neonCLUSTER Deployment

on Ubuntu 16.04 LTS

# Overview

This document starts out by describing how the scripts and other assets located within this source folder are to be used to create neonCLUSTER development and production clusters on Ubuntu 16.04 LTS.

A [Docker Swarm](https://docs.docker.com/swarm/overview/) cluster is a collection of Linux host machines, each running a Docker Engine (in swarm mode), as well as potentially Consul, Vault, and TD-Agent. Docker (swarm mode) creates a virtual Docker Engine across all of the cluster hosts and handles the assignment, deployment, and management of containers across the cluster. Docker refers to this as container *scheduling*. Swarm scheduling is relatively sophisticated and provides for distributing containers across hosts for fault tolerance, relocating containers on failure, accounting for container/hardware as well as container/container affinity. Finally, Swarm can handle the deployment of applications that consist of multiple containers.

# Virtual Machine Templates

Prepared VHDX base images with Linux installed are available on Amazon S3 at:  
  
<https://s3-us-west-2.amazonaws.com/neonforge/neoncluster/ubuntu-16.04.3-prep.vhdx.zip>  
<https://s3-us-west-2.amazonaws.com/neonforge/neoncluster/ubuntu-16.04.latest-prep.vhdx.zip>

neon-cli can download and cache this image for when constructing a neonCLUSTER. Note that the initial credentials for these images are:

username: **sysadmin**  
password: **sysadmin0000**

# Creating a Development neonCLUSTER

1. Optional but **Highly Recommended**: Setup an **apt-cache-ng** server to cache Ubuntu/Debian software packages. This will dramatically reduce the amount of traffic on your network when installing and upgrading a cluster.  
     
   Here’s the detailed: [manual](https://www.unix-ag.uni-kl.de/~bloch/acng/html/)  
   1. Follow steps 4-6 above to create an Ubuntu VM for the cache. Configure **RAM=1024MB** and **CPU=4**.
   2. **SSH** into the server and configure its host name like:

sudo bash  
echo apt-cache > /etc/hostname

* 1. The service’s IP address must be statically assigned and/or be resolvable via DNS:  
     1. Edit /etc/network/interfaces to look something like this:  
          
        auto eth0

iface eth0 inet static

address 10.50.0.10

netmask 255.255.255.0

gateway 10.50.0.1

broadcast 10.50.0.255

* + 1. Edit /etc/resolvconf/resolv.conf.d/base to specify the name servers:  
         
       nameserver 8.8.8.8

nameserver 8.8.4.4

* 1. Install the **apt-get-cacher-ng** server ([manual](https://www.unix-ag.uni-kl.de/~bloch/acng/html/index.html)):  
     1. apt-get update  
        apt-get install -yq apt-cacher-ng
     2. We need to configure the cache to pass-thru SSL requests:  
        Edit **/etc/apt-cacher-ng/acng.conf** by running the following command:  
          
        echo "PassThroughPattern:^.\*:443$" >> /etc/apt-cacher-ng/acng.conf
  2. **Restart** the server to pick up the host name, network, and **apt-cacher-ng** configuration changes.  
       
     Note: You can manage the cache via: http://<apt-cache-server>:3142/acng-report.html
  3. Optional: **neon-cli** will automatically configure cluster nodes to use the cache if **PackageCache** is set in the cluster configuration file. You can also configure any other Ubuntu servers to use the cache by running the commands below (substituting the cache server’s IP address or FQDN):  
       
     echo 'Acquire::http::proxy "http://APT-CACHE-IP:3142";' > \  
      /etc/apt/apt.conf.d/02proxy

1. **Cluster Configuration**:  
     
   You’ll be using the neon-cli command line tool to configure your cluster. The latest release binary is located in the Neon source repository in the **~/Build** folder (which will be on the path if you followed the developer setup instructions in **~/Setup-Build.docx** document).  
   1. Create a Neon cluster definition file. This file is formatted as JSON, potentially augmented with comments, variables, and conditionals. Run the command below to obtain a sample cluster definition. The sample includes comments describing the file format and will be a good start for defining your own cluster.  
        
      neon sample
   2. Start all of your node VMs. Wait a couple minutes for the servers to start to avoid a package manager race condition. You’ll have to try again if one of your nodes fails to configure with a log message like:  
        
      E: Could not get lock /var/lib/dpkg/lock - open (11: Resource temporarily unavailable)

E: Unable to lock the administration directory (/var/lib/dpkg/), is another process using it?

* 1. Create a **folder** like C:\VM\log to hold setup log files.
  2. Prepare you cluster by creating the Hyper-V virtual machines using this command, passing the path to your cluster configuration file:  
       
     neon cluster prepare ^  
      -u=sysadmin ^  
      -p=sysadmin0000 ^  
      --log=c:\docker\neon.log ^  
      --max-parallel=4 ^  
      CLUSTER-DEF
  3. Setup your cluster via:  
       
     neon setup-cluster ^  
      --user=sysadmin ^  
      --password=sysadmin0000 ^  
      –-log=C:\VM\log ^  
      CLUSTER-DEF

You’ll be prompted for the user name and password if these aren’t provided as command line options.

* 1. The tool will provision the cluster nodes over the course of a few minutes. Once this is complete, you’ll want to verify that the cluster is functional.
  2. Open a DOS command window.  
     1. Run the command below to open an SSH connection using PuTTY to the first cluster manager:  
          
        neon ssh
     2. Run the following command in PuTTY to see what containers are running on the manager:  
          
        docker ps  
          
        and then this command and to discover the services running across the cluster:  
          
        docker service ls