neonCLUSTER Node Template VHDX

for Ubuntu 16.04 LTS

Introduction

This document details the steps required to configure a Template VHDX suitable for building neonCLUSTERs for development, testing, staging, and production.

VHDX Template Creation

Follow these instructions to create an Ubuntu-16.04 VHDX that can be used to quickly instantiate new Ubuntu VMs for development and test or to use to generate a PXE boot image for staging and production. Note that these steps will only be **rarely necessary** to regenerate the VHDX from scratch. Most of the time, you’ll be able to quickly **clone the pre-built VHDX** downloaded from Amazon S3 from (where **#** is the Ubuntu revision):

<https://s3.amazonaws.com/neonforge/neoncluster/ubuntu-16.04.#-prep.vhdx.zip>

or the latest version from:  
  
 <https://s3.amazonaws.com/neonforge/neoncluster/ubuntu-16.04.latest-prep.vhdx.zip>

**Setup Instructions**

1. Download the **Ubuntu-16.04 Server ISO** from (where **#** is the desired revision):   
     
   <https://s3.amazonaws.com/neonforge/neoncluster/ubuntu-16.04.#-server-amd64.iso>
2. Open the **Hyper-V Manager** and step through the **New Virtual Machine Wizard**:  
   1. Name the VM **ubuntu-16.04-prep**.
   2. Configure **Generation 1**Note: I tried using generation 2 but I was unable to clone VMs by making copies of the template VHD when I did this. The cloned VMs wouldn’t boot.
   3. Then **1024MB RAM**.
   4. Set the networking connection to a switch with external access (e.g. **Intel® Ethernet Connection (2) I218- Virtual Switch** or **Default Switch)**
   5. Create a **127GB VHD** (this seems like a reasonable size for test development but can be customized). Locate this in a known directory where you’ll be able to find it. Name the disk:  
        
      **ubuntu-16.04-prep.vhdx**
   6. Installation Options: Configure to **boot** from the downloaded **Ubuntu ISO** and then press **Next/Finish**.
   7. Select the new VM in the Hyper-V manager, select settings, click **Processor** in the left panel and set **4 Virtual Processors**.
   8. Press **OK** to close Settings.
   9. Optional: Recent versions of Windows 10 (as of August 2017) configure new VMs to **automatically checkpoint** their virtual hard drives (which is annoying). You can **disable this** by executing the following command (replacing **VM-NAME** with the target VM):  
        
      powershell Set-VM -CheckpointType Disabled -AutomaticCheckpointsEnabled 0 -Name VM-NAME
3. **Start the VM** and then **double-click** to **connect** via the Hyper-V Manager.
4. Press enter to select **English**.
5. Press enter to **Install Ubuntu Server**.
6. Press enter twice to select **English** and **United States**.
7. Press enter three times to select **English (US) keyboard**.
8. A bunch of stuff will be installed.
9. Make sure the host name is **ubuntu** and **Enter**.  
     
   \*\* **Do not change this** to avoid breaking subsequent scripts.
10. Create the **neon** **account:**Full Name: **sysadmin**  
    UID: **sysadmin**

PWD: **sysadmin0000**Note: These are the DEV credentials, use **secure credentials for PROD**.

1. **Do not encrypt the home directory**: Press **Enter** to skip.
2. Select **Yes** and configure the default **Time Zone** (a later script will change this to UTC).
3. Press **Up-Arrow** and then **Enter** to select **Guided – use entire disk** (**do not setup LVM!**).
4. Press **Enter** to accept the device changes.
5. Press **TAB** and **Enter** to select **Yes** to **Write the changes to disks** andconfigure **standard partitions** (not LVM).
6. Wait for the system to install.
7. Press Enter to **skip proxy** configuration.
8. More software will be installed.
9. Press **Enter** to select **No automatic updates** (I figure we’ll want to control when upgrades happen).
10. Check **OpenSSH Server** in the **Software selection screen** by pressing **Space** and then press **Enter** to continue.
11. Even more software is installed.
12. Press **Enter** to Install the **GRUB** boot loader.
13. Press **Enter** to **Reboot**.
14. **Login** with the credentials you specified earlier to verify that the VM works.
15. Optional: Use the command below to discover the VM’s **IP address** for the **eth0** interface and connect via PuTTY or another terminal program that allows for easy copy and pasting of commands. Then connect to the server.

ifconfig

1. Run this command to start bash with root permissions:  
     
   sudo bash
2. Run the following command to modify **sudo** behavior so it doesn’t request passwords, making remote configuration possible:  
     
   echo "%sudo ALL=NOPASSWD: ALL" > /etc/sudoers.d/nopasswd

**Restart** the VM and log back in. Then run the following command to verify that **sudo** no longer requests a password.  
  
sudo bash

1. Run the following commands to install the ZIP package:  
     
   apt-get update  
   apt-get install -yq zip
2. Install the Hardware Enablement (HWE) kernel and related daemons to support better integration with virtualization:  
     
   apt-get install -yq linux-virtual-lts-xenial linux-tools-virtual-lts-xenial linux-cloud-tools-virtual-lts-xenial
3. Optional (but **not recommended**): Install any upgraded packages (I generally don’t do this here and upgrade during cluster provisioning instead):  
     
   apt-get upgrade -yq
4. Run the following command to clear the **cached DHCP** leases and shutdown:  
     
   rm -rf /var/lib/dhcp/\*  
   shutdown -h now
5. ZIP the disk image and then use the AWS Console to **Upload** the image to the location below (where **#** is the revision) and grant **public read access**. Note that you need to ZIP the image using the **neon-cli** because the Windows file system ZIP feature uses Deflate64 compression which is not compatible with **neon-cli** (and other tools). Be patient; zipping is very slow:

neon zip create PATH-TO-VHDX PATH-TO-ZIP

Then upload the ZIP file to:  
  
<https://s3.amazonaws.com/neonforge/neoncluster/ubuntu-16.04.#-prep.vhdx.zip>

and if this is the latest version, to:  
  
<https://s3.amazonaws.com/neonforge/neoncluster/ubuntu-16.04.latest-prep.vhdx.zip>  
  
**Be sure to make these files public on AWS!**