Prepared for:

Northwestern University

HIPAA Compliant CloudHPC

POC Build Document

X-ISS

9800 Northwest Freeway, Suite 305

Houston, TX 77092

713-862-9200

www.x-iss.com

1. Overview

This document contains technical information about the setup of the HIPAA compliant CloudHPC proof of concept, and tutorials on management and usage of said systems. The build document of a production system should look similar to what is shown in this document.

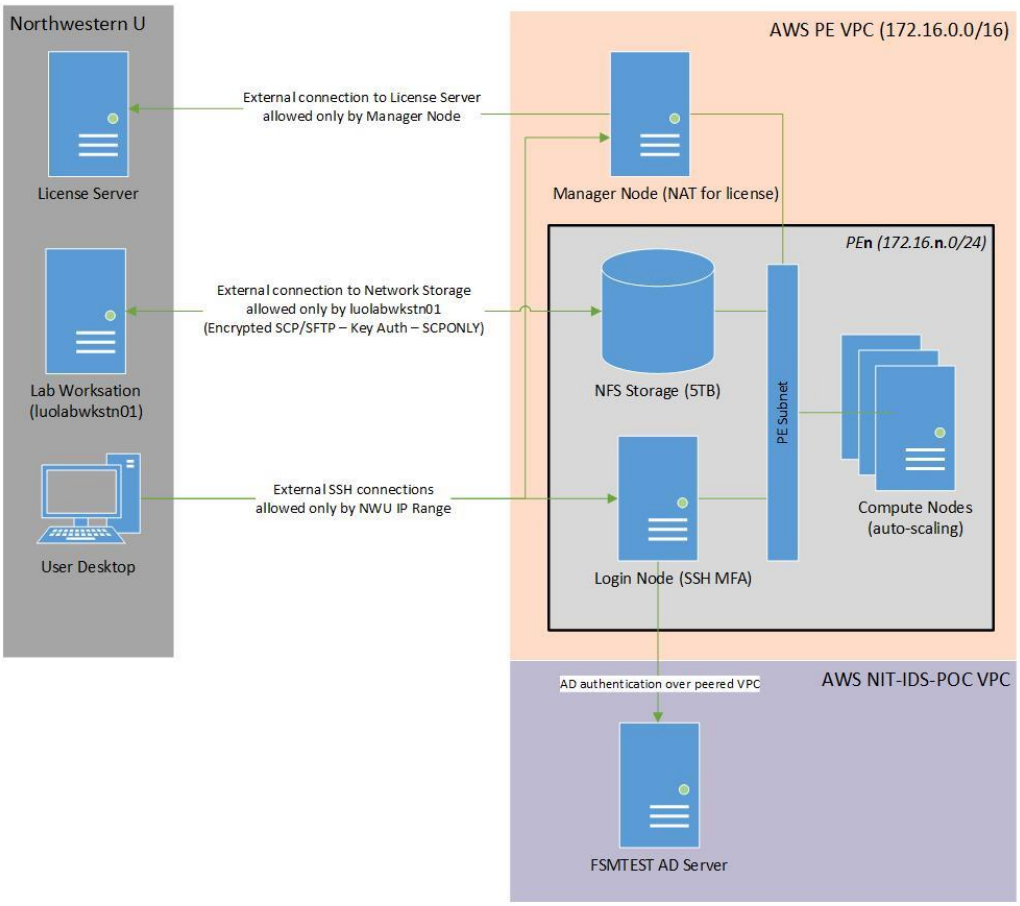
a. Glossary

- POC – Proof of concept.
- AWS – Amazon Web Services: Cloud provider.
- VPC – Virtual Private Cloud: Partitioning of cloud environment into a manageable space.
- PE – Protected Environment: Contained HIPAA compliant environment.
- AMI – Amazon Machine Image: Image saved on amazon and used for deployment.
- IAM – Identity and Access Management: Used to limit access to AWS resources.

b. Features and Design

Some of the features of the POC design are:

- AWS CfnCluster product used for deployment.
    - Multiple Clusters per PE, to support different instance types – CPU and GPU.
    - Auto-scaling of compute resources, both up & down, based on job queue times.
- Security
    - Multi-factor authentication, using Active Directory and DUO.
    - User authorization configuration per cluster, using Linux PAM modules.
    - Network access control, using AWS security groups.
    - Data encrypted at rest and in motion, using EBS encryption and OpenSSH.
        - SCP/SFTP from on-premise server to PE Storage, locked down to SCPONLY shell.
        - AMIs and configurations frozen and encrypted.
    - Auditing tools provided, including CloudWatch, CloudTrail, and CloudConfig.
- POC Specific Versions:
    - Operating System: Centos 7.2
    - Job Scheduler: OpenLAVA
    - Applications: R 3.3.1, Matlab R2016a, Python 2.7.6
        - Modules environment used to manage application versions.



c. User Workflow

This workflow assumes:

- The user has an account in FSMTEST AD Server and DUO service.
- HIPAA compliant LAB workstation at NU has SSH key access to PE Storage in AWS.
- Cluster has been deployed and relevant users added as local users on Login Node.

The following workflow is expected from users:

- Copy data from LAB workstation at NU to PE storage in AWS using SCP/SFTP (*SSHFS*).
- SSH to Login Node and pass multi-factor authentication steps.
- Set up batch job and submit to scheduler.
- Compute nodes will spin up/down automatically to match resource requirements.
- Copy data from PE storage in AWS to LAB workstation at NU.

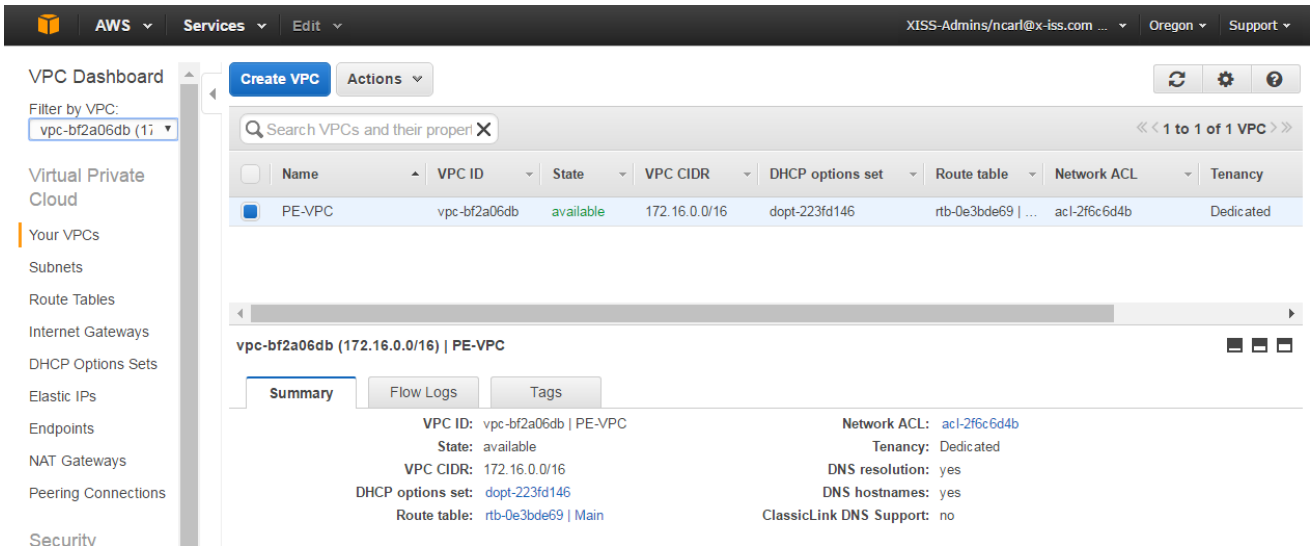
Assuming the above batch job workflow is standardized, it may be possible to automate job submission when data is fed to the PE storage node in AWS.

