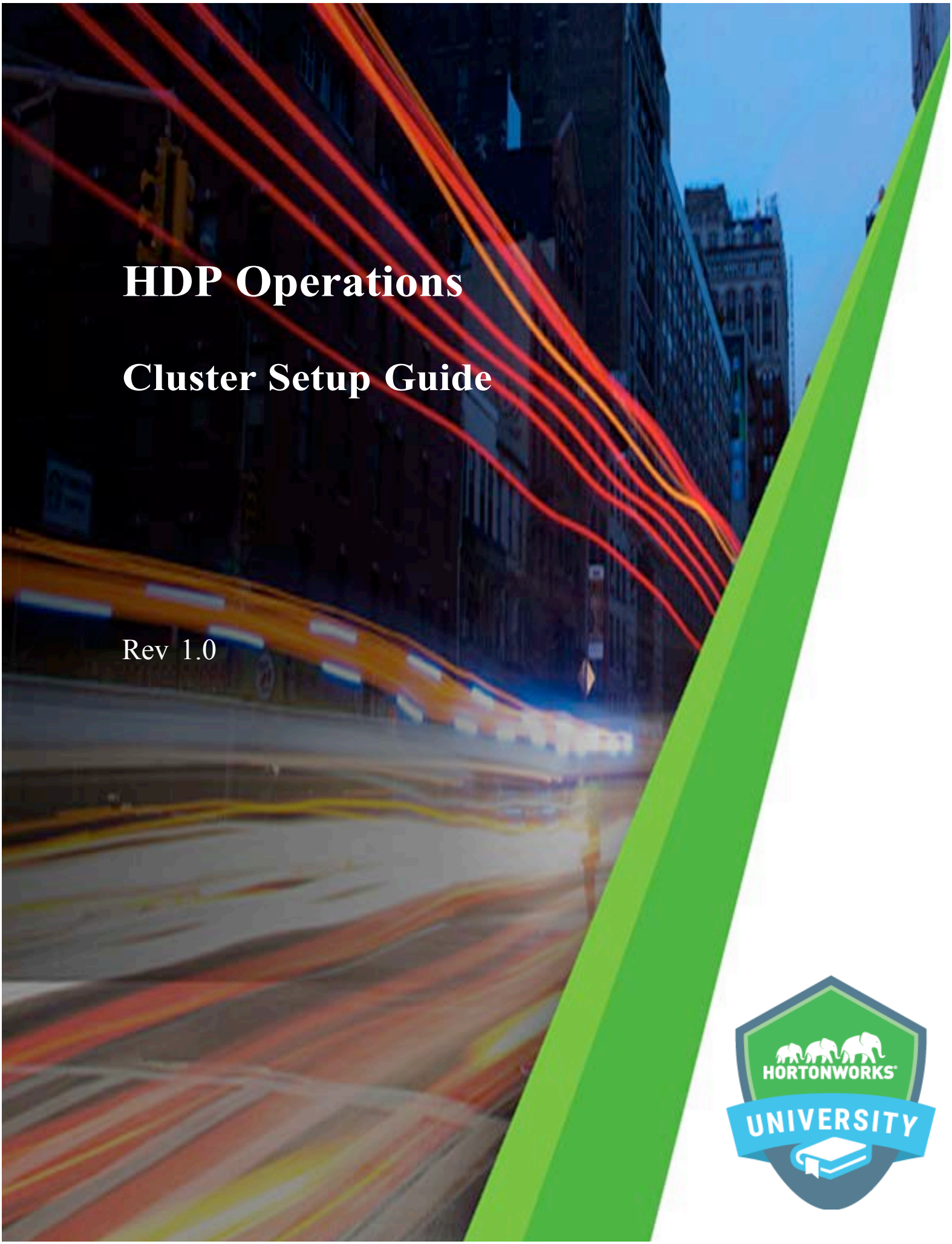


HDP Operations

Cluster Setup Guide

Rev 1.0

HORTONWORKS UNIVERSITY

Create the Operations Clusters in AWS

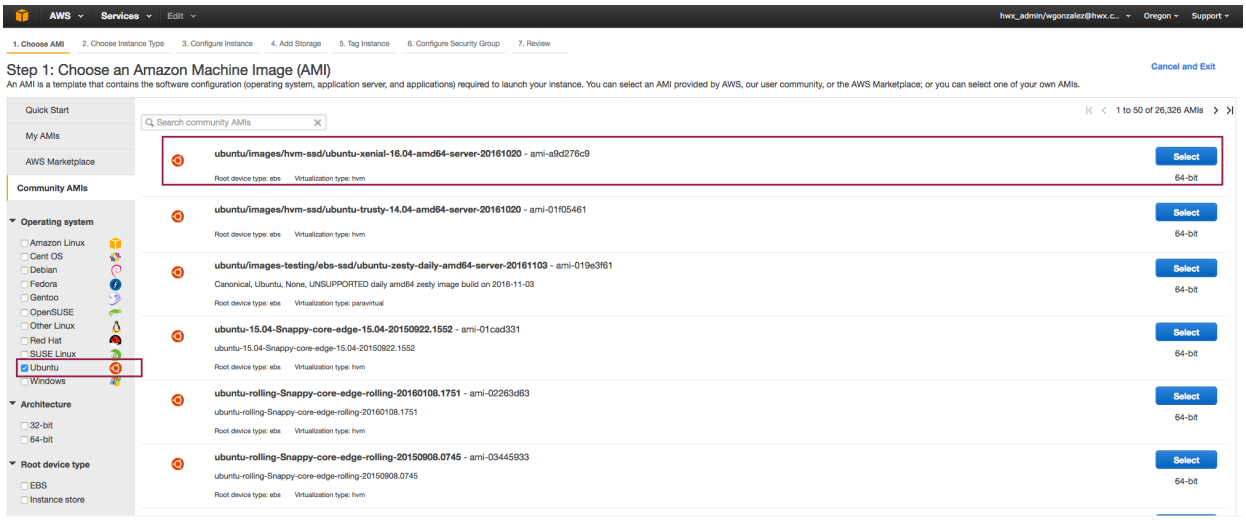
Objective:	This is an instruction guide to setup HDP Operations Clusters for Admin1, Admin2 & Security trainings.
Before You Begin:	• You need to have an AWS account to launch clusters. • You need to have access to the AWS EC2 console. • You need to ask Hortonworks to share the Windows Server AD Image with you (please contact hwuniversity@hortonworks.com).

