sure that you have Xquartz running on your local machine, and that you have given permission to display back from the cyclecloud server.

On your local terminal: `host +`

Test the display

```
display
```

## 4.9 Install Anaconda on the /shared/build directory

Follow instructions available here: [Install Anaconda on Linux](#)

```
cd /shared/build/
```

```
sudo yum install libXcomposite libXcursor libXi libXtst libXrandr alsa-lib mesa-libEGL_
↳ libXdamage mesa-libGL libXScrnSaver
curl -O https://repo.anaconda.com/archive/Anaconda3-2023.09-0-Linux-x86_64.sh
```

select installation directory as /shared/build/anaconda3

add to your .cshrc

```
set path = ($path /shared/build/anaconda3/bin)
# The base environment is activated by default
conda config --set auto_activate_base True
```

Install additional packages

```
conda install netcdf4
conda install pyproj
conda install cartopy
```

## 4.10 QA CMAQ

Quality Assurance: Comparing the output of two CMAQ runs.

### 4.10.1 Quality Assurance Checks for Successful CMAQ Run on CycleCloud

**run m3diff to compare the output data for two runs that have different values for NPCOL**

```
cd /shared/data/output
ls */*ACONC*
```

```
setenv AFILE output_v54_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_3x96/CCTM_ACONC_v54_cb6r5_
↳ ae7_aq_WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
setenv BFILE output_v54_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_4x96/CCTM_ACONC_v54_cb6r5_
↳ ae7_aq_WR413_MYR_gcc_2018_12US1_4x96_20171222.nc
```

```
m3diff
```

hit return several times to accept the default options

```
grep A:B REPORT
```

Should see all zeros. There are some non-zero values. It appears to have all zeros if the domain decomposition is the same NPCOL, here, NPCOL differs (10 vs 16) Did a test to determine if removing the compiler option -march=native would result in zero differences if NPCOL differs. This seems to work on CycleCloud, but did not work on Parallel Cluster.

Verify that you have loaded the gcc and openmpi modules.

```
module avail
```

```
module load gcc-9.2.0
```

```
module load mpi/openmpi-4.1.5
```

Verify the compiler version:

```
gcc --version
```

Output

```
gcc (GCC) 9.2.0
```

Evaluation of the Makefiles on Cyclecloud:

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc
grep FSTD Makefile
```

Output (note there is no longer the option -march=native that was causing differences in answers:

```
FSTD = -O3 -funroll-loops -finit-character=32 -Wtabs -Wsurprising -ftree-vectorize -
↪ftree-loop-if-convert -finline-limit=512
```

Check NPCOL NPROW for each run script

```
grep 'NPCOL =' run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.3x96.ncclassic.csh
```

```
@ NPCOL = 16; @ NPROW = 18
```

```
grep 'NPCOL =' run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.4x96.ncclassic.csh
```

Output

```
@ NPCOL = 16; @ NPROW = 24
```

```
grep A:B REPORT
```

output

```
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
A:B 0.00000E+00@( 1, 0, 0) 0.00000E+00@( 1, 0, 0) 0.00000E+00 0.00000E+00
```

(continues on next page)

(continued from previous page)

```

A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00
A:B  0.000000E+00@( 1,  0,  0)  0.000000E+00@( 1,  0,  0)  0.000000E+00  0.000000E+00

```

If CMAQ were compiled with `-march=native`, then you would see differences in the output if NPCOL was different, see previous version of this tutorial for more information

## 4.10.2 R analysis scripts

Run the following R script to create box plots and spatial plots showing difference between two CMAQ runs.

Note: requires that the R scripts and packages. See earlier instructions.

edit the R script to specify the `sim1.dir`, `sim1.file` and `sim2.dir`, `sim2.file` to correspond to the Benchmark cases that have been run.

Then run the R scripts!

```

cd /shared/cyclecloud-cmaq/qa_scripts
Rscript compare_EQUATES_benchmark_output_CMAS_cyclecloud.r

```

Note, if you have a + symbol in the log filename, the script will fail. Rename 5.4+ to 5.4plus to use the scripts.

View the Operating System

```
cat /etc/os-release
```

Output:

```

NAME="CentOS Linux"
VERSION="7 (Core)"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="7"
PRETTY_NAME="CentOS Linux 7 (Core)"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:7"
HOME_URL="https://www.centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"

```

(continues on next page)

(continued from previous page)

```

CENTOS_MANTISBT_PROJECT="CentOS-7"
CENTOS_MANTISBT_PROJECT_VERSION="7"
REDHAT_SUPPORT_PRODUCT="centos"
REDHAT_SUPPORT_PRODUCT_VERSION="7"

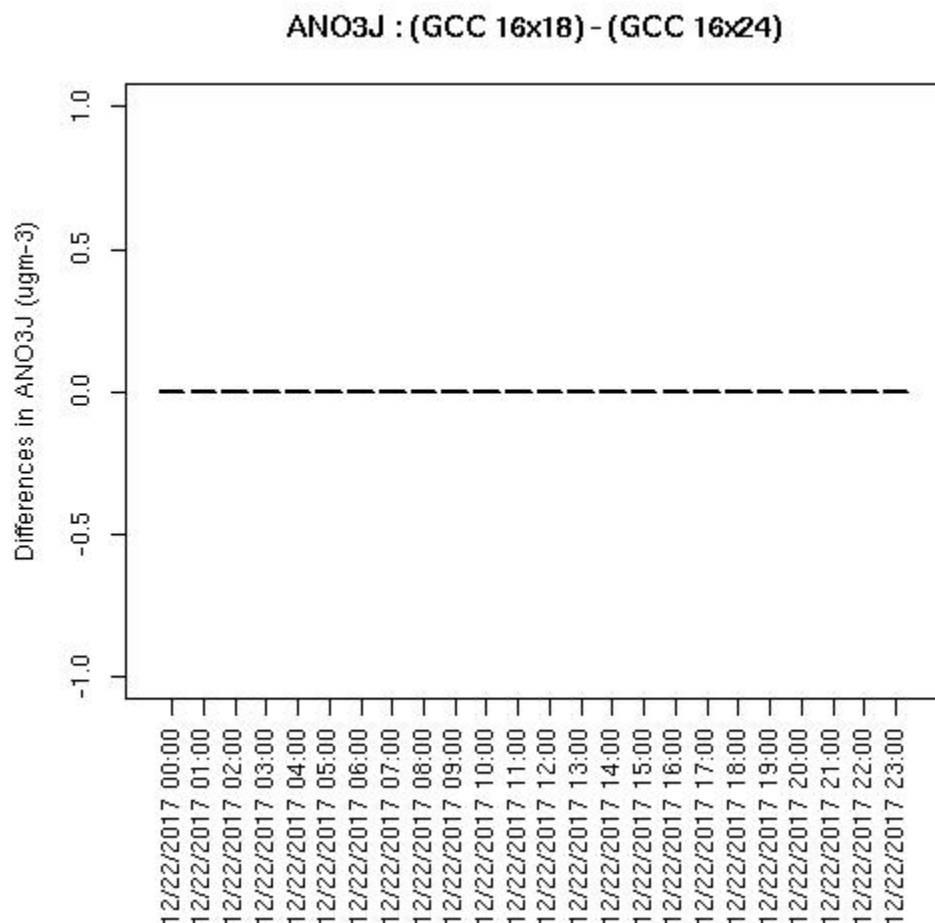
```

Example output plots are available for the CONUS Benchmark in the following directory

When NPCOL is fixed, we are seeing no difference in the answers.

Example comparison of 10x18 compared to 9x10 with the -march=native compiler flag removed

Box Plot for ANO3J when -march=native compiler flag is removed



Box plot shows no difference between ACONC output for a CMAQv5.4 run using different PE configurations as long as NPCOL is fixed (this is true for all species that were plotted (AOTHRJ, CO, NH3, NO2, O3, OH, SO2), or when not using march=native in the compiler flag

Spatial plots were not created by the script, as there were not differences between the output files.

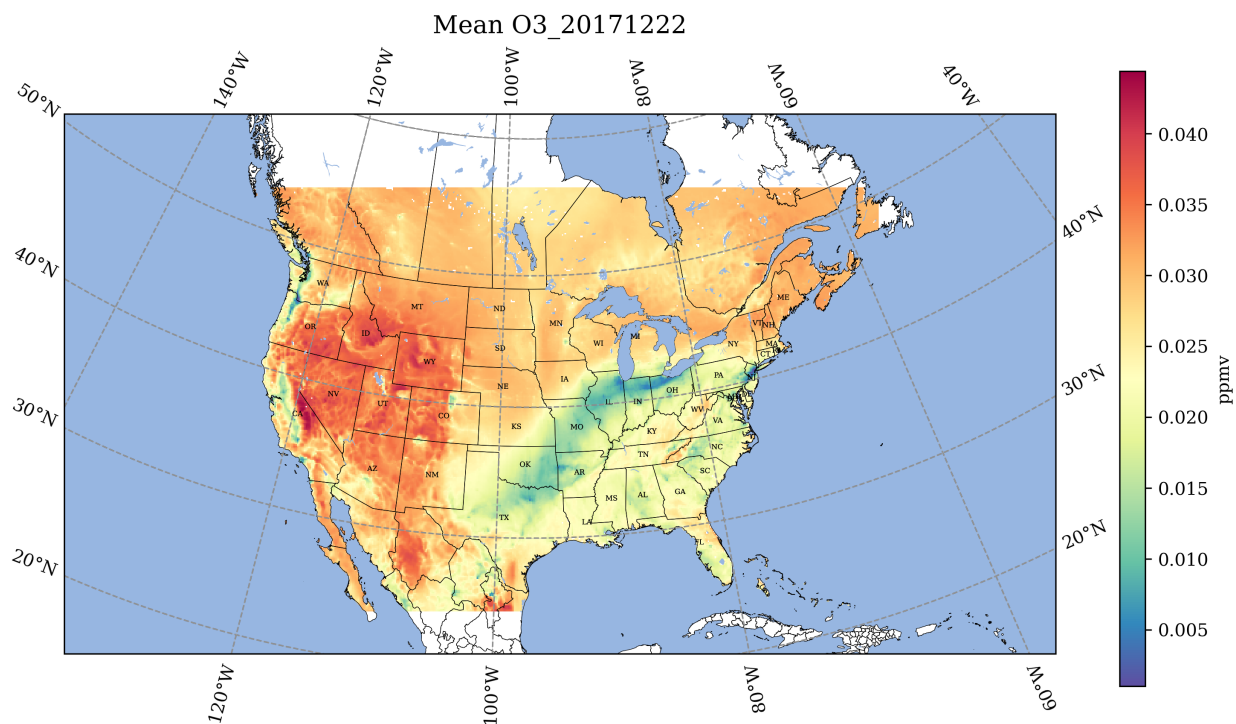


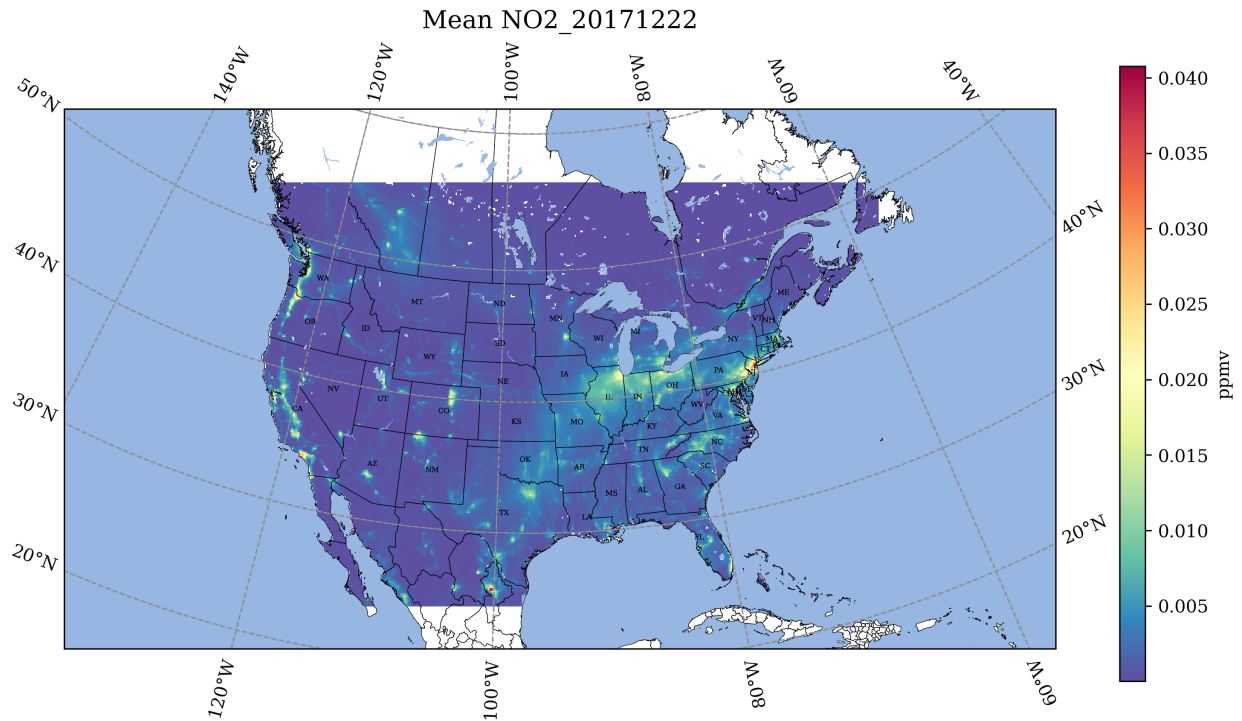
### 4.10.3 Jupyter Notebook

Use Jupyter Notebook to plot the daily average of O3 and NO2.