2. VPC Setup

a. VPC Settings

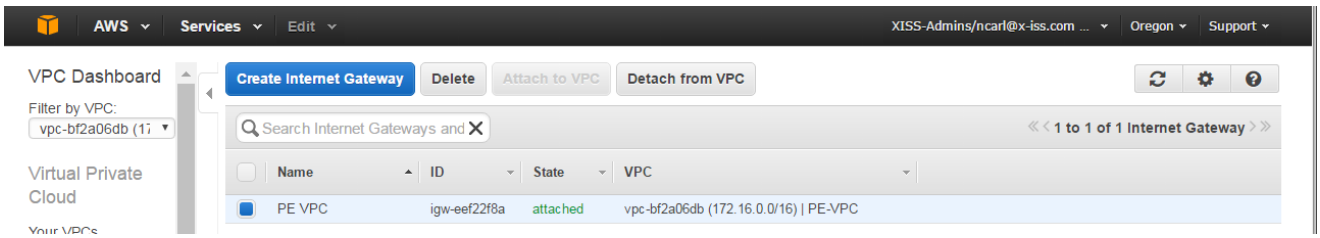
Create a VPC with the following settings:

- Default Tenancy.
  - Dedicated Tenancy is used for POC, but default is preferred for flexibility.
- DNS resolution/hostnames set to “yes”
- DHCP options are default generated set.
- VPC CIDR network range that can handle at least two subnets.
  - Size so that one of the subnets can scale to contain all cluster nodes.



b. Internet Gateway

Create an internet gateway and attach to the VPC created in the section 2.a. above.

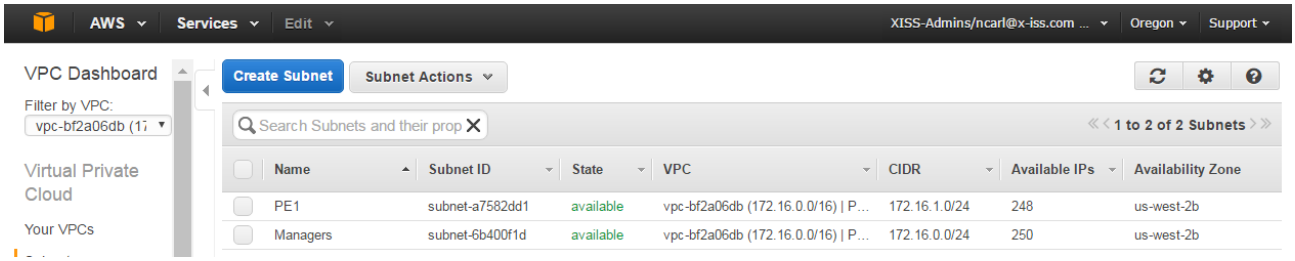


c. Subnets

Two subnets must be created to facilitate routing tables, for NAT, required for MATLAB licensing. It is safe for multiple Protected Environments to use the same PE subnet, as Security Groups will control access.

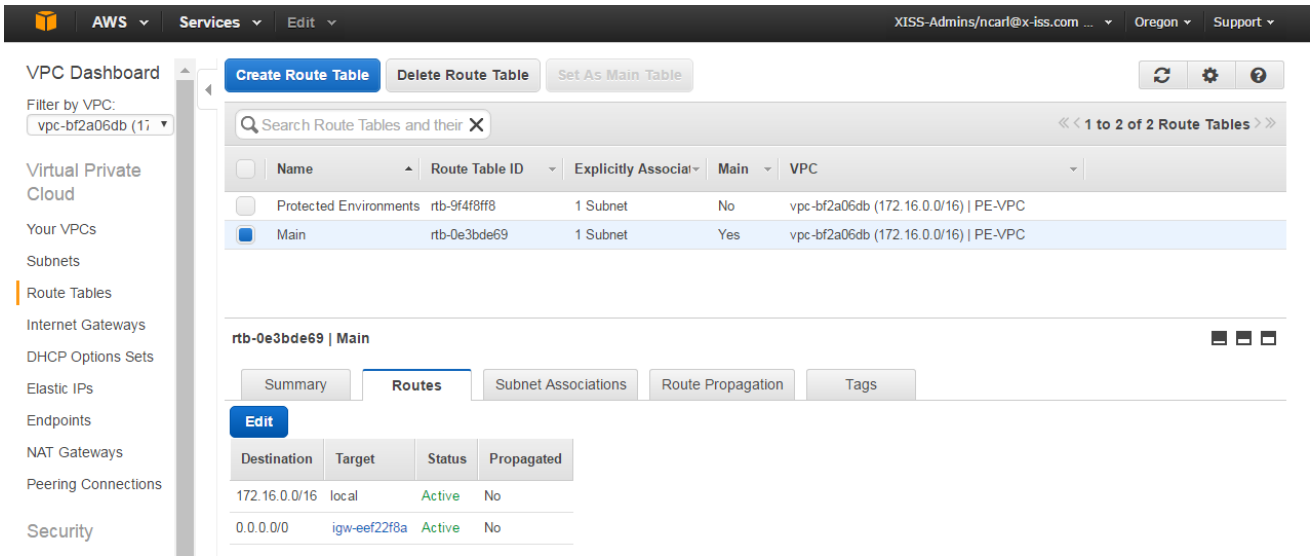
The subnets created for POC are designated for:

- Manager node, named “Managers”
- Protected environment, named “PE1” in POC



d. Routing Tables - 1

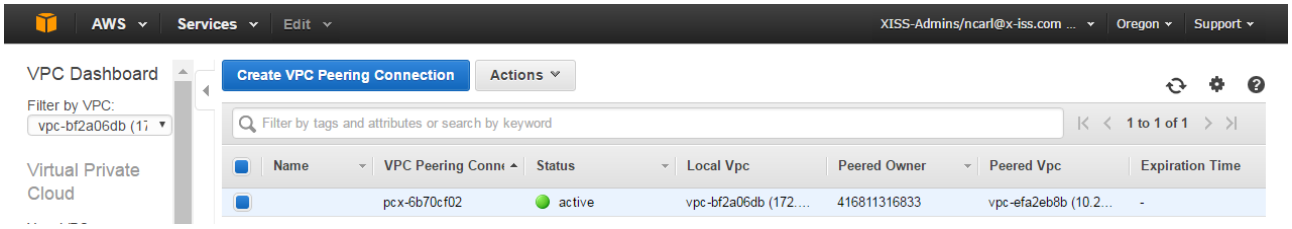
Set the Main routing table set to use internet gateway created in section 2.b. This routing table is associated with the Managers subnet.



e. VPC Peering

Configure the VPC with peering to the Northwestern shared services VPC, which contains the Active Directory server used for testing.