Step 1: Prepare an AWS CLI Server to Deploy AWS Clusters

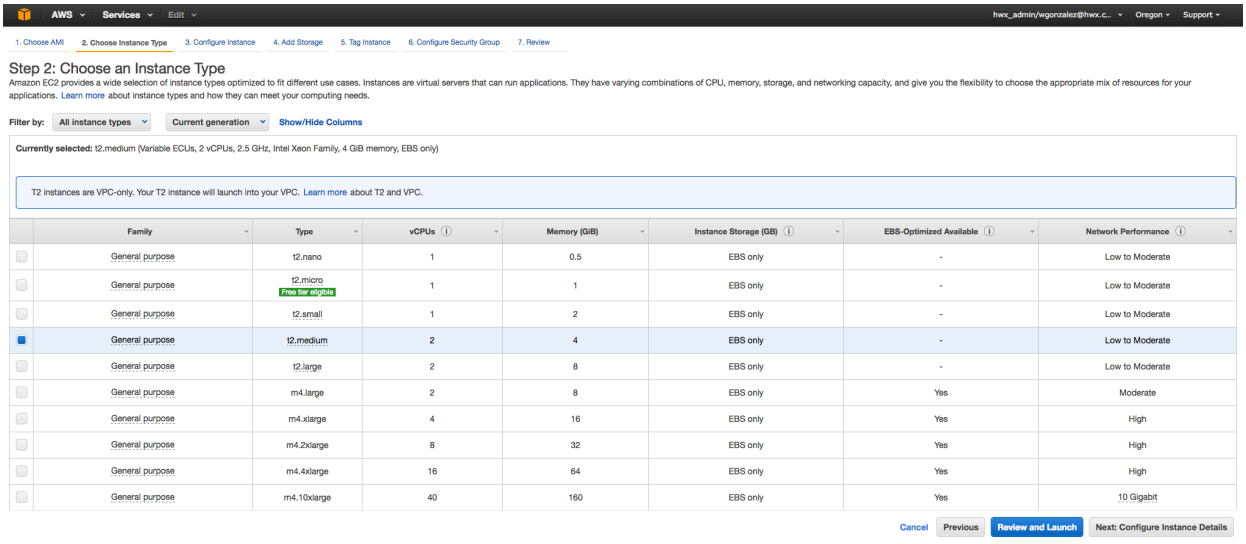
1.1. If you already have a server that has AWS CLI installed, you can jump to Step 2.

1.2. Launch an Ubuntu AWS image. This step can also be performed on other OS, although we are only giving Ubuntu as an example.

1.2.1. From your AWS console, choose an Ubuntu AMI in your zone similar to: Ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20161020:



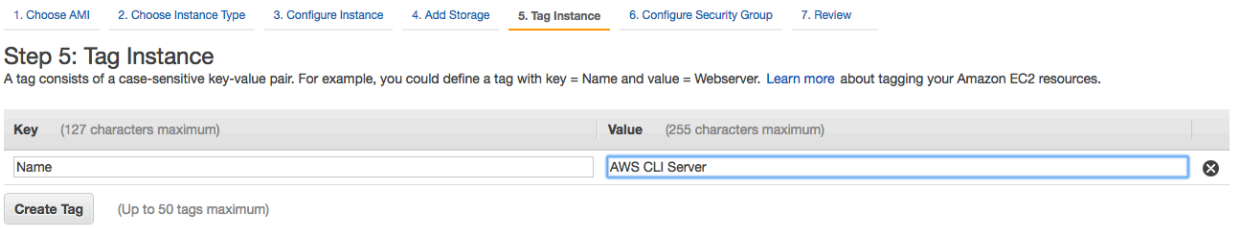
1.2.2. For the instance type, select t2.medium:



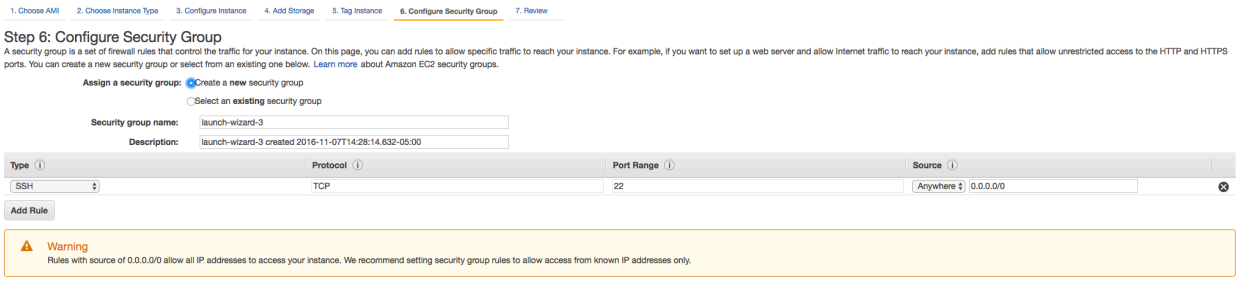
1.2.3. On the 'Step 3 - Configure Instance' steps you can take most of the defaults, just make sure you enable 'Auto-assign Public IP'

1.2.4. Take defaults on 'Step 4 – Add storage'

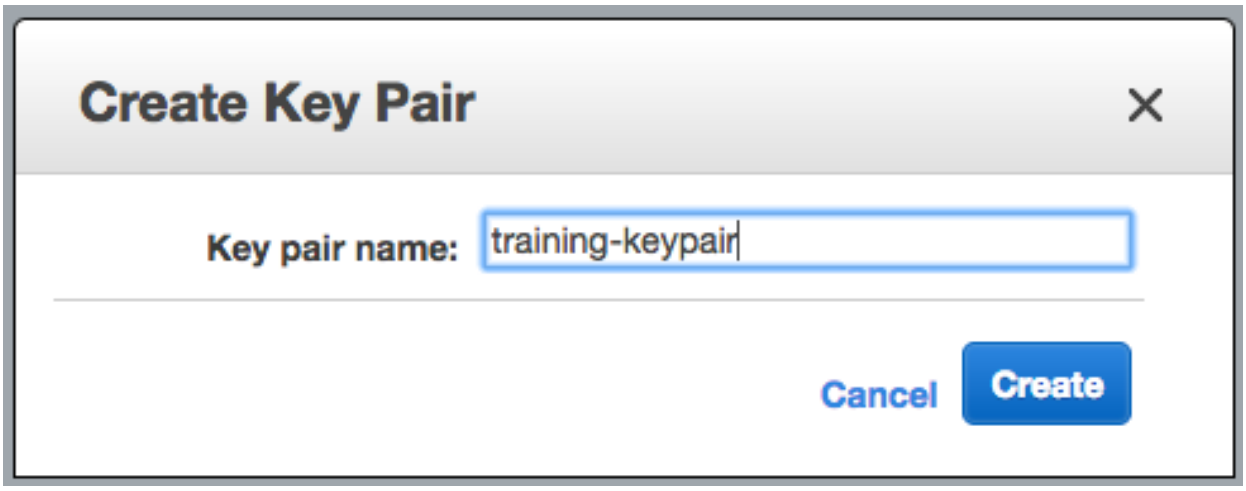
1.2.5. Name your instance appropriately on 'Step 5 – Tag Instance':



1.2.6. On 'Step 6 – Configure Security Group' create a new group with SSH access, then review and Launch:



1.2.7. Create a NEW key-pair for accessing your server. Name your keypair "training-keypair." The new keypair will be downloaded locally.