```
cd /shared/cyclecloud-cmaq/notebook
jupyter notebook
```

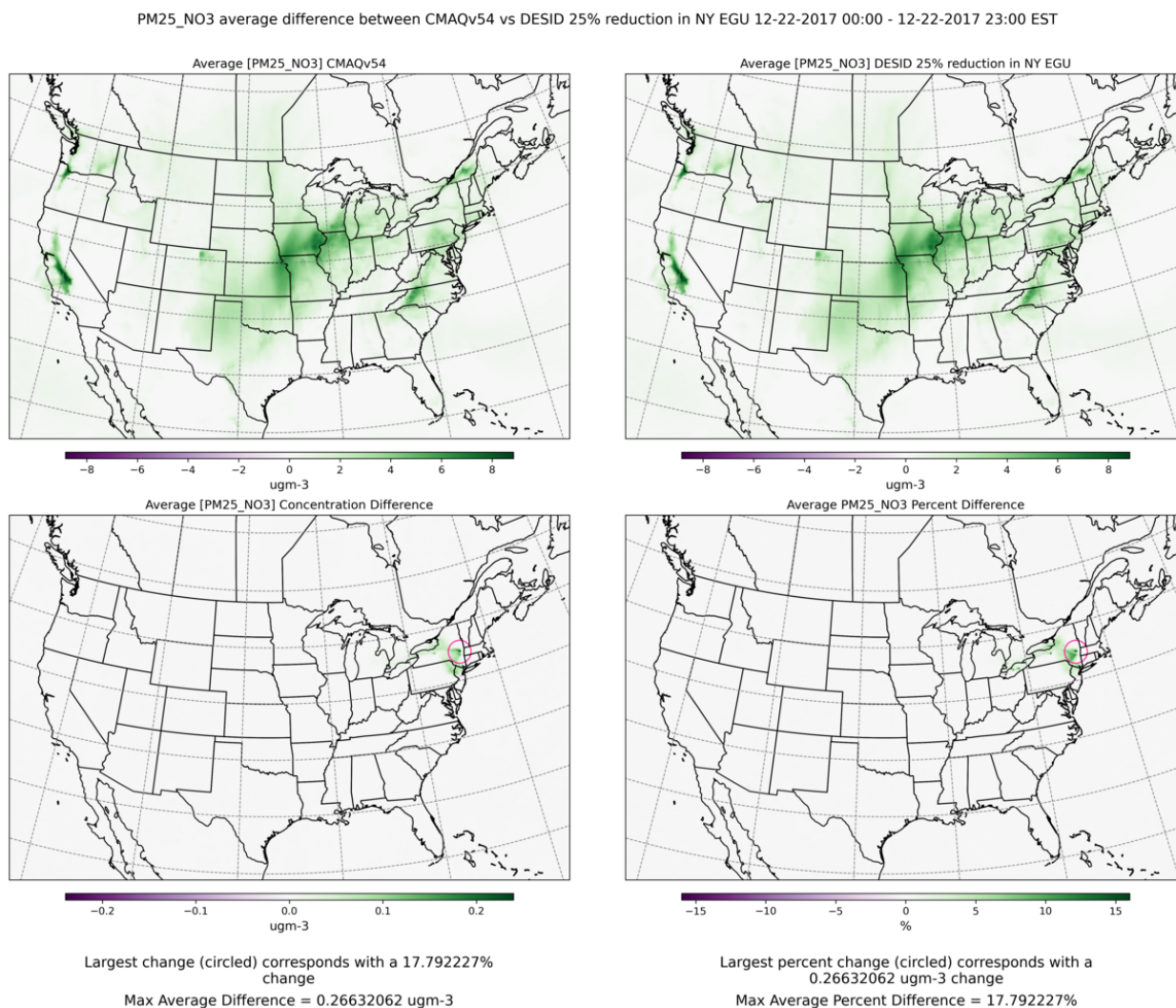
You will need to edit the path to the file, and the filenames. Output:





**Use Jupyter Notebook to plot the averages, average differences difference and average percent differences**

This notebook produces a panel of plots per species where the top two plots are averages and then the bottom two are average differences and average percent differences



## 4.11 Compare Timing of CMAQ Routines

Compare the timing of CMAQ Routines for two different run configurations.

### 4.11.1 Parse timings from the log file

#### Compare CONUS CycleCloud Runs

**Note:** CycleCloud Configurations can impact the model run times.

It is up to the user, as to what model run configurations are used to run CMAQ on the CycleCloud. The following configurations may impact the run time of the model.

- For different PE configurations, using 36 cpus out of 120 cpus on HBv120s
  - [ ] 6x6 #SBATCH -nodes=2, #SBATCH -ntasks-per-node=18

- [ ] 6x6 #SBATCH -nodes=1, #SBATCH -ntasks-per-node=36
- For different PE configurations, using 96 cpus out of 120 cpus on HBv120s
  - [ ] 8x12 #SBATCH -nodes=1, #SBATCH -ntasks-per-node=96
  - [ ] 16x12 #SBATCH -nodes=2, #SBATCH -ntasks-per-node=96
  - [ ] 16x18 #SBATCH -nodes=3, #SBATCH -ntasks-per-node=96
  - [ ] 16x24 #SBATCH -nodes=4, #SBATCH -ntasks-per-node=96
- For different filesystems /shared versus /beeond
  - [ ] input data copied to /shared
  - [ ] input data copied to /beeond

## Edit the R script

First check to see what log files are available:

```
ls -lrt /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/*.log
```

Modify the name of the log file to match what is available on your system.

```
cd /shared/pcluster-cmaq/qa_scripts
vi parse_timing.beeond.v54+.2018.12US1.r
```

Edit the following section of the script to specify the log file names available on your ParallelCluster

```
library(RColorBrewer)
library(stringr)
#sens.dir <- '/shared/cyclecloud-cmaq/run_scripts/HB120v3_singleVM/'
sens.dir <- '/shared/cyclecloud-cmaq/run_scripts/HB120v3_12US1_CMAQv54plus/'
base.dir <- '/shared/cyclecloud-cmaq/run_scripts/HB120v3_12US1_CMAQv54plus/'
#files <- dir(sens.dir, pattern = 'run_cctm5.4plus_Bench_2018_12US1_cb6r5_ae6_20200131_
↳MYR.192.8x12pe.2days.beeond.1x96.log')
files <- dir(sens.dir, pattern = 'run_cctm5.4plus_Bench_2018_12US1.96.12x8pe.2day.
↳cyclecloud.shared.codifix.log')
#b.files <- dir(base.dir, pattern = 'run_cctm5.3.3_Bench_2016_12US2.96.12x8pe.2day.
↳cyclecloud.shared.codemod.nopin.redo.log')
b.files <- c('run_cctm5.4plus_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.192.16x12pe.2days.
↳20171222start.2x96.log', 'run_cctm5.4plus_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.288.
↳16x18pe.2days.20171222start.3x96.log', 'run_cctm5.4plus_Bench_2018_12US1_cb6r5_ae6_
↳20200131_MYR.384.16x24pe.2days.20171222start.4x96.log' )
#Compilers <- c('intel','gcc','pgi')
Compilers <- c('gcc')
# name of the base case timing. I am using the current master branch from the CMAQ_Development
↳repository.
# The project directory name is used for the sensitivity case.
#base.name <- c('data_pin','lustre_pin','shared_pin')
base.name <- c('cmaq5.4plus_beeond_192', 'cmaq5.4plus_beeond_288', 'cmaq5.4plus_beeond_
↳384' )
sens.name <- c('cmaq5.4plus_beeond_96')
```

## R Timing Plot

Use `parse_timing.r` script to examine timings of each process in CMAQ

```
cd qa_scripts
Rscript parse_timing.beeond.v54+.2018.12US1.r
```

Timing Plot Comparing Total Run Time of CMAQv5.4 on 1, 2, 3, 4 Compute Nodes using HB120rs\_v3 or HB176\_v4 using Shared, Beeond and Lustre Filesystem; created by Manish Soni

Note: Click on Full Screen to see plot (icon in lower right corner, see icon circled in red in the image below the plot)

## 4.12 Copy Output to S3 Bucket

Copy output from CycleCloud to an S3 Bucket

### 4.12.1 Copy Output Data and Run script logs to S3 Bucket

Note, you will need permissions to copy to a S3 Bucket. see S3 Access Control

Currently, the bucket listed below has ACL turned off see S3 disable ACL

See example of sharing bucket across accounts. see Bucket owner granting cross-account permissions

#### Copy scripts and logs to /shared

The CTM\_LOG files do not contain any information about the compute nodes that the jobs were run on. Note, it is important to keep a record of the NPCOL, NPROW setting and the number of nodes and tasks used as specified in the run script: `#SBATCH --nodes=16 #SBATCH --ntasks-per-node=8` It is also important to know what volume was used to read and write the input and output data, so it is recommended to save a copy of the standard out and error logs, and a copy of the run scripts to the OUTPUT directory for each benchmark.

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
cp run*.log /shared/data/output
cp run*.csh /shared/data/output
```

#### Examine the output files

```
cd /shared/data/output/output_v54_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_3x96
ls -lrt
```

output:

```
-rw-rw-r--. 1 lizadams lizadams 14839992 Mar  8 00:27 CCTM_BSOILOUT_v54_cb6r5_ae7_aq_
  WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams  52713180 Mar  8 00:27 CCTM_MEDIA_CONC_v54_cb6r5_ae7_aq_
  WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams 2253034072 Mar  8 00:27 CCTM_DRYDEP_v54_cb6r5_ae7_aq_
  WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams 1831415812 Mar  8 00:27 CCTM_WETDEP1_v54_cb6r5_ae7_aq_
```

(continues on next page)

(continued from previous page)

```

↪WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams 178430048 Mar 8 00:27 CCTM_CONC_v54_cb6r5_ae7_aq_
↪MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams 2924988384 Mar 8 00:27 CCTM_ACONC_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams 1989523012 Mar 8 00:27 CCTM_AELMO_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams 4323159700 Mar 8 00:27 CCTM_CGRID_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171222.nc
-rw-rw-r--. 1 lizadams lizadams 2378548 Mar 8 00:27 CCTM_BUDGET_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171222.txt
drwxrwxrwx. 2 lizadams lizadams 24576 Mar 8 00:29 LOGS
-rw-rw-r--. 1 lizadams lizadams 14839992 Mar 8 00:57 CCTM_BSOILOUT_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 52713180 Mar 8 00:57 CCTM_MEDIA_CONC_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 2253034072 Mar 8 00:57 CCTM_DRYDEP_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 1831415812 Mar 8 00:57 CCTM_WETDEP1_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 178430048 Mar 8 00:57 CCTM_CONC_v54_cb6r5_ae7_aq_WR413_
↪MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 2924988384 Mar 8 00:57 CCTM_ACONC_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 1989523012 Mar 8 00:57 CCTM_AELMO_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 4323159700 Mar 8 00:57 CCTM_CGRID_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.nc
-rw-rw-r--. 1 lizadams lizadams 2378548 Mar 8 00:57 CCTM_BUDGET_v54_cb6r5_ae7_aq_
↪WR413_MYR_gcc_2018_12US1_3x96_20171223.txt

```

Check disk space

```
du -sh
```

Output

```
26G
```

## Copy the output to an S3 Bucket

Examine the example script

```
cd s3_scripts
cat s3_upload.HBv3-120_v54.csh
```

output:

```
#!/bin/csh -f
# Script to upload output data to S3 bucket
# NOTE: a new bucket needs to be created to store each set of cluster runs
```

(continues on next page)

(continued from previous page)

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
cp run*.log /shared/data/output
cp run*.csh /shared/data/output

aws s3 mb s3://hbv3-120-v54-compute-conus-output
aws s3 cp --recursive /shared/data/output/ s3://hbv3-120-v54-compute-conus-output/
```

If you do not have permissions to write to the s3 bucket listed above, you will need to edit the script to specify the s3 bucket that you have permissions to write to. In addition, edit the script to include a new date stamp, then run the script to copy all of the CMAQ output and logs to the S3 bucket.

