# Distributed Data Platform on Microsoft Azure Virtual Machines

This framework will automate many of the steps for creating a distributed data cluster on Microsoft Azure Virtual Machines.

This framework will automate the generation of the Azure based infrastructure, and walk through the manual steps to configure the environment. The framework is a blend of Azure CLI, bash scripts coupled with manual configuration.

The following instructions were tested on Ubuntu 12.04.3 LTS, Oracle Linux 6 and CentOS 6.3/6.5.

This document will be used by those who prefer using Linux or Mac for deploying Windows Azure Linux VMs.

# Overview

1. Plan the size, configuration, and naming standards for your system. Update the clustersetup.sh file with information about your cluster
2. Update clustersetup.sh with cluster settings
3. Most clusters or farms have one node that is used to manage the other nodes, we will refer to that as the “management node” throughout this document. Execute createmgmtnode.sh
   1. Create the Affinity Group (if it doesn’t exist)
   2. Create the Virtual Network (if it doesn’t exist)
   3. Create the Storage Accounts (if they don’t exist)
   4. Create the Management virtual machine
4. Execute createcloneimage.sh. This step is only necessary if you are creating a new customer image for your nodes.
   1. Create the Clone Node
5. Manually configure the Management
   1. Set root passwords
   2. Set up passwordless SSH between the Management Node and the Clone Node
   3. Set various server configurations to meet software requirements
   4. Add disk mount script
6. Deprovision the Clone node to enable it for use as an image.
   1. Update waagent.conf
   2. Run waagent –deprovision
7. Execute capturecloneimage.sh
   1. Create the Windows Azure Clone Image
8. Execute createclusternode.sh
   1. Creates multiple Windows Azure Virtual Machines using the Clone Node image
   2. Creates the script hosts.txt to update etc/hosts file on each node in the cluster
   3. Creates the script hostscript.sh which mounts data drives and updates the hosts files on each node in the cluster
9. Restart the Management Node
10. Update hosts and mount drives on each node
11. Install management and installation software on Management node (ie Hortonworks HDP, Cloudera Manager, etc)
12. Install software on cluster

# Dependencies

* Windows Azure Cross Platform Command Line Tools: <http://www.windowsazure.com/en-us/documentation/articles/xplat-cli/>
* SSH client.
* Windows Azure Subscription. Steps below assist with creating the storage account and containers.

# Planning

Determine the cluster requirements before executing the scripts:

1. How many nodes in the cluster
2. Virtual machine size
3. Number of cloud services
4. Number of storage accounts
5. Naming conventions for cloud services, storage accounts and virtual machines
6. Virtual network configuration

If needed, update the subscription quotas to support the cluster configuration virtual machine cores, storage accounts, cloud services, etc: <http://azure.microsoft.com/en-us/support/options/>

# Preparation

## Windows Azure Cross Platform Command Line Tools

This section provides steps for setting up your development PC to leverage Windows Azure command line tools for deployments.

### MAC OSX Mavericks

For Mac you need to install brew. It can be installed by running the command shown below from the terminal window. Detailed instructions are at <http://brew.sh/>

Install node by executing brew install node

Install Azure CLI by executing npm install –g azure-cli

Test that Azure CLI was installed by executing azure –v

You should see version 0.7.4 or greater

ruby -e "$(curl -fsSL https://raw.github.com/Homebrew/homebrew/go/install)"

brew update

brew doctor

brew install node

npm install -g azure-cli

azure –v

brew install jq

Jq is a light weight command line json processor available for many platform.

### Ubuntu

Here are the instructions for install Azure CLI on Ubuntu. For a different flavor of Linux, some of the set up commands will need the corresponding changes.

## Install node.js

|  |
| --- |
| sudo apt-get install -y python-software-properties python g++ make  sudo add-apt-repository ppa:chris-lea/node.js  sudo apt-get update  sudo apt-get install nodejs |

For reference, see

<http://askubuntu.com/questions/49390/how-do-i-install-the-latest-version-of-node-js>

[Installing-Node.js-via-package-manager](https://github.com/joyent/node/wiki/Installing-Node.js-via-package-manager).

## Install CLI

|  |
| --- |
| sudo npm install -g azure-cli  # check version to ensure CLI is properly installed  azure -v # should be 0.7.4 or greater |

Once Azure CLI has been installed the rest of the instructions are same for MAC and Linux.

