Air pollution modeling with CMAQ & WRF- CMAQ on AWS & AWS-Graviton2

The following guide provides instructions on how to launch EC2 instances and clusters running CMAQ & WRF-CMAQ with AMIs from the AWS Marketplace. These AMIs work either on single instances or in conjunction with AWS ParallelCluster for cluster creation. For users interested in working with a Linux desktop environment, the AMIs already include the extra elements and open the necessary ports; section II provides further details on how to create graphical sessions. This guide also includes descriptions on more advanced solutions such as AMI tailoring or storage options, which might be of interest to CMAQ users.

AWS offers a very large variety of services, which grow almost on a weekly basis. Many of our customers who are new to cloud environments and AWS use videoconferencing to get extra support. If either you are getting familiar with the AWS environment or wish to learn how to use the AMIs, feel free to contact us (support@odyhpc.com) to schedule any such session or for any other question.

I - Launching EC2 instances running CMAQ

The following instructions discuss how to launch EC2 instances running CMAQ and WRF-CMAQ.

1. **Account creation** – Air pollution modeling with CMAQ & WRF-CMAQ on AWS & AWS-Graviton2 with WRF can be used by any user with an active AWS account. If you are new to AWS, follow the instructions at https://aws.amazon.com/premiumsupport/knowledge-center/create-and-activate-aws-account/ to create a new account. You will also need to create an IAM user (https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_create.html) and assign the required permissions to create keypairs and launch instances (https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_change-permissions.html). You do not need the access & secret keys for running CMAQ on a single instance, but you will need them for more complex actions such as cluster launch or downloading data from S3 buckets. Either way, keep these keys safely as required by AWS.
2. **Subscribing to the AMI** - The AMIs with the preinstalled software can be downloaded from the AWS Marketplace (https://aws.amazon.com/marketplace/). Choose the right architecture (x86_64 or AArch64) depending on whether you want to use Intel or Graviton2 and subscribe. Make sure to understand the charges for AWS infrastructure and for the AMI.
3. **Launching instances** - Once your subscription is active, you can launch available instances based on your choices of architecture and configuration. More information about launching EC2 instances is available at https://docs.aws.amazon.com/quickstarts/latest/vmlaunch/step-1-launch-instance.html. The AMI is available in most regions and AZs. Contact us if you wish to use CMAQ on AWS GovCloud (US). A few tips for instance launch follow:
    3.1. Spot instances are available per the usual conditions.
    3.2. The AMIs are close to 100 GB but only about 10 GB are free. Users have several options to increase available space. Two easy options are to increase the storage at the time of launching instances or to use EBS volumes. The latter can be particularly useful for large datasets to be recycled with different instances and clusters. Section IV describes these options in greater detail.
    3.3. Choose the SG (Security Group) according to you own needs. As a minimum, it should have port 22 open. This is the default configuration.

3.4. Once you have selected the region, you will need to use your own existing keypair or create a new one for launching CMAQ.

4. **Connection to the instance** – In order to connect to your instance, you must use a SSH client such as PuTTY, MobaXterm or a similar app (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html). Your username is 'centos' and you enjoy superuser privileges.
5. **Running the apps** – The AMI comes with 4 precompiled flavors of CMAQ: standard CMAQ (CCTM_v533.exe), CMAQ-DDM3D (CCTM_DDM3D_v533.exe), CMAQ-ISAM (CCTM_ISAM_v533.exe), and WRF-CMAQ (wrf.exe). These executables are at the subdirectories /home/centos/CMAQ/CCTM/scripts/BLD_CCTM_v533_gcc, /home/centos/CMAQ-ISAM/CCTM/scripts/BLD_CCTM_v533_gcc???, /home/centos/CMAQ/CCTM/scripts/BLD_CCTM_v533_gcc, and /home/centos/WRF-CMAQ/CMAQ-v5.3.3/CCTM/scripts/WRF-4.3/test/em_real, respectively. We recommend using different approaches depending on whether you intend on running the app in single instances or clusters. The following lines describe how to use the former, whereas running jobs in clusters requires the use of Slurm (see specific directives for clusters in section V). For those relatively new to AWS, we also recommend gaining some experience running CMAQ in single instances before jumping into clusters, which are more complex to manage.

Advanced options

6. **Preprocessing tools** - The AMIs come with several preprocessing tools, which can be used to prepare your case as in any other Linux system. Contact us if you wish to use any other tool not available in the AMI.
   6.1. CMIO, BCON, ICON and CREATE_OMI are included in the CMAQ distribution. The compiled executables are available at the /home/centos/CMAQ/PREP subdirectory. To run the scripts, you can choose between the csh version included with the standard CMAQ distribution or a bash file.
   6.2. SMOKE is available at the /home/centos/PREPRO/SMOKE subdirectory.
7. **Postprocessing tools** – To use any postprocessing tool, it's better to use a graphical session as discussed in section II. Contact us if you wish to use any other tool not available in the AMI.
   7.1. VERDI. After opening an X-windows as described in section II, you can create a VERDI session by typing verdi.sh from the /home/centos/POSTPRO/VERDI_2.1 subdirectory.
   7.2. IDV v 6.0 also is included in the AMI.

1 – Test case – Southeast U.S. benchmark

The AMIs come with all the files necessary to run the Southeast U.S. benchmark. Running the test case as-it-is simply requires moving into the subdirectory and running the script:

```
$ cd /home/centos/CMAQ/CCTM/scripts
$ ./CMAQ533_ncores64_12SE.sh

$ cd /home/centos/CMAQ-ISAM/CCTM/scripts
$ ./ISAM533_ncores64_12SE.sh

$ cd /home/centos/CMAQ-DDM3D/CCTM/scripts
$ ./DDM3D533_ncores64_12SE.sh
```

Here, it is important to notice the CMAQ flavor to be run and the number of MPI ranks that must be equal or lesser than the number of cores for the selected instance. The test-case, which can be used either as it-is or slightly modified, can help users to either get acclimated with the AWS environment and the AMI itself. It is possible to use scripts in either csh or bash; however, we strongly recommend using bash for complex scripts as they are a better match with the OS and the AWS environment.