```
./s3_upload.HBv3-120_v54.csh
```

## 4.13 Logout and Delete CycleCloud

Logout and delete the CycleCloud when you are done to avoid incurring costs.

### 4.13.1 Link to Azure Instructions on how to logout and delete cyclecloud

How to Terminate Cluster Resources

Tutorial to Clean-up Cluster Resources

## 4.14 Performance Optimization

Timing information and scaling plots to assist users in optimizing the performance of their Cycle Cloud HPC Cluster.

Performance Optimization for Single Virtual Machine

### 4.14.1 Right-sizing Compute Nodes for a Single Virtual Machine.

Selection of the compute nodes depends on the domain size and resolution for the CMAQ case, and what your model run time requirements are. Larger hardware and memory configurations may also be required for instrumented versions of CMAQ including CMAQ-ISAM and CMAQ-DDM3D. Running on a single virtual machine requires that the user know how CMAQ scales for the domain of interest.



### 4.14.2 An explanation of why a scaling analysis is required for Single Node

Quote from the following link.

“IMPORTANT: The optimal value of `-nodes` and `-ntasks` for a parallel code must be determined empirically by conducting a scaling analysis. As these quantities increase, the parallel efficiency tends to decrease. The parallel efficiency is the serial execution time divided by the product of the parallel execution time and the number of tasks. If multiple nodes are used then in most cases one should try to use all of the CPU-cores on each node.”

---

**Note:** For the scaling analysis that was performed with CMAQ, the parallel efficiency was determined as the runtime for the smallest number of CPUs divided by the product of the parallel execution time and the number of additional cpus used. If smallest NPCOLxNPROW configuration was 18 cpus, the run time for that case was used, and then the parallel efficiency for the case where 36 cpus were used would be  $\text{parallel efficiency} = \text{runtime\_18cpu} / (\text{runtime\_36cpu} * 2) * 100$

---

**See also:**

Scaling Analysis - see section on Multinode or Parallel MPI Codes

Azure HBv3-120 Pricing

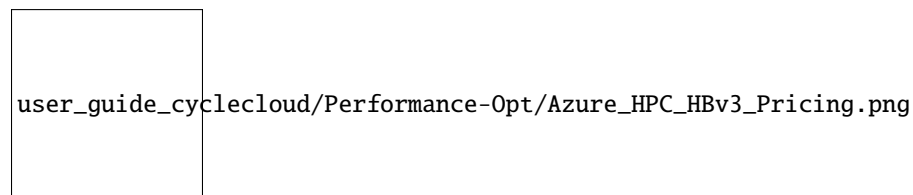


Table 1. Azure Instance On-Demand versus Spot Pricing (price is subject to change)

Instance Name	CPUs	RAM	Memory Band-width	Network Band-width	Linux On-Demand Price	Linux Spot Price
HBv3-120	120	448 GiB	350 Gbps	200 Gbps(Infiniband)	\$3.6/hour	\$.36/hour

Table 2. Timing Results for CMAQv5.4+ 2 Day CONUS2 Run on Single Virtual Machine HBv120 (120 cpu per node) I/O on /shared directory (UPDATE)



CPU	Nodes	NP	CD	Day1	Day2	Total	CPU	SBAT	Data	Equa	Spot	Equa	On-	com-	i/o
	by-	PRO	W	Tim-	Tim-	Time	Hours	day	Im-	tion	Cost	tion	De-	piler	dir
	CPU	W		(sec)	(sec)			clu-	ported	using		using	mand	flag	
								sive	or	Spot		On	Cost		
									Copied	Pricing		Demand			
16	1x16	4x4	10374.66	10.67	19685.33	34	no		copied	\$ .36/hr * 1 nodes * 5.468 =	7.87	3.6/hr * 1 nodes * 5.468 =	19.68	with - march=native com- piler flag	shared/data
36	1x36	6x6	5102.89	714.96	9817.85	36	no		copied	\$ .36/hr * 1 nodes * 2.72 =	3.92	3.6/hr * 1 nodes * 2.72 =	9.79	with - march=native com- piler flag	/shared/data
72	1x72	8x9	3130.73	2747.3	5878.03	15	no		copied	\$ .36/hr * 1 nodes * 1.63 =	2.35	3.6/hr * 1 nodes * 1.63 =	5.87	with - march=native com- piler flag	/shared/data
90	1x90	9x10	2739.38	2417.26	5156.64	15	no		copied	\$ .36/hr * 1 nodes * 1.43 =	2.06	3.6/hr * 1 nodes * 1.43 =	5.15	with - march=native com- piler flag	/shared/data
120	1x120	10x12	2646.52	2374.25	5020.76	973	no		copied	\$ .36/hr * 1 nodes * 1.3946 =	2.01	3.6/hr * 1 nodes * 1.39 =	5.00	with - march=native com- piler flag	/shared/data

Total HBv3-120 compute cost of Running Benchmarking Suite using SPOT pricing = \$1.4/hr

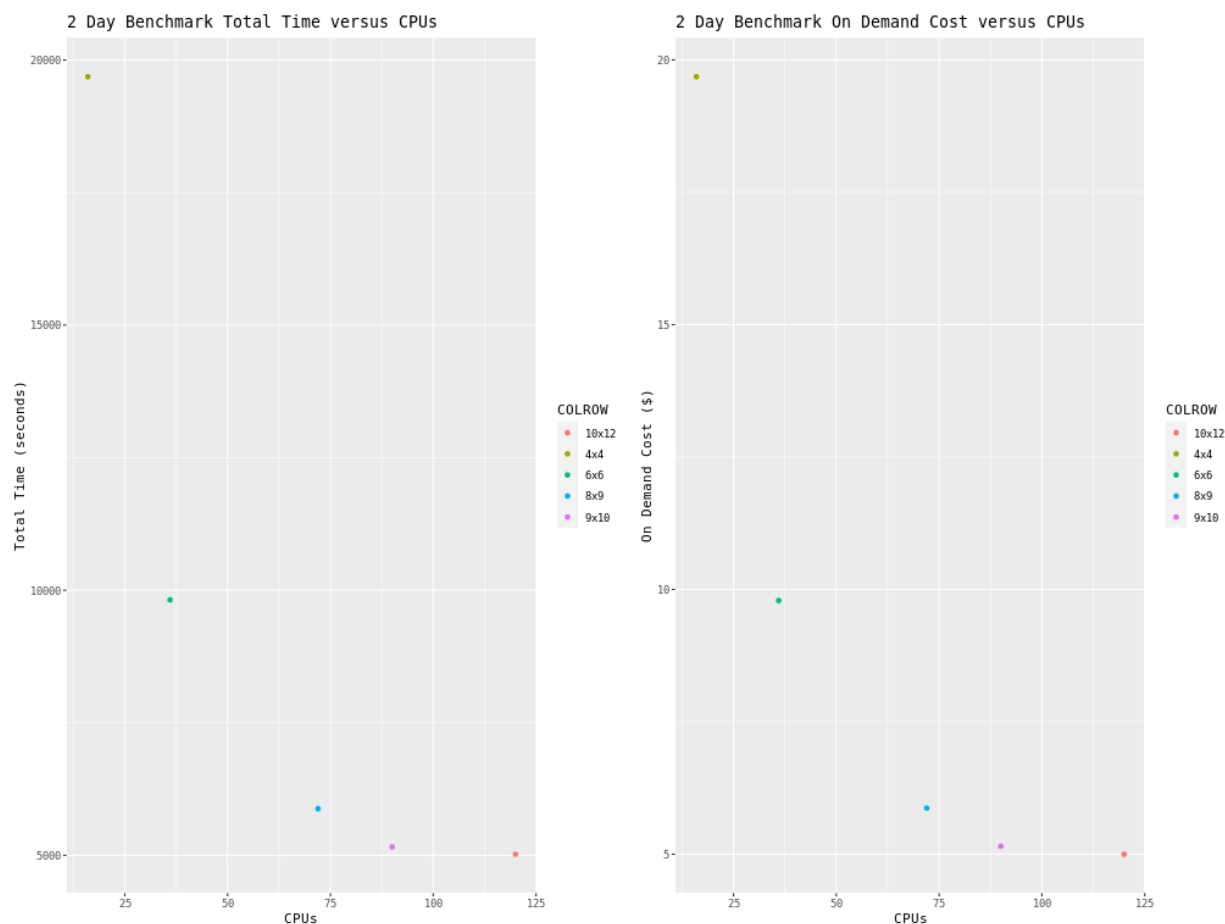
Total HBv3-120 compute cost of Running Benchmarking Suite using ONDEMAND pricing = \$3.6/hr

Savings is ~ 60% for spot versus ondemand pricing for HBv3-120 compute nodes.

Azure Spot and On-Demand Pricing

### 4.14.3 Benchmark Scaling Plots using Single Virtual Machine HBv120

Figure 1. Plot of Time and On Demand Cost versus CPU



Performance Optimization for Cycle Cloud

### 4.14.4 Right-sizing Compute Nodes for the CycleCloud

Selection of the compute nodes (virtual machines) depends on the domain size and resolution for the CMAQ case, the CMAQ Version, the model run time requirements, and the disks (beefond, lustre, or shared) used for input and output. Larger hardware and memory configurations may also be required for instrumented versions of CMAQ including CMAQ-ISAM and CMAQ-DDM3D. The CycleCloud allows you to run the compute nodes only as long as the job requires, and you can also update the compute nodes as needed for your domain.

### 4.14.5 An explanation of why a scaling analysis is required for Multinode or Parallel MPI Codes

Quote from the following link.

“IMPORTANT: The optimal value of `–nodes` and `–ntasks` for a parallel code must be determined empirically by conducting a scaling analysis. As these quantities increase, the parallel efficiency tends to decrease. The parallel efficiency is the serial execution time divided by the product of the parallel execution time and the number of tasks. If multiple nodes are used then in most cases one should try to use all of the Cores on each node.”

**Note:** For the scaling analysis that was performed with CMAQ, the parallel efficiency was determined as the runtime for the smallest number of Cores divided by the product of the parallel execution time and the number of additional cpus used. If smallest NPCOLxNPROW configuration was 18 cpus, the run time for that case was used, and then the parallel efficiency for the case where 36 cpus were used would be  $\text{parallel efficiency} = \text{runtime\_18cpu} / (\text{runtime\_36cpu} * 2) * 100$

#### See also:

Scaling Analysis - see section on Multinode or Parallel MPI Codes

Example Slurm script for Multinode Runs

### 4.14.6 Slurm Compute Node Provisioning

Azure CycleCloud relies on SLURM to make the job allocation and scaling decisions. The jobs are launched, terminated, and resources maintained according to the Slurm instructions in the CMAQ run script. The CycleCloud Web Interface is used to set the identity of the head node and the compute node, and the maximum number of compute nodes that can be submitted to the queue.

Number of compute nodes dispatched by the slurm scheduler is specified in the run script using `#SBATCH –nodes=XX` `#SBATCH –ntasks-per-node=YY` where the maximum value of tasks per node or YY limited by many Cores are on the compute node.

As an example:

For HB120rs\_v3, there are 120 Cores/node, so maximum value of YY is 120 or `–ntask-per-node=120` For many of the runs that were done, we set `–ntask-per-node=96` so that we could compare to the Parallel Cluster, and to avoid oversubscribing the cores.

If running a job with 192 processors, this would require the `–nodes=XX` or `XX` to be set to 2 compute nodes, as  $96 \times 2 = 192$ .

The setting for NPCOLxNPROW must also be a maximum of 192, ie. 16 x 12 or 12 x 16 to use all of the Cores in the Cycle Cloud HPC Node.

If running a job with 240 processors, this would require the `–nodes=XX` or `XX` to be set to 2 compute nodes, as  $120 \times 2 = 240$ .

Azure HBv3-120 Pricing

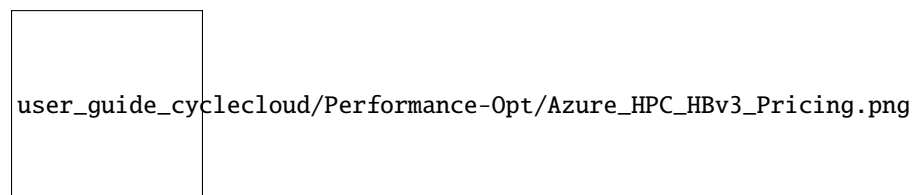


Table 1. Azure Instance On-Demand versus Spot Pricing (price is subject to change)

Instance Name	Cores	RAM	Memory Band-width	Network Band-width	Linux On-Demand Price	Linux Spot Price
HB120rs_v3	120	448 GiB	350 Gbps	200 Gbps(Infiniband)	\$3.6/hour	\$.36
HB176_v4	176	656 GiB	780 Gbps	400 Gbps(Infiniband)	\$7.2/hour	\$.41

Azure HBv3-series Specifications Azure HBv4-series Specifications

Note, check to see what processors were used

```
lscpu
```

Core (logs in March 2023)