## Authenticate using Certificate

The development machine will use a certificate (self-signed) to authenticate against Windows Azure Management services. For details, see [How to install the Windows Azure Cross-Platform Command-Line Interface](http://www.windowsazure.com/en-us/documentation/articles/xplat-cli/#install).

|  |
| --- |
| azure account download |

A browser will pop up and prompt for login. After a successful login, you will be prompted to download and save a file with file name extension .publishsettings. This publish-settings file contains the certificate with private key, and will be used in the next step. Please note that a new management certificate is automatically created using this step. The details of this newly created certificate can also be viewed in the portal under Settings/Management Certificates section.

If the development machine is a server without a UI (user interface), the URL will be displayed on the Unix prompt along with an informational message; see below.

|  |
| --- |
| $ azure account download  info: Executing command account download  info: Launching browser to http://go.microsoft.com/fwlink/?LinkId=254432  help: Save the downloaded file, then execute the command  help: account import <file>  info: account download command OK |

Copy the URL and use and go to a different machine which does have a browser enabled. Login using a browser and download the publish-settings file. In this case, the publish-settings file will have to be copied over to the development machine before proceeding to the next step.

|  |
| --- |
| azure account import <path-to-publish-settings-file> |

If everything goes through well up to this point, you should see a hidden folder .azure in your current working directory on the development machine. This directory should contain a certificate in .pem format and other files for connecting to your Azure subscription. For example:

|  |
| --- |
| ls -la .azure  total 20  drwxrw-r-- 2 azureuser azureuser 4096 Jan 29 21:26 ./  drwxr-xr-x 8 azureuser azureuser 4096 Jan 29 21:26 ../  -rw-rw-r-- 1 azureuser azureuser 105 Jan 29 21:26 config.json  -rw------- 1 azureuser azureuser 2740 Jan 29 21:26 managementCertificate.pem  -rw------- 1 azureuser azureuser 3962 Jan 29 21:26 publishSettings.xml |

If you logon to the Windows Azure Management Portal using a browser, and navigate to Settings 🡪 Management Certificates page, you should see a certificate (with public key) already installed there. At this point the development machine is configured for interacting with the Windows Azure Subscription.

To verify that connectivity has been established, run the command shown below to list the Account/Subscription that will be accessible from this machine.

|  |
| --- |
| azure account list |

# Distributed Data Platform on Microsoft Azure Virtual Machines

## Cluster Configuration: Update clustersetup.sh

### Development PC

Get all the scripts and files from GitHub <https://github.com/devopscloudorg/azure-cdh/tree/master/bash>

You will need to edit the clustersetup.sh in a text editor. This file contains all the settings necessary to create a cluster. The following section provides details for each variable in clustersetup.sh.

### Affinity Group

An Affinity Group will help you deploy your compute and storage account together. An example of affinity group settings are as follows. Note that the affinityGroupName must be globally unique.

#Affinty group helps you keep your storage and compute in the same region

#Identify the region where affinity group should be created.

#choices are valid values are "East US", "West US", "East Asia", "Southeast Asia", "North Europe", "West Europe"

export affinityGroupName=ddpWest

export affinityGroupLocation="West US"

export affinityGroupLabel="AG for DDP Sample"

export affinityGroupDescription="Affinity for DDP Sample"

### Virtual Network

Virtual machines in a cluster need to communicate with each other. We recommend putting all the virtual machines in the same virtual network. Virtual network can have one of more subnets.

You can learn more about Azure Virtual Network online: <http://www.windowsazure.com/en-us/documentation/services/virtual-network/>

If the virtual network already exists, input the existing settings in the configuration file. If the virtual network does not exist, it will be created based on these settings.

#setting related to virtual network

#address space allows 192.168.0.0, 10.0.0.0 and 172.16.0.0 ip address ranges

#virtual network faq: http://msdn.microsoft.com/en-us/library/windowsazure/dn133803.aspx#setting related to

export vnetName=ddp01

export vnetAddressSpace=172.16.0.0

export vnetCidr=17

export subnetName=App

export subnetAddressSpace=172.16.1.0

export subnetCidr=17

### Azure Storage Account

The storageAccountName is the primary storage account used for the data disks and OS disks of the Management node, and the OS disks of the cluster nodes. The storage accounts listed under storageAccountList are used to store the data disks for the cluster nodes.