Something that needs to be kept in mind when running HPC simulations is memory requirements. CMAQ is not particularly memory hungry compared to other HPC apps. The test case is suitable to run on single instances with 8 GB of memory or higher. If memory is insufficient, the simulation will break down with an error message like:

```
--------------------------------------------------------------------------
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 2 with PID 0 on node ip-172-31-3-9 exited on signal 9 (Killed).
--------------------------------------------------------------------------
```

If memory is sufficient and everything else goes without a problem, the app will complete the simulation and the output files will be in the `/home/centos/CMAQ/data` subdirectory unless otherwise indicated. It is easily feasible to modify the test case in many ways to test different aspects of the simulations.

2 – WRF-CMAQ – Southeast U.S. benchmark

The AMIs also include the files necessary to run the WRF-CMAQ Southeast U.S. benchmark. In a similar fashion to CMAQ, running the benchmark requires moving into the subdirectory and running the script:

```
$ cd /home/centos/CMAQ/CCTM/scripts
$ ./CMAQ533_ncores64_12SE.sh
```

Running the WRF-CMAQ benchmark is much more computationally demanding than running the CMAQ benchmark and it takes about 5 times the previous time. However, our measurements also show significant gains and an excellent scalability when using large instances.

II. Using a Linux desktop environment

CMAQ users exploring the option of migrating to the public cloud cite the lack of a graphical environment as one of the major drawbacks associated with this option. To overcome this challenge with a simple procedure, the AMIs are fully ready to start a Linux desktop environment session. Launching the AMI will also open the necessary ports so that users do not need to modify SGs or worry about other networking setups.

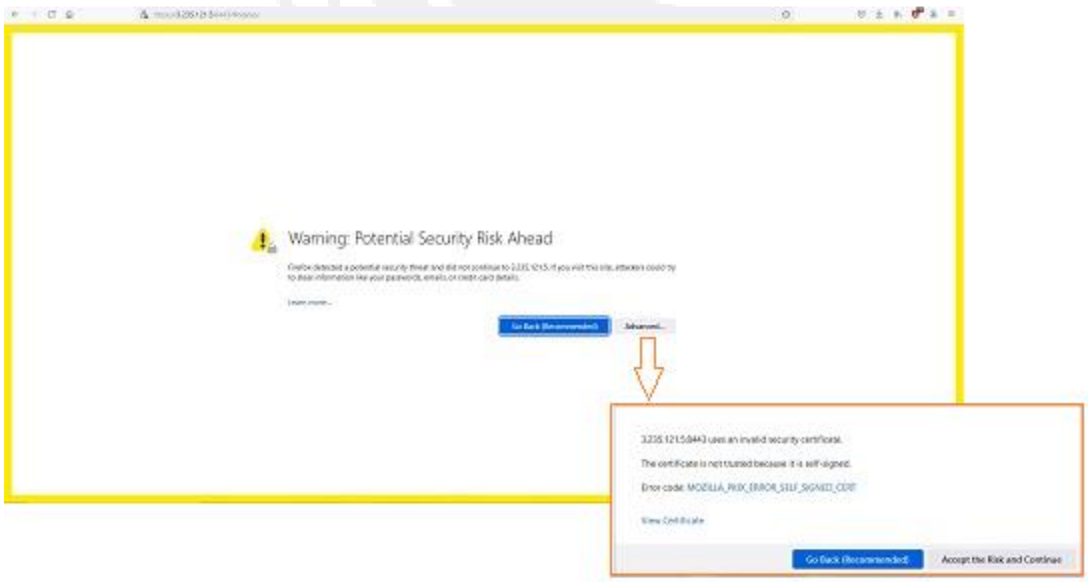
The AMIs utilize a high-performance display app named NICE-DCV (https://docs.aws.amazon.com/dcv/index.html) and currently owned by AWS. In the case of single instances, the only steps to initialize the desktop environment are to start the server and initialize the session[1] from a terminal:

```
$ sudo systemctl start dcvserver
$ dcv create-session custom_name_for_your_session
```

To check if the session is running and its characteristics, you can simply type:

```
$ dcv list-sessions
$ dcv create-session custom_name_for_your_session
```

The session will be available in most web browsers at https://External_IP_for_your_instance:8443/#custom_name_for_your_session.
The first 2 screens will inform you of a potential certificate issue; this is normal as the app uses a self-certificate with NGINX.



Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to 3.235.121.5. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

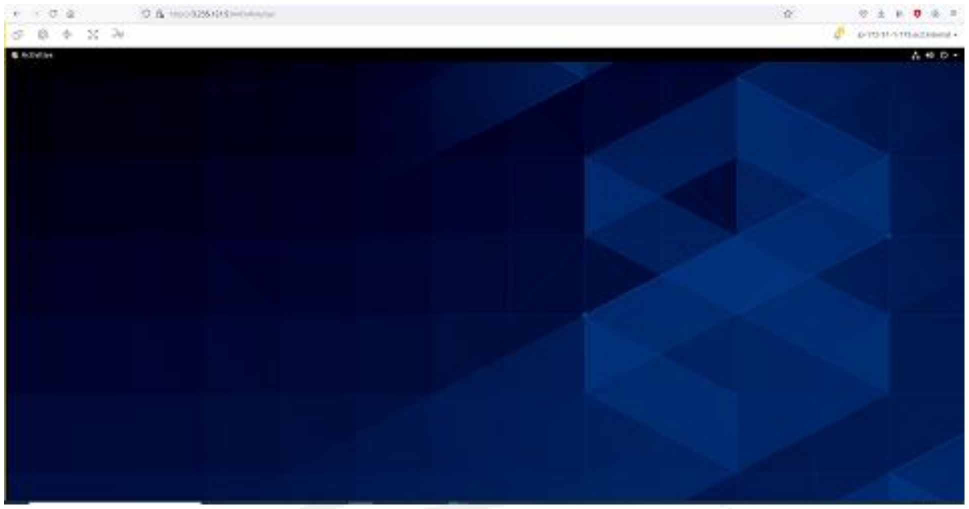
3.235.121.5:8443 uses an invalid security certificate.

The certificate is not trusted because it is self-signed.