```
Vendor ID:      AuthenticAMD
CPU family:     25
Model:         1
Model name:     AMD EPYC 7763 64-Core Processor
Stepping:      1
CPU MHz:       3021.872
BogoMIPS:      4890.85
Virtualization: AMD-V
Hypervisor vendor: Microsoft
Virtualization type: full
L1d cache:     32K
L1i cache:     32K
L2 cache:      512K
L3 cache:      32768K
NUMA node0 CPU(s): 0-3
```

Table 2. Timing Results for CMAQv5.4+ 2 Day 12US1 (CONUS) Run on Cycle Cloud with D12v2 schedulare node and HBv3-120 Compute Nodes (120 cpu per node), I/O on /mnt/beeond

Cores	Nodes	Nodes	SQL	Day1	Day2	To-	Com-	SBAT	Equa-	Spot	Equa-	On-	com-	In-	cpu
		Cores	ROW	Tim-	Tim-	tal-	pute	clu-	tion	Cost	tion	De-	piler	put-	Mhz
				(sec)	(sec)	Time	Node	sive	us-		using	mand	flag	Data	
							Hours	/day	ing		On	Cost			
									Spot		Demand				
									Pricing		Pricing				
96	1	1x96	8x12	3278	93800	77079	60983	no	\$.36/hr	.708	\$3.6/hr	7.077	with-	Beeond	872
									* 1		* 1		out -	=native	
									nodes		nodes		march		
									* 1.966		* 1.966		com-		
									hr =		hr =		piler		
													flag		
192	2	2x96	16x12	2027	82241	64269	46593	no	\$.36/hr	.854	\$3.6/hr	8.54	with-	Beeond	872
									* 2		* 2		out -	=native	
									nodes		nodes		march		
									* 1.186		* 1.186		com-		
									hr =		hr =		piler		
													flag		
288	3	3x96	16x18	1562	71692	63255	30452	no	\$.36/hr	.976	\$3.6/hr	9.76	with-	Beeond	872
									* 3		* 3		out -	=native	
									nodes		nodes		march		
									* .904		* .904		com-		
									hr =		hr =		piler		
													flag		
384	4	4x96	16x24	1356	51474	22830	70893	no	\$.36/hr	1.13	\$3.6/hr	11.3	with-	Beeond	872
									* 4		* 4		out -	=native	
									nodes		nodes		march		
									* .786		* .786		com-		
									hr =		hr =		piler		
													flag		

Total HBv3-120 compute cost of Running Benchmarking Suite using SPOT pricing = .36/hr

Total HBv3-120 compute cost of Running Benchmarking Suite using ONDEMAND pricing = \$3.6/hr

For CentOS or Ubuntu Linux in East US Region.

Savings is ~ 90% for spot versus ondemand pricing for HBv3-120 compute nodes.

Azure Spot and On-Demand Pricing Azure Spot and On-Demand Pricing

Table 3. Timing Results for CMAQv5.4+ 2 Day 12US1 (CONUS) Run on Cycle Cloud with D12v2 schedulare node and HBv2-120 Compute Nodes (120 cores per node), I/O on /mnt/beeond

Cores	Nodes	Nodes Core	SOL ROW	Day1 Tim- ing (sec)	Day2 Tim- ing (sec)	To- tal- Time	Com- pute Node Hours	SBATCH clu- sive day	Equa- tion us- ing Spot Pric- ing	Spot Cost	Equa- tion using On De- mand Pric- ing	On- De- mand Cost	com- piler flag	In- put- Data	Pin
96	1	1x96	12x8	3400.93	437.96	3838.89	50	no	\$1.89/hr * 1 nodes * \$.36 = \$0.36	\$0.68	1.89/hr * 1 nodes * 3.6 = \$6.804	6.804	no	Becondo	no
192	2	2x96	16x12	1954.61	220.53	2175.14	538	no	\$1.07/hr * 2 nodes * \$.36 = \$0.72	\$0.77	1.07/hr * 2 nodes * 3.6 = \$7.704	7.704	no	Becondo	no

Table 4. Timing Results for CMAQv5.4+ 2 Day 12US1 (CONUS) Run on Cycle Cloud with D12v2 schedulare node and HB176\_v3 Compute Nodes (176 cores per node), I/O using Becondo

Cores	Nodes	Nodes Core	SOL ROW	Day1 Tim- ing (sec)	Day2 Tim- ing (sec)	To- tal- Time	Com- pute Node Hours	SBATCH clu- sive day	Equa- tion us- ing Spot Pric- ing	Spot Cost	Equa- tion using On De- mand Pric- ing	On- De- mand Cost	com- piler flag	In- put- Data	Pin
160	1	1x176	16x10	2062.92	2235.34	4298.26	597	no	\$1.19/hr * 1 nodes * \$.41 = \$0.41	\$0.487	1.19/hr * 1 nodes * 7.2 = \$8.568	8.568	no	becondo	no
320	2	2x176	16x20	1644.41	1728.33	3372.74	468	no	\$.938/hr * 1 nodes * \$.41 = \$0.38	\$0.769	.938/hr * 2 nodes * 7.2 = \$13.51	13.51	no	becondo	no

### 4.14.7 Benchmark Scaling Plots using CycleCloud

Note: the numbers on the plot surrounded by a box indicates the number of Compute Nodes or Virtual Machines.

Figure 1. Plot of Scaling per Core

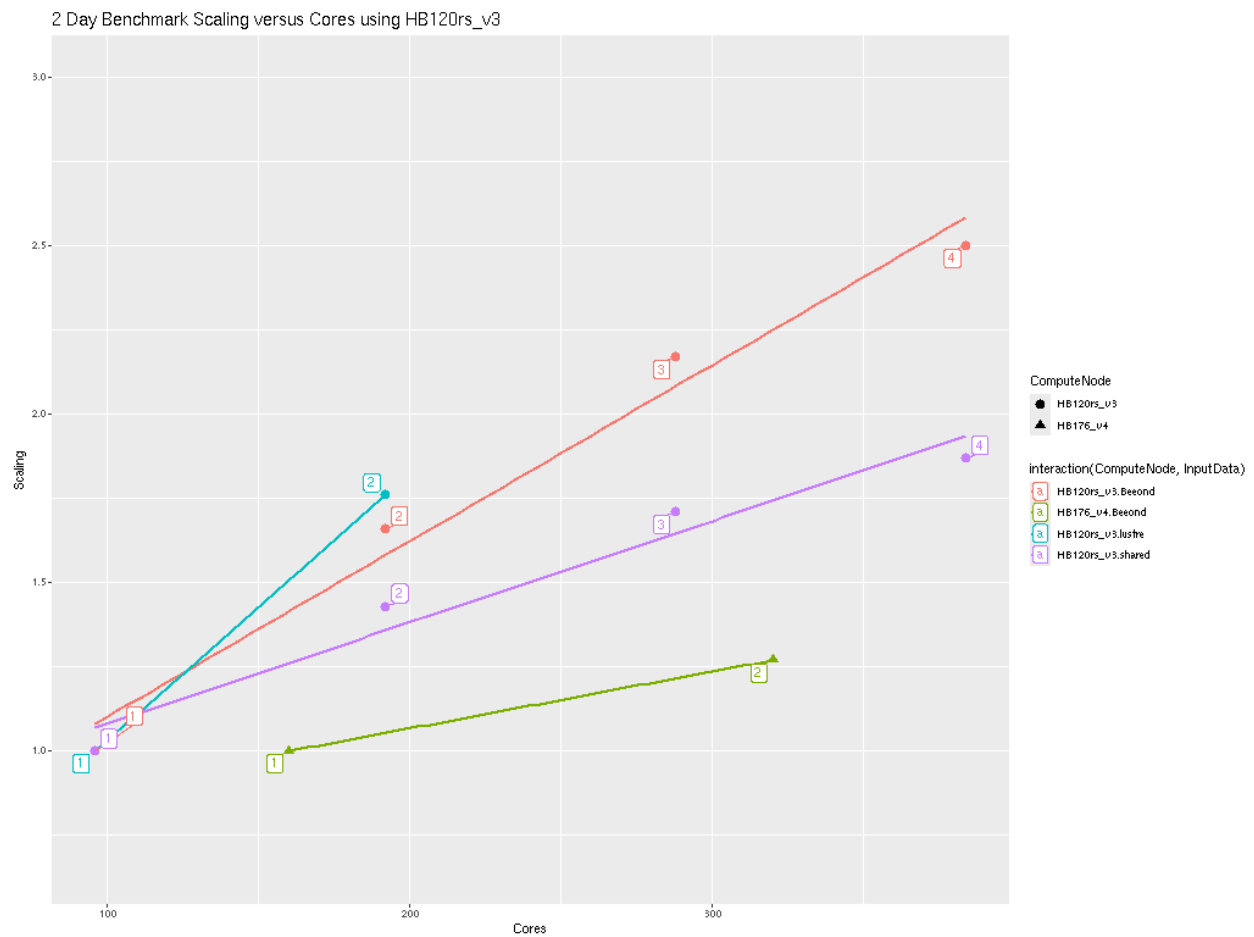


Figure 2. Plot of Total Time and On Demand Cost versus Cores for HB120\_v3 and HB176\_v4

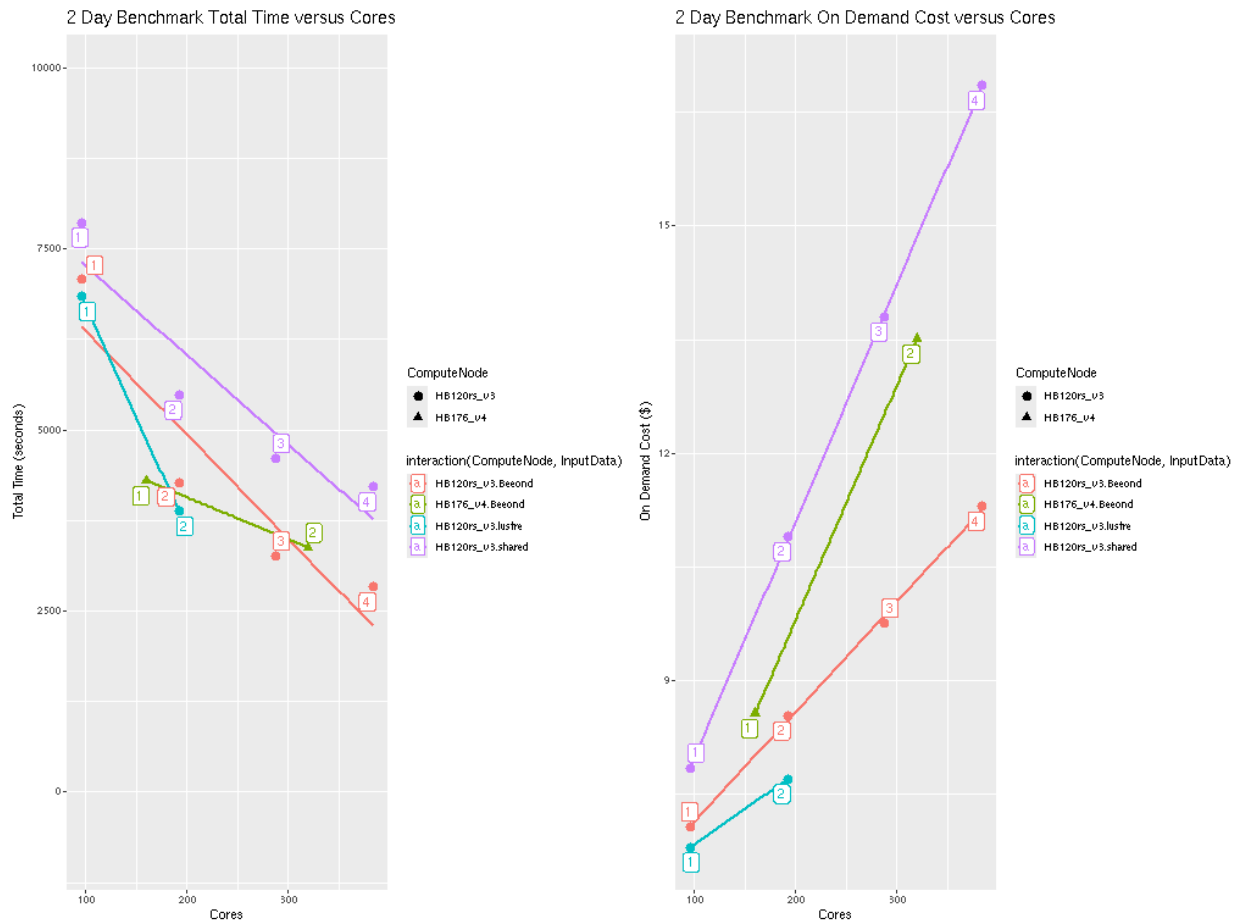
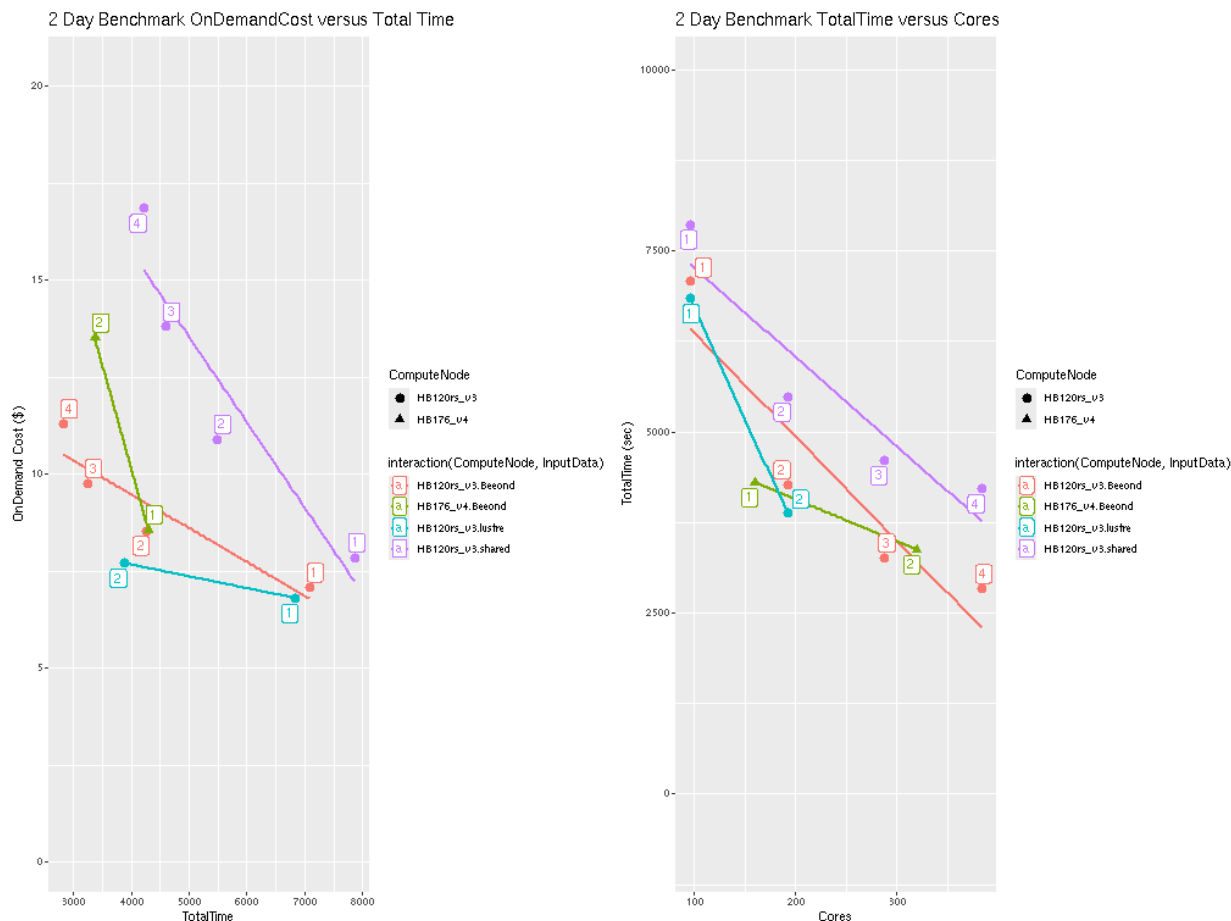


Figure 3. Plot of On Demand Cost versus Total Time for HB120\_v3 and HB176\_v4





Note CMAQ scales well up to ~ 288 Cores for the CONUS domain. As more cores are added beyond 288 cores, the CMAQ gets less efficient at using all of them.

Scheduler node D12v2 compute cost = Will be charged for the scheduler for the entire time that the CycleCloud HPC Cluster is running ( creation to deletion) = 6 hours \* \$0.7/hr = \$ ? using spot pricing, 6 hours \* \$7/hr = \$? using on demand pricing.

## Annual Run Estimates

Using 288 cpus on the Cycle Cloud Cluster, it would take 1 week to run a full year, using 3 HBv3-120 compute nodes, at a cost of \$1782 using ondemand nodes, or \$178.2 using interruptible spot nodes. (Note, spot nodes have not been tried yet in this tutorial.)

Table 5. Extrapolated Cost for CMAQv5.4 Annual Simulation based on 2 day 12US1 CONUS benchmark, without pinning

Virtual Machine	Nodes	Cores	SPOT \$/hr	On-Demand \$/hr	2 day time seconds	2 day time hours	Annual Cost Equation	Total Core hours	Annual Cost Spot	Annual Cost OnDemand	Days to Complete Annual Simulation
HB120_v3	3	96	\$.36	\$3.6	7079.60	1.96	$1.96/2 * 365 = 359$ hours/node * 1 node	359	\$129	\$1292	14.9
HB120_23	3	192	\$.36	\$3.6	4269.40	1.19	$1.19/2 * 365 = 216$ hours/node * 2 nodes	432	\$155.8	\$1558	9
HB120_33	3	288	\$.36	\$3.6	3255.3	.904	$.904/2 * 365 = 165$ hours/node * 3 nodes	495	\$178.2	\$1782	6.8
HB120_43	3	384	\$.36	\$3.6	2830.70	.786	$.786/2 * 365 = 143.5$ hours/node * 4 nodes	574	\$206.7	\$2066	5.95
HB176_v4	4	160	\$.41	\$7.2	4298.2	1.19	$1.19/2 * 365 = 217.9$ hours/node * 1 node	218	\$89.3	\$1569	9.04
HB176_24	4	320	\$.41	\$7.2	3372.7	0.94	$.94/2 * 365 = 171.55$ hours/node * 2 node	343	\$140.7	\$2470.3	7.2

Azure SSD Disk Pricing Azure SSD Disk Pricing

Table 6. Shared SSD File System Pricing

Storage Type	Storage throughput	Max IOPS (Max IOPS w/ bursting)	Pricing (monthly)	Pricing	Price per mount per month (Shared Disk)
Persistent 1TB	200 MB/s/TB	5,000 (30,000)	\$122.88/month	\$6.57	

### Lustre File System Pricing

Table 7. Lustre File System Pricing \*note, there isn't currently a method that starts and stops the Lustre Filesystem as part of the CycleCloud start and stop, so there is a danger of leaving the lustre file system on for long periods of time. It is recommended that you use the Beeond Filesystem, where we get similar performance, but at zero cost.

Storage Type	Storage throughput	Storage Capacity Availability (in multiples of)	Cost per GiB/hr	Cost/month for minimum capacity available
Standard tier	125 MB/s	16,000 GiB	.000198	\$2312
Premium tier	250 MB/s	8,000 GiB	.000287	\$1676
Ultra tier	500 MB/s	4,000 GiB	.000466	\$1361

According to the Azure calculators, the price varies by I/O Speed, and different tiers have different minimum storage size requirements.:

Calculations: 16000 GiB (17 TB) x 730 Hours x 0.000198Per GiB/hour = \$2312 / month 8000 GiB (9 TB) x 730 Hours x \$0.000287 Per GiB/hour = \$1676 / month 4000 GiB (4.3 TB) x 730 Hours x 0.000466Per GiB/hour = \$1361 / month

#### **CycleCloud and ParallelCluster Price Comparison of Cost Estimate for Annual Simulation (Filesystem + Compute)**

Table 8. Price Estimate for Annual Simulation (Filesystem + Compute)

Vendor	Cluster Name	Resource Type	Virtual Machine	Nodes	Cores	Min Storage	Spot Price	On-Demand \$/hr	CMAC day run-time (sec)	CMAC day run-time (hr)	5.7 Annual Equiv. (hr)	7.4 Annual Cost (\$)	Annual Cost (\$)	Annual Cost (\$)	Storage Cost (\$)	Storage Cost (\$)	Storage Cost (\$)	Days to Complete	Total Cost (\$)	Total Cost (\$)	Cost Savings			
Azure	CycleCloud	Compute	HB120_v2	8	288		\$0.36	\$3.60	255	0.904	164	165.02	8.25	82				6.9	\$2,462	\$184	\$615			
Azure	CycleCloud	Login	Standard_D8as_v4	8			N/A	\$0.0068	0.68	0.808	580	336.36	1.5	\$2										
Azure	CycleCloud	Scheduler	Standard_D4s_v3	4			N/A	\$0.19	510	0.68	0.808	580	336.36	1.5	\$63									
Azure	CycleCloud	NFS Storage	Premium SSD LRS			1	\$0.0001	100							\$0.0363	056								
Azure	CycleCloud	Lustre Storage	Ultra Storage (500 MB/s/TiB)			4000	\$0.0004	66							\$307.60	7765								
Azure	CycleCloud	Beacon	2 * 960 GB NVMe (block)												\$0									
AWS	ParallelCluster	Compute	hpc7g.1xlarge	12			N/A	\$1.68	509	0.87	497	497.49	127.13	168				7.4	\$1,006		\$81.9			
AWS	ParallelCluster	Scheduler	d7g.large 2				N/A	\$0.07	019	0.69	0.88	885	364	225	73									
AWS	ParallelCluster	Storage	Shared FS: GP3			1	\$0.0001	0959							\$0.0389	9812								
116							Chapter 4. Why might I need to use Azure Virtual Machine or CycleCloud?																	

## Assumptions for Price Estimate for Annual Simulation (Filesystem + Compute)

- Assuming that you have an annual simulation turn-around time requirement of < 8 days (less than 3787 seconds for 2 day benchmark)
- Assuming you have the scheduler and filesystems available for 2 \* the length of the compute node time for building CMAQ, installing input data, and copying output data to S3 bucket.
- Note, SPOT pricing is not available for AWS hpc7g.16xlarge
- Note, SPOT pricing is not recommended for the scheduler node
- Note, instructions for how to use SPOT pricing on Azure are not yet available
- Note, have not replicated using the Beeond Filesystem on AWS
- Note, assuming Lustre Filesystem is used at least as long as the scheduler node
- Note, Lustre Filesystem is created before Azure CycleCloud, and would need manual deletion after the run, recommend using Beeond Filesystem due to level of difficulty of provisioning Lustre Filesystem on CycleCloud
- Assuming that you have the scheduler node running 2x longer than the compute nodes

Parallel Cluster Performance Timings and Cost Estimates are available from the CMAQ on AWS Tutorial ParallelCluster Cost Estimate

## Recommended Workflow

Post-process monthly save output and/or post-processed outputs to archive storage at the end of each month.

Goal is to develop a reproducible workflow that does the post processing after every month, and then copies what is required to archive storage, so that only 1 month of output is stored at a time on the /shared/data scratch file system. This workflow will help with preserving the data in case the cluster or scratch file system gets pre-empted.

## 4.15 Additional Resources

- Additional resources for Cycle Cloud

### 4.15.1 Cycle Cloud Resources

Links to additional resources

1. Create, Customize and manage an HPC cluster in Azure with CycleCloud Azure High Performance Computing
2. Paper on HPC Computing on Azure using Cycle Cloud: High Performance Computing on Azure using Cycle-Cloud
3. Link on how to run GEOS-Chem on Cloud: Geos-Chem on the Cloud
4. Paper on HPC on Cloud: Enabling High-Performance Cloud Computing for Earth Science Modeling on Over a Thousand Cores: Application to the GEOS-Chem Atmospheric Chemistry Model
5. Comparative Benchmarking on Cloud: Comparative benchmarking of cloud computing vendors with high performance linpack
6. Information about Azure Open Dataset Program: Azure Open Datasets
7. Tutorial on Getting Started with GEOS Chem: Getting Started with GEOS Chem Tutorial

8. Git repo for Auto-scaling Slurm CycleCloud Cluster [Git Repo](#) to set up Auto-scaling Slurm CycleCloud Cluster
9. WRF, CMAQ & CAMx VM Image on Azure HPC available for free on Azure Marketplace (not using Cycle Cloud) [WRF, CMAQ & CAMx VM Image on Azure HPC \(not using Cycle Cloud\)](#)

### 4.15.2 Help Resources for CMAQ

1. CMAS Center Forum
2. EPA CMAQ Website
3. UNC CMAS Center Website

### 4.15.3 Resources from Azure for diagnosing issues with running the Cycle Cloud

#### Issues

Please open a GitHub issue for any feedback or issues:

There is also an active community driven Q&A site that may be helpful:

All Git Repositories matching CycleCloud

CycleCloud itself is installed as an application server on a Virtual Machine (VM) in Azure that requires outbound access to Azure Resource Provider APIs. CycleCloud then starts and manages VMs that form the High Performance Computing (HPC) systems — these typically consist of the HPC scheduler head node(s) and compute nodes.

Azure CycleCloud Documentation

Create an Virtual Machine for the CycleCloud 8.2 Image and then from that VM configure a cycle cloud host instance which will create a Cycle Cloud Scheduler. Use your credentials to ssh into the scheduler.

Follow these quickstart instructions to create CycleCloud.

Quickstart CycleCloud from Marketplace VM

Set up and use Managed Identities

Set up and use Managed Identities

List of error messages encountered on Cycle Cloud

List of error message one encounters on Cycle Cloud.

### 4.15.4 Frequently Asked Questions

Q. How do you figure out why a job is not successfully running in the slurm queue

A. Check the logs available in the following link

Slurm on CycleCloud