All storage account names must be lower case and globally unique.

This example will create a total of three storage accounts: The primary storage account named ddp0 and two additional storage accounts ddp1 and ddp2.

#storage account settings

#name of the primary storage account for the management node, images, and data node OS disks.

#list the array of storage accounts to store the data disks for the cluster nodes

export storageAccountName=ddp0

export storageAccountList=(ddp1 ddp2)

### Virtual Machines

The virtual machine settings will drive the configuration of the Management node, Clone node and some aspects of the Cluster nodes. All virtual machines in the cluster will have the same adminUserName and adminPassword.

The vmNamePrefix will drive the name of the Management node (vmNamePrefix + 0) and the Clone node (vmNamePrefix + c). Different naming conventions may be used for the cluster nodes based on settings later in the ClusterConfig.xml.

The cloudServicePrefix will drive the name of the cloud service for both the Management node and the Clone node. The cloudServicePrefix must be globally unique.

#cloud service settings

#Prefix for all cloud services. This will also be used as the name of the primary cloud service.

export cloudServicePrefix=

#virtual machine settings

export vmNamePrefix=

export adminUserName=

export adminPassword=

### Hosts Files

Two files are generated during the execution of the process: hosts.txt and hostscript.sh. The hosts.txt is updated with the host name and IP address of the Management node and all cluster nodes. The hostscript.sh file writes commands that will update all hosts in the cluster and create the filesystem on each virtual machine. By default, these files are generated in the same location as the scripts.

#This script will be generated and it will be used to mount data drives in each node in the cluster. It will also copy /etc/hosts file to each node

mntscript="hostscript.sh"

#This file will generate hosts file that can be appended to /etc/hosts on each node.

hostsfile="hosts.txt"

### Management Node and Clone Node

All Management node and Clone node specific settings are in the Management Node and Clone Node section.

Cluster management software is installed on the Management node (sometimes referred to as an Operations node or Edge node by OSS partners). The Clone node is the master image that will be used to stamp out virtual machines later in the process.

The Management node and Clone node are created based on a gallery image. Windows Azure provides virtual machine images that are supported by Microsoft or other vendors. If you need to get a list of which images are available you can run the command:

Azure vm image list

Valid instanceSize settings are available online: <http://msdn.microsoft.com/en-us/library/azure/dn197896.aspx>. Most distributed data platform software will require a minimum virtual machine size of ExtraLarge or larger.

The installerport will create an endpoint used by the OSS software.

Specify the number of disks and size of disks to attach to the Management node. Disks are not attached to the Clone node, as they are attached later in the process after the cluster nodes are generated.

#Name of the image you will use to create your management node and clone node virtual machines

export galleryimageName=5112500ae3b842c8b9c604889f8753c3\_\_OpenLogic-CentOS-65-20140415

#Size of the Virtual machine. Valid sizes are extrasmall, small, medium, large, extralarge, a5, a6, a7

#we recommend extra large or higher

export instanceSize=A7

#endpoint port to open for software installers on the Management Node (ie Amabari, Cloudera Manager)

export installerport=8080

#Size of the data disk you want to attach to the Management Node. You will typically attach at least 1 disk.

export diskSizeInGB=500

export numOfDisks=2

### Clone Image

After the Management node and the Clone node have been updated with the proper settings to optimize for the distributed data platform software, the Clone node is stopped and captured as an image. The name of the image is based on the value cloneimageName, and cloneimageLabel is a descriptive label for the image.

#Name and label of the custom image you will use to create your cluster nodes

export cloneImageName=ddpc

export cloneImageLabel=”DDP Clone”

### Cluster Nodes

The cluster nodes are generated from the Clone image.

Total nodes (nodeCount) is the total number of virtual machines to create for all roles in the cluster. How these nodes are used in the cluster will be determined when the OSS software is installed. For example, decisions on which virtual machines will serve as the name nodes in Hadoop are made when installing Hadoop. In this example, the nodeCount value should include the name node(s) + data nodes needed for the cluster.

Valid instanceSize settings are available online: <http://msdn.microsoft.com/en-us/library/azure/dn197896.aspx>. Most distributed data platform software will require a minimum virtual machine size of ExtraLarge.

Specify the number of disks and size of disks to attach to the Cluster nodes. The filesystem will be configured after the Cluster nodes are created.