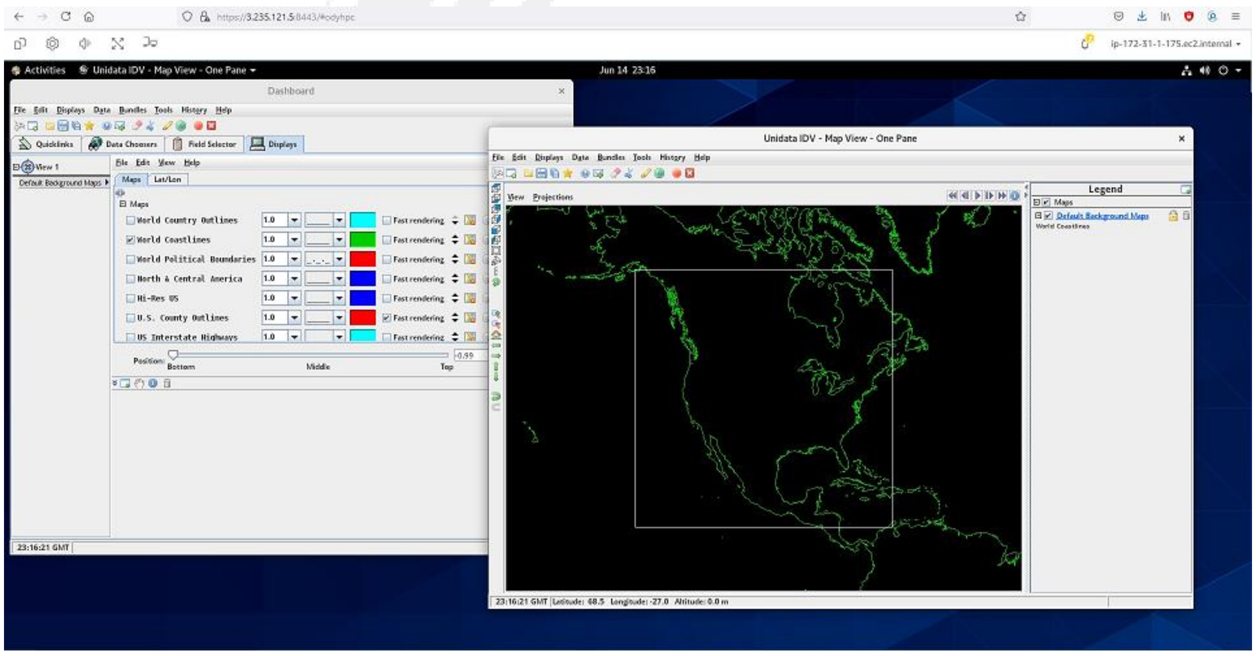
Error code: MOZILLA_PKIX_ERROR_SELF_SIGNED_CERT

[1] Graphical sessions consume resources quickly, particularly memory. Although it is possible to start a session with 4 GB of memory, we recommend using an instance with a minimum memory of 8GB or even more if any visualization app is going to be used.

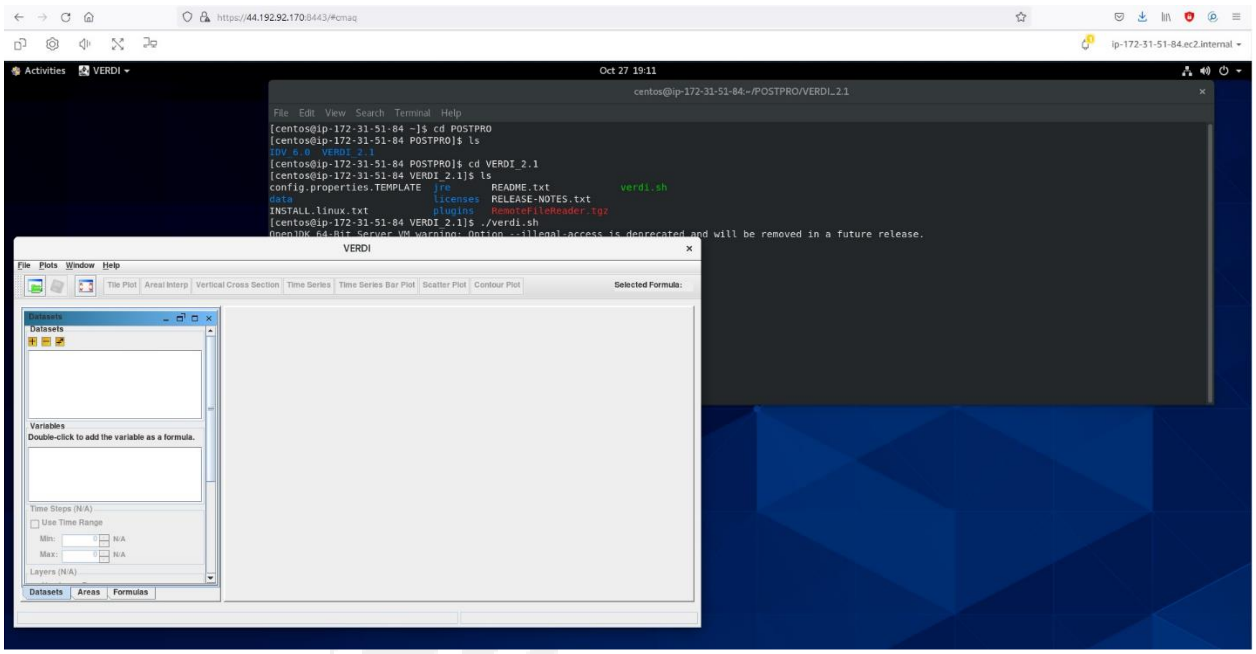
After clicking on 'Accept Risk and Continue,' the GNOME welcome screen will appear:



If, for example, IDV is installed and opened, the app can be used as with any other graphical desktop:



Opening a VERDI session looks like:



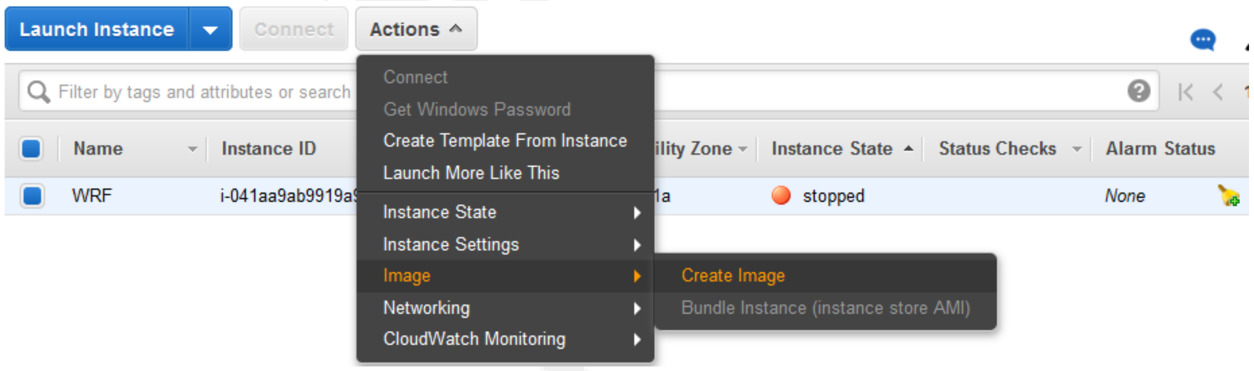
```
[centos@ip-172-31-51-84 ~]$ cd POSTPRO
[centos@ip-172-31-51-84 POSTPRO]$ ls
IDV_6.0  VERDI_2.1
[centos@ip-172-31-51-84 POSTPRO]$ cd VERDI_2.1
[centos@ip-172-31-51-84 VERDI_2.1]$ ls
config.properties.TEMPLATE  jre       README.txt            verdi.sh
data                        licenses  RELEASE-NOTES.txt
INSTALL.linux.txt           plugins   RemoteFileReader.tgz
[centos@ip-172-31-51-84 VERDI_2.1]$ ./verdi.sh
OpenJDK 64-Bit Server VM warning: Option --illegal-access is deprecated and will be removed in a future release.
```

III. Tailoring your AMI

The CMAQ AMI is built with the latest stable version of the app and with the U.S. Southeast case. If a new case is going to be reused several times or you wish to store it long-term, we strongly recommend using AWS EBS (Elastic Block Storage) and AWS S3 (Simple Storage Service), respectively.

Before launching the instance and building the new case, it is important to assess how much space will be needed. Most of our custom AMIs have little free space (usually around 10 or 20 GB) but, at the time of launching an instance, it is feasible to increase the available space. Using the console provides you with this feature in Step 4. You can select between increasing the size of the root volume or adding a new volume. The former is easy to select, and no further step is needed, whereas the latter requires mounting the new volume but has the advantage of dedicated EBS to that case (see the storage options section for more details).

Once the new case is available and any other tailoring is complete, you can build your own AMI by clicking on 'Creating Image' as shown in the screenshot or using the CLI.

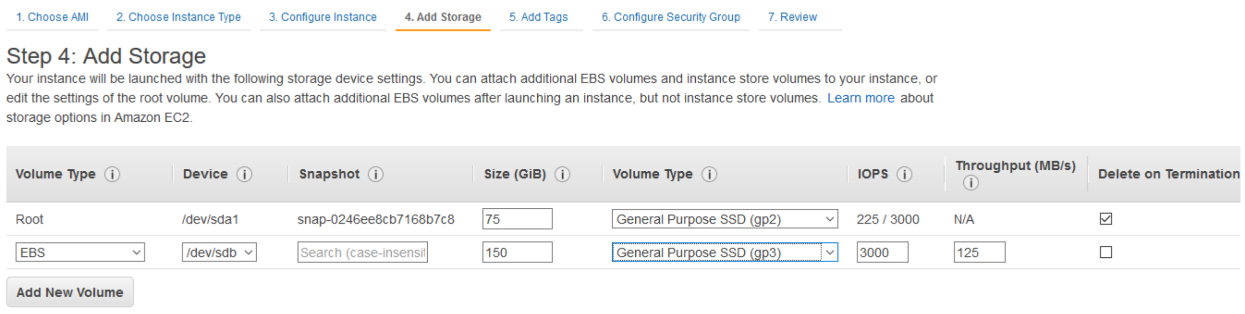


The new AMI is available only to IAMs in your account. It has the same features as the original AMI including its availability for launching clusters. However, and unlike the original AMI, it is available only in the region where created.

IV. Storage options

One of the many advantages of using AWS infrastructure is the availability of many storage solutions meeting different needs and criteria. It is not the purpose of this section to cover every available option, which has its own voluminous documentation, but to describe a few options of interest to CMAQ users. The two most common options are EBS (Elastic Block Store) and S3 (Simple Storage Service), which should meet most demands. Two more advanced options include the use of Lustre filesystems for high performance I/O attached to instances or clusters, and Amazon EFS (Elastic File System), which can provide a general filesystem attached to several instances or clusters.

The use of EBS is the most common as AWS AMIs are imprinted in this format at the time of instance creation. The AMI contains, the OS, apps, and dependencies, and constitute the root volume associated to the instance. In the case of AMIs containing CMAQ, the minimum root volume is usually around 100 GB with little free space. When launching an instance, increasing the available disk space can be accomplished either through increasing the root volume size (the system will not accept a smaller size) or by attaching another EBS to the instance as shown in the screenshot.



Step 4: Add Storage

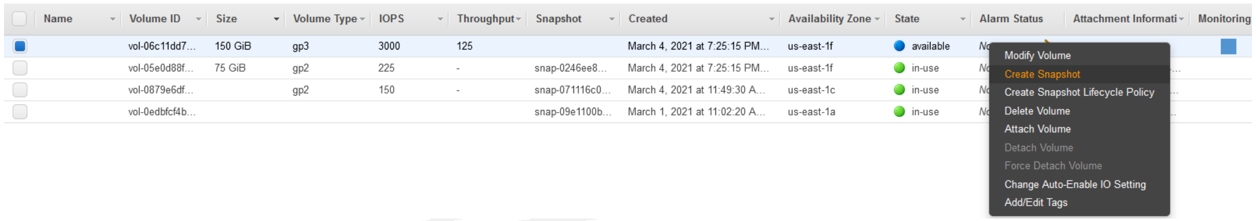
Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination
Root	/dev/sda1	snap-0246ee8cb7168b7c8	75	General Purpose SSD (gp2)	225 / 3000	N/A	✓
EBS	/dev/sdb	Search (case-insensi	150	General Purpose SSD (gp3)	3000	125	☐

In this example, we are adding 150 GB of disk space with the caveat that our choice is of the type gp3, which offers better performance than the standard gp2 type. A discussion on the technical specifications of different EBS types is available, for example, at https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html. The advantage of using this extra EBS volume (instead of simply increasing the root volume) is that the data in this filesystem can be used separately from the instance including its attachment to other instances or clusters. This extra space is not immediately available, but it needs to be mounted. An example of how to mount this EBS volume on the data subdirectory would be:

```
$ lsblk                              ! To check the name of the volume
$ sudo file -s /dev/nvme1n1
$ sudo mkfs -t xfs /dev/nvme1n1
$ sudo mkdir data
$ sudo mount /dev/nvme1n1 data
$ sudo chown centos:centos data
```

After creating or running any case, and to reuse or store this block, we can simply unmount it `$ sudo umount -d /dev/nvme1n1`, detach the volume and create a snapshot from the console volume menu.