```
vi /var/log/slurmctld/resume.log
```

If the resume step is failing, there will also be a `vi /var/log/slurmctld/resume_fail.log`

Q. Is there a tutorial on how to use SLURM?

A. Yes

Slurm Tutorial

### 4.15.5 Computing on the Cloud References

WRF Cloud Computing Paper

NOAA Cloud-based Warn-on-Forecast

## 4.16 Future Work

### 4.16.1 List of ideas for future work

#### Azure Cycle Cloud

1. Test creation of Cycle Cloud on new account to verify instructions for user account permission settings and cycle cloud options to login, build and run CMAQ.
2. Figure out how to install slurm scheduler on AlmaLinux Virtual machine for HBv120. Determine if there is an overhead cost or penalty for running jobs within slurm versus interactively.
3. Figure out how to streamline install scripts and have them load while the Virtual Machine or Cycle Cloud instance is procured using cloud-init.
4. Figure out how to save the machine image with the software installed and re-use for the creation of a new Virtual Machine.
5. Learn how to set alarms or alerts or automatic shut-down of Virtual Machine if costs exceed budget.
6. Learn how to use spot instances for running CMAQ for both a single Virtual Machine and also for the compute nodes in Cycle Cloud. I ran into difficulty using a spot node machine for the Cycle Cloud VM Application Server. (it may make it difficult to terminate the Cycle Cloud)
7. Look into the Azure Open Data Program. Is there any reason to save the input data for the benchmark there in addition to on the AWS S3 Bucket?

#### Documentation

1. Finalize documentation and implement a version for the documentation in github.
2. Learn how to use the Cost Explorer in Azure and create screenshots showing costs for the single Virtual Machine HBv120 benchmarks and also for the Azure Cycle Cloud.
3. Document the screenshot that shows the Cycle Cloud VM that is the Application Server. This VM has a public IP address from which you can see the Cycle Cloud and on the left column see all of the Clusters that have been created, and in the center panel, see the status of those clusters. Note, that even if all of the clusters listed in the Cycle Cloud Application Server have been terminated, you can restart them, and have access to the input data and software.
4. I think it is a best practice to leave the Application Server running, but terminate the Cycle-Cloud Clusters. I don't know what happens if you "Stop" the Application Server, Can you later restart the Application Server? It would likely use a new IP address, but would it allow you to see all of the Cycle-Cloud Clusters again?
5. Add a poll to gauge usefulness of AWS and Azure Tutorials
6. Does your group have an azure account,
7. What is your experience level with Virtual Machine on Azure
8. What is your experience level with CycleCloud on Azure
9. What is your experience with using the AWS Parallel Cluster Tutorial

## 4.17 Contribute to this Tutorial

The community is encouraged to contribute to this documentation. It is open source, created by the CMAS Center, under contract to EPA, for the benefit of the CMAS Community.

### 4.17.1 Contribute to Azure-cmaq Documentation

Please take note of any issues and submit to Github Issue

---

**Note:** At the top of each page of the documentation, there is also an pencil icon, that you can click. It will create a fork of the project on your github account that you can make edits and then submit a pull request.

---

Intermediate Tutorial



Edit this page

If you are able to create a pull request, please include the following in your issue:

- pull request number

If you are not able to create a pull request, please include the following in your issue:

- section number
- description of the issue encountered
- recommended fix, if available

## 4.18 Optional instructions for Creating CycleCloud Cluster using /shared and /lustre

Warning - these are being kept to preserve the timings, but the method used to create the cluster is outdated.

Use portal.azure.com to create CycleCloud Cluster

### 4.18.1 Advanced Tutorial

- Learn how to install CMAQ software and underlying libraries, copy input data, and run CMAQ.

#### Modify Cyclecloud CMAQ Cluster

If you make changes to the nodes you must run the following commands

```
cd /opt/cycle/slurm
sudo ./cyclecloud_slurm.sh remove_nodes
sudo ./cyclecloud_slurm.sh scale
```

The above commands will remove the hpc nodes from the cluster and then rescale the cluster to use the new nodes that were specified when you edited the cluster.



This is required if you change the identity of the hpc VM. An example: if you change Standard\_HB120-64rs\_v3, a EPYC virtual machine containing 64 pes, to Standard\_HB120rs\_v3, an EPYC virtual machine containing 120 pes.

## Install CMAQ and pre-requisite libraries on linux on Alinux

### Login to updated cluster

(note, replace the centos.pem with your Key Pair)

```
ssh -v -Y -i ~/[your_azure].pem [scheduler-node-ip-address]
```

### Change shell to use .tcsh

```
sudo usermod -s /bin/tcsh azureuser
```

### Log out and then log back in to activate the tcsh shell

### Optional Step to allow multiple users to run on the CycleCloud Cluster

Add group name to users

```
sudo groupadd cmaq
```

Add the new group for each user

```
sudo usermod -a -G cmaq azureuser
```

Logout and log back in to reset the new group

Set the group to be default group for files created by the user

```
sudo usermod -g cmaq azureuser
```

logout and log back in to have it take effect

### Check to see if the group is added to your user ID

