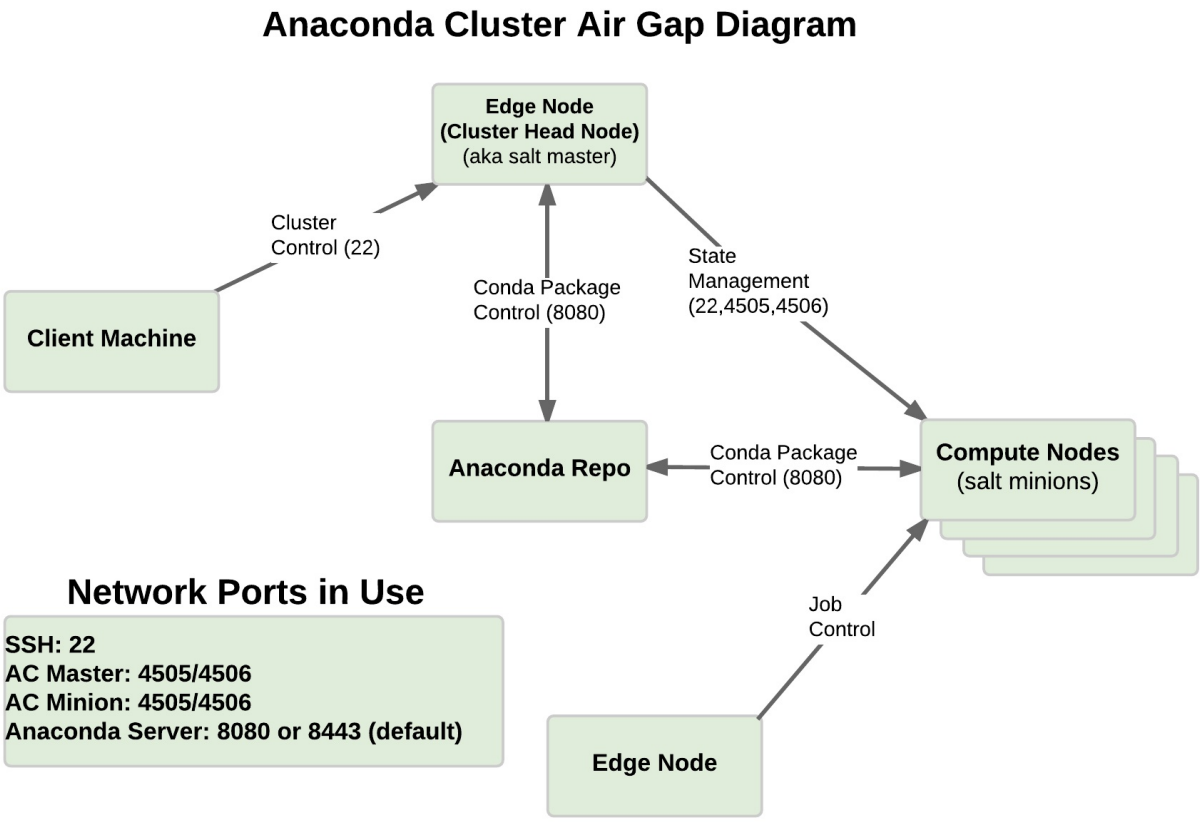
Anaconda Cluster Runbook

Anaconda Cluster is a resource management tool for Anaconda that allows users to easily create, provision, and manage bare-metal or cloud-based clusters. It enables management of Conda environments on clusters and provides integration, configuration, and setup management for Hadoop. Supported platforms include Amazon Web Services Elastic Compute Cloud (EC2), physical machines, or even a collection of virtual machines. The following runbook walks through the steps needed to install Anaconda Cluster. This runbook is designed for two audiences: those who have direct access to the internet for installation and those where such access is not available or restricted for security reasons. For these restricted a.k.a. "Air Gap" environments, Continuum ships the entire Anaconda product suite on portable storage medium or as a downloadable TAR archive. Where necessary, additional instructions for Air Gap environments are noted. If you have any questions about the instructions, please contact your sales representative or Priority Support team, if applicable, for additional assistance.

The following runbook walks through the steps needed to install a basic Anaconda Cluster comprised of a management ""client"" machine and four cluster nodes:

- **Client machine:** Where the Anaconda Cluster is configured from. Typically an IT-managed machine.
- **Head node (1):** Controls the software, file/folder and environment state of compute nodes
- **Compute nodes (3):** Where jobs are run. Managed by the head node.