	Name	Volume ID	Size	Volume Type	IOPS	Throughput	Snapshot	Created	Availability Zone	State	Alarm Status	Attachment Informati	Monitoring
■		vol-06c11dd7...	150 GiB	gp3	3000	125		March 4, 2021 at 7:25:15 PM...	us-east-1f	● available	N		
☐		vol-05e0d88f...	75 GiB	gp2	225	-	snap-0246ee8...	March 4, 2021 at 7:25:15 PM...	us-east-1f	● in-use	N	...	
☐		vol-0879e6df...		gp2	150	-	snap-071116c0...	March 4, 2021 at 11:49:30 A...	us-east-1c	● in-use	N	...	
☐		vol-0edbfcf4b...					snap-09e1100b...	March 1, 2021 at 11:02:20 A...	us-east-1a	● in-use	N	...	

Also, it is important to notice that we did not select 'Delete on Termination' previously so terminating the instance will not delete the volume and the meter will keep running until its deletion from the console or alternatively the CLI.

S3 is more adequate for long-term storage, backup, and access to large databases. To access non-public buckets belonging to your organization, it is necessary to configure your credentials with `$ aws configure`, which will ask for the IAM credentials (aws_access_key_id and aws_secret_access_key). There is plenty of online help and discussions on AWS S3, plus the official documentation (https://docs.aws.amazon.com/s3/index.html), which is voluminous.

V. Launching AWS clusters running CMAQ

This section provides instructions on how to launch HPC clusters using custom AMIs. It is meant to be a complement to the official AWS ParallelCluster documentation (https://docs.aws.amazon.com/parallelcluster/), which describes exhaustively all AWS ParallelCluster features. If you are familiar with AWS ParallelCluster, many of these steps will be familiar. If this is your first venture with AWS ParallelCluster, enjoy the journey and follow these instructions to get you started. Launching an HPC cluster is more involved than launching a single instance so feel free to contact us at any time or schedule a video conference if you would prefer a walkthrough the different steps.

The process of launching an HPC cluster can be divided into several steps:
Basic functions

1. Preliminary steps
2. Cluster design
3. Launching the cluster and connecting to it
4. Submitting jobs to the cluster scheduler

Advanced functions

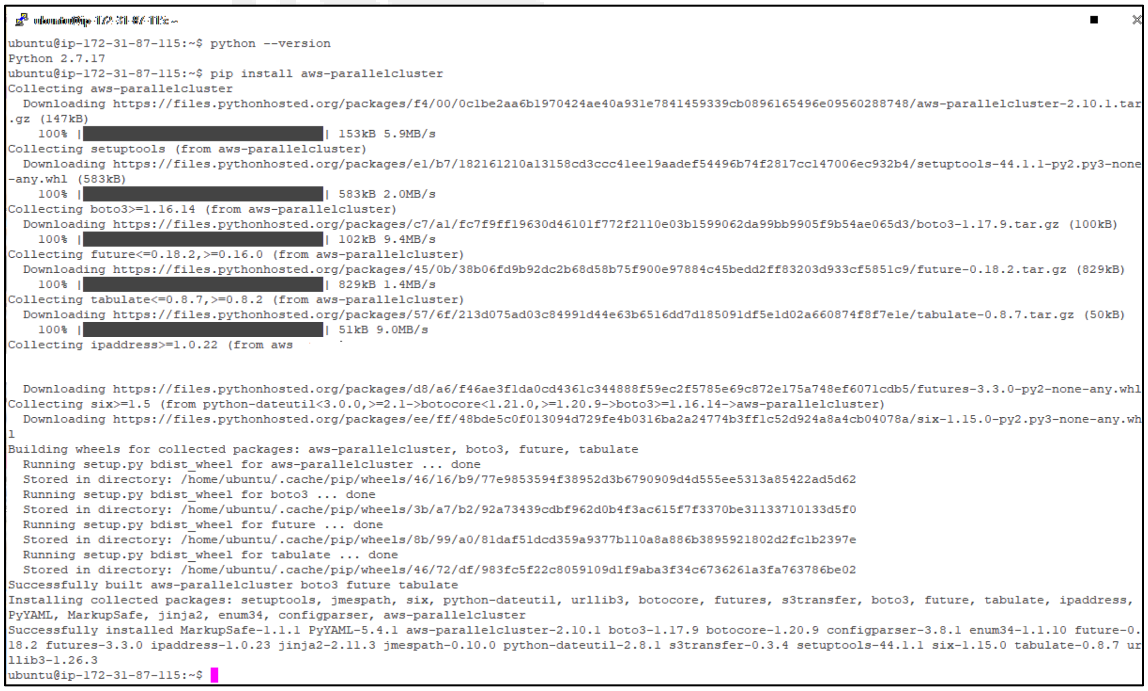
5. Using the master instance for high-performance display
6. Monitoring the cluster

The basic functions are easy to complete and will get your cluster up and running in a relatively short period of time.