1.3. Install AWS CLI on the newly provisioned AWS CLI Server

1.3.1. After the instance finishes launching (check 'Instance State' and 'Status check' columns on the EC2 Console), try to ssh to the instance using a terminal window (if connecting from MacOS) or puTTY (if connecting from Windows). For instance, from a MacOSX terminal, type (substitute < … > for the actual values):

```
$ ssh -i /<directory where key-pair was installed>/training-
keypair.pem ubuntu@<instance IP>
```

1.3.2. Install the AWS CLI as described here:

http://docs.aws.amazon.com/cli/latest/userguide/installing.html#install-bundle-other-os

1.3.3. After installing the AWS CLI, make sure the 'aws' command works:

```
$ aws
usage: aws [options] <command> <subcommand> [<subcommand> ...]
[parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help
aws: error: the following arguments are required: command
$
```

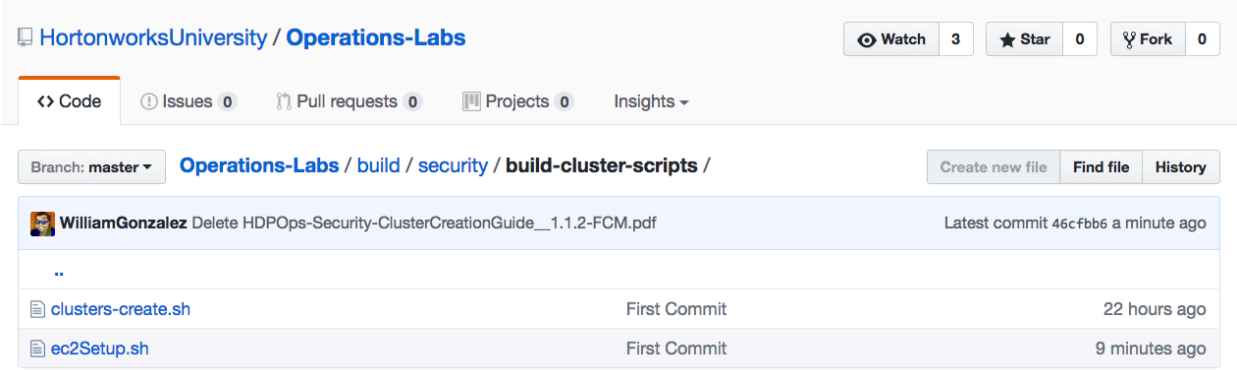
Step 2: Download and Install Operations Cluster Create scripts into AWS CLI Server

2.1. Upload scripts from the relevant GitHub Repo for this guide (see step below).

2.1.1. Operations Clusters (Admin1, Admin2, Security) please see:

https://github.com/HortonworksUniversity/Operations-Labs/tree/master/build/security/build-cluster-scripts

There are 2 scripts found in that directory:



2.1.2. You can use wget from your AWS CLI Server to get those 2 files:

```
$ wget https://raw.githubusercontent.com/HortonworksUniversity/Operations-Labs/master/build/security/build-cluster-scripts/clusters-create.sh

$ wget https://raw.githubusercontent.com/HortonworksUniversity/Operations-Labs/master/build/security/build-cluster-scripts/ec2Setup.sh
```

2.1.3 Copy or Move the training-keypair.pem to the same directory in AWS CI Server where the other 2 files from GitHub were saved.

```
If you are using a Windows machine to connect to the AWS instance:  $ c:\>"c:\Program Files (x86)\PuTTY\pscp.exe" -l ubuntu scp c:\Users\myUserName\Documents\*.* <IP>:
```

2.1.4 If you are using a MacOS machine to connect to the AWS instance:

```
$ scp -i ./NEW-AWS-CLI-Server-keypair.pem <directory where you downloaded github script files>/* ubuntu@<IP>:~
```

2.2 Login to the AWS CLI Server again and ensure the 3 files got copied to AWS CLI Server appropriately (this includes the keypair not shown in screenshot below):

```
$ $ ls -al /home/ubuntu/
total 44
drwxr-xr-x 4 ubuntu ubuntu 4096 Nov  7 20:59 .
drwxr-xr-x 3 root   root   4096 Nov  7 19:34 ..
-rw------- 1 ubuntu ubuntu   34 Nov  7 20:57 .bash_history
-rw-r--r-- 1 ubuntu ubuntu  220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Aug 31  2015 .bashrc
drwx------ 2 ubuntu ubuntu 4096 Nov  7 20:00 .cache
-rwx------ 1 ubuntu ubuntu 1527 Nov  7 20:59 clusters-create.sh
-rwx------ 1 ubuntu ubuntu 4958 Nov  7 20:59 ec2Setup.sh
-rw-r--r-- 1 ubuntu ubuntu  655 Jun 24 15:44 .profile
drwx------ 2 ubuntu ubuntu 4096 Nov  7 19:34 .ssh
```

2.3 Make the scripts executable:

```
$ chmod 755 /home/ubuntu/*.sh
```

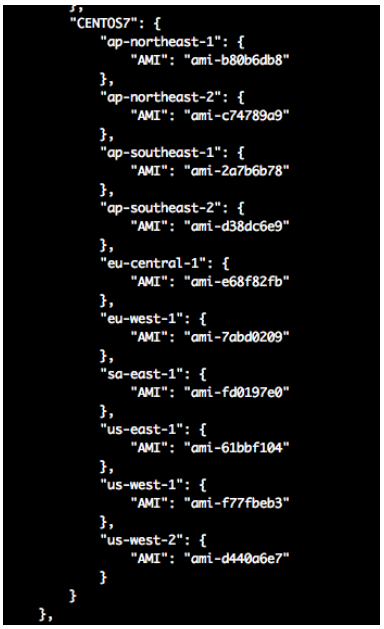
2.4 Add the JSON template to the current directory

2.4.2 Into the same directory in AWS CLIServer above, wget the file found in

```
$ wget https://github.com/HortonworksUniversity/Operations-
Labs/blob/master/build/security/cloudformation.json
```