All cluster nodes are created with a naming convention based on the vmNamePrefix followed by a number between 1 and the value in numNodes. All cluster nodes will be created in cloud services that are named based on a naming convention based on cloudServicePrefix followed by a number between 1 and the value in numCloudServices.

#These settings are for nodes in the cluster

#Number of nodes in your cluster

export nodeCount=10

#Number of cloud services to create for the cluster nodes. One additional cloud service is created for the management node and clone image.

export numCloudServices=2

#Size of the nodes in the cluster. Valid sizes are extrasmall, small, medium, large, extralarge, a5, a6, a7

export clusterinstanceSize=A7

#Size of the data disk you want to attach to the VM you are creating. You will typically attach at least 1 disk

#Number of disks you want to attach. Small VM can have 2 disks, medium can have 4, large can have 8 and extralarge can have 8 data disks

export clusterdiskSizeInGB=1000

export clusternumOfDisks=4

export clustervmNamePrefix=ddp

export clustercloudServicePrefix=ddp

## Create the Management Node and Clone Node

From your development PC run the bash scripts createmgmtnode.sh and createclonenode.sh

These scripts will use the settings defined in clustersetup.sh to create your virtual machines for the Management node and Clone node.

#### Sample Execution Script

#On your Development PC

createmgmtnode.sh

createclonenode.sh

## Manually configure the Management and Clone Node

Verify the two virtual machines have completed provisioning and are running.

The Management Node is named the value supplied in the clustersetup setting vmNamePrefix followed by 0, and the Clone Node is named the value supplied in the setting vmNamePrefix followed by “c”.

In addition to the DNS name, you may need the SSH public endpoint, private IPs, and public IPs of the Management node and the Clone node to proceed with this section. The virtual machine IP addresses can be found by viewing the properties of the virtual machine in the [Microsoft Azure Management Portal](https://manage.windowsazure.com/). The SSH endpoint is available by select ENDPOINTS in the upper menu of the virtual machine.

Use your SSH client to log into the management node.

**ssh <adminUserName>@<Cloud Service Name>.cloudapp.net –p <Management Node SSH port>**

You will be prompted for password. Enter the value of adminPassword defined in clustersetup.sh file.

### Set root passwords

Set the root passwords on both the Management node and Clone node. You are currently connected to the Management node.

Elevate to root. Enter the password for the adminUserName when prompted. Update the root password with the command **passwd**. Enter the new password when prompted.

**sudo -s**

**passwd**

SSH to the Clone Node. Enter the password (adminPassword) for the adminUserName when prompted.

**ssh <adminUserName>@<Cloud Service Name>.cloudapp.net –p <Clone SSH port>**

Update the root password. Enter the new password when prompted.

**sudo -s**

**passwd**

With both root passwords set, the next step will configure the nodes for the cluster.

### Set up passwordless SSH between the Management Node and the Clone Node

In the Management node generate the key. This must be done under the context of root.

**ssh <Cloud Service Name>.cloudapp.net -p <Management SSH port>**

**ssh-keygen**

Accept the default file location when prompted (press enter). Press enter (twice) to create the key without a passphrase. The public key is stored in .ssh/id\_rsa.pub, and the private key is id\_rsa.

Copy the key to the Clone node and Management node (self-referencing), enter the root password when prompted. The following example connects to the nodes with DNS. The screenshot demonstrates connecting to the node with the private IP.

**ssh-copy-id -i /root/.ssh/id\_rsa.pub "root@<Cloud Service Name>.cloudapp.net -p <Clone SSH Port>"**

**ssh-copy-id -i /root/.ssh/id\_rsa.pub "root@<Cloud Service Name>.cloudapp.net -p <Management SSH Port>"**



To test the keys were set up correctly, type the following and validate that you are not prompted for a password.

**ssh <Cloud Service Name>.cloudapp.net -p <Management SSH Port>**

**ssh <Cloud Service Name>.cloudapp.net -p <Clone SSH Port>**

Note that the passwordless ssh is not reciprocal. You will be prompted for a password when you are returning to the Management Node from the Clone Node.

Return to the Management node to configure the final settings.

**ssh <Cloud Service Name.cloudapp.net -p <Management SSH Port>**

### Mount Disks Script

The script makefilesystem.sh shell script will create partitions and filesystems for the cluster nodes. This script will be executed later in the process. This step will load the script to the virtual machines.