1. Preliminary steps

AWS has developed AWS ParallelCluster, which renders launching HPC clusters in the cloud significantly easier than it was a few years ago. If you are already familiar with AWS ParallelCluster, you can skip this step. Before proceeding, make sure that you have available a copy of the following data: the name of the region where the cluster will be launched along with one of the keypairs available in that region, the AMI ID for that region, and the IAM credentials (aws_access_key_id and aws_secret_access_key).

a. AWS ParallelCluster Installation - AWS ParallelCluster can be installed either in virtualized or non-virtualized environments. It runs on Linux, Windows or macOS. Although the AWS ParallelCluster team recommends using a virtualized environment, we have installed AWS ParallelCluster in non-virtualized environments and found no problem. Whichever are your choices for environment and OS, your system must have python and pip (or python3 and pip3) installed. If that is not the situation, you can follow specific instructions for any combination of environment and OS available at https://docs.aws.amazon.com/parallelcluster/latest/ug/install.html.
Once pip is installed, the command `$pip/pip3 install aws-parallelcluster` will install the app and some dependencies. The system will show the progress of the installation and list the installed apps at the end of it.



```
ubuntu@ip-172-31-87-115:~$ python --version
Python 2.7.17
ubuntu@ip-172-31-87-115:~$ pip install aws-parallelcluster
Collecting aws-parallelcluster
  Downloading https://files.pythonhosted.org/packages/f4/00/0c1be2aa6b1970424ae40a931e7841459339cb0896165496e09560288748/aws-parallelcluster-2.10.1.tar
.gz (147kB)
    100% |████████████████████████████████| 153kB 5.9MB/s
Collecting setuptools (from aws-parallelcluster)
  Downloading https://files.pythonhosted.org/packages/e1/b7/182161210a13158cd3ccc41ee19aadef54496b74f2817cc147006ec932b4/setuptools-44.1.1-py2.py3-none
-any.whl (583kB)
    100% |████████████████████████████████| 583kB 2.0MB/s
Collecting boto3>=1.16.14 (from aws-parallelcluster)
  Downloading https://files.pythonhosted.org/packages/c7/a1/fc7f9ff19630d46101f772f2110e03b1599062da99bb9905f9b54ae065d3/boto3-1.17.9.tar.gz (100kB)
    100% |████████████████████████████████| 102kB 9.4MB/s
Collecting future<=0.18.2,>=0.16.0 (from aws-parallelcluster)
  Downloading https://files.pythonhosted.org/packages/45/0b/38b06fd9b92dc2b68d58b75f900e97884c45bedd2ff83203d933cf5851c9/future-0.18.2.tar.gz (829kB)
    100% |████████████████████████████████| 829kB 1.4MB/s
Collecting tabulate<=0.8.7,>=0.8.2 (from aws-parallelcluster)
  Downloading https://files.pythonhosted.org/packages/57/6f/213d075ad03c84991d44e63b6516dd7d185091df5e1d02a660874f8f7e1e/tabulate-0.8.7.tar.gz (50kB)
    100% |████████████████████████████████| 51kB 9.0MB/s
Collecting ipaddress>=1.0.22 (from aws


  Downloading https://files.pythonhosted.org/packages/d8/a6/f46ae3f1da0cd4361c344888f59ec2f5785e69c872e175a748ef6071cdb5/futures-3.3.0-py2-none-any.whl
Collecting six>=1.5 (from python-dateutil<3.0.0,>=2.1->botocore<1.21.0,>=1.20.9->boto3>=1.16.14->aws-parallelcluster)
  Downloading https://files.pythonhosted.org/packages/ee/ff/48bde5c0f013094d729fe4b0316ba2a24774b3ff1c52d924a8a4cb04078a/six-1.15.0-py2.py3-none-any.wh
l
Building wheels for collected packages: aws-parallelcluster, boto3, future, tabulate
  Running setup.py bdist_wheel for aws-parallelcluster ... done
  Stored in directory: /home/ubuntu/.cache/pip/wheels/46/16/b9/77e9853594f38952d3b6790909d4d555ee5313a85422ad5d62
  Running setup.py bdist_wheel for boto3 ... done
  Stored in directory: /home/ubuntu/.cache/pip/wheels/3b/a7/b2/92a73439cdbf962d0b4f3ac615f7f3370be31133710133d5f0
  Running setup.py bdist_wheel for future ... done
  Stored in directory: /home/ubuntu/.cache/pip/wheels/8b/99/a0/81daf51dcd359a9377b110a8a886b3895921802d2fc1b2397e
  Running setup.py bdist_wheel for tabulate ... done
  Stored in directory: /home/ubuntu/.cache/pip/wheels/46/72/df/983fc5f22c8059109d1f9aba3f34c6736261a3fa763786be02
Successfully built aws-parallelcluster boto3 future tabulate
Installing collected packages: setuptools, jmespath, six, python-dateutil, urllib3, botocore, futures, s3transfer, boto3, future, tabulate, ipaddress,
PyYAML, MarkupSafe, jinja2, enum34, configparser, aws-parallelcluster
Successfully installed MarkupSafe-1.1.1 PyYAML-5.4.1 aws-parallelcluster-2.10.1 boto3-1.17.9 botocore-1.20.9 configparser-3.8.1 enum34-1.1.10 future-0.
18.2 futures-3.3.0 ipaddress-1.0.23 jinja2-2.11.3 jmespath-0.10.0 python-dateutil-2.8.1 s3transfer-0.3.4 setuptools-44.1.1 six-1.15.0 tabulate-0.8.7 ur
llib3-1.26.3
ubuntu@ip-172-31-87-115:~$
```

b. AWS ParallelCluster Configuration
The first step is to set up the AWS credentials if you have not done it yet. The command `$ aws configure` will ask for the IAM's access key ID and secret access key along with the default region.
The second step is to begin the configuration with the `$ pcluster configure` command. The system will ask you a few standard questions. We recommend answering most everything by default except when prompted about the automatic creation of a Virtual Public Cloud (VPC), which requires deciding whether to create a new VPC dedicated to clusters or use an existing VPC. If you select the former, we recommend choosing a public subnet for the head node and private subnets for the compute nodes; once the information is introduced, the system will automatically create several VPC resources that will also appear in the VPC menu of the AWS console. If the choice is to use an existing VPC and subnet, you will need their IDS as discussed in the 'cluster design' subsection. In either case, the command `pcluster` will create a subdirectory (e.g. `.parallelcluster`) with a config file in it. You can substitute this file a bit later, but it contains some identification parameters that will be needed during the cluster design phase.
The third step is simply to check the version with `$ pcluster version` (without any dash), which returns the AWS ParallelCluster version. It is strongly encouraged to match the running AWS ParallelCluster version with the version used to create the AMI. This piece of data is detailed in the awspcluster_version.txt file located at `/home/centos/CLUSTERs` subdirectory. AMIs created with an older pcluster version usually do not work well with a newer version of pcluster. Therefore, and if the versions do not match, we recommend changing the pcluster version with $pip/pip3 install aws-parallelcluster==[awsparallelcluster_version_used_to_create_AMI].