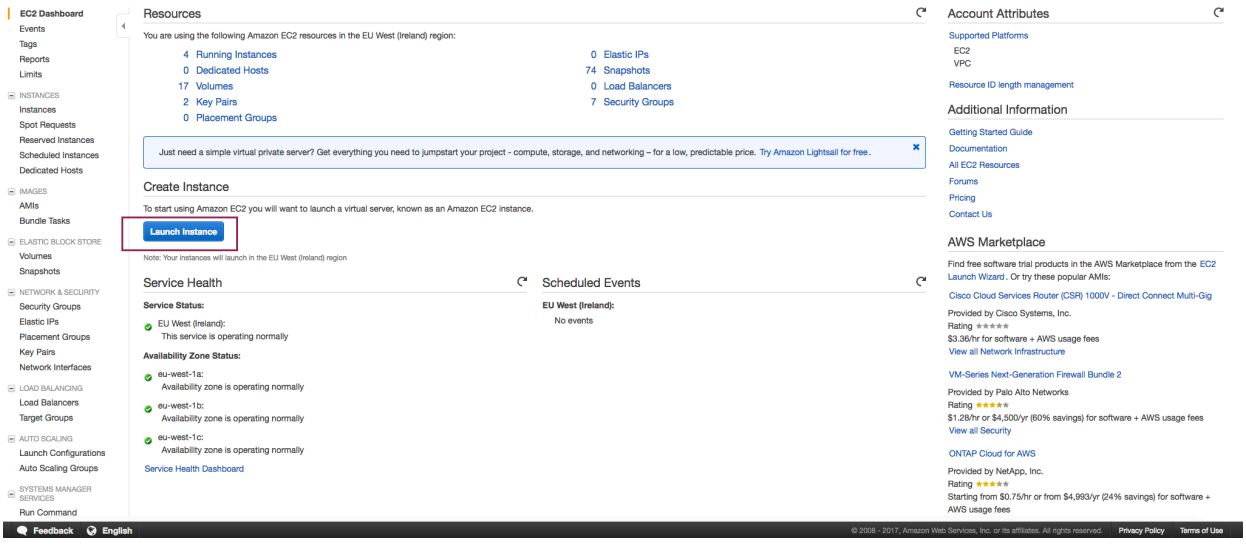
2.4.3 To clarify: there should now be 4 files in your folder in the AWS CLI Server: clusters-create.sh, ec2Setup.sh, cloudformation.json and the training-keypair that was downloaded from an earlier step.

2.4.4 The json file defines, among other things the AWS AMIs that will be use as base images for the nodes.

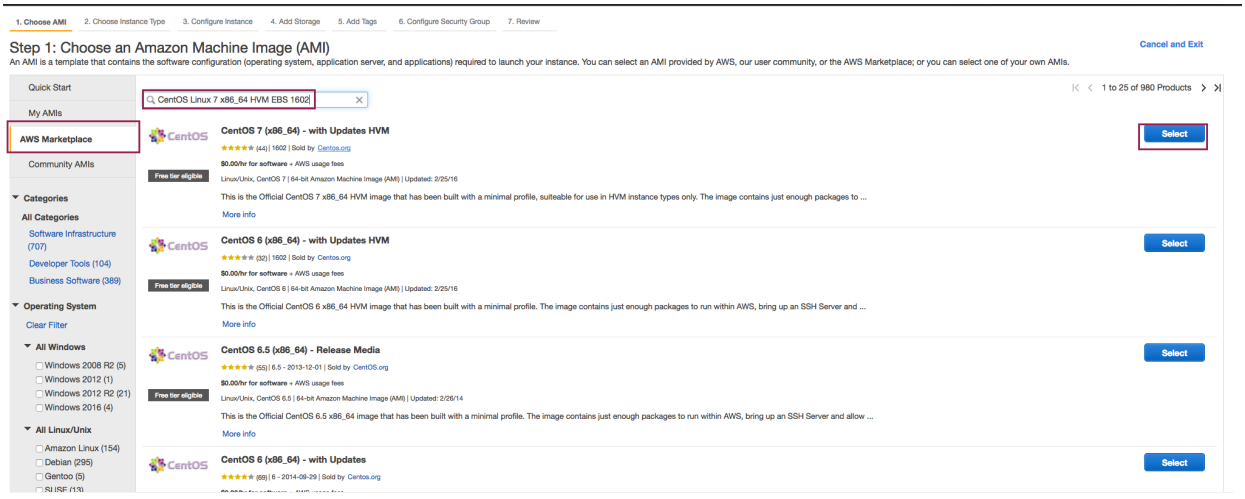


2.4.5 These AMIs listed in the screenshot are HWX Private images and cannot be shared. You need to change these AMIs into free “AWS Marketplace” AMIs, one for each region that you want to launch a cluster in. The subsequent steps describe how to obtain and customize these AMIs.

2.4.6 For this example, we will use Ireland AWS Region. You need to replicate these steps for each region that you want to launch clusters in. In Ireland region EC2 dashboard, select “Launch Instances”