For the POC, we are peering with the Northwestern development shared services VPC to access the FSMTEST active directory server.



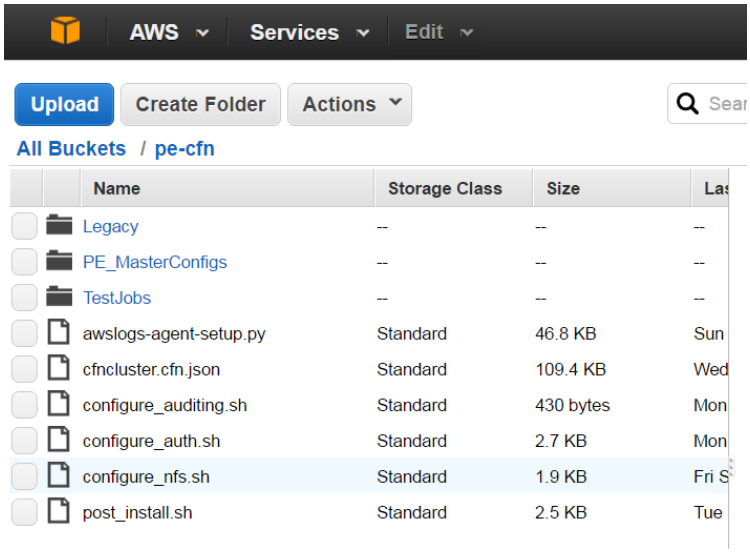
f. Roles

The following roles are required and can be used across all VPCs:

- **PE_Deploy** role with 'AdministratorAccess' policy attached, for CfnCluster to perform deployments and auto-scaling.
- **PE_Logs** role with 'CloudWatchLogsFullAccess' policy attached, which will get attached to all nodes that log to CloudWatch service.

g. S3

The "pe-cfn" S3 bucket is used for CFN Cluster configuration and custom deployment scripts. Permissions are set to allow Authenticated AWS Users.

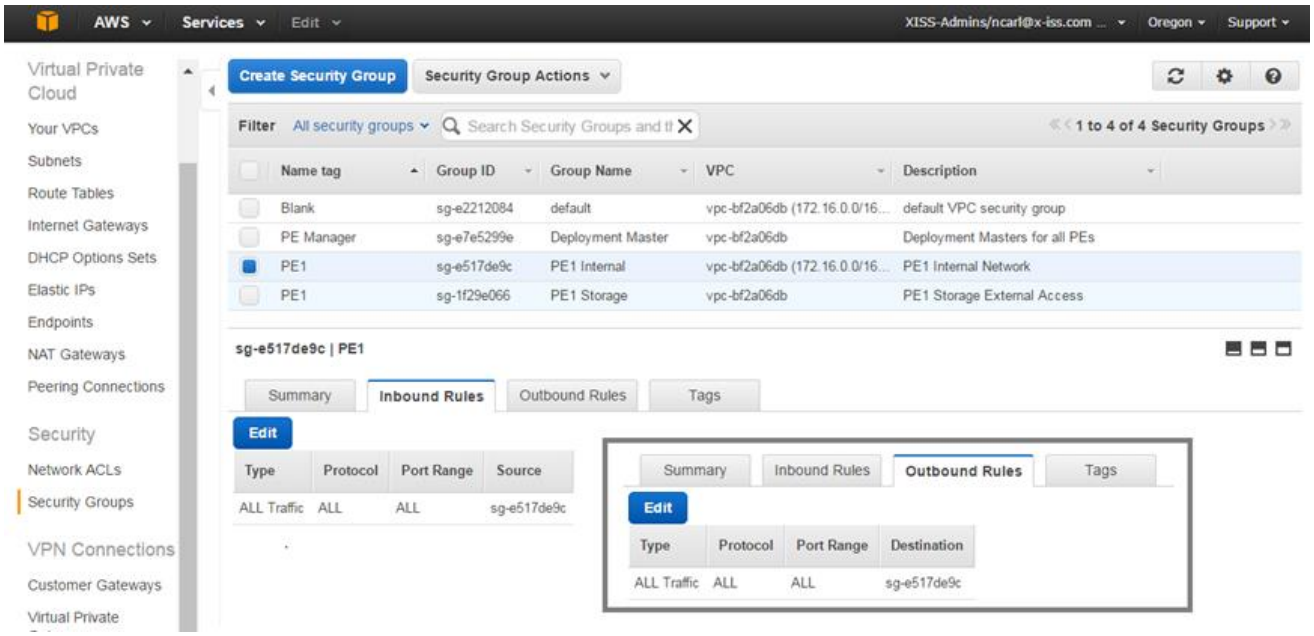


3. Protected Environment Setup

a. Security Groups

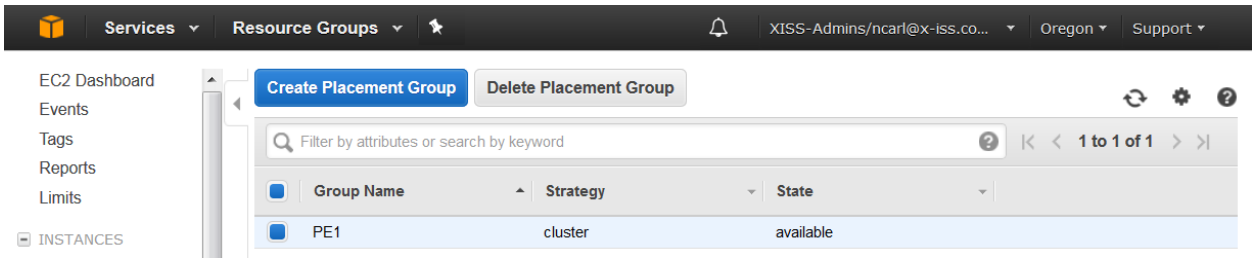
The following security groups are used. Recommend tagging each these security groups with the name of the protected environment.

- **PE Manager** rule allows all access outbound. Add SSH inbound rules for IP of any administrators that need access to deploy clusters.
- **PE Storage** rule allows all access outbound. Add SSH inbound rule for IP of HIPAA compliant servers where data is to be transferred from (LAB Workstation).
- **PE Internal** rule allows access between nodes that are a part of the same security group; required for PE Storage and PE Manager to interface with cluster nodes.



b. Placement Group

Create a placement group for your PE, which will allow for your clusters and PE Storage to take advantage of enhanced networking connectivity.