2. Cluster design

AWS ParallelCluster provides great flexibility in the configuration of HPC clusters, which results in an exceptionally large number of potential designs. Hence, this step is the most involved but completing the mandatory adjustments suffice to launch a cluster with the integrated apps ready for production purposes.

Launching a cluster requires a configuration script describing cluster details. Instead of using the default file created at the time of running 'pcluster configure,' we recommend modifying your own (if you are an experienced AWS ParallelCluster user) or using one of the templates found at the CLUSTERs subdirectory in the AMI. As the number of different designs can grow over time, it is convenient to use your own naming & archival conventions; then, copy the file with the desired configuration into the 'config' file located in the base subdirectory that was created when running $pcluster configure.

The CLUSTERs subdirectory in the AMI also contains a file describing the cluster characteristics that will be launched for each script. These templates can be used with minimal modifications or recycled for further cluster tailoring.

The provided scripts in the AMI require a minimum of changes before launching. Some of these changes are mandatory, while others are optional.

- Mandatory adjustments
- Optional changes
- Advanced options

1. Mandatory adjustments

The parameters discussed below constitute the minimum modifications necessary to launch a cluster. If this is the first-time using AWS ParallelCluster, pick one of the configuration files included in the AMI and complete the mandatory adjustments. After completing these steps, you should be ready to launch a cluster following the steps described in section iii.

[aws] section

The parameter aws_region_name is set to *us-east-1* by default. It should be modified accordingly if the cluster is going to be launched in any other region.

The parameters aws_access_key_id and aws_secret_access_key must be set with the account credentials.

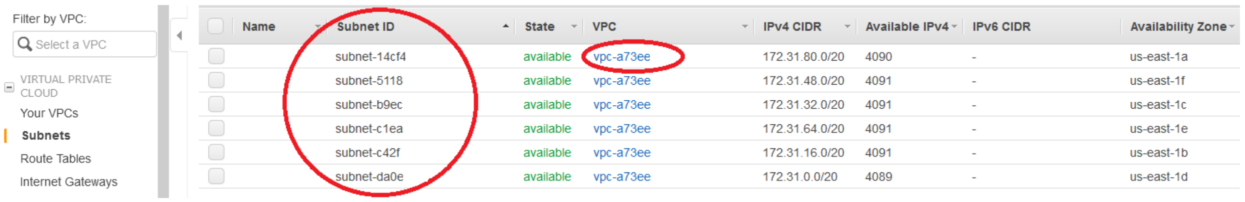
[cluster cmaqcluster] section

The parameter key_name must be set to the name of your own keypair in the selected region.

The parameter custom_ami must be set to the desired value. Here, it is important to keep in mind that although the AMI is the same across all regions, AWS assigns different IDs to each region. The $ pcluster create command will produce an error message if it finds any discrepancy between AMI ID and region.

[vpc cmaqfvpc] section

This step assumes that that the cluster will use an existing VPC and subnet associated with your account. See advanced options for other network configurations. The IDs for these resources can be easily found in the VPC submenu of the console:



Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	IPv6 CIDR	Availability Zone
	subnet-14cf4	available	vpc-a73ee	172.31.80.0/20	4090	-	us-east-1a
	subnet-5118	available	vpc-a73ee	172.31.48.0/20	4091	-	us-east-1f
	subnet-b9ec	available	vpc-a73ee	172.31.32.0/20	4091	-	us-east-1c
	subnet-c1ea	available	vpc-a73ee	172.31.64.0/20	4091	-	us-east-1e
	subnet-c42f	available	vpc-a73ee	172.31.16.0/20	4091	-	us-east-1b
	subnet-da0e	available	vpc-a73ee	172.31.0.0/20	4089	-	us-east-1d

Copy these IDs and set the parameters master_subnet_id and vpc_id in the configuration script with them. Notice that depending on your choice of subnet, the cluster will launch in a specific AZ.

b. Optional changes

The following parameters allow users to modify the cluster size and choose spot instances.

[cluster] section

master_instance_type: The default instance for this parameter is c5.large. The master instance does not perform any computation, but it hosts the scheduler and other services. Hence, it can be a small instance (e.g. c5.large or m5.large) unless it is also being used for graphics processing (see advanced options).

compute_instance_type: These are the instances where the apps run on and should be chosen accordingly.

To set the cluster size, there are 2 groups of parameters. The first group uses initial_queue_size, max_queue_size & maintain_initial_size; the second group includes the min_vcpus, desired_vcpus and max_vcpus parameters. We prefer the first option, but the CLUSTERs subdirectory also has templates using the second group.

Case 1

Cluster with a constant number of 4 c5.12xlarge compute instances:

```
[cluster cmaqcluster]
…
master_instance_type = c5.large
compute_instance_type = c5.12xlarge
cluster_type = spot
disable_hyperthreading = true
initial_queue_size = 4
max_queue_size = 4
maintain_initial_size = true
…
```

This script would launch the cluster with spot instances. Alternatively, it could read:

```
[cluster cmaqcluster]
…
master_instance_type = c5.large
compute_instance_type = c5.12xlarge
cluster_type = spot
disable_hyperthreading = true
min_vcpus = 192
desired_vcpus = 192
max_vcpus = 192
…
```

Case 2

For arm-powered clusters, both master and compute must be AArch64 instances. For example, the script for a cluster with 4 c5.12xlarge compute instances would read:

```
[cluster cmaqarmcluster]
…
master_instance_type = a1.large
compute_instance_type = c6g.12xlarge
cluster_type = ondemand
initial_queue_size = 4
max_queue_size = 4
maintain_initial_size = true
…
```

or

```
[cluster cmaqarmcluster]
…
master_instance_type = a1.large
compute_instance_type = c6g.12xlarge
cluster_type = ondemand
min_vcpus = 192
desired_vcpus = 192
max_vcpus = 192
…
```

Case 3

If, instead of having a fixed number of instances, the objective is to have a variable number of instances controlled by the scheduler, the script would read:

```
[cluster cmaqcluster]
…
master_instance_type = c5.large
compute_instance_type = c5.12xlarge
cluster_type = spot
disable_hyperthreading = true
initial_queue_size = 2
max_queue_size = 8
maintain_initial_size = false
…
```

With this script, the cluster launches with 2 instances, and will terminate or spin up more instances (up to 8) depending on the scheduler load.