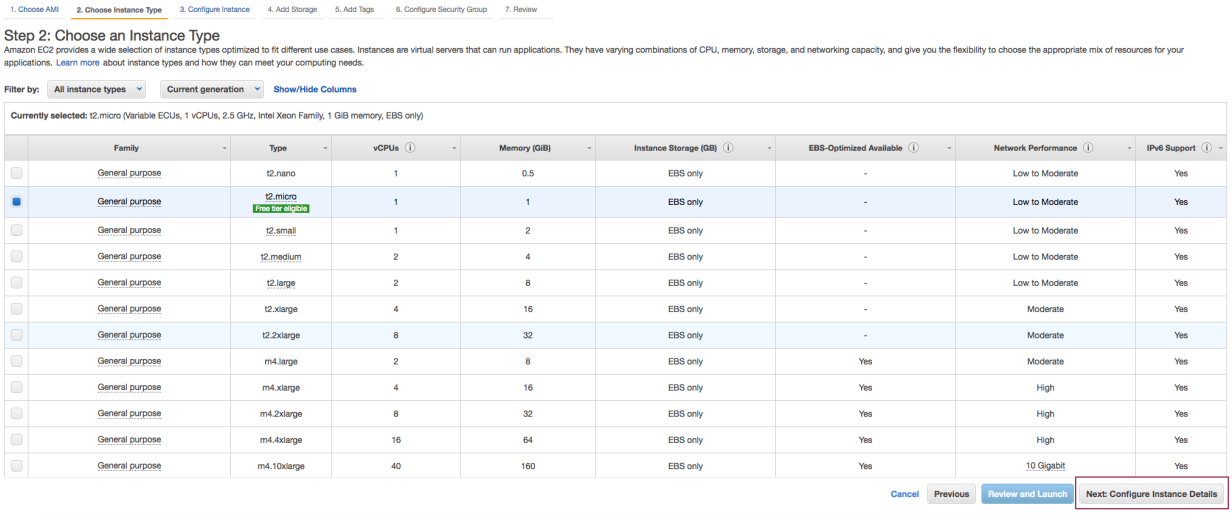


2.4.7 Select AWS Marketplace and search for this string “CentOS Linux 7 x86_64 HVM EBS 1602”

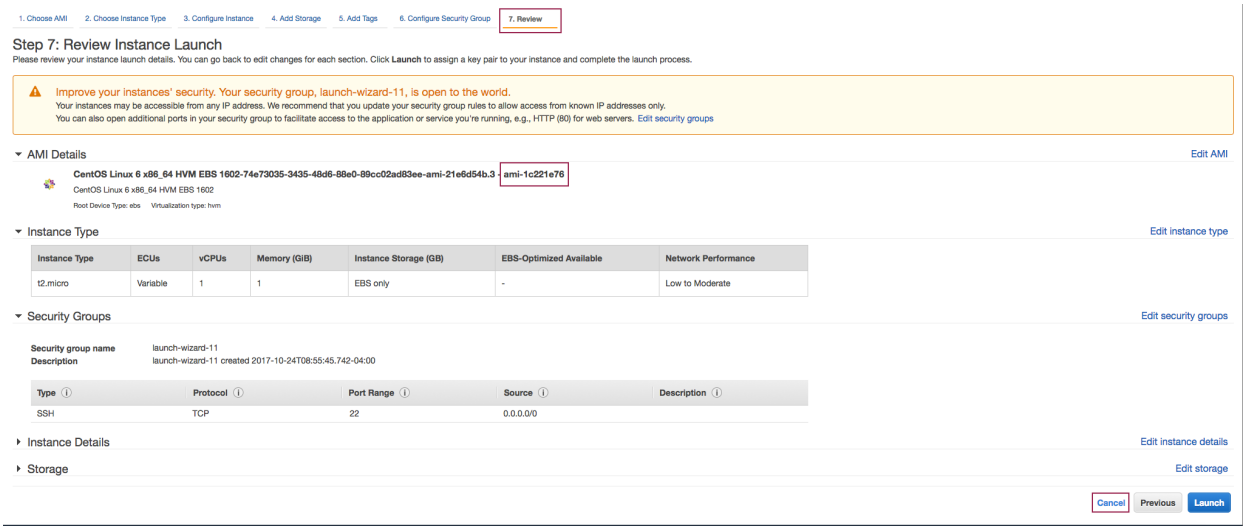


2.4.8 Unfortunately, the AMI-ID id of the CentOS instance is not provided in this screen. To get the AMI ID, you need to go thru each step of launching, and you can obtain the AMI-Id in the last step (7), see the step below for an example (but do not actually “launch” the instance as it is not needed, we only want to obtain the Base CentOS AMI-ID – see next step).

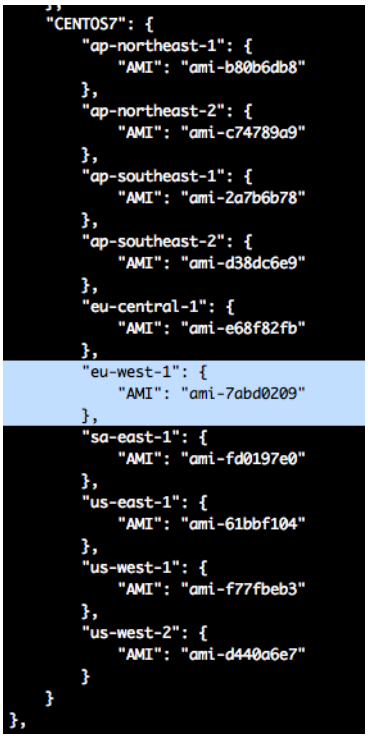
2.4.9 You will notice that the launch procedure has 7 steps. Keep hitting next and taking all defaults until the 7th step:



2.4.10 In the last step 7-Review you will see the ami-id. Since we are not interested in launching it from here, only to obtain the ami-id, take note of the ami-id and cancel the launch.



2.5 Now that you know the base CentOS ami-id to use in your clusters, go back and edit the cloudformation.json file obtained from github earlier and enter the correct AMI id for your region as obtained from the step above:



2.6 Update the clusters-create.sh script to your parameters

2.6.2 Make a backup copy of the clusters-create.sh script first

2.6.3 There is a line that points to the github copy of the json file that needs to be changed to point to the local json file updated in the last step:

Change from:

```
--template-body
https://raw.githubusercontent.com/HortonworksUniversity/Operatio
ns-Labs/master/build/security/cloudformation.json \
```

to:

```
--template-body file:////home//ubuntu//cloudformation.json \
```

2.6.4 In this example, my json file is in "/home/ubuntu/cloudformation.json" Notice the double slash syntax for directories.

2.6.5 Now you need to configure your clusters-create.sh to your settings. The next steps show you how.

Step 3: Launch Operations Cluster(s)

3.1 Please review and update ec2Setup.sh. The next several steps describe how.

3.1.1 Edit lines 9-29 in the ec2Setup.sh script

```
$ vi /home/ubuntu/ec2Setup.sh
```

Here is a screenshot of the first few lines of the script:

```
1  #!/bin/bash
2
3  ##############################
4  # CREATE OPERATIONS CLUSTERS #
5  ##############################
6
7  # 1. DEFINE VARIABLES
8  # 1.1 global vars
9  export AWS_CLI_HOME=/usr/local/aws
10 PATH=$PATH:$AWS_CLI_HOME/bin
11 export AWS_ACCESS_KEY_ID=<your aws key id>
12 export AWS_SECRET_ACCESS_KEY=<your aws secret access key>
13
14 # 1.2 job specific variables -- change these as required
15 export AWS_DEFAULT_REGION=us-west-2
16 export EC2_URL=https://ec2.us-west-2.amazonaws.com
17 export SEC_GROUP=<your aws sec group>
18 export AMI=ami-0e3fde6e
19 export AMI2=ami-b35346ca
20 export AMI3=ami-f8574281
21 export SUBNET=<your aws subnet id>
22 export TRAINING_NAME=BASE-NAME-FOR-YOUR-CLUSTERS
23 export FIRST_CLUSTER_LABEL=100
24 export NO_OF_VMs=1
25 export NO_OF_ADDTL_NODES=3
26 export INSTANCE_TYPE="m4.large"
27 export ADD_AD_SERVER=true
28 export ADD_XR_SERVER=true
29 export DEPLOY_CLUSTER=false
```

3.1.2 Lines 9-12 are normally not need to change, but verify that the aws command is at /usr/local/aws.

3.1.3 Change lines 11-12 to reflect your AWS account key id and secret key

3.1.4 Change lines 15-16 to reflect the AWS region where you want the clusters to be created.

3.1.5 Line 17 represents a AWS Security group that you need to create that exposes ports (or IPs) for the cluster. This Sec-Group will be used for all cluster nodes as well as any AD/KDC servers (if required). To setup the AWS Sec Group ports, please see table in step 3.1.15 or 3.1.16.

3.1.6 Lines 18- 19 are the AMI IDs for the Windows AD Server, and the Cross-realm servers. These images will need to be shared with your AWS account in order for you to use them. These images live in the us-west-2 (Oregon) region. If you do not have these AMIs shared with you, or you need them on another AWS region, please contact hwuniversity@hortonworks.com.

3.1.7 Line 21 is an AWS VPC subnet that needs to be created in your AWS region. For info on how to create a VPC Subnet, please refer to: http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Subnets.html

3.1.8 Line 22 is the Basic Name of the clusters (one for each training class). Since this setup creates a multi-node clusters, this line represents the cluster PREFIX. Normally clusters will be appended a NUMBER for each student after this basic name, for instance: My-Cluster-100, My-Cluster-101, etc., where the postfix 100 & 101 refer to a separate clusters per student. For instance there will be several instances named "My-Cluster-100" that all refer to a single student's cluster.

3.1.9 Line 23 is the numeric POSTFIX that will be appended to the Cluster Name Prefix described in the previous step. In the screenshot, the postfix for the first cluster, for instance, is "100" and will be incremented by 1 for each cluster, thus you can end up with clusters each named "My-Cluster-100," "My-Cluster-101," "My-Cluster-102" etc.