Copy the makefilesystem.sh script from your development PC to both the Management node and Clone node. You can use scp to copy this file as shown below.

**scp –P <Management SSH Port> makefilesystem.sh**  [**root@<Cloud Service Name>.cloudapp.net:**](mailto:root@management_node_hostname.cloudapp.net:)**makefilesystem.sh**

### Update Server Configuration Settings

SSH to the Management node and Clone node to configure the nodes. Execute the following to move and secure the makefilesystem.sh script, and install and use dos2unix to convert makefilesystem.sh file to a Linux friendly format. The final steps will configure prerequisite server settings.

1. Connect to the Management node **as root**. Execute the following commands on the management node. Choose the block based on the node’s Linux version.

**CentOS, Oracle**

**mkdir /root/scripts**

**mv makefilesystem.sh /root/scripts/makefilesystem.sh**

**chmod 755 /root/scripts/makefilesystem.sh**

**yum -y install dos2unix**

**dos2unix /root/scripts/makefilesystem.sh /root/scripts/makefilesystem.sh**

**#disable iptables**

**chkconfig iptables off**

**/etc/init.d/iptables stop**

**setenforce 0**

**#start ntp service**

**yum -y install ntp**

**chkconfig ntpd on**

**ntpdate pool.ntp.org**

**echo 'vm.swappiness = 0' >> /etc/sysctl.conf**

**sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config**

**echo never > /sys/kernel/mm/transparent\_hugepage/enabled**

**echo never > /sys/kernel/mm/redhat\_transparent\_hugepage/defrag**

**echo "echo never > /sys/kernel/mm/transparent\_hugepage/enabled">>/etc/rc.local**

**echo "echo never > /sys/kernel/mm/redhat\_transparent\_hugepage/defrag">>/etc/rc.local**

**Ubuntu**

**mkdir /root/scripts**

**mv makefilesystem.sh /root/scripts/makefilesystem.sh**

**chmod 755 /root/scripts/makefilesystem.sh**

**apt-get -y install dos2unix**

**dos2unix /root/scripts/makefilesystem.sh /root/scripts/makefilesystem.sh**

**#disable firewall**

**ufw disable**

**#start ntp service**

**apt-get -y install ntp**

**chkconfig ntpd on**

**ntpdate pool.ntp.org**

**echo 'vm.swappiness = 0' >> /etc/sysctl.conf**

**sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config**

**echo never > /sys/kernel/mm/transparent\_hugepage/enabled**

**echo never > /sys/kernel/mm/redhat\_transparent\_hugepage/defrag**

**echo "echo never > /sys/kernel/mm/transparent\_hugepage/enabled">>/etc/rc.local**

**echo "echo never > /sys/kernel/mm/redhat\_transparent\_hugepage/defrag">>/etc/rc.local**

1. SSH to the Clone node **as root** and repeat the previous step.
2. Restart the Management node to disable SELINUX, which is required for software installation (ie Ambari and Cloudera Manager).

Verify that the directory scripts exists in root and makefilesystem.sh is the only file in the directory.

Review the output from the scripts to verify all steps succeeded.

### Deprovision Clone Node With Windows Azure Linux Agent

SSH to the Clone Node as root. **This will not be run on the Management Node!** Set up the virtual machine for provisioning as an image. Open the waagent.conf file.

**vi /etc/waagent.conf**

If you are new to vi, type i to enter insert mode. Navigate to the correct line in the file and update. Press Esc to exit insert mode, type :wq to save changes and return to the command prompt.

Change the following settings then exit vi:

**Provisioning.DeleteRootPassword=n**

**Provisioning.RegenerateSshHostKeyPair=n**

Run the Windows Azure Linux Agent.

**waagent -deprovision**

Return to the Management Node as root, entering the password when prompted.

**ssh <Cloud Service Name>.cloudapp.net –p <Management SSH Port>**

## Capture the Image

From your development PC run the bash script capturecloneimage.sh.

This script will use the settings defined in clustersetup.sh to capture the Clone virtual machine as an image. Upon successful completion this script will display detailed information about the Clone image.

#### Sample Execution Script

#On your Development PC

capturecloneimage.sh

## Create the Cluster

On your development PC run the bash script createclusternodes.sh to create the virtual machines for the cluster. On completion, it will create the virtual machines and the hosts.txt and hostscript.sh files needed in the next step.