Case 4

When using any of m5dn.24xlarge, m5n.24xlarge, m5zn.12xlarge, m5zn.metal, c5n.18xlarge, c5n.metal, c6gn.16xlarge, r5dn.24xlarge or r5n.24xlarge as compute instances, it is important to tell AWS ParallelCluster that network communications must use EFA. The scripts for Example 1 should look like as follows:

```
[cluster cmaqcluster]
…
master_instance_type = c5.large
compute_instance_type = c5n.18xlarge
cluster_type = spot
enable_efa = compute
disable_hyperthreading = true
initial_queue_size = 4
max_queue_size = 4
maintain_initial_size = false
…
```

```
[cluster cmaqarmcluster]
…
master_instance_type = c6g.large
compute_instance_type = c6gn.16xlarge
cluster_type = ondemand
enable_efa = compute
initial_queue_size = 4
max_queue_size = 4
maintain_initial_size = false
…
```

Using instances with EFA capabilities is strongly recommended for very large workloads.

c. Advanced options

AWS ParallelCluster allows further tailoring depending on the end-user needs. Here, we discuss a few options that can be combined with odyhpc AMIs.

- Enable hyperthreading

  Although not recommended for most workloads, it is possible to enable hyperthreading for Intel-powered insatnces by simply erasing the 'disable_hyperthreading' line in the configuration script.
- Adding storage

  In a similar way to single instances, it is feasible and easy to increase the available storage for the cluster.
  - Increase the base volume size. This is the easiest solution for simply running computations that require more space than available in the AMI. It only requires changing the values of master_root_volume_size and compute_root_volume_size to the desired capacity.
  - Attach EBS volume

    This option requires adding the name of the EBS volume to the cluster

    ```
    [cluster cmaqcluster]
    …
    ebs_settings = cmaqebs1
    …
    ```

    and specifying the characteristics of the EBS volume in an added section

    ```
    [ebs cmaqebs1]
    shared_dir = vol1
    ebs_snapshot_id = snap-xxxxx
    volume_type = gp3
    ```
  - Attach Lustre filesystem

    This option also requires adding the Lustre filesystem to the cluster

    ```
    [cluster cmaqcluster]
    …
    fsx_settings = fscmaq
    …
    ```

    and specifying the characteristics

    ```
    [ebs fscmaq]
    shared_dir = /fsx
    storage_capacity = 3600
    imported_file_chunk_size = 1024
    export_path = s3://bucket/folder
    import_path = s3://bucket
    ```

3. Launch and connection to the cluster

Once the scrip file is ready, copy it to your AWS ParallelCluster base directory and start the cluster with `$ pcluster create cmaqcluster`. The launch will go through several stages, as reported on the screen. This is not required but you can further visualize progress on the AWS console. For example, the following screenshot shows the on-going process while launching a small cluster with the master instance and 4 compute instances.

These are names given by AWS ParallelCluster. Do not log in onto the master instance until the process is complete even if the console shows all the instances running. The screen will indicate completion with a message like:

```
Status: parallelcluster-c1cluster - CREATE_COMPLETE
MasterPublicIP: 100.26.65.162
ClusterUser: centos
MasterPrivateIP: 172.31.38.235
```

You can connect to the master instance and perform a preliminary check that everything is up and running with `$ sinfo`, which should show information about the compute instances.

When you are finished with your computations, end the cluster with `$ pcluster delete cmaqcluster`. Manually deleting cluster resources is feasible but strongly not recommended; it is easy to forget some of the many resources used by AWS ParallelCluster and you will still need to run `$ pcluster delete wrfcluster` before launching any other cluster.

4. Submitting jobs to the cluster scheduler

Clusters run with the Slurm workload manager (submitting jobs with mpirun is not allowed unlike on single instances). If you are unfamiliar with this scheduler, there are many online tutorials and blogs in addition to the own Slurm website (https://slurm.schedmd.com/documentation.html). To simplify matters, there are several Slurm scripts in the BATCH subdirectory ready for job submission, which can be submitted with $ sbatch cmaq_256.sbatch or $ sbatch cmaqGrav_256.sbatch (for Graviton2 clusters). This command will run the case with 256 MPI ranks.

5. Using the master instance for high-performance display

One of the advantages of using AWS for HPC computations is the availability of more flexible & cheaper options for storage and post-processing purposes than traditional on-premises infrastructure. Many HPC apps require high-performance visualization capabilities as part of their post-processing tasks. In addition to a wide selection of instances with acceleration capabilities, AWS offers NICE-DCV (https://aws.amazon.com/hpc/dcv/) that is a high-performance remote display protocol useful for HPC and other graphically intensive workloads. A single instance can host NICE-DCV but the master node of a cluster can also serve as the hosting hardware. To accomplish this task, the cluster configuration script must be modified accordingly. The 'configuration_script_details' file in the CLUSTERs subdirectory provides the details for examples (e.g. DCV_1) where the master instance also hosts NICE-DCV.

Hosting NICE-DCV is usually more computationally expensive than most other tasks handled by the master instance. It is therefore recommended to select an instance larger than when no hosting NICE-DCV. It is impossible to provide specific rules for this selection as there are too many variables that impact performance. Users will have to try and test different choices depending on their goals. However, we can provide some general recommendations to begin the process. When using x86_64 architecture, the choice will depend on the intended graphic-processing use. If nothing or very slight, an instance with larger memory (e.g. r5 series) should suffice. If the purpose is graphic intensive, we recommend trying P2 (Nvidia K-80) and P3 (Nvidia V-100) instances. For extreme graphic processing capabilities, P4 instances with Nvidia A-100 are also available, but users should be aware of the price tag as they are some of the most expensive instances offered by AWS.

6. Monitoring the cluster

AWS provides several tools to monitor and analyze usage of its many resources. The most widely used of these tools is CloudWatch, which covers much of AWS infrastructure. Although CloudWatch can be used to monitor several cluster variables, HPC clusters are complex structures that benefit from more sophisticated monitoring tools. To accomplish this goal, WRF clusters can be launched with several add-ons facilitating the access to Grafana dashboards and Prometheus monitoring tools. These add-ons are activated by including a postprocessing script that automatically loads these features during cluster launch; examples on how to do this are included in the CLUSTERs subdirectory (MONI_XXX files). The procedure to launch the cluster does not change and, once this is complete, the dashboards will be available via any web browser at https://your_public_ip. After acknowledging that you understand that the connection uses a self-certificate (see page 6 for more details), a menu page will appear.