```
id
```

**Make the /shared/build directory**

```
sudo mkdir /shared/build
```

**Change ownership to your username**

```
sudo chown $USER /shared/build
```

**Make the /shared/cyclecloud-cmaq directory**

```
sudo mkdir /shared/cyclecloud-cmaq
```

**Change ownership to your username**

```
sudo chown $USER /shared/cyclecloud-cmaq
```

**Install git**

```
sudo yum install git
```

**Install the cluster-cmaq git repo to the /shared directory**

```
cd /shared  
git clone -b main https://github.com/CMASCenter/cyclecloud-cmaq.git cyclecloud-cmaq
```

```
cd cyclecloud-cmaq
```

**Optional - Change the group to cmaq recursively for the /shared directory/build**

```
sudo chgrp -R cmaq /shared/build
```

**Check what modules are available on the cluster**

```
module avail
```

### Load the openmpi module

```
module load mpi/openmpi
```

### Verify the gcc compiler version is greater than 8.0

```
gcc --version
```

output:

```
gcc (GCC) 9.2.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

### Change directories to install and build the libraries and CMAQ

```
cd /shared/cyclecloud-cmaq
```

### Build netcdf C and netcdf F libraries - these scripts work for the gcc 8+ compiler

```
./gcc_netcdf_cluster.csh
```

A `.cshrc` script with `LD_LIBRARY_PATH` was copied to your home directory, enter the shell again and check environment variables that were set using

```
cat ~/.cshrc`
```

If the `.cshrc` wasn't created use the following command to create it

```
cp dot.cshrc.cyclecloud ~/.cshrc
```

### Execute the shell to activate it

```
csh
env
```

### Verify that you see the following setting

```
echo $LD_LIBRARY_PATH
```

output:

```
/opt/openmpi-4.1.5/lib:/opt/gcc-9.2.0/lib64
```

### Build I/O API library

```
./gcc_ioapi_cluster.csh
```

### Build CMAQ

note, the primary difference is the location of the openmpi libraries on cyclecloud, /opt/openmpi-4.1.5/lib and include, /opt/openmpi-4.1.5/include

```
./gcc_cmaq_v54+.csh
```

Check to see that the cmaq executable has been built

```
ls -lrt /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/* .exe
```

Output

```
-rwxr-xr-x. 1 lizadams cmaq 152920040 Jan  9 18:49 /shared/build/openmpi_gcc/CMAQ_v54/  
↪CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
```

If it fails due to an issue with finding mpi, you will need to edit the gcc\_cmaq\_cyclecloud.csh script to point to the location of the mpi library and bin directory.

The following path is specified in the config\_cmaq\_cyclecloud.csh script, and may need to be updated. To find the mpi paths, use the command:

```
which mpirun
```

```
setenv MPI_INCL_DIR    /opt/openmpi-4.1.5/include      #> MPI Include_  
↪directory path  
setenv MPI_LIB_DIR     /opt/openmpi-4.1.5/lib          #> MPI Lib directory_  
↪path
```

## Install CMAQ and pre-requisite libraries on linux on SUSE Linux

### Login to updated cluster

(note, replace the centos.pem with your Key Pair)

```
ssh -v -Y -i ~/[your_azure].pem [scheduler-node-ip-address]
```

### Change shell to use .tcsh

```
sudo usermod -s /bin/tcsh azureuser
```

### Log out and then log back in to activate the tcsh shell

### Optional Step to allow multiple users to run on the CycleCloud Cluster

Add group name to users

```
sudo groupadd cmaq
```

Add the new group for each user

```
sudo usermod -a -G cmaq azureuser
```

Logout and log back in to reset the new group

Set the group to be default group for files created by the user

```
sudo usermod -g cmaq azureuser
```

logout and log back in to have it take effect

### Check to see if the group is added to your user ID

```
id
```

### Make the /shared/build directory

```
sudo mkdir /shared/build
```

**Change ownership to your username**

```
sudo chown $USER /shared/build
```

**Make the /shared/cyclecloud-cmaq directory**

```
sudo mkdir /shared/cyclecloud-cmaq
```

**Change ownership to your username**

```
sudo chown $USER /shared/cyclecloud-cmaq
```

**Install git**

```
sudo zypper install git
```

**Install the cluster-cmaq git repo to the /shared directory**

```
cd /shared  
git clone -b main https://github.com/CMASCenter/cyclecloud-cmaq.git cyclecloud-cmaq
```

```
cd cyclecloud-cmaq
```

**Optional - Change the group to cmaq recursively for the /shared directory/build**

```
sudo chgrp -R cmaq /shared/build
```

**Check what modules are available on the cluster**

```
module avail
```

Output

```
gnu/7      gnu/11 (L,D)
```

## Install openmpi

```
sudo zypper install openmpi openmpi-devel
module load mpi/openmpi-4.1.5
```

## Install gnu-compilers

```
sudo zypper in gnu-compilers-hpc
```

**Load the gcc copiler - note, this may have been automatically loaded by the openmpi module**

```
module load gcc-9.2.0
```

**Verify the gcc compiler version is greater than 8.0**

```
gcc --version
```

output:

```
gcc (GCC) 9.2.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

**Change directories to install and build the libraries and CMAQ**

```
cd /shared/cyclecloud-cmaq
```

**Build netcdf C and netcdf F libraries - these scripts work for the gcc 8+ compiler**

```
./gcc_netcdf_cluster.csh
```

**A .cshrc script with LD\_LIBRARY\_PATH was copied to your home directory, enter the shell again and check environment variables that were set using**

```
cat ~/.cshrc`
```

If the `.cshrc` wasn't created use the following command to create it

```
cp dot.cshrc.cyclecloud ~/.cshrc
```

Execute the shell to activate it

```
csh
env
```

Verify that you see the following setting

```
echo $LD_LIBRARY_PATH
```

output:

```
/opt/openmpi-4.1.5/lib:/opt/gcc-9.2.0/lib64
```

**Build I/O API library**

```
./gcc_ioapi_cluster.csh
```

**Build CMAQ**

note, the primary difference is the location of the openmpi libraries on cyclecloud, `/opt/openmpi-4.1.5/lib` and include, `/opt/openmpi-4.1.5/include`

```
./gcc_cmaq_v54+.csh
```

Check to see that the cmaq executable has been built

```
ls -lrt /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/BLD_CCTM_v54_gcc/*.exe
```

Output

```
-rwxr-xr-x. 1 lizadams cmaq 152920040 Jan  9 18:49 /shared/build/openmpi_gcc/CMAQ_v54/
↪CCTM/scripts/BLD_CCTM_v54_gcc/CCTM_v54.exe
```

If it fails due to an issue with finding mpi, you will need to edit the `gcc_cmaq_cyclecloud.csh` script to point to the location of the mpi library and bin directory.

The following path is specified in the `config_cmaq_cyclecloud.csh` script, and may need to be updated. To find the mpi paths, use the command:

```
which mpirun
```



```

setenv MPI_INCL_DIR      /opt/openmpi-4.1.5/include      #> MPI Include_
↪directory path
setenv MPI_LIB_DIR      /opt/openmpi-4.1.5/lib          #> MPI Lib directory_
↪path

```

## Configuring selected storage and obtaining input data

### Install AWS CLI to obtain data from AWS S3 Bucket

see <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

```

cd /shared/build
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install

```

### edit .cshrc file to add /usr/local/bin to path

```
vi ~/.cshrc`
```

add /usr/local/bin to the set path line

Run csh at the command line

### Verify you can run the aws command

```
aws --help
```

If not, you may need to logout and back in.

Set up your credentials for using s3 copy (you can skip this if you do not have credentials)

```
aws configure
```

### Azure Cyclecloud install input on the /shared/data directory

```
sudo mkdir /shared/data
```

## Change ownership

```
sudo chown azureuser /shared/data
```

```
ls /shared/data
df -h
```

Output:

```
/dev/mapper/vg_cyclecloud_builtinshared-lv0 1000G 66G 935G 7% /shared
```

## Use the following aws cp command to copy the CONUS input data from the CMAS s3 bucket

Modify the script if you want to change where the data is saved to. Script currently uses /shared/data

```
cd /shared/data
aws s3 --no-sign-request --region=us-east-1 cp --recursive s3://cmas-cmaq/CMAQv5.4_2018_
↪12US1_Benchmark_2Day_Input .
```

check that the resulting directory structure matches the run script

Note, this input data requires 85 GB of disk space

```
cd /shared/data/2018_12US1
du -sh
```

output:

```
85G .
```

CMAQ Cycle Cloud is configured to have 1 Terrabytes of space on the /shared filesystem, to allow multiple output runs to be stored.

## Copy the run scripts from the CycleCloud repo to run on HBv120

Note, the run scripts are tailored to the Compute Node. This assumes the cluster was built with HC44rs compute nodes.

Change directories to where the run scripts are available from the git repo.

```
cd /shared/data/2018_12US1/CMAQ_v54+_cb6r5_scripts
```

Copy the run scripts to the run directory

```
cp *.csh /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/`
```

## Edit the run script to run on 192 pes

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/
```

```
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic.csh
```

Note, it will take about 3-5 minutes for the compute nodes to start up This is reflected in the Status (ST) of PD (pending), with the NODELIST reason being that it is configuring the partitions for the cluster

## Check the status in the queue

```
squeue
```

output:

```
[lizadams@CMAQSlurmHC44rsAlmaLinux-scheduler scripts]$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
2	hpc	CMAQ	lizadams	CF	0:02	2	CycleCloud8-5-hpc-[1-2]

After 5 minutes the status will change once the compute nodes have been created and the job is running

```
squeue
```

output:

```
[lizadams@CMAQSlurmHC44rsAlmaLinux-scheduler scripts]$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
6	hpc	CMAQ	lizadams	R	0:37	2	CycleCloud8-5-hpc-[1-2]

The 192 pe job should take 85 minutes to run (42 minutes per day)

Note, if the job does not get scheduled, examine the slurm logs

```
sudo vi /var/log/slurmctld/slurmctld.log
sudo vi //var/log/slurmctld/resume.log
```

## check the timings while the job is still running using the following command

```
cd /shared/data/output/output_v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_2x96_classic_
→shared
grep 'Processing completed' CTM_LOG_001*
```

output:

```
Processing completed... 27.3354 seconds
Processing completed... 5.3785 seconds
Processing completed... 5.4735 seconds
Processing completed... 5.4295 seconds
Processing completed... 5.4600 seconds
Processing completed... 5.5127 seconds
```

(continues on next page)

(continued from previous page)

Processing completed...	5.4453	seconds
Processing completed...	5.4866	seconds
Processing completed...	5.4551	seconds
Processing completed...	5.4535	seconds
Processing completed...	5.4729	seconds
Processing completed...	7.7710	seconds

When the job has completed, use tail to view the timing from the log file.

```
tail -n 30 /shared/build/openmpi_gcc/CMAQ_v54+_classic/CCTM/scripts/run_cctm5.4+_Bench_
↪2018_12US1_cb6r5_ae6_20200131_MYR.192.16x12pe.2day.20171222start.2x96.shared.log
```

output:

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 192
All times are in seconds.

Num Day Wall Time
01 2017-12-22 2549.7
02 2017-12-23 2752.4
Total Time = 5302.10
Avg. Time = 2651.05
```

Check whether the scheduler thinks there are cpus or vcpus

```
sinfo -lN
```

output:

```
Wed Jan 10 19:24:35 2024
NODELIST          NODES PARTITION      STATE CPUS    S:C:T MEMORY TMP_DISK_
↪WEIGHT AVAIL_FE REASON
cyclecloudlizadams-hpc-pg0-1      1      hpc*   allocated 96    96:1:1 443596      0
↪ 1      cloud none
cyclecloudlizadams-hpc-pg0-2      1      hpc*   idle~ 96    96:1:1 443596      0
↪ 1      cloud none
cyclecloudlizadams-hpc-pg0-3      1      hpc*   idle~ 96    96:1:1 443596      0
↪ 1      cloud none
```

(continues on next page)

(continued from previous page)

cyclecloudlizadams-hpc-pg0-4	1	hpc*	idle~ 96	96:1:1	443596	0
→ 1 cloud none						
cyclecloudlizadams-hpc-pg1-1	1	hpc*	idle~ 96	96:1:1	443596	0
→ 1 cloud none						
cyclecloudlizadams-htc-1	1	htc	idle~ 2	2:1:1	3072	0
→ 1 cloud none						
cyclecloudlizadams-htc-2	1	htc	idle~ 2	2:1:1	3072	0
→ 1 cloud none						
cyclecloudlizadams-htc-3	1	htc	idle~ 2	2:1:1	3072	0
→ 1 cloud none						
cyclecloudlizadams-htc-4	1	htc	idle~ 2	2:1:1	3072	0
→ 1 cloud none						
cyclecloudlizadams-htc-5	1	htc	idle~ 2	2:1:1	3072	0
→ 1 cloud none						

### Edit the run script to run on 96 pes

```
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x96.ncclassic.retest.shared.csh
```

### Check the timing after run completed

```
tail -n 30 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.96.8x12pe.2day.
→20171222start.1x96.shared.log
```

### Output

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 96
All times are in seconds.

Num Day Wall Time
01 2017-12-22 3744.5
02 2017-12-23 4184.8
Total Time = 7929.30
Avg. Time = 3964.65
```

## Copy the run scripts from the CycleCloud repo

Note, the run scripts are tailored to the Compute Node.