Requirements

Hardware Requirements

- Client machine
  - 2+GB RAM, 2+CPU cores
- Head and Compute nodes
  - 8+GB RAM, 8+CPU cores

- Storage: Head and Compute nodes
  - ~1GB for Anaconda Cluster software

Software Requirements

- RHEL/CentOS 6.7 (Other operating systems are supported, however this document assumes RHEL or CentOS 6.7)
- MongoDB version 2.6
- Anaconda Repo Server (Anaconda Cloud or local Anaconda Repo)
  - If using a local Anaconda Repo, Anaconda Cluster channel has been mirrored to **http://your.anaconda.server:8080/anaconda-cluster**

Security Requirements

- Client machine
  - Passwordless SSH access to Head and Compute nodes
- Head and Cluster nodes
  - Matching user account/credentials on all nodes with passwordless sudo enabled
- SELinux in Permissive mode

Network Requirements

TCP Ports

- Inbound TCP 22 (SSH) from client machine to all nodes
- Inbound TCP 4505,4506 (salt)
  - from client machine to all nodes
  - from Head node to Compute nodes
- Outbound TCP 443 from all machines to Anaconda Cloud or local Anaconda Server

Other Requirements

Assuming the above requirements are met, there are no additional dependencies necessary for Anaconda Repo.

Assumptions

- '**admin**' user exists on client machine and all cluster nodes
- Client machine has an IP address of 172.31.56.123
- Head node has an IP address of 172.31.60.133
- Cluster nodes have IP addresses of 172.31.55.76, 172.31.55.77 and 172.31.55.78
- Those with **Air Gap** installations have already completed the local **Anaconda Repo** installation and have a working local **Anaconda Repo** and Anaconda Cluster channel is available at **http://your.anaconda.server:8080/anaconda-cluster**

Air Gap vs. Regular Installation

As stated previously, this document contains installation instructions for two audiences: those with internet access on the destination server(s) and those who have no access to internet resources. Many of the steps below have two sections: "**Air Gap Installation**" and "**Regular Installation**". Those without internet access should follow the **Air Gap Installation** instructions and those with internet access should follow **Regular Installation** instructions.

Prepare the Client Machine

Anaconda Cluster is managed by one or more client machines with the anaconda-cluster packages installed, along with the cluster configuration information. The client machine orchestrates the head node to install packages, run commands and operational tasks on the compute nodes.

Create an SSH keypair as the admin user

```
 ssh-keygen
Generating public/private rsa key pair.
```

```
Enter file in which to save the key (/home/admin/.ssh/id_rsa):
Created directory '/home/admin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/admin/.ssh/id_rsa.
Your public key has been saved in /home/admin/.ssh/id_rsa.pub.
The key fingerprint is:
d0:67:26:cd:3f:7c:f1:7e:b1:f4:fb:2b:86:2b:5e:c2 admin@localhost.localdomain
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|       . o       |
|      . o *   .  |
|       . = o   o |
|        S   + .o.|
|         .    o..+|
|          E ..  .+|
|           .o. o  o|
|         ....o .o+|
+-----------------+
[admin@localhost ~]$
```

NOTE: Save the contents of ~admin/.ssh/id_rsa.pub to a text file.

Download Miniconda

- **Air Gap Installation:**

```
curl 'http://your.anaconda.server:8080/static/extras/miniconda/Miniconda-latest-Linux-x86_64.sh' > Minic
```

- **Regular Installation:**

```
curl 'http://repo.continuum.io/miniconda/Miniconda-latest-Linux-x86_64.sh' > Miniconda.sh
```

Install Miniconda

```
bash Miniconda.sh
```

Review and accept the license terms:

```
Welcome to Miniconda 3.19.3 (by Continuum Analytics, Inc.)
In order to continue the installation process, please review the license agreement. Please, press ENTER t
```

Accept the default location or specify an alternative:

```
Miniconda will now be installed into this location:
/home/admin/miniconda2  -Press ENTER to confirm the location
-Press CTRL-C to abort the installation
-Or specify a different location below
 [/home/admin/miniconda2] >>>" [Press ENTER]
 PREFIX=/home/admin/miniconda2
installing: python-2.7.10-0
...
Python 2.7.10 :: Continuum Analytics, Inc.
creating default environment... installation finished.
```

Update the admin user's path (prepending /home/admin/miniconda2):

```
Do you wish the installer to prepend the Miniconda install location to PATH in your /home/admin/.bashrc
```

For the new path changes to take effect, "source" your .bashrc or start a new bash shell:

```
source ~/.bashrc
```

Use conda to download and install Anaconda Cluster

Add the anaconda-cluster and anaconda channels:

- **Air Gap Installation:** Install the Anaconda Cluster channel from the local Anaconda Repo server.

```
conda config --add channels http://your.anaconda.server:8080/conda/anaconda-cluster
conda config --add default_channels http://your.anaconda.server:8080/conda/anaconda --system
```

- **Regular Installation:** Install the Anaconda Cluster channel from Anaconda Cloud.

```
export TOKEN=<your Anaconda Cloud token>
conda config --add channels http://conda.anaconda.org/t/$TOKEN/anaconda-cluster
```

Install the anaconda-cluster packages:

```
conda install anaconda-cluster
```

Run the acluster command to initialize Anaconda Cluster:

```
acluster
```

NOTE: this command creates the **~/.acluster** directory

Prepare Head and Compute Nodes

In order to communicate with the Head and Compute nodes, the admin user needs to be able to SSH to each node without being prompted for a password. To allow this but still maintain a level of security, we'll use public key authentication from the client machine to the nodes. Additionally, we'll allow the admin user to execute root level commands via sudo, in order to install system packages, update system files, etc. NOTE: The steps below should be repeated on the Head node and all Compute nodes.