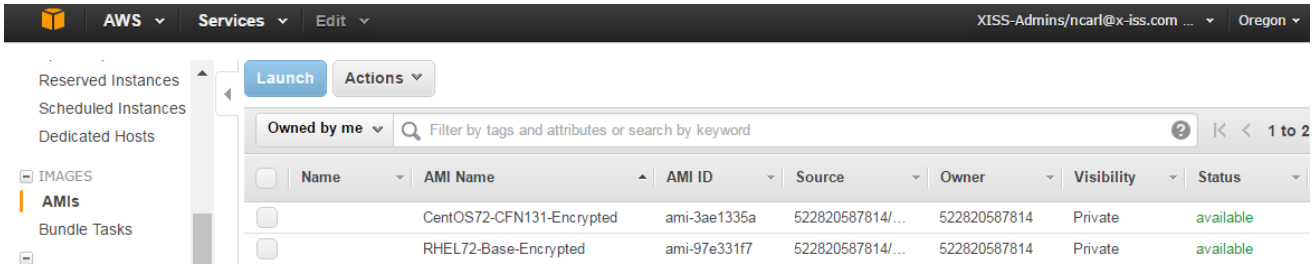
c. Elastic IPs

Allocate an Elastic IP for the PE Manager and PE Storage nodes. Login nodes will automatically have newly created Elastic IPs assigned during deployment.

d. Encrypted AMI

Instance root volumes are not encrypted by default, but must be encrypted for HIPAA compliance.

See section 5.d. “Updating and Encrypting AMIs” for more information on this step.

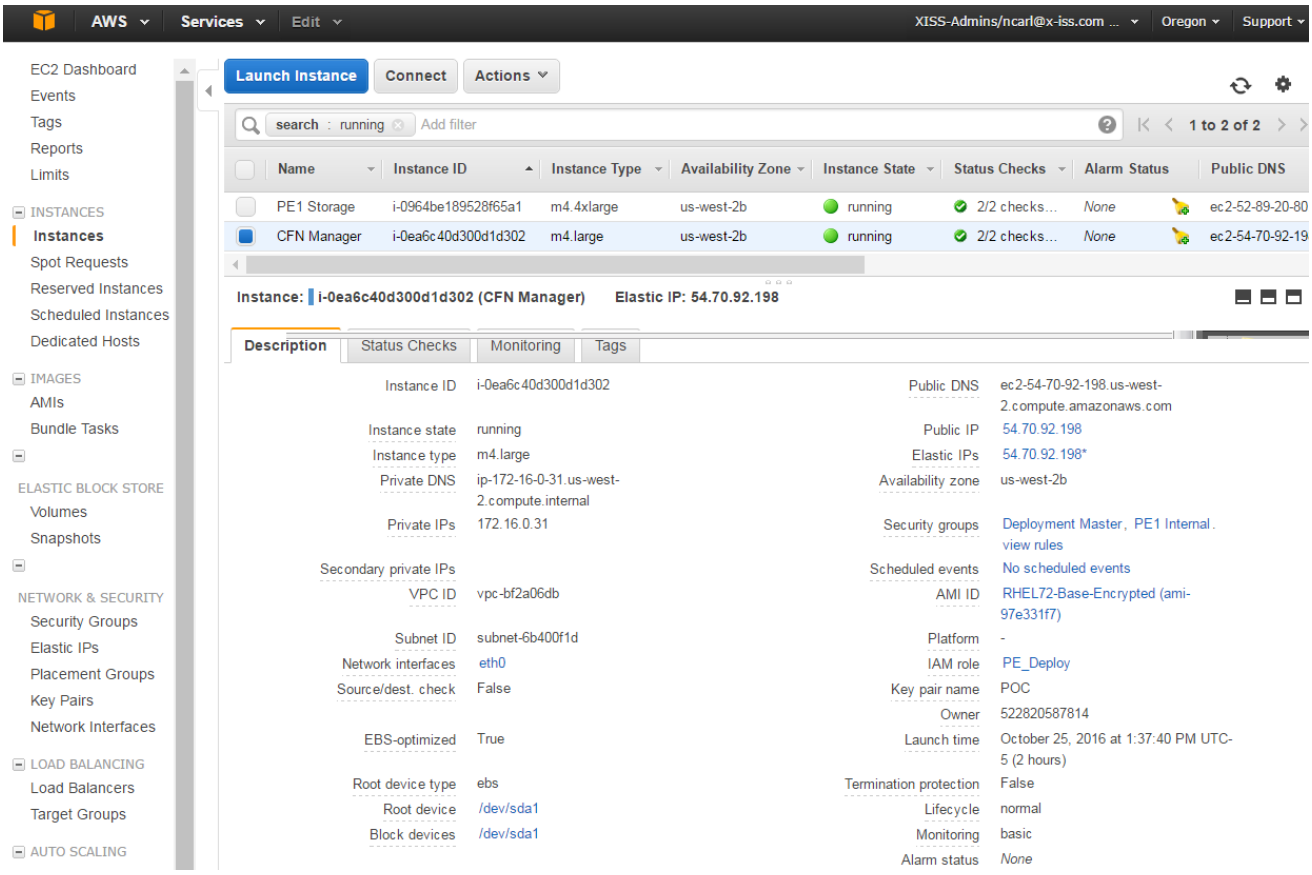


e. PE Manager – Instance Information

Create an instance for the Manager node with following settings.

- Instance Type: small instance type (t2.micro; m4.large if dedicated tenancy)
- Tenancy: default tenancy is sufficient since no PHI will go through this instance.
- VPC: is the one created in step 2.a.
- Subnet: Managers subnet created in step 2.c.
- IAM Role: PE Manager role created in step 2.f.
- Security Groups: PE Manager, PE Internal created in step 3.a.
- Placement Group: NOT required (nor recommended)
- Elastic IP: one of the two created in step 3.c.
- AMI: Encrypted RHEL AMI created in step 3.d.
- New SSH Key if not created in previous steps; save this key.

Below is the POC PE Manager node for example.



f. PE Manager – CfnCluster Setup

To install and configure CFN cluster on the PE Manager node, the following steps and associated commands are executed.

i. Base CfnCluster Install

```
sudo yum -y install python-setuptools.noarch wget yum
sudo easy_install cfncluster
```

ii. CfnCluster Configuration Files

```
mkdir .cfncluster #For cfncluster-cli.log
aws s3 cp s3://pe-cfn/Deploy_PE1/config.pe1.cpu . #Ex. Config
aws s3 cp s3://pe-cfn/Deploy_PE1/config.pe1.gpu . #Ex. Config
# Previously saved SSH key
aws s3 cp s3://pe-cfn/PE_MasterConfigs/POC.pem .ssh/POC.pem
chmod 600 .ssh/POC.pem
```

iii. Disable SELINUX and reboot

```
sudo sed -e 's/SELINUX=.*/SELINUX=disabled/' -i
/etc/selinux/config
shutdown -r now
```

iv. Configure Auditing

```
wget https://s3-us-west-2.amazonaws.com/pe-
cfn/configure_auditing.sh -P /tmp
sh /tmp/configure_auditing.sh PE1_Manager
```

g. PE Manager – NAT for licenses

To configure the PE Manager node to NAT license requests, the following steps and associated commands are executed. This may be set up in a server in Shared Services VPC, to simplify routing/subnet rules in Protected Environment VPC.

i. Disable source/dest checking on instance

http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_NAT_Instance.html#EIP_Disable_SrcDestCheck

