neonCLUSTER Node Preparation

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** uploaded to Amazon S3 from:

<https://s3.amazonaws.com/neon-research/images/ubuntu-16.04-prep.vhdx.zip>

**Setup Instructions**

1. Download the **Ubuntu-16.04 Server ISO** from:   
     
   <https://s3.amazonaws.com/neon-research/images/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. Networking connection = **Intel® Ethernet Connection (2) I218- Virtual 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.
3. **Start the VM** and then **connect** to it 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. Press **Enter** to accept the default hostname **ubuntu**.  
     
   \*\* **Do not change this** to avoid breaking subsequent scripts.
10. Create the **admin** **account:**Full Name: **spot**  
    UID: **spot**

PWD: **WagTheDog!**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. Run this command to start bash with root permissions:  
      
    sudo bash
16. Run the following command to modify **sudo** behavior so it doesn’t request passwords to make 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. Ubuntu configures unattended security upgrades by default. We’re going to disable this capability for the base image since we’ll want to manage upgrades ourselves in production. Run this command:  
     
   apt-get remove -yq unattended\_upgrades
2. Run the following commands to install the ZIP utility:  
     
   apt-get update  
   apt-get install -yq zip
3. Use the command below to discover the VM’s **IP address** for the **eth0** interface and make a note of it for the next step:

ifconfig

1. Run the following command to clear the **cached DHCP** leases:  
     
   rm /var/lib/dhcp/\*
2. Optional: **Shutdown** the VM down and make a copy VHDX file as **ubuntu-16.04-clean.vhdx**. This file can be used as a short-cut when debugging or modifying the neon prep-node command implementation. **Restart** the VM afterwards and continue with the steps below.
3. On your Windows workstation, run the neon-cli command below to prepare the NeonCloud template VM. You’ll need to pass the IP address from the previous step and you’ll be prompted for the admin credentials you specified above.  
     
   neon prepare <node-ip>

**Highly Recommended:** The step above creates an image that can run anywhere but will download Linux packages from the Internet. You can dramatically reduce external network traffic for large clusters or development clusters that are often regenerated by deploying an  
**apt-cache-ng** server on your local network. See **Ubuntu-16.04 neonCLUSTER Deploy.docx** for detailed instructions. Then add the –package-cache option to the prepare command:  
  
neon prepare <node-ip> --package-cache=http://<package-cache>:3142

1. Wait for the VM to stop.
2. ZIP the disk image and then use the AWS Console to **Upload** the image to the location below and grant **public read access**:  
     
   <https://s3.amazonaws.com/neon-research/images/ubuntu-16.04-prep.vhdx.zip>

# Debugging VHDX

To ease the creating and debugging of this template, a VHDX has been created with the base Ubuntu-16.04 image with **sudo** configured to not ask for passwords.

This VHDX can be obtained from:

<https://s3.amazonaws.com/neon-research/images/ubuntu-16.04-clean.vhdx.zip>

You can then complete the remaining configuration steps manually as described above or use the  
**neon-cli** tool:

neon -u=spot -p=WagTheDog! prepare <server-ip>