Cloudfish Classroom CFNCluster

https://cfncluster.readthedocs.io/en/latest/index.html

Configuration

File: ./cfncluster/config

```
[cluster Cloudfish-Classroom]
vpc_settings = US_EAST_HPC
key_name = CloudFish
master_instance_type = t2.micro
compute_instance_type = t2.micro
master_root_volume_size = 100
compute_root_volume_size = 15
initial_queue_size = 1
max_queue_size = 8
maintain_initial_size = false
cluster_type = ondemand
base_os = centos7
scheduler = slurm
#placement_group = DYNAMIC
#placement = cluster
tags = {"Environment": "HPC Demo","Application":
"Cloudfish-Classroom","Stack": "Cloudfish-Classroom"}
shared_dir = /modeling
ebs_settings = modeling_classroom_blank
s3_read_write_resource = arn:aws:s3:::cloudfish-classroom.chesapeakebay.net
post_install = s3://bluefish-scripts/compute-post-install-classroom.sh

[ebs modeling_classroom_blank]
volume_type = gp2
volume_size = 100

[global]
update_check = true
sanity_check = true
cluster_template = Cloudfish-Classroom

[aliases]
ssh = ssh {CFN_USER}@{MASTER_IP} {ARGS}
```

Create a new cluster

```
cfncluster create Cloudfish-Classroom
```

Host Customization

Cluster Customizer Script

```
#!/bin/bash
## ==================== ##
## John Massey          ##
## 2/24/2017            ##
## Attain               ##
## Post Start-Up Script ##
##  for Compute Nodes   ##
## ==================== ##

# Step 1 Make sure CSH works correctly with Symlinks
echo "set symlinks = expand" > /etc/profile.d/symlinks.csh
chown 644 /etc/profile.d/symlinks.csh

# Step 2, configure cron job to clear sys memory cache
# Removed for testing (may not be necessary)
#echo -e "*/30 * * * * root sync; echo 3 > /proc/sys/vm/drop_caches\n" >
/etc/cron.d/drop-caches

# Step 3 Install Pre-Requesit software for WQSTM Model
yum -y install readline-devel hdf5 netcdf udunits2 nco libcurl-devel
netcdf-devel \
antlr hdf5-openmpi hdf5-openmpi-devel dos2unix libXfixes-devel
libXdamage-devel \
libdrm-devel libXxf86vm-devel mesa-libGL-devel mesa-libGLU-devel
compat-libstdc++ \
ghostscript ilmbase OpenEXR-libs ghostscript-devel libtiff-devel librsvg2
jasper-devel \
libwmf-lite ImageMagick ImageMagick-devel ImageMagick-c++
ImageMagick-c++-devel libtool-ltdl emacs nano

# Step 4 Configure ModuleFiles for Scripts, Modules stored in NFS Shared
High Speed Disk.
#ln -s /modeling/tools/pgi/modulefiles /etc/modulefiles/pgi
#ln -s /modeling/tools/modulefiles/netcdf /etc/modulefiles/netcdf
#ln -s /modeling/tools/modulefiles/intel /etc/modulefiles/intel
#ln -s /modeling/tools/modulefiles/hdf5 /etc/modulefiles/hdf5
#ln -s /modeling/tools/modulefiles/gri /etc/modulefiles/gri
#ln -s /modeling/tools/modulefiles/intel-mpi/intel-mpi_2017.2.174
/etc/modulefiles/mpi/intel-mpi_2017.2.174

# Step 5 Configure users and groups for Munge/Slurm to function properly.
groupadd modeling -g 1002

for i in student{1..20}; do
    useradd -m -p $(openssl passwd -1 $i"123#") -s /bin/bash -G modeling $i
    echo "user $i added successfully!"
```

```
done

for i in presenter{1..2}; do
    useradd -m -p $(openssl passwd -1 $i"123#") -s /bin/bash -G modeling $i
    echo "user $i added successfully!"
done

for i in ta{1..5}; do
    useradd -m -p $(openssl passwd -1 $i"123#") -s /bin/bash -G modeling $i
    echo "user $i added successfully!"
done

# Step 6 Configure /etc/security/limits.conf for mpi processing
aws s3 cp s3://bluefish-scripts/security-limits.conf
/etc/security/limits.conf
chown root:root /etc/security/limits.conf
chmod 644 /etc/security/limits.conf

#EOF
```