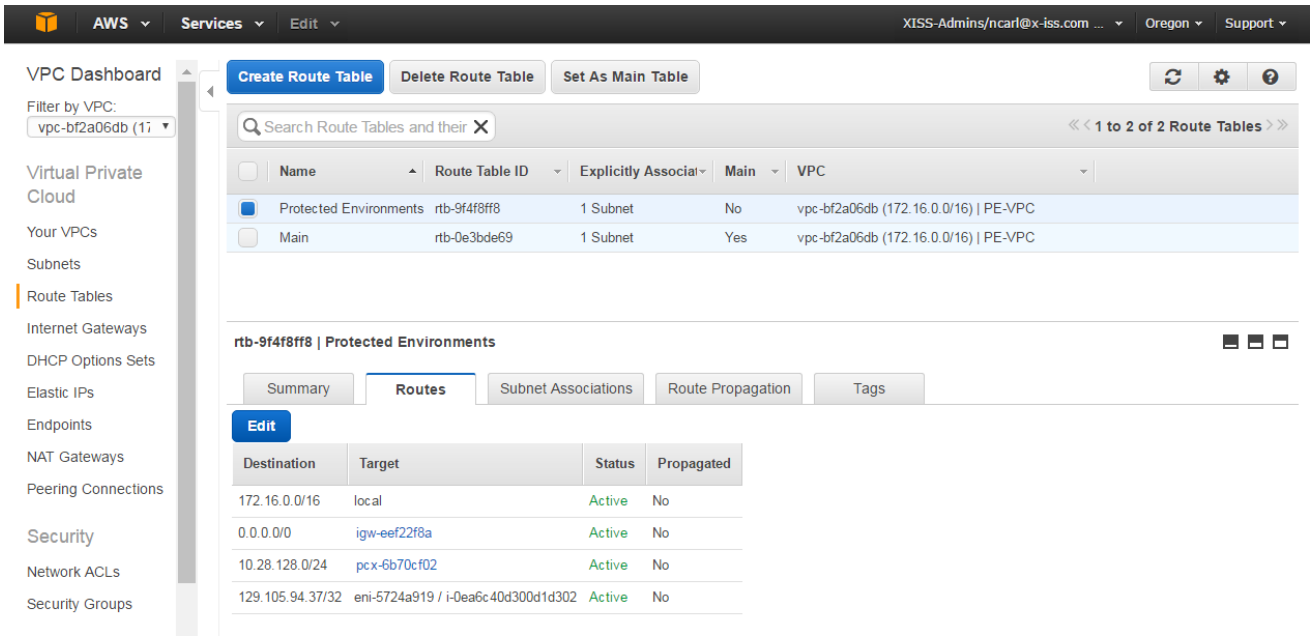
ii. Configure IPtables to perform NAT Masquerade

```
yum -y install iptables-services
systemctl enable iptables
echo 'net.ipv4.ip_forward = 1' >> /etc/sysctl.conf
sysctl -p /etc/sysctl.conf

cat << 'EOF' > /etc/sysconfig/iptables
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A POSTROUTING -o eth0 MASQUERADE
COMMIT
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
COMMIT
EOF

systemctl restart iptables
```

h. Routing Tables - 2

Set up the Protected Environment routing table to contain the following. It is safe to assign many PE subnets to the same routing table, as the security groups will control access.

- Peering connection to shared VPC (pcx)
- Interface of PE Master for license NAT (eni)

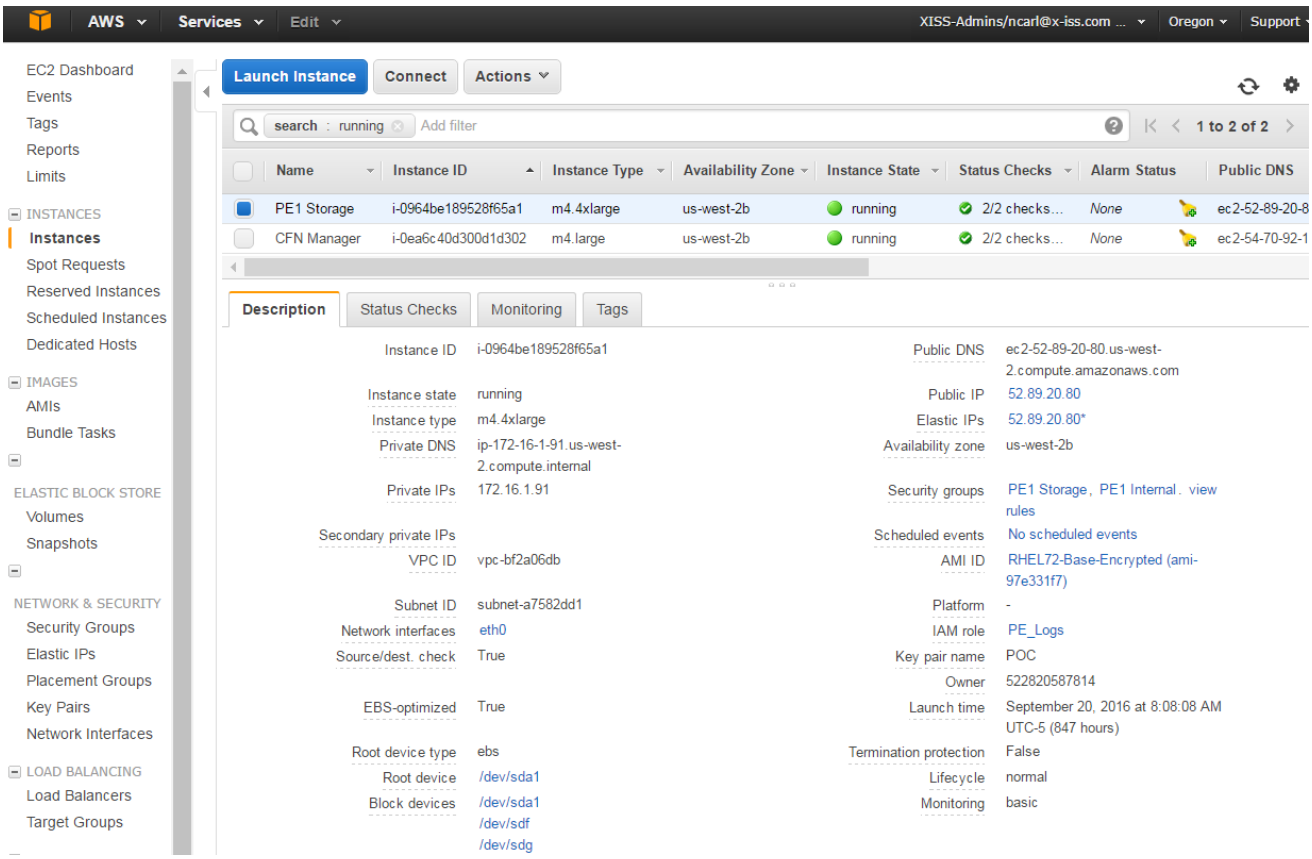


i. PE Storage – Instance Information

Create an instance for the PE Storage node with following settings.

