Greenplum + Ambari on AWS in Minutes.

AWS Configuration

1. Create four or more d2.8xlarge instances for Greenplum. Use CentOS 6 (x86_64) - with Updates HVM

   Put these instances into their own VPC and subnet, configure the instances to be assigned public IPs (these can be removed after installation), and add them to a placement group. Next set the hard disk space to something more reasonable, and give these instances a name.

   A security group will need to be defined for these instances. All traffic between instances in the security group should be allowed. You may also want to add SSH, which can be removed after installation is complete.

   Review the instance settings, make sure to either create or use a key you have, and launch the instances.

2. Create a t2.medium instance for Ambari. Use CentOS 6 (x86_64) - with Updates HVM

   Add this instance to the same VPC and subnet, configure the instance to be assigned a public IP, set the hard disk space, and name the instance.

   Add this instance to the security group created in the previous step, review the settings, and launch the instance. The Ambari server should also allow connections on port 8080/tcp, so we'll want to define a second security group which allows port 8080/tcp and SSH from your IP. Apply this security group along with the first one to the Ambari host.

Configure Hosts

1. First copy the key used during the creation of the EC2 instances to the Ambari server and the server which will be the mdw host.

2. (On Ambari host as root) Download the ambari-server-bootstrap.sh script and run it.

660 4TH ST. STE. 176 | SAN FRANCISCO, CA 94107 | T. 415.890.5764 | AMBARI@ZDATAINC.COM | AMBARI.ZDATAINC.COM

The script does the following:

- Installs wget, openssl, git, ntp, createrepo, and httpd Configures Ambari repository
- Installs Ambari through yum
- Installs the zData Ambari stack
- Disables iptables
- Disables transparent_hugepage Sets scheduler to deadline
- Disables SElinux
- Configures ntpd
- Creates a local repository
- Downloads two other scripts for configuration

3. (On Ambari host as root) In the empty file named hosts, created by the script in the previous step, add each Greenplum instance's IP followed by its FQDN, and a short hostname. One instance per line.

   It should look similar to:

- 10.0.0.9 ambari.aws ambari
- 10.0.0.10 mdw.aws mdw
- 10.0.0.11 sdw1.aws sdw1
- 10.0.0.12 sdw2.aws sdw2
- 10.0.0.13 sdw3.aws sdw3

4. (On Ambari host as root) Now run ./gpdb-cluster-bootstrap.sh {your key file}.

   This script will loop through each host in the file hosts, SSH into each instance using the instance's IP and the key file given as a parameter, and perform the following:

- Copies the file host to the instance
- Copies the key file to the instance
- Copies the gpdb-host-bootstrap.sh script to the instance Adds all the specified hosts to the machine's hostfile Sets the hostname of the machine to the given FQDN
- Finally it runs the gpdb-host-bootstrap.sh file, and waits for all hosts to be configured.
- The gpdb-host-bootstrap.sh script does the following:
- Installs mdadm, unzip, ntp, opensll, wget, and xfsprogs Configures scheduler to deadline
- Disables transparent_hugepage
- Disables SELinux
- Removes memory limits
- Allow root login, and password login (these are necessary for installation, but can be disabled after).
- Disables iptables
- Configures NTP
- Configures two raid devices, each containing half the ephemeral storage disks, the raids /dev/md0 and /dev/md1 are mounted at /data1 and /data2, respectively.
- These raids are added to the fstab with specific attributes (notably noatime, inode64, and allocsize=16m are set)
- The deadline scheduler is set for these raids.

Setting up Ambari

1. (On Ambari host as root) Run ambari-server setup

   The defaults should work fine.
   We recommend changing the database password to something more complex though. To do this enter the advanced database configuration section, use defaults except for password.

2. After Ambari has been configured, you should be able to connect to the Ambari server on port 8080 in your web browser and see the login page for Ambari.

Configuring Greenplum

1. Login to the Ambari web interface using the default admin credentials: admin / admin.
2. Start the cluster creation wizard by clicking 'Launch Install Wizard'.
3. Name the cluster.
4. Select the stack to use for the cluster. For this example you'll want to use the zData stack.

   Toggle down the 'Advanced Repository Options' and set the Local-PHD- Repo to http://ambari.aws/phd. The local repository you're referencing here was created by the ambari-server-bootstrap.sh script.
5. Now enter a list of all the hosts' FQDNs. They should look like:

   ambari.aws
   mdw.aws
   sdw[1-3].aws

   Use the private key downloaded from AWS during instance creation.

6. Select the services you wish to install. For this demo we recommend just selecting Greenplum.
7. Configure the Greenplum master and standby servers. The master will go on mdw.aws, and the standby will go on sdw1.aws. Normally the standby would have its own server, but we'll put it on a segment for this demonstration.
8. Set the sdw[1-3] hosts as Greenplum segments, and add the Client tools to all but the Ambari host.
9. Now you must configure your Greenplum installation, select the Greenplum tab if it isn't already.

In Advanced greenplum-env

- Make sure to first accept the Greenplum license agreement, available to read on their website.
- Verify the data_directory, make sure the directories referenced are correct.
- Configure the path to the previously uploaded Greenplum installation archive.
- Make sure to move the master_data_directory to one of the raid devices. The path should be something like /data1/master.
- Set the segments_per_host, for a d2.8xlarge we recommend 16.

660 4TH ST. STE. 176 | SAN FRANCISCO, CA 94107 | T. 415.890.5764 | AMBARI@ZDATAINC.COM | AMBARI.ZDATAINC.COM

In Advanced greenplum-mirroring

- Set `enable_mirroring` to true.
- Verify the `mirror_data_directory` value is correct.

1. Finally review your configurations, and deploy Greenplum.

Testing Greenplum

Now we are ready to test out Greenplum! First verify Greenplum is running in the Ambari web interface. If it is not, you can start it selecting 'Service Actions' in the top right, and Start. Go ahead and SSH into mdw.aws (or into ambari.aws, and then into mdw.aws), login as gpadmin, and run gpstate. If everything looks good, you should now have a fully functioning Greenplum cluster running entirely on AWS!

At this point you can remove the public IPs assigned to the four Greenplum instances, as well as the security group rule allowing direct SSH connections to Greenplum instance.