3.1.10 Line 24 is the number of Clusters (one per student) to be created. Each cluster can have 2 or more nodes (one Ambari + x additional nodes).

3.1.11 Line 25 is the number of "additional" nodes (besides Ambari) to be created per cluster. For the Operations training, this is either 2 or 3. As per screenshot, this default is "3" and this will create an Ambari node + 3 Additional nodes (total 4 nodes) per student/cluster.

3.1.12 Line 26 is the AWS instance type. We have found that "m4.large" is acceptable for this class.

3.1.13 Line 27 is the flag whether to create an AD Server or not. For the Security Training this is required; for Admin trainings, this may be optional. Leave as "true." Please note that only 1 AD server is created/needed per class. Similarly, line 28 is used in case that Kerberos cross-realm authentication is desired. This will create 2 servers: one MIT-KDC and one Win-AD for x-realm auth. Normally you will only need to set to true either line 27 or 28.

3.1.14 After making the appropriate changes, save the ec2Setup.sh script.

3.1.15 AWS Security Group Port list

These are the ports and protocols needed for Operations trainings.

Type	Protocol	Port	Source
SSH	TCP	22	0.0.0.0/0
HTTP	TCP	80	0.0.0.0/0
LDAP	TCP	389	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0
Custom TCP Rule	TCP	3000	0.0.0.0/0
Custom TCP Rule	TCP	3359	0.0.0.0/0
RDP	TCP	3389	0.0.0.0/0
Custom TCP Rule	TCP	4000	0.0.0.0/0
Custom TCP Rule	TCP	6080	0.0.0.0/0
Custom TCP Rule	TCP	6083	0.0.0.0/0
Custom TCP Rule	TCP	8080	0.0.0.0/0
Custom TCP Rule	TCP	8081	0.0.0.0/0
Custom TCP Rule	TCP	8088	0.0.0.0/0
Custom TCP Rule	TCP	8443	0.0.0.0/0
Custom TCP Rule	TCP	8444	0.0.0.0/0
Custom TCP Rule	TCP	8983	0.0.0.0/0
Custom TCP Rule	TCP	9995	0.0.0.0/0
Custom TCP Rule	TCP	11000	0.0.0.0/0
Custom TCP Rule	TCP	15000	0.0.0.0/0
Custom TCP Rule	TCP	19888	0.0.0.0/0
Custom TCP Rule	TCP	50070	0.0.0.0/0
Custom TCP Rule	TCP	50075	0.0.0.0/0
Custom TCP Rule	TCP	59100	0.0.0.0/0
Custom TCP Rule	TCP	61888	0.0.0.0/0
All traffic	All	All	sg-xxxxxxxx (THIS security group)
All ICMP - IPv4	All	N/A	0.0.0.0/0

3.1.16 If you prefer, you can also assign all ports to be opened to only your IP (or student IP), this may work better since it opens all ports but only to a single IP. For example, if your external IP is 127.1.33.5, then you can configure the aws security group as:

Type	Protocol	Port Range	Source	Description
All TCP	TCP	0 - 65535	127.1.33.5	My-External IP
All UDP	UDP	0 - 65535	127.1.33.5	My-External IP
All ICMP - IPv4	All	N/A	127.1.33.5	My-External IP

3.2 Run the ec2Setup.sh script:

```
$ /home/ubuntu/ec2Setup.sh
```

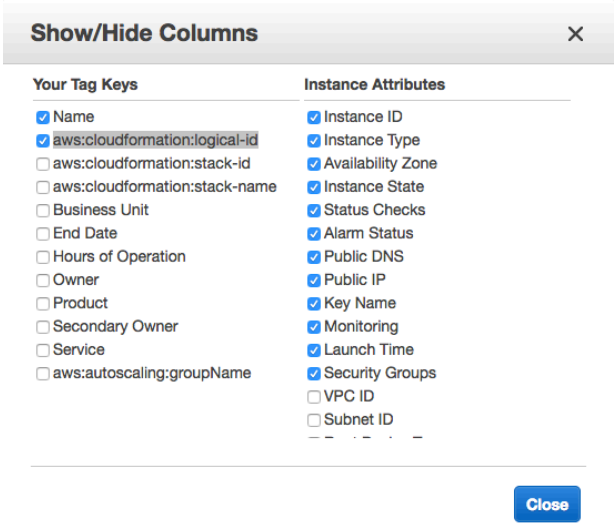
3.3 This will launch an AWS CloudFormation job that takes several minutes, please be patient (it may take up to 5 minutes per cluster to be build). You will eventually see an output similar to this:

```
[root@ip-10-0-0-11]# ./ec2Setup.sh
1. Creating new clusters in region us-west-2...
   This may take several minutes, please wait.

   Creating Win-AD Instance

   Creating Cross-real auth MIT-KDC Server

   Creating Cross-real auth AD Server

2. List of newly created Instances:
i-00d5682ef38d7a0da i-099b25bb38df537f2

3. Final List of Hosts:

 HDP-ADMIN-I-100 : AdditionalNodes : 34.210.203.38/172.30.0.234
 HDP-ADMIN-I-100 : AdditionalNodes : 52.38.208.114/172.30.0.227
 HDP-ADMIN-I-100 : AdditionalNodes : 34.215.132.78/172.30.0.10
 HDP-ADMIN-I-100 : AmbariNode----- : 34.212.144.101/172.30.0.15
 HDP-ADMIN-I     : WIN AD SERVER   : 35.161.59.176/172.30.0.25
 HDP-ADMIN-I     : MIT-KDC-XREALM  : 52.38.67.78/172.30.0.88
 HDP-ADMIN-I     : AD-XREALM       : 34.215.78.230/172.30.0.172
```