- Instance Type: larger instance type to handle NFS load (m4.4xlarge)
- Tenancy: dedicated tenancy is required if PHI will go through this instance.
- VPC: is the one created in step 2.a.
- Subnet: either of the subnets created in step 2.c.
- IAM Role: PE Logs role created in step 2.f.
- Security Groups: PE Storage, PE Internal created in step 3.a.
- Placement Group: is recommended; use one created in step 3.b.
- Elastic IP: the remaining of the two created in step 3.c.
- AMI: Encrypted RHEL AMI created in step 3.d.
- SSH Key created in previous steps; save this key.
- Create a couple EBS volumes (must be encrypted)
    - One for apps (~20+GB)
    - One for data (as big as requested)

Below is the POC PE Manager node for example.



j. PE Storage – NFS Server Setup

To configure the PE Storage node, the following steps and associated commands are executed.

i. Configure Filesystems

```
mkfs.xfs /dev/xvdf
mkfs.xfs /dev/xvdg
echo "/dev/xvdf /apps xfs defaults 0 0" >> /etc/fstab
echo "/dev/xvdg /project xfs defaults 0 0" >> /etc/fstab
mkdir /apps /project
mount -a
```

ii. Install and Configure NFS

```
yum -y install nfs-utils
systemctl enable rpcbind
systemctl enable nfs-server
sed -e 's/RPCNFSDARGS=.*/RPCNFSDARGS="-N 4"/' \
    -e 's/#RPCNFSDCOUNT=.*/RPCNFSDCOUNT=32/' \
    -i /etc/sysconfig/nfs
echo "/apps 172.16.1.0/24(rw,async,no_subtree_check)" >>
/etc/exports
```

```
echo "/project 172.16.1.0/24(rw,async,no_subtree_check)" >>
/etc/exports
sed -e 's/SELINUX=.*/SELINUX=disabled/' -i /etc/selinux/config
shutdown -r now
```

iii. Configure Authentication for Data Transfer (SCPONLY)

```
sed -e 's/^# Cipher.*/Ciphers aes128-ctr,aes192-ctr,aes256-
ctr/' \
   -e 's/^#PermitRootLogin.*/PermitRootLogin no/' \
   -i /etc/ssh/sshd_config

wget https://s3-us-west-2.amazonaws.com/pe-
cfn/Common/scponly.tgz -P /tmp
sudo tar -zxvf /tmp/scponly.tgz -C /usr/local
echo "/usr/local/bin/scponly" >> /etc/shells

useradd -s /usr/local/sbin/scponlyc -u 9999 project
chmod 775 /project
chown root:project /project
```

iv. SSH Key

Have a key generated on LAB workstation and copy into /home/project/.ssh/authorized_keys on PE storage node.

```
chown -R project. /home/project
chmod 700 /home/project/.ssh
chmod 600 /home/project/.ssh/authorized_keys
```

v. Configure Auditing

```
wget https://s3-us-west-2.amazonaws.com/pe-
cfn/configure_auditing.sh -P /tmp
sh /tmp/configure_auditing.sh PE1_secure_logs
```

4. CfnCluster Setup

a. Configuration Files

Some overview information about CfnCluster configuration files.

- CFN Cluster uses a configuration file located at ~/.cfncluster/config by default, but we're not using it. This is to force us to define a configuration when creating a cluster.
- You do not need to specify AWS credentials because of the PE_Deploy role assigned to PE Manager.
- More in-depth explanation of configuration file is located here: http://cfncluster.readthedocs.io/en/latest/configuration.html

b. Example: config.pe_cpu

The following is an example CfnCluster configuration file used in the POC.

- These first sections are at the beginning of the file, and are where you define your region and some global config.

```
[aws]
aws_region_name = us-west-2

[global]
update_check = true
sanity_check = true
cluster_template = PE1CPU
```

- *Note:* ***References*** *to other sections of configuration must come* ***before*** *the sections to which they are referring. For example,* `cluster_template = PE1CPU` above refers to `[cluster PE1CPU]` further down the configuration file.

- These next sections outline the specific cluster configuration. It includes instance and scheduler specifications.

```
####### Cluster #######
[cluster PE1CPU]
cluster_type = ondemand
tenancy = dedicated
key_name = POC

base_os = centos7
custom_ami = ami-3ae1335a
ebs_settings = custom
shared_dir = /cluster
template_url = https://pe-
cfn.s3.amazonaws.com/Common/cfncluster.cfn.json
post_install = https://pe-
cfn.s3.amazonaws.com/Deploy_PE1/post_install.sh

scheduler = openlava
initial_queue_size = 1
max_queue_size = 5
maintain_initial_size = true
scaling_settings = custom

# This is the only other difference between PEs
vpc_settings = PE1
placement_group = PE1
```

```
# This is only difference between clusters in the same PE
master_instance_type = c4.4xlarge
compute_instance_type = c4.4xlarge
```

- These sections are at the end of the file, and are more specific configurations referenced from the cluster config above.

```
####### General #######
[vpc PE1]
vpc_id = vpc-bf2a06db
master_subnet_id = subnet-a7582dd1
additional_sg = sg-e517de9c
ssh_from = 165.124.223.0/24

[ebs custom]
volume_size = 10
encrypted = true

[scaling custom]
scaling_cooldown = 600
scaling_period = 60
scaling_evaluation_periods = 2
scaling_threshold = 1
scaling_adjustment = 1
scaling_threshold2 = 60
scaling_adjustment2 = 10
```

c. Multiple Cluster Configurations

The easiest way to manage multiple configurations is with multiple configuration files.

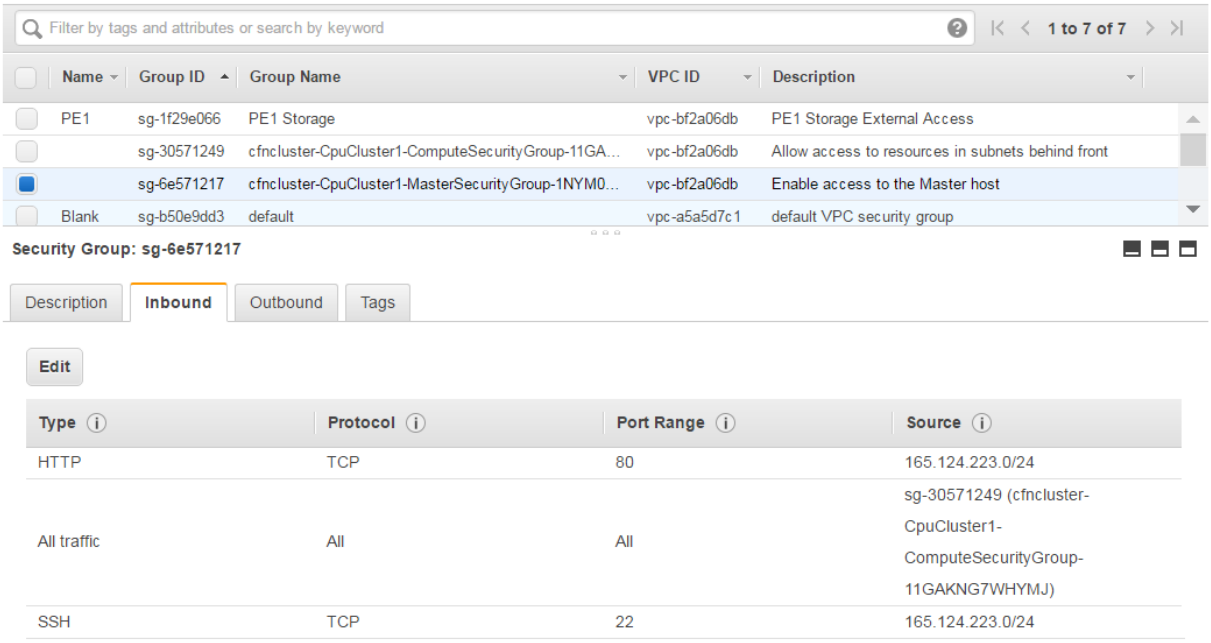
The following is an example of differences between CPU and GPU cluster.

```
[ec2-user@ip-172-16-0-31 ~]$ diff config.pe1.cpu
config.pe1.gpu
7c7
< cluster_template = PE1CPU
---
> cluster_template = PE1GPU
10c10
< [cluster PE1CPU]
---
> [cluster PE1GPU]
35c35
< compute_instance_type = c4.4xlarge
---
> compute_instance_type = g2.2xlarge
```

