# Using Vagrant to setup Scale Environments

**Christopher D. Maestas**
**Senior Architect – Spectrum Scale, IBM Systems**

IBM Storage & SDI

# Running Spectrum Scale in a Vagrant Environment

# Replicate a repeatable Scale environment

- Yes, we have a VM
- Stemmed from work to do an IBM Scale GUI Lab
  - Spin a VM with an RedHat based OS and kickstart file
  - Use install toolkit and latest version of Scale!
  - Tied to VMWare workstation

```
sudo genisoimage -U -r -v -T -J -joliet-long -V "CentOS 7 x86_64" -volset "CentOS-7.4" -A "CentOS-7.4" -b isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -eltorito-alt-boot -e images/efiboot.img -no-emul-boot -o ISONAME .
```

# What is vagrant and why??




Build and manage virtual machines on the fly

Plugins to configuration management utilities like:

ansible, chef , puppet, salt …

Scale runs anywhere but you need:

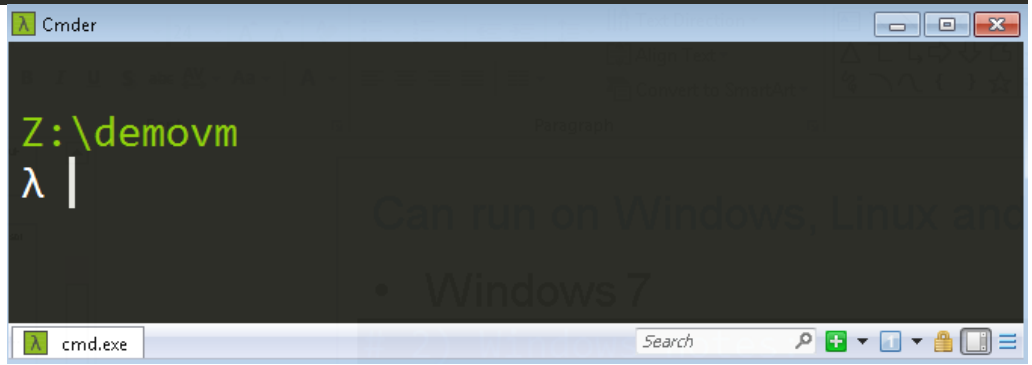
1. an OS installed
2. time and name resolution working
3. working network

# Can run on Windows, Linux and OS X

- Windows 7 needs powershell update

```
# 2) Windows notes:
#    * use cmder http://cmder.net/ (suggest Full version)
#    * Need powershell greater than 2.0
#        https://technet.microsoft.com/en-us/scriptcenter
#
```

- Cmder
  - Includes ssh
  - Bash/git …
- Linux and OS X are fine



# Tested Hypervisors

- Virtualbox
  - Runs the published Scale and Archive VMs today
  - Scale Vagrant files tested on Linux and Windows
- KVM/libvirt
  - No problems with RHEL7, some testing with RHEL6

```
# 3) Hypervisors - recommend VirtualBox
#   * tested Virtualbox for Win7/Win10 and Linux
#   + Linux has also been tested with libvirt
#   - Testing needs to be done for VMWare and Hyper-V
#      Basically need to know how to add an external disk and share it
#
```

# Tested Hypervisors

- VMWare
  - Working with Scale community
  - Have some initial prototypes for Fusion and Workstations

# Vagrant Mini-HowTo

- Everything starts with vagrant
  - To ssh: vagrant ssh VMNAME
  - To start: vagrant up
  - To halt: vagrant halt
  - To re-provision: vagrant destroy
- The main definition is in a file called
  - Vagrantfile – ruby syntax
- To cry or start from scratch: rm –fr $HOME/.vagrant.d

# Setup plugins and add default OS to use

- Certain plugins help with
  - Hosts file update
    - vagrant plugin install vagrant-hosts
- if using Virtualbox, run
  - vagrant plugin install vagrant-vbguest
- else if using libvirt, run
  - vagrant plugin install \ vagrant-libvirt
  - Sometimes trouble starting libvirt vms, so restart it
    - systemctl restart libvirtd

```
sh-4.2$ vagrant plugin list
vagrant-hosts (2.8.0)
vagrant-libvirt (0.0.43)
```