Import SSH public key

Using your editor of choice, copy the contents of **~admin/.ssh/id_rsa.pub** (from step XYZ above) from the Client machine to **~/admin/.ssh/authorized_keys**

Your **~/admin/.ssh/authorized_keys** will look something like this:

```
cat ~admin/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA6NF8iallvQVp22WDkTkyrtvp9eWW6A8YVr+kz4TjGYe7gHzIw+niNltGEFHzD8+v1I2YJ6oXe
```

Make sure ~/admin/.ssh/authorized_keys has the proper permissions:

```
chmod 600 ~admin/.ssh/authorized_keys
```

Enable passwordless sudo

Add the following line to the bottom of /etc/sudoers to allow the admin user to run commands via sudo without entering a password:

```
admin ALL = (ALL) NOPASSWD: ALL
```

Configure the Cluster

An Anaconda Cluster consists of two primary pieces of information; the cluster profile and the cluster provider. Both of these live within the ~/admin/.acluster directory structure. In a nutshell, the profile describes the layout of the cluster: number of nodes, user to connect with, conda channels, plugins to install, etc. The provider describes *how* the cluster is provisioned and built; most importantly, cloud vs. bare-metal.

Define a Cluster Configuration

We're going to create a simple cluster configuration using the head node and 3 compute nodes we configured previously. Create the ~admin/.acluster/profiles.d/demo-cluster.yaml with the following content:

- **Air Gap Installation:**

```
 name: demo-cluster
provider: bare_metal
num_nodes: 4
node_id: bare
node_type: bare
user: admin
machines:
    head:
- 172.31.60.133
    compute:
- 172.31.55.76
- 172.31.55.77
- 172.31.55.78
conda_channels:
 - http://your.anaconda.server:8080/conda/anaconda-cluster
 - http://your.anaconda.server:8080/conda/anaconda
anaconda_url: http://your.anaconda.server:8080/static/extras/Miniconda-latest-Linux-x86_64.sh
default_channels: http://your.anaconda.server:8080/conda/anaconda
plugins:
 - conda:
     install_prefix: /opt/anaconda
     conda_sh: false
     conda_acl: true
```

- **Regular Installation:**

```
 name: demo-cluster
provider: bare_metal
num_nodes: 4
node_id: bare
node_type: bare
user: admin
machines:
    head:
- 172.31.60.133
    compute:
- 172.31.55.76
- 172.31.55.77
- 172.31.55.78
plugins:
 - conda:
     install_prefix: /opt/anaconda
     conda_sh: false
     conda_acl: true
```

*Note: * More information about cluster profiles can be found here.

Add the following to ~admin/.acluster/providers.yaml:

```
bare_metal:
  cloud_provider: none
  private_key: ~/.ssh/id_rsa
```

More information about cluster providers can be found here.

Create the Cluster

Now that the cluster has been defined, we're ready to create it. Use the acluster create command to start the cluster. "demo" is the name of the new cluster and "demo-cluster" is the name of the cluster profile (from ~admin/.acluster/profiles.d/demo-cluster.yaml).

```
 acluster create demo -p demo-cluster

Creating cluster
No license file found matching /home/admin/.acluster/cluster*.lic
Using unlicensed limits.
Number of existing nodes: 0
Number of requested nodes: 4
Licensed nodes: 4
License is valid for the current number of nodes.
INFO: Creating new cluster "demo" with profile "demo-cluster"
INFO: Creating 4 instances
INFO: Checking SSH connection
INFO: Successfully created instances
INFO: Cluster info: {'ips': ['172.31.60.133', '172.31.55.76', '172.31.55.77', '172.31.55.78'], 'user': 'admin
Saving cluster file
No license file found matching /home/admin/.acluster/cluster*.lic
Using unlicensed limits.
Cluster "demo": 4 nodes
Number of existing nodes: 4
Number of requested nodes: 0
Licensed nodes: 4
License is valid for the current number of nodes.
Checking ssh connection
INFO: Checking SSH connection
Checking sudo
Bootstraping conda
INFO: Checking conda installation
INFO: Installing miniconda

Syncing formulas

Installing plugin 1/1: conda
INFO: Disconnecting from all active servers
Disconnecting from 172.31.60.133... done.
Disconnecting from 172.31.55.76... done.
Disconnecting from 172.31.55.77... done.
Disconnecting from 172.31.55.78... done.
Done
```