#### Sample Execution Script

#On your dev PC

createclusternodes.sh

## Update Cluster Machines

After the createclusternodes.sh has completed SCP the files clustersetup.sh, updatehosts.sh, hosts.txt and hostscript.sh to the Management node. Secure the hosts.txt and hostscript.sh files.

From the Management node, execute updatehosts.sh to update /etc/hosts with the values in hosts.txt.

From the Management node execute hostscript.sh. This script executes makefilesystem.sh on each node in the cluster to mount the data drives and update /etc/hosts.

#### Sample Execution Script

#On your dev PC

scp –P <Management SSH Port> hosts.txt root@<Cloud Service Name>.cloudapp.net:hosts.txt

scp –P <Management SSH Port> hostscript.sh root@<Cloud Service Name>.cloudapp.net:hostscript.sh

scp –P <Management SSH Port> updatehosts.sh root@<Cloud Service Name>.cloudapp.net:updatehosts.sh

scp –P <Management SSH Port> clustersetup.sh root@<Cloud Service Name>.cloudapp.net:clustersetup.sh

#On your management node logged in as root

chmod 755 hostscript.sh

chmod 755 updatehosts.sh

#Reads the hosts.txt and updates /etc/hosts file on the management node

updateHosts.sh

#Update the hosts file on each node in the cluster and mounts the data drives

hostscript.sh

At the completion of the script a set of virtual machines is ready for the software installation.

To test that the virtual machines were generated correctly, connect to the Management node and SSH to a few of the machines to verify the host names were set and to verify passwordless ssh is working:

**ssh <hostname>**

# Appendix A: Installing Ambari

Open an SSH session to the Management Node.

If you are using the gallery image “Oracle Linux 6.4.0.0.0” install wget.

**yum install wget**

Download and install Ambari:

**wget http://public-repo-1.hortonworks.com/ambari/centos6/1.x/updates/1.4.2.104/ambari.repo**

**cp ambari.repo /etc/yum.repos.d**

**yum install ambari-server**

After the installation has completed, run the setup:

**ambari-server setup**

Verify Ambari is started. Execute the following command on the Management Node:

**ambari-server start**

## Install HDP

Open the browser and navigate to http://<Management Node Hostname>:8080

When prompted, log in as admin with password admin.

In general, follow the prompts to install. The only tricky part I would point out is in the install screen to input hosts and the key. You will paste the private key from the Management Node:

**cat .ssh/id\_rsa**

Highlight the full key and press enter to copy to the clipboard. You will then paste this in the web interface.

# Appendix B: Installing Cloudera Manager and CDH

This is a summary of the instructions from Cloudera’s online documentation: <http://www.cloudera.com/content/cloudera-content/cloudera-docs/CM4Ent/latest/Cloudera-Manager-Installation-Guide/cmig_install_path_A.html?scroll=cmig_topic_6_5>

Open an SSH session in PuTTY to the Management Node using root.

If you are using the gallery image “Oracle Linux 6.4.0.0.0” install wget.

**yum install wget**

Download and install Cloudera Manager. The following will install the latest version of Cloudera Manager 5.x. Check with Cloudera’s website for other versions

**wget http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin**

**chmod u+x cloudera-manager-installer.bin**

**./cloudera-manager-installer.bin**

From Windows Azure Management Portal, navigate to the management node virtual machine and select Endpoints. Add an endpoint for installation software (7180 for Cloudera) if it wasn’t added in the earlier scripts.

Open the browser and navigate to http://<cloud service name for management node>.cloudapp.net:7180

When prompted, log in to Cloudera Manager in the browser as admin with password admin. You can change the password after the installation is complete, but you cannot rename admin.

In general, follow the prompts to install. You will use the hostname, not the full <cloudservice of management node>.cloudapp.net, for the list of hosts. For example for a cluster with machines following a naming convention of hadoopvm1, hadoopvm2… hadoopvm7 you can input hadoopvm[0-7].

Once you confirm that the cluster is installed, configured, and running get the system production ready.

Enable iptables for Firewall protection

# /etc/init.d/iptables start #http://www.cyberciti.biz/faq/turn-on-turn-off-firewall-in-linux/

? enable selinux??