Change directories to where the run scripts are available from the git repo.

```
cd /shared/cyclecloud-cmaq/run_scripts/HB120v3_12US1_CMAQv54plus
```

Copy the run scripts to the run directory

```
cp *.csh /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/`
```

## Run the CONUS Domain on 176 pes

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/
```

```
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x176.ncclassic.csh
```

Note, it will take about 3-5 minutes for the compute nodes to start up This is reflected in the Status (ST) of PD (pending), with the NODELIST reason being that it is configuring the partitions for the cluster

## Check the status in the queue

```
squeue
```

output:

```
[lizadams@CMAQSlurmHC44rsAlmaLinux-scheduler scripts]$ squeue
      JOBID PARTITION   NAME     USER ST       TIME  NODES NODELIST(REASON)
          2        hpc     CMAQ  lizadams CF        0:02      1 CycleCloud8-5-hpc-1
```

After 5 minutes the status will change once the compute nodes have been created and the job is running

```
squeue
```

output:

```
      JOBID PARTITION   NAME     USER ST       TIME  NODES NODELIST(REASON)
          6        hpc     CMAQ  lizadams R        0:37      5
↪cmaqslurmhc44rsalinux-hpc-pg0-[1-5]
```

The 176 pe job should take 85 minutes to run (42 minutes per day)

Note, if the job does not get scheduled, examine the slurm logs

```
sudo vi /var/log/slurmctld/slurmctld.log
sudo vi //var/log/slurmctld/resume.log
```

check the timings while the job is still running using the following command

```
grep 'Processing completed' CTM_LOG_001*
```

output:

```
Processing completed... 29.9047 seconds
Processing completed... 4.7678 seconds
Processing completed... 4.8123 seconds
Processing completed... 4.7888 seconds
Processing completed... 4.7633 seconds
Processing completed... 4.8243 seconds
```

When the job has completed, use tail to view the timing from the log file.

```
tail -n 30 /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts/run_cctm5.4+_Bench_2018_12US1_
↳cb6r5_ae6_20200131_MYR.176.16x11pe.2day.20171222start.1x176.log
```

output:

Check whether the scheduler thinks there are cpus or vcpus

```
sinfo -lN
```

output:

```
Tue Jan 09 19:11:04 2024
NODELIST          NODES PARTITION      STATE CPUS    S:C:T MEMORY TMP_DISK WEIGHT_
↳AVAIL_FE REASON
CycleCloud8-5-hpc-1    1    hpc*  allocated# 176    176:1:1 747110      0      1  _
↳cloud none
CycleCloud8-5-hpc-2    1    hpc*    idle~ 176    176:1:1 747110      0      1  _
↳cloud none
CycleCloud8-5-hpc-3    1    hpc*    idle~ 176    176:1:1 747110      0      1  _
↳cloud none
CycleCloud8-5-hpc-4    1    hpc*    idle~ 176    176:1:1 747110      0      1  _
↳cloud none
CycleCloud8-5-htc-1    1    htc    idle~ 2      2:1:1 3072        0      1  _
↳cloud none
CycleCloud8-5-htc-2    1    htc    idle~ 2      2:1:1 3072        0      1  _
↳cloud none
```

### 4.18.2 CMAQv5.4+ Benchmark on HB176\_v4 compute nodes and shared

Run CMAQv5.4+ on a using pre-loaded software and input data on shared HB176\_v4 Cycle Cloud.

CMAQv4+ CONUS Benchmark Tutorial using 12US1 Domain

#### Use Cycle Cloud pre-installed with CMAQv5.4+ software and 12US1 Benchmark data.

Step by step instructions for running the CMAQ 12US1 Benchmark for 2 days on a Cycle Cloud The input files are \*.nc4, which requires the netCDF-4 compressed libraries. Instructions are provided below on how to use load module command to obtain the correct version of the I/O API, netCDF-C, netCDF-Fortran libraries.

#### Log into the new cluster

**Note:** Use your username and credentials to login

```
`ssh -Y username@IP-address``
```

Load the modules

```
module avail
```

Output:

```
----- /usr/share/Modules/
↪modulefiles -----
amd/aocl      dot      module-git  modules  mpi/hpcx-v2.16  mpi/impi_2021.9.0  mpi/
↪mvapich2-2.3.7-1  mpi/openmpi-4.1.5  use.own
amd/aocl-4.0  gcc-9.2.0  module-info  mpi/hpcx  mpi/impi-2021  mpi/mvapich2      mpi/
↪openmpi      null
```

#### Download the input data from the AWS Open Data CMAS Data Warehouse.

Do a git pull to obtain the latest scripts in the cyclecloud-cmaq repo.

```
cd /shared/data
aws s3 --no-sign-request --region=us-east-1 cp --recursive s3://cmas-cmaq/CMAQv5.4_2018_
↪12US1_Benchmark_2Day_Input .
```

#### Verify Input Data

```
cd /shared/data
mkdir /shared/data/CMAQ_Modeling_Platform_2018
cd /shared/data/CMAQ_Modeling_Platform_2018
ln -s ../2018_12US1 .
cd /shared/data/CMAQ_Modeling_Platform_2018/2018_12US1`
```

```
du -h
```

Output



```

44K      ./CMAQ_v54+_cracmm_scripts
40K      ./CMAQ_v54+_cb6r5_scripts
1.6G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/cmv_c1c2_12
2.4G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/cmv_c3_12
5.1G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/merged_nobeis_norwc
1.4G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/othpt
1.3G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/pt_oilgas
6.7M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptagfire
255M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptegu
19M      ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire
2.9M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire_grass
3.0M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire_othna
5.9G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptnonipm
18G      ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready
3.5G     ./emis/cb6r3_ae6_20200131_MYR/premerged/rtc
3.5G     ./emis/cb6r3_ae6_20200131_MYR/premerged
22G      ./emis/cb6r3_ae6_20200131_MYR
60K      ./emis/emis_dates
22G      ./emis
2.3G     ./epic
13G      ./icbc/CMAQv54_2018_108NHEMI_M3DRY
17G      ./icbc
41G      ./met/WRFv4.3.3_LTNG_MCIP5.3.3_compressed
41G      ./met
4.0G     ./misc
697M     ./surface
85G      .

```

## Examine CMAQ Run Scripts

Copy the run scripts. Note, there are different run scripts depending on what compute node is used. This tutorial assumes hpc6a-48xlarge is the compute node.

```

cd /shared/cyclecloud-cmaq/run_scripts/HB176v4
cp *.csh /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/

```

**Note:** The time that it takes the 2 day CONUS benchmark to run will vary based on the number of CPUs used, and the compute node that is being used, and what disks are used for the I/O (shared or lustre). The Benchmark Scaling Plot for hbv176\_v4 on shared (include here).

Examine how the run script is configured

```

cd /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/
head -n 30 run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x176.ncclassic.csh

```

```

#!/bin/csh -f

## For CycleCloud 176pe
## data on /shared data directory

```

(continues on next page)

(continued from previous page)

```
## https://dataverse.unc.edu/dataset.xhtml?persistentId=doi:10.15139/S3/LDTWKH
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=176
#SBATCH --exclusive
#SBATCH -J CMAQ
#SBATCH -o /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/run_cctm5.4+_Bench_2018_
↳ 12US1_cb6r5_ae6_20200131_MYR.176.16x11pe.2day.20171222start.1x176.log
#SBATCH -e /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/run_cctm5.4+_Bench_2018_
↳ 12US1_cb6r5_ae6_20200131_MYR.176.16x11pe.2day.20171222start.1x176.log

# ===== CCTMv5.4.X Run Script =====
# Usage: run.cctm >&! cctm_2018_12US1.log &
#
# To report problems or request help with this script/program:
#         http://www.epa.gov/cmaq      (EPA CMAQ Website)
#         http://www.cmascenter.org    (CMAS Website)
# =====

# =====
#> Runtime Environment Options
# =====

echo 'Start Model Run At ' `date`

#> Toggle Diagnostic Mode which will print verbose information to
#> standard output
setenv CTM_DIAG_LVL 0
```

---

**Note:** In this run script, slurm or SBATCH requests 1 node, each node with 176 pes

---

Verify that the NPCOL and NPROW settings in the script are configured to match what is being requested in the SBATCH commands that tell slurm how many compute nodes to provision. In this case, to run CMAQ using on 176 cpus (SBATCH --nodes=1 and --ntasks-per-node=176), use NPCOL=16 and NPROW=11.

```
grep NPCOL run_cctm_2018_12US1_v54_cb6r3_ae6.20171222.csh
```

Output:

```
setenv NPCOL_NPROW "1 1"; set NPROCS    = 1 # single processor setting
@ NPCOL  = 16; @ NPROW = 11
@ NPROCS = $NPCOL * $NPROW
setenv NPCOL_NPROW "$NPCOL $NPROW";
```

## Submit Job to Slurm Queue to run CMAQ on 2 nodes

```
cd /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x176.ncclassic.csh
```

## Check status of run

squeue

Output:

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
-------	-----------	------	------	----	------	-------	------------------

It takes about 5-8 minutes for the compute nodes to spin up, after the nodes are available, the status will change from CF to R.

## Successfully started run

squeue

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
10	hpc	CMAQ	lizadams	R	1:08:04	2	CycleCloud8-5-hpc-[3-4]

## Timings 2x176

Check on the log file status

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
grep -i 'Processing completed.' run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.352.16x22pe.2day.20171222start.2x176.log`
```

Output:

Processing completed...	38.3631 seconds
Processing completed...	2.2522 seconds
Processing completed...	2.2842 seconds
Processing completed...	2.2733 seconds
Processing completed...	2.2939 seconds
Processing completed...	2.2799 seconds
Processing completed...	2.2699 seconds
Processing completed...	2.2948 seconds
Processing completed...	2.2759 seconds
Processing completed...	2.2810 seconds
Processing completed...	2.2850 seconds
Processing completed...	2.2935 seconds
Processing completed...	2.3114 seconds

(continues on next page)

(continued from previous page)

```
Processing completed... 2.2939 seconds
Processing completed... 3.2520 seconds
```

Once the job has completed running the two day benchmark check the log file for the timings.

```
tail -n 18 run_cctm5.4+_Bench_2018_12US1_M3DRY_cb6r3_ae6_20200131_MYR.256.16x16pe.2day.
↪20171222start.log`
```

Output:

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 352
All times are in seconds.

Num Day Wall Time
01 2017-12-22 2103.6
02 2017-12-23 2183.0
Total Time = 4286.60
Avg. Time = 2143.30
```

### Submit a run script to run on the shared volume

#### Submit a 176 pe job 1 nodes x 176 cpus on the EBS volume /shared

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x176.ncclassic.csh
```

### Timings 1x176

```
cd /shared/data/output/output_v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176
grep -i 'Processing completed.' CTM_LOG_000*
```

Output:

```
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↪Processing completed... 33.6960 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↪Processing completed... 3.8317 seconds
```

(continues on next page)

(continued from previous page)

```
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9250 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9163 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9316 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9393 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9570 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9355 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9535 seconds
CTM_LOG_000.v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x176_20171223:
↳Processing completed... 3.9619 seconds
```

```
cd /shared/build/openmpi_gcc/CMAQ_v54/CCTM/scripts
tail -n 18 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.176.16x11pe.2day.
↳20171222start.1x176.log
```

Output:

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 176
All times are in seconds.

Num Day Wall Time
01 2017-12-22 2573.5
02 2017-12-23 2720.5
Total Time = 5294.00
Avg. Time = 2647.00
```

### 4.18.3 CMAQv5.4+ Benchmark on HBv3\_120 compute nodes and lustre

Run CMAQv5.4+ on a using pre-loaded software and input data on Lustre using HBv3\_120 Cycle Cloud.

CMAQv4+ CONUS Benchmark Tutorial using 12US1 Domain

#### Use Cycle Cloud pre-installed with CMAQv5.4+ software and 12US1 Benchmark data.

Step by step instructions for running the CMAQ 12US1 Benchmark for 2 days on a Cycle Cloud The input files are \*.nc4, which requires the netCDF-4 compressed libraries. Instructions are provided below on how to use load module command to obtain the correct version of the I/O API, netCDF-C, netCDF-Fortran libraries.

#### This method relies on obtaining the code and data from blob storage.

---

**Note:** Information about how to share a snapshot of a Blob Storage account: [Share from Blob Storage Account](#)

---

You will need to copy the snapshot and create a new blob storage, and then use your Blob Storage as the backup to your Lustre Filesystem.

#### Use a configuration file from the github by cloning the repo to your local machine

```
cd /lustre
sudo mkdir cyclecloud-cmaq
sudo chown username cyclecloud-cmaq
git clone -b main https://github.com/CMASCenter/cyclecloud-cmaq
```

```
cd cyclecloud-cmaq
```

#### Lustre - Request Public Preview

---

**Note:** Information about the Public Preview for Azure Managed Lustre see: [Azure Managed Lustre Benchmarking Lustre](#)

---

See information on how to join: [Azure Managed Lustre - Registration form link](#)

#### Create Lustre Server

#### Blob Storage - Lustre hierarchical storage management

#### Update Cycle Cloud

#### Log into the new cluster

---

**Note:** Use your username and credentials to login

---

```
ssh -Y username@IP-address
```

## Verify Software

The software is pre-loaded on the /lustre volume of the CycleCloud.

```
ls /lustre/build
```

Load the modules

```
module avail
```

Output:

```
----- /usr/share/Modules/
↪modulefiles -----
amd/aocl      dot      module-git  modules  mpi/hpcx-v2.9.0  mpi/impi_2021.2.0 ↪
↪mpi/mvapich2-2.3.6  mpi/openmpi-4.1.1  use.own
amd/aocl-2.2.1  gcc-9.2.1  module-info  mpi/hpcx  mpi/impi-2021    mpi/mvapich2      ↪
↪mpi/openmpi      null

----- /shared/build/Modules/
↪modulefiles -----
hdf5-1.10.5/gcc-9.2.1  ioapi-3.2_20200828/gcc-9.2.1-hdf5  ioapi-3.2_20200828/gcc-9.2.1-
↪netcdf  netcdf-4.8.1/gcc-9.2.1
```

Load the modules for the netCDF-4 compressed libraries.

```
module load ioapi-3.2_20200828/gcc-9.2.1-hdf5
```

output:

Change the group and ownership permissions on the /lustre/data directory

```
sudo chown ubuntu /lustre/data
sudo chgrp ubuntu /lustre/data
```

Create the output directory

```
mkdir -p /lustre/data/output
```

## Download the input data from the AWS Open Data CMAS Data Warehouse.

Do a git pull to obtain the latest scripts in the cyclecloud-cmaq repo.