d. Dynamic Security Groups

The following security groups get created by CfnCluster on deployment.

- Both “Compute” and “Master” group rules allow all access outbound.
- “Compute” group rules allow access between compute nodes.
- “Master” group allows access from compute nodes.
- Additional “Master” group rules come from the following config definition:
  `ssh_from = 165.124.223.0/24`
- Modify the “Master” group to change external access to login node after deployment.



e. Post Script Configurations

The post_install.sh script runs on every instance deployment and configures:

- NFS shares from storage server (all nodes).
- Authentication
  - Duo MFA (Login node only)
  - AD Kerberos setup (Login node only)
  - SSH Lockdown (Login node only)
  - Project share group (Login node only)
  - PAM files synchronization (all nodes)
- Auditing
  - Cloudwatch agent to collect Linux secure logs (Login only)

5. Day to day administration

a. CfnCluster Deployment

To deploy a cluster, run creation against configuration file.

```
cfncluster -c config.pe1.cpu create ClusterName
```

- CFN Cluster will set up additional roles and all required services for the cluster to deploy and auto-scale with queued jobs.
- The post_install.sh script is run ON EVERY NODE
    - This includes both the login and compute nodes.
    - Logic is built into the script to determine node type, and control what functions run on desired nodes.

To remove a cluster, run the following command. Note: This will not remove PE Storage or Manager.

```
cfncluster delete ClusterName
```

b. Starting and stopping instances

Instance dependencies:

- PE Storage instance is required for any clusters.
- PE Manager instance is required only when running 'cfncluster' commands create or delete. Clusters become self-sufficient for auto-scaling once deployed.

Stopping instances:

- SSH to Manager node and run: `cfncluster stop ClusterName`
- Log into AWS and navigate the EC2 Instances page.
- Select the Manager and Storage nodes.
- At the top of the page, click Actions → Instance State → Stop

Starting instances:

- Log into AWS and navigate the EC2 Instances page.
- Select the Manager and Storage nodes.
- At the top of the page, click Actions → Instance State → Start
- Wait for instances to fully start up.
- SSH to Manager node and run: `cfncluster start ClusterName`

The main reason to 'stop/start' a cluster vs. 'delete/create' is to keep the same ElasticIP.

c. Adding Users

The following steps are run on the cluster login node after it has been created by CfnCluster.

- Create user to match the Active Directory user and place that user in the project group to provide share access: `useradd -G project <username>`
- You do not need to define password, as only AD/DUO will be used for authentication.
- The user will be synced to compute nodes within 2 minutes.

d. Updating and Encrypting AMIs

The following steps are required for updating AMIs that are used to process/store PHI.

- Deploy an instance using desired AMI
- Log in with key and perform required updates
- Create Image (AMI) of Instance; this version is unencrypted
- Select and copy newly created AMI, and select the encryption option
- Remove the unencrypted version of this new AMI
- Update CFN Cluster configuration to use new encrypted AMI

e. Installing Applications

The following conventions should be followed for application installs in AWS PE.

- Installs can be done on the login node for the PE.
- Applications should be installed into the /apps.
- Module environment files for the applications should be placed into /apps/modulefiles.
- Since /apps should be exported from the PE Storage node, they persist and are shared across clusters in the same PE.

f. Post Script Modifications

Instance modifications can be made by downloading the relevant post_install.sh script, making changes, and uploading back to s3.

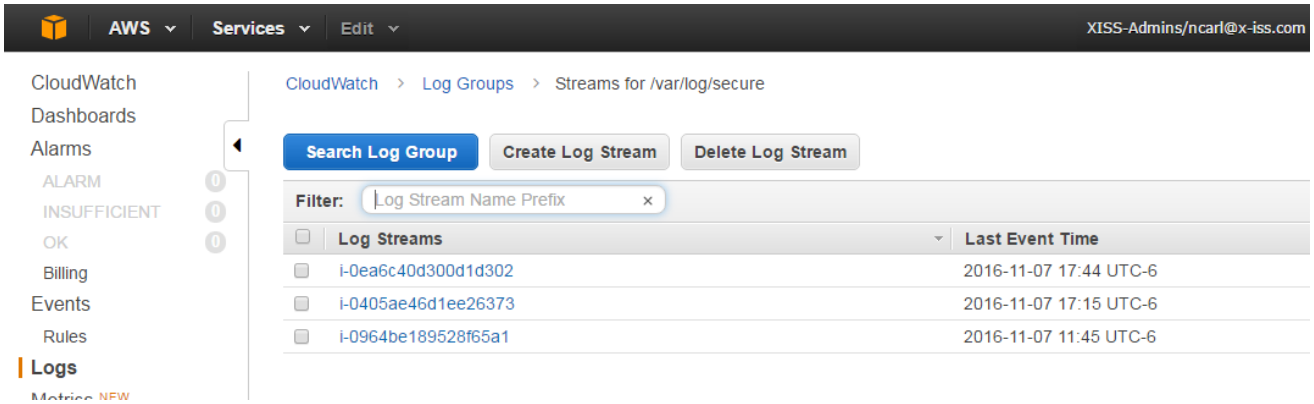
```
$ aws s3 cp s3://pe-cfn/Deploy_PE1/post_install.sh .
$ vim post_install.sh   # Make modifications
$ aws s3 cp --acl public-read post_install.sh s3://pe-
cfn/Deploy_PE/post_install.sh
```

g. CloudWatch

CloudWatch is where /var/log/secure is collected from external facing servers for auditing.

To view logs, navigate to AWS “CloudWatch” service.