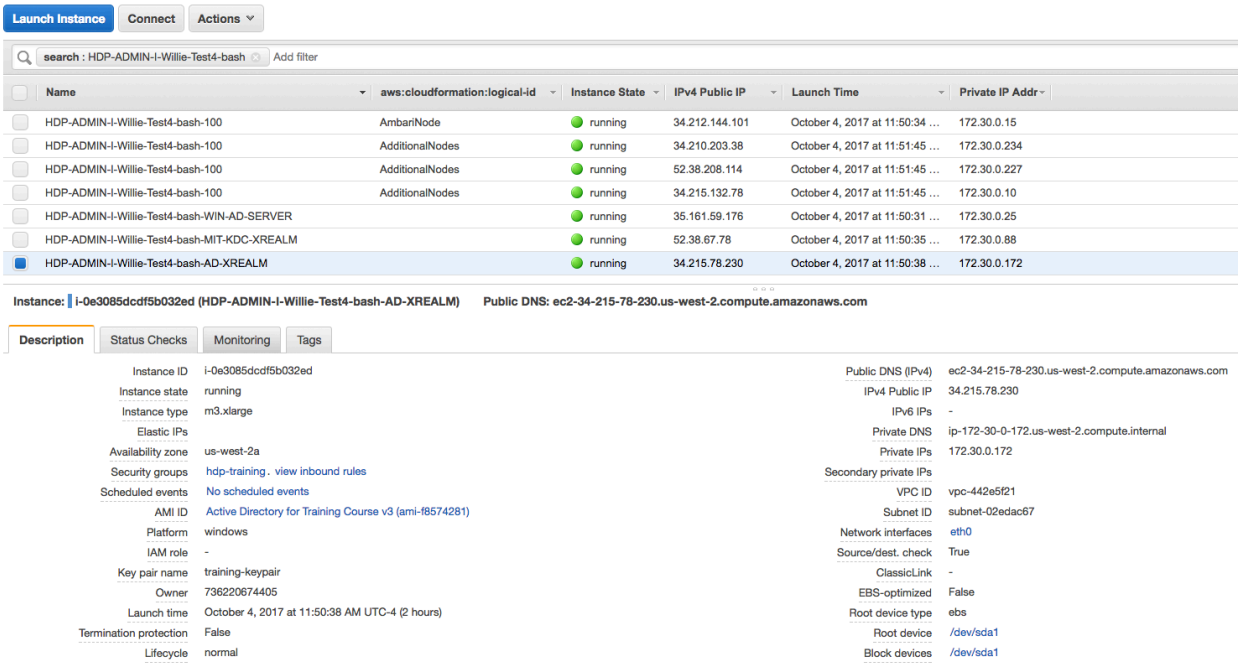
3.4 In this example, I created 1 cluster named “HDP-ADMIN-I” with an Ambari server, plus 3 additional nodes + Win AD server + 2 cross-realm servers. In this output, each node has external/internal IPs. The Ambari node, for instance, has an external IP: 34.212.144.101 and an internal IP: 172.30.0.15. Newer versions of this script may print DNS names instead of IPs (which is preferable for the labs).

3.5 You can also see all IPs, DNS and instance information in the EC2 Console:

3.5.1 In the EC2 console add a column for aws:cloudformation:logical-id:

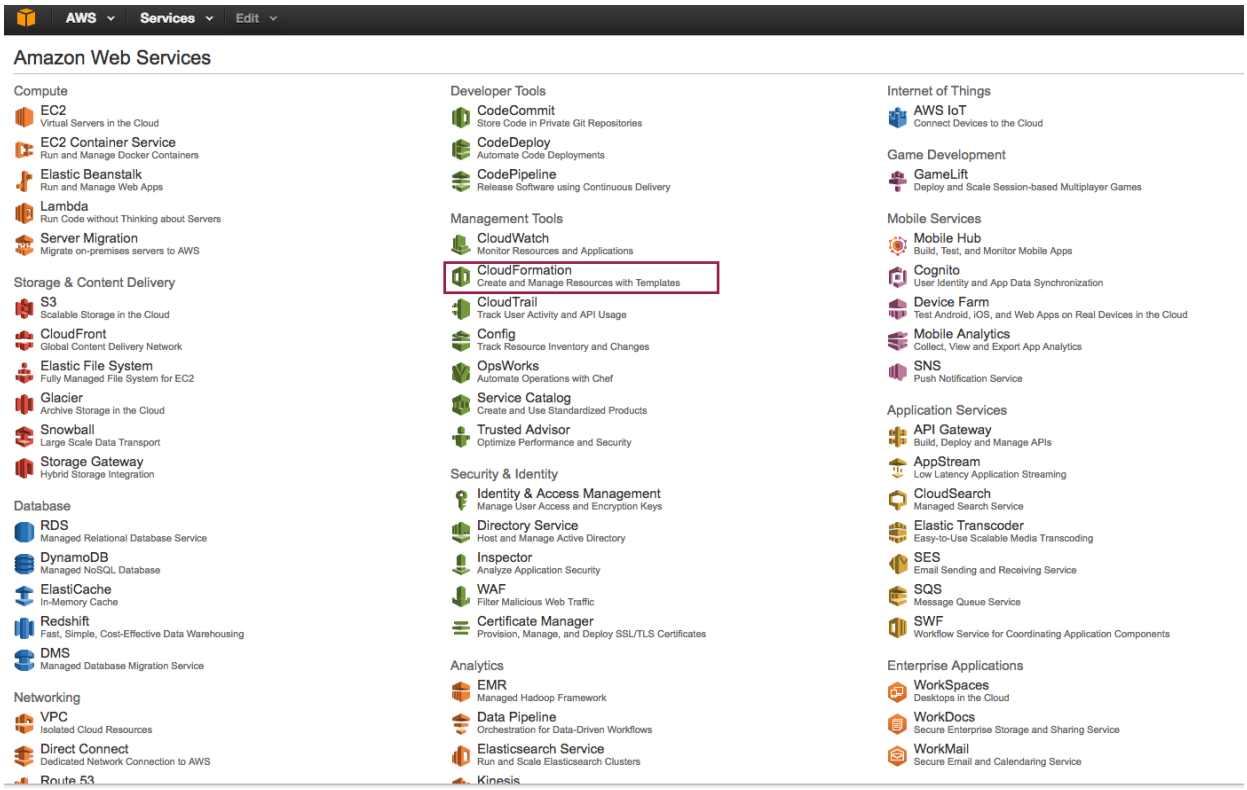


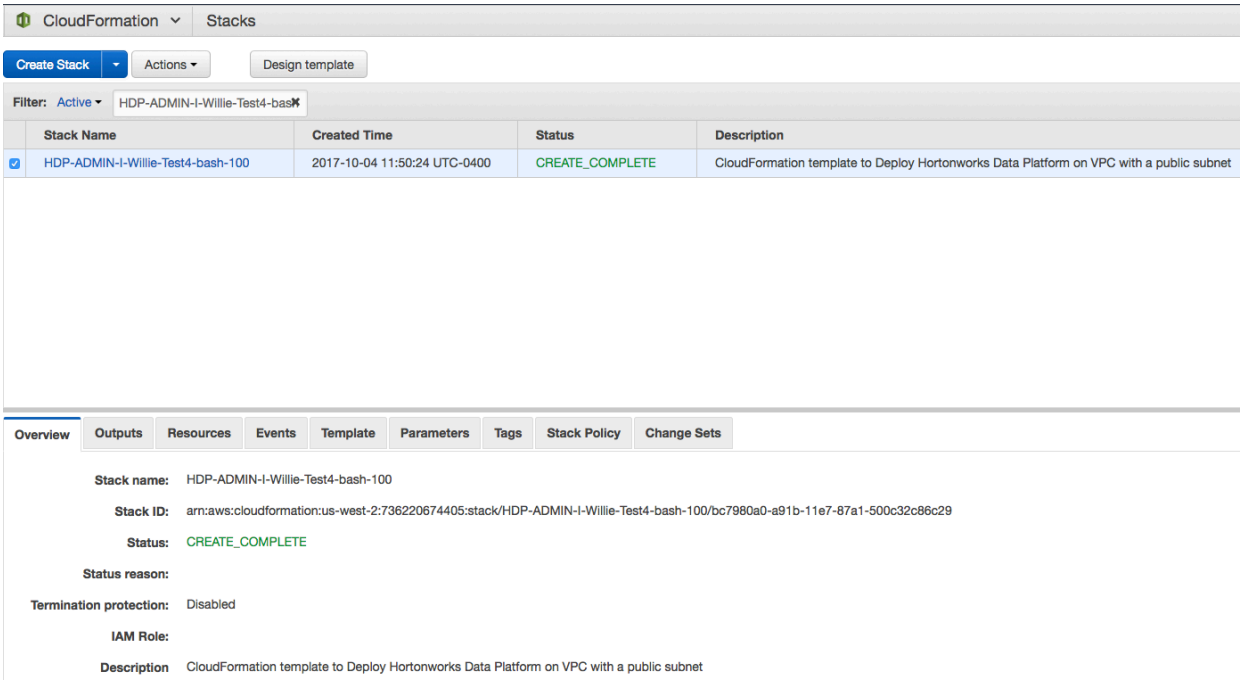
3.5.2 Search for your cluster name and you will see the created nodes:



3.5.3 In this screenshot, you can see One cluster, HDP-ADMIN-I-Willie-Test4-bash-100 (for one student), that has 1 Ambari node + 3 Additional Nodes + 1 AD Server + 2 cross realm servers.

3.6 Also, you can check the additional CloudFormation Administration console:





3.7 from the ec2Script.sh output. You are now ready to start the labs.

3.8 After your clusters have been successfully created, you can Stop (but not terminate) the AWS CLI Server to limit your AWS costs.

3.9 Do not stop/terminate the actual cluster nodes since they have a feature to auto-relaunch on stop/termination. To properly terminate the clusters, please see Step5.

3.10 If you need to "stop but not terminate" clusters because you anticipate some downtime, there is a separate procedure not explained in this guide. Please write to hwuniversity@hortonworks.com for a walk-thru of this.

Step 4: Test cluster connectivity

4.1 ssh to each cluster node via Terminal window (if MacOS) or puTTY (if Windows) using the previously downloaded training-keypair. Make sure that you can ssh and that nodes are responding to ping between each other.

4.2 To access the WIN-AD server or the Cross-real Win AD box, you will need the Remote Desktop application (Windows) or Microsoft Remote Desktop.app (Mac OS). The credentials for the server are user/password = Administrator/pqysjWG9V(F.

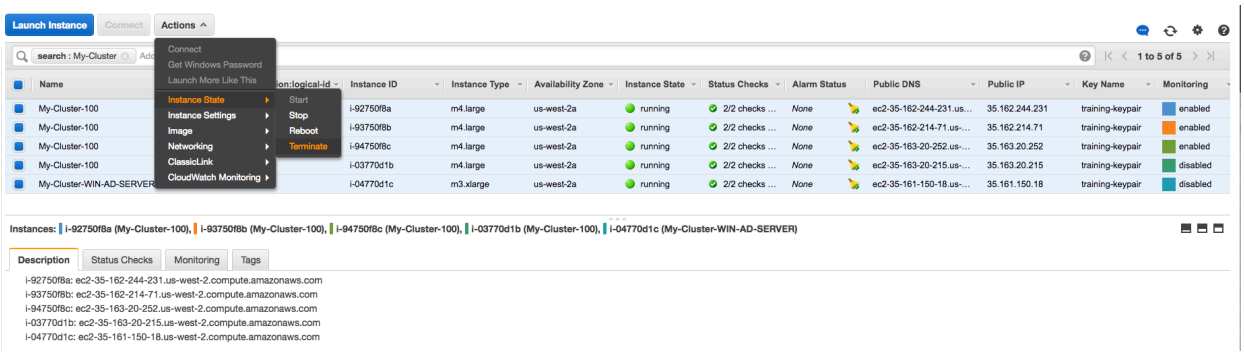
4.3 To access the Cross-realm MIT-KDC box, use the training-keypair to ssh to it as described in 4.1 above.

Shut Down Clusters after class completion

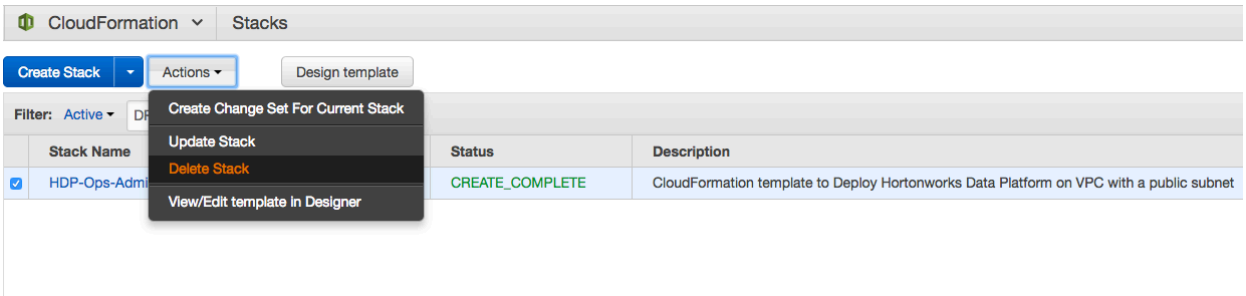
Step 5: How to shut down the clusters on the last day after the end of class.

5.1 Normally you should terminate instances via the EC2 console and selecting "Terminate." However, these clusters were created with AWS CloudFormation, which has a feature to "RELAUNCH" terminated/stopped instances. To avoid the automatic re-launching of instances you should perform the following steps.

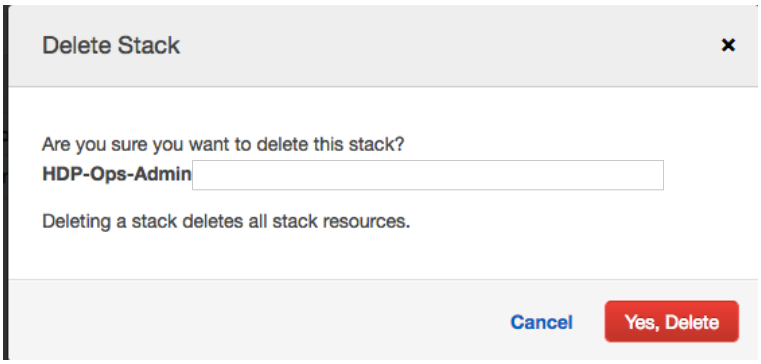
5.2 In the EC2 Console select all the nodes from all the class clusters to be deleted and "Terminate" them:



5.3 <u>In addition to the above</u>, go to the CloudFormation Console and select the clusters to be deleted:



5.4 Confirm delete. After a few minutes the Clusters will be gone from both AWS and CloudFormation Consoles.



Classes Available Worldwide Through Our Partners

Study Options Worldwide

In combination with our partner providers, classes are often available in numerous locations across the world.

Private On-site Training

Hortonworks training in-house covers all of our basic coursework, and provides a more intimate setting for 6 or more students.
Contact us for more details