```
cd /lustre/cyclecloud-cmaq
git pull
```

```
cd /shared/cyclecloud-cmaq/s3_scripts
./s3_copy_nosign_2018_12US1_conus_cmas_opendata_to_lustre_20171222_cb6r3.csh
```

## Verify Input Data

```
cd /lustre/data_lim/CMAQ_Modeling_Platform_2018/2018_12US1
du -h
```

### Output

```
40K      ./CMAQ_v54+_cb6r5_scripts
44K      ./CMAQ_v54+_cracmm_scripts
1.6G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/cmv_c1c2_12
2.4G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/cmv_c3_12
5.1G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/merged_nobeis_norwc
1.4G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/othpt
1.3G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/pt_oilgas
6.7M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptagfire
255M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptegu
19M      ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire
2.9M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire_grass
3.0M     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptfire_othna
5.9G     ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready/ptnonipm
18G      ./emis/cb6r3_ae6_20200131_MYR/cmaq_ready
3.5G     ./emis/cb6r3_ae6_20200131_MYR/premerged/rwc
3.5G     ./emis/cb6r3_ae6_20200131_MYR/premerged
22G      ./emis/cb6r3_ae6_20200131_MYR
60K      ./emis/emis_dates
22G      ./emis
2.3G     ./epic
13G      ./icbc/CMAQv54_2018_108NHEMI_M3DRY
17G      ./icbc
41G      ./met/WRFv4.3.3_LTNG_MCIP5.3.3_compressed
41G      ./met
4.0G     ./misc
697M     ./surface
85G      .
```

## Examine CMAQ Run Scripts

The run scripts are available in two locations, one in the CMAQ scripts directory.

Another copy is available in the cyclecloud-cmaq repo.

Copy the run scripts from the repo. Note, there are different run scripts depending on what compute node is used. This tutorial assumes hpc6a-48xlarge is the compute node.

```
cp /shared/cyclecloud-cmaq/run_scripts/2018_12US1_CMAQv54plus/run_cctm_2018_12US1_v54_
  ↪cb6r3_ae6.20171222.csh /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/
```

**Note:** The time that it takes the 2 day CONUS benchmark to run will vary based on the number of CPUs used, and the compute node that is being used, and what disks are used for the I/O (shared or lustre). The Benchmark Scaling Plot for hbv3\_120 on lustre and shared (include here).



Examine how the run script is configured

```
head -n 30 /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/run_cctm_2018_12US1_v54_
  ↪ cb6r3_ae6.20171222.csh
```

```
#!/bin/csh -f

## For CycleCloud 120pe
## data on /lustre data directory
## https://dataverse.unc.edu/dataset.xhtml?persistentId=doi:10.15139/S3/LDTWKH
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=64
#SBATCH --exclusive
#SBATCH -J CMAQ
#SBATCH -o /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/run_cctm5.4+_Bench_2018_
  ↪ 12US1_M3DRY_cb6r3_ae6_20200131_MYR.256.16x16pe.2day.20171222start.log
#SBATCH -e /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts/run_cctm5.4+_Bench_2018_
  ↪ 12US1_M3DRY_cb6r3_ae6_20200131_MYR.256.16x16pe.2day.20171222start.log

# ===== CCTMv5.4.X Run Script =====
# Usage: run.cctm >&! cctm_2018_12US1.log &
#
# To report problems or request help with this script/program:
#         http://www.epa.gov/cmaq      (EPA CMAQ Website)
#         http://www.cmascenter.org    (CMAS Website)
# =====

# =====
#> Runtime Environment Options
# =====

echo 'Start Model Run At ' `date`

#> Toggle Diagnostic Mode which will print verbose information to
#> standard output
setenv CTM_DIAG_LVL 0
```

---

**Note:** In this run script, slurm or SBATCH requests 4 nodes, each node with 64 pes, or  $4 \times 64 = 576$  pes

---

Verify that the NPCOL and NPROW settings in the script are configured to match what is being requested in the SBATCH commands that tell slurm how many compute nodes to provision. In this case, to run CMAQ using on 256 cpus (SBATCH --nodes=4 and --ntasks-per-node=64), use NPCOL=16 and NPROW=16.

```
grep NPCOL run_cctm_2018_12US1_v54_cb6r3_ae6.20171222.csh
```

Output:

```
#> Horizontal domain decomposition
if ( $PROC == serial ) then
    setenv NPCOL_NPROW "1 1"; set NPROCS    = 1 # single processor setting
else
```

(continues on next page)

(continued from previous page)

```
@ NPCOL = 16; @ NPROW = 16
@ NPROCS = $NPCOL * $NPROW
setenv NPCOL_NPROW "$NPCOL $NPROW";
endif
```

### To run on the Shared Volume a code modification is required.

Note, we will use this modification when running on both lustre and shared.

Copy the BLD directory with a code modification to wr\_conc.F, wr\_aconc.F, and opaconc.F to your directory.

```
`cp -rp /shared/cyclecloud-cmaq/BLD_CCTM_v54+_gcc_codefix /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts`
```

### Build the code by running the makefile

```
cd /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts
```

Check to see you have the modules loaded

```
module list
```

Currently Loaded Modulefiles:

```
1) gcc-9.2.1 2) mpi/openmpi-4.1.1 3) hdf5-1.10.5/gcc-9.2.1 4) ioapi-3.2_20200828/
↪ gcc-9.2.1-hdf5
```

Run the Make command

```
make
```

Verify that the executable has been created

```
ls -lrt CCTM_v54+.exe
```

### Submit Job to Slurm Queue to run CMAQ on Lustre

```
cd /shared/build/openmpi_gcc/CMAQ_v54+/CCTM/scripts
```

```
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.2x96.ncclassic.retest.csh
```

### Check status of run

```
squeue
```

Output:

It takes about 5-8 minutes for the compute nodes to spin up, after the nodes are available, the status will change from CF to R.

## Successfully started run

squeue

## Once the job is successfully running

Check on the log file status

```
grep -i 'Processing completed.' run_cctm5.4+_Bench_2018_12US1_M3DRY_cb6r3_ae6_20200131_MYR.256.16x16pe.2day.20171222start.log
```

Output:

Once the job has completed running the two day benchmark check the log file for the timings.

```
tail -n 18 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.192.16x12pe.2day.20171222start.2x96.log
```

Output:

```
OUTDIR | /lustre/data_lim/output/output_v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_
↳ 12US1_2x96_classic

=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day:   2017-12-23
Number of Simulation Days: 2
Domain Name:      12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers:  35
Number of Processes: 192
    All times are in seconds.

Num  Day      Wall Time
01   2017-12-22  1813.3
02   2017-12-23  2077.7
    Total Time = 3891.00
    Avg. Time = 1945.50
```

## Submit a run script to run on the shared volume

To run on the shared volume, you need to copy the input data from the s3 bucket to the /shared volume. You don't want to copy directly from the /lustre volume, as this will copy more files than you need. The s3 copy script below copies only two days worth of data from the s3 bucket. If you copy from /lustre directory, you would be copying all of the files on the s3 bucket.

```
cd /shared/data
aws s3 --no-sign-request --region=us-east-1 cp --recursive s3://cmaq-cmaq/CMAQv5.4_2018_
↪12US1_Benchmark_2Day_Input .
```

## Submit a 96 pe job 1 nodes x 96 cpus on the EBS volume /shared

```
cd /shared/build/openmpi_gcc/CMAQ_v54+_classic/CCTM/scripts
sbatch run_cctm_2018_12US1_v54_cb6r5_ae6.20171222.1x96.ncclassic.retest.shared.csh
```

## The per-processor log files are being sent to the output directory

```
cd /shared/data/output/output_v54+_cb6r5_ae7_aq_WR413_MYR_gcc_2018_12US1_1x96_classic_
↪retest
grep -i 'Processing completed.' CTM_LOG_000*
```

Output:

```
Processing completed... 11.1089 seconds
Processing completed... 8.5691 seconds
Processing completed... 8.5596 seconds
Processing completed... 8.5419 seconds
Processing completed... 8.4918 seconds
Processing completed... 8.4257 seconds
Processing completed... 8.3264 seconds
Processing completed... 8.2567 seconds
Processing completed... 8.3922 seconds
Processing completed... 8.0141 seconds
Processing completed... 8.2161 seconds
Processing completed... 10.3496 seconds
```

```
tail -n 18 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.96.8x12pe.2day.
↪20171222start.1x96.shared.log
```

Output:

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
```

(continues on next page)

(continued from previous page)

```

Number of Grid Cells:      4803435 (ROW x COL x LAY)
Number of Layers:         35
Number of Processes:       96
    All times are in seconds.

```

```

Num  Day      Wall Time
01   2017-12-22  3744.5
02   2017-12-23  4184.8
    Total Time = 7929.30
    Avg. Time = 3964.65

```

Going to test running on 96 cpu/node on 2 nodes

```

sbatch run_cctm_2018_12US1_v54_cb6r3_ae6.20171222.2x96.csh

```

getting an error

```

Processing Day/Time [YYYYDDD:HHMMSS]: 2017356:095500
Which is Equivalent to (UTC): 9:55:00 Friday, Dec. 22, 2017
Time-Step Length (HHMMSS): 000500
    VDIFF completed...      2.5918 seconds
    COUPLE completed...     0.1093 seconds
    HADV completed...       1.7682 seconds
    ZADV completed...       0.1838 seconds
    HDIFF completed...      0.1922 seconds
    DECOUPLE completed...   0.0270 seconds
    PHOT completed...       0.1236 seconds
    CLDPROC completed...    1.2388 seconds
    CHEM completed...       0.6030 seconds
    AERO completed...       0.7680 seconds
    Master Time Step
    Processing completed...  7.6066 seconds

    ==> Data Output completed... 1.9993 seconds

Processing Day/Time [YYYYDDD:HHMMSS]: 2017356:100000
Which is Equivalent to (UTC): 10:00:00 Friday, Dec. 22, 2017
Time-Step Length (HHMMSS): 000500

```

```

-----
Primary job terminated normally, but 1 process returned
a non-zero exit code. Per user-direction, the job has been aborted.
-----

```

```

-----
mpirun noticed that process rank 7 with PID 0 on node cyclecloudlizadams-hpc-pg0-1
exited on signal 9 (Killed).
-----

```

```

30348.546u 996.456s 12:12.83 4277.2%      0+0k 9708555+14629912io 1223pf+0w

```

```

*****
** Runscript Detected an Error: CGRID file was not written. **
** This indicates that CMAQ was interrupted or an issue **

```

(continues on next page)

(continued from previous page)

```

**      exists with writing output. The runscript will now      **
**      abort rather than proceeding to subsequent days.      **
*****
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day:   2017-12-23
Number of Simulation Days: 1
Domain Name:      12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers:  35
Number of Processes: 192
    All times are in seconds.

Num  Day      Wall Time
01   2017-12-22  7
      Total Time = 7.00
      Avg. Time = 7.00
slurmstepd: error: Detected 2 oom-kill event(s) in StepId=47.batch cgroup. Some of your
↳ processes may have been killed by the cgroup out-of-memory handler.

```

Trying a run after converting the nc4 files to nc classic compressed files to see if the out-of-memory is due to a memory leak in the nc4 compressed libraries.

steps:

1. rebuild the CMAQv5.4+ code with uncompressed libraries

```
module load ioapi-3.2_20200828/gcc-9.2.1-netcdf
```

2. create a new project directory to compile the code

```
cd /shared/build/CMAQ_REPO_v54+
git pull
```

edit bldit\_project.csh to use

```
set CMAQ_HOME = /shared/build/openmpi_gcc/CMAQ_v54+_classic
```

Run bldit\_project.csh

```
./bldit_project.csh
```

Change directories to the new build location.

```
cd /shared/build/openmpi_gcc/CMAQ_v54+_classic
```

Copy the config\_cmaq.csh script from the v533 version, as that has the paths specified for the netCDF classic and I/O API libraries

```
cp /shared/build/openmpi_gcc/CMAQ_v533/config_cmaq.csh .
```

Edit to specify repo.

```
setenv CMAQ_REPO $BUILD/CMAQ_REPO_v54+
```

Run the bldit\_cctm.csh script

```
cd /shared/build/openmpi_gcc/CMAQ_v54+_classic/CCTM/scripts
./bldit_cctm.csh gcc |& tee bldit_cctm.log
```

Verify that the executable was created

```
ls -rlt */*.exe
```

Output:

```
-rwxrwxr-x. 1 lizadams lizadams 152894280 Mar 10 20:12 BLD_CCTM_v54+_gcc/CCTM_v54+.exe
```

Modify the run script to change the .nc4 extension to .nc and submit job

```
sbatch run_cctm_2018_12US1_v54_cb6r3_ae6.20171222.2x96.ncclassic.csh
```

This ran successfully to completion without getting a memory leak error.

Output

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 192
All times are in seconds.

Num Day Wall Time
01 2017-12-22 1767.5
02 2017-12-23 1991.3
Total Time = 3758.80
Avg. Time = 1879.40
```

### Submit a minimum of 2 benchmark runs

NOTE, trying to reproduce this run on HBv120 Cluster am getting slower times.

```
tail -n 30 run_cctm5.4+_Bench_2018_12US1_cb6r5_ae6_20200131_MYR.192.16x12pe.2day.
↪20171222start.2x96.shared.log
```

Output

```
=====
***** CMAQ TIMING REPORT *****
=====
Start Day: 2017-12-22
End Day: 2017-12-23
Number of Simulation Days: 2
Domain Name: 12US1
Number of Grid Cells: 4803435 (ROW x COL x LAY)
Number of Layers: 35
Number of Processes: 192
    All times are in seconds.

Num  Day      Wall Time
01  2017-12-22  2549.7
02  2017-12-23  2752.4
    Total Time = 5302.10
    Avg. Time = 2651.05
```

Ideally, two CMAQ runs should be submitted to the slurm queue, using two different NPCOLxNPROW configurations, to create output needed for the QA and Post Processing Sections in Chapter 6.