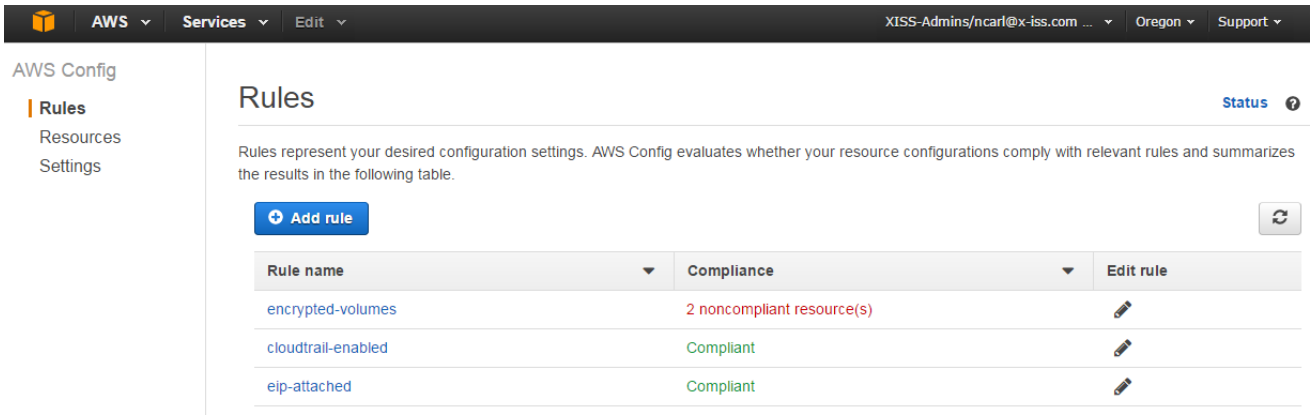
- Click “Logs” on the left pane.
- Click the log group defined for the PE.
- A stream is created for every login node and will persist until explicitly deleted.



h. CloudConfig

AWS Config is used to check for compliance on items like drive encryption. To use:

- Navigate to the AWS “Config” service.
- Look for labels showing noncompliance.



- Click on the relevant rule name to investigate further.


Click on the icon to view configuration details for the resource when it was last evaluated with this rule.

Resource type	Resource identifier	Compliance	Last successful invocation	Last successful evaluation	Config timeline
EC2 Volume	vol-09080388e2fc5adc9	Noncompliant	September 16, 2016 5:03:24 PM	September 16, 2016 5:03:24 PM	
EC2 Volume	vol-0fb649acd0488d6b3	Noncompliant	September 16, 2016 5:03:23 PM	September 16, 2016 5:03:24 PM	
EC2 Volume	vol-00816ade9cdec13de	Compliant	September 16, 2016 5:03:23 PM	September 16, 2016 5:03:24 PM	

6. New Environments

a. New Protected Environment Setup

When setting up a new PE, the following steps are optional:

- Creating a new Subnet, if current subnet is large enough.
- Creating a new encrypted AMI.
- Creating a new PE Manager node.

To set up a new PE, follow sections 3 and 4 of this document. Yellow highlights are shown where configuration will most likely need modification.

Configuration should be copied from previous examples and changed to fit the new PE. You can copy the post_install.sh from the other PE with the following command:

```
$ aws s3 cp s3://pe-cfn/Deploy_PE1/post_install.sh .
$ aws s3 cp --acl public-read post_install.sh s3://pe-
cfn/Deploy_NewPE/post_install.sh
```

b. New VPC Setup

When setting up a new VPC, the following steps are optional:

- Creating IAM Roles
- Creating a new S3 bucket for configurations.

To set up a new VPC, follow section 2 of this document, then continue to follow section 6.a. to set up a new protected environment.

If creating a new S3 bucket, copy the “Common” folder to the new S3 bucket.

7. Known Issues/Caveats

a. Placement Groups

Placement groups are less stable when a mix of instances types, or more common instance types, are deployed into them.

At times when deploying clusters into a placement group, our deployments failed. To investigate why, we navigated to AWS CloudFormation service page and discovered the following “Insufficient Capacity” message.

2016-11-09	Status	Type	Logical ID	Status reason
▸ 17:36:11 UTC-0600	ROLLBACK_IN_PROGRESS	AWS::CloudFormation::Stack	cfncluster-CpuCluster1	The following resource(s) failed to create: [MasterServer]. . Rollback requested by user.
▾ 17:36:10 UTC-0600	CREATE_FAILED	AWS::EC2::Instance	MasterServer	Instance i-00702d1af77337e99 failed to stabilize. Current state: terminated. Reason: Server.InsufficientInstanceCapacity: Insufficient capacity to satisfy instance request

At the time, we were using smaller ‘c4.large’ instance type for the login nodes. Amazon recommended we use the same instance type for login nodes as we were using for CPU compute nodes: ‘c4.4xlarge’. This keeps the placement group as a near homogeneous environment.

Using a ‘c4.4xlarge’ does seem to be overkill for login purposes. This leaves some room for user testing on that node. It may be desirable to open eight cores on this node to job submission, to delay the need for scaling up additional compute nodes for small workloads.

b. Job Schedulers

Some job schedulers are not currently optimized for dynamic environments, such as the one described in this document.

OpenLAVA and Torque both have been tested to have issues when scaling. Symptoms show when the compute node has finished deploying, but the login node sees it as ‘unavail’ or ‘offline’. In both of these cases, a simple restart of the compute node scheduler daemons fixed the issue. Other schedulers may have issues but have not been tested through CfnCluster deployments.

To resolve these issues with schedulers in auto-scaling in CfnCluster, the ‘fix_scheduler’ function and script was added to the post_install.sh script. This installs as a cron entry that runs a health check on the scheduler and restarts if there is an issue with scaling. This function will need to be updated with the correct scheduler type, to match CfnCluster config definition.

c. S3 Security Rules

When modifying S3 security rules to lock down buckets, we recommend creating a new bucket and test the security rules on that bucket first. This will show you if your rules lock you out completely, since doing so on your main bucket will put the PE out of order until someone with a root AWS account can remove the rules.

8. Recommendations before going live

a. Access

Access needs to be set up for administrators:

- AWS portal access to cloud POC resources
- Key or local user, with sudo access on PE manager and clusters login nodes.

Currently users get added manually after cluster deployment. It is possible to set up a post_install script to add users based on a list stored in S3 for the cluster.

b. Licenses

Matlab uses ACLs to allow only certain IP addresses to check out a license from the server. Setting up a license server or proxy (NAT) in the shared services VPC would allow use across multiple VPCs and PEs and simplify architecture by removing the NAT requirement inside the VPC, which also removes the requirement of multiple subnets.

Before going to production, the above conciderations should be made and access audited to make sure ACLs are up to date.

c. Benchmarks

Users should collect a few different cases of jobs, that have known run times in the current environment, and run the jobs in the cloud POC environment to determine the performance differences between the two solutions.

d. Workflow Optimization

Once users have tried the system and understand the architecture, we recommend looking at the workflows and determining if parts of it could be automated. If the workflow is standardized and only the inputs change, it could be possible to have jobs automatically submitted based on certain criteria like new input files being put in the cloud storage. These types of changes could simplify user workflows and even remove the requirement of having users log into the cloud systems directly.