# Setup a local box to work from

- Select your hypervisor (recommend virtualbox or libvirt)
  - Add centos/7 vagrant box
    - vagrant box add centos/7
    - vagrant box list
- You should see centos/7 listed

```
sh-4.2$ vagrant box list
centos/7                (libvirt, 1802.01)
```

# Vagrant file - Clients and Protocol nodes



```
clients=2
(1..clients).each do |i|
  config.vm.define "scaleclients#{i}" do |scaleclients|
    scaleclients.vm.network "private_network", ip: "192.168.123.3#{i+2}"
    scaleclients.vm.synced_folder ".", "/vagrant", disabled: true
    scaleclients.vm.synced_folder "./root/", "/root/", owner: "root", group: "root"
    scaleclients.vm.provision :shell, path: "../../libexec/clientsprovision.sh"
  end
end
```

```
protoservers=2
(1..protoservers).each do |i|
  config.vm.define "scaleproto#{i}" do |scaleproto|
    scaleproto.vm.network "private_network", ip: "192.168.123.2#{i+2}"
    scaleproto.vm.synced_folder ".", "/vagrant", disabled: true
    scaleproto.vm.synced_folder "./root/", "/root/", owner: "root", group: "root"
    scaleproto.vm.provision :shell, path: "../../libexec/protoprovision.sh"
  end
end
```

Vagrantfile is Ruby code

# Vagrant file – libvirt SNC vs Shared



```
scalensd.vm.provider :libvirt do |libvirt, override|
  libvirt.storage :file, :size => '5G', :type => 'raw'
  libvirt.storage :file, :size => '5G', :type => 'raw'
end
```

```
scalesharednsd.vm.provider :libvirt do |libvirt, override|
  libvirt.storage :file, :size => '10G', :allow_existing => true, :path => 'sharednsd1.raw', :shareable => true, :type => 'raw'
  libvirt.storage :file, :size => '10G', :allow_existing => true, :path => 'sharednsd2.raw', :shareable => true, :type => 'raw'
end
```

# Shared libvirt vs Virtualbox



```
sharednsdservers=2
(1..sharednsdservers).each do |i|
  config.vm.define "scalesharednsd#{i}" do |scalesharednsd|
    scalesharednsd.vm.host_name = "scalesharednsd#{i}"
    scalesharednsd.vm.network "private_network", ip: "192.168.123.2#{i}"

    scalesharednsd.vm.provider :libvirt do |libvirt, override|
      libvirt.storage :file, :size => '10G', :allow_existing => true, :path => 'sharednsd1.raw', :shareable => true, :type =>
      libvirt.storage :file, :size => '10G', :allow_existing => true, :path => 'sharednsd2.raw', :shareable => true, :type =>
    end

    scalesharednsd.vm.provider :virtualbox do |vbox, override|
      port = 1
      sharednsdiskcontroller="NSDSataController"
      disks = [ "sharednsdiska.vdi", "sharedns
      disks.each do |disk|
        needsharedattach = "." + disk + "_need
        if not File.exists?(disk) or File.exists?(ne
          if not File.exists?(disk)
            vbox.customize ['createhd', '--filename', disk, '--Var          , 'Fixed', '--size', 10 * 1024]
            vbox.customize ['modifyhd', disk, '--type', 'shareable']
            if port == 1
              vbox.customize ['storagectl', :id, '--name', sharednsdiskcontroller, '--add', 'sata', '--portcount', disks.lengt
            end
            vbox.customize ['createhd', '--filename', needsharedattach, '--size', 1]
            vbox.customize ['storageattach', :id,  '--storagectl', sharednsdiskcontroller, '--port', port, '--device', 0, '--ty
          else
            if port == 1
              vbox.customize ['storagectl', :id, '--name', sharednsdiskcontroller, '--add', 'sata', '--portcount', disks.lengt
            end
            vbox.customize ['storageattach', :id,  '--storagectl', sharednsdiskcontroller, '--port', port, '--device', 0, '--ty
            vbox.customize ['closemedium', 'disk', needsharedattach, '--delete']
          end
        end
        port = port + 1
      end
    end
```

KVM VS Virtualbox

# Virtualbox SNC



```
scalensd.vm.provider :virtualbox do |vbox, override|
  port = 1
  nsdiskcontroller="NSDSataController"
  disks = [ "scalensd#{i}nsdiska.vdi", "scalensd#{i}nsdiskb.vdi" ]
  disks.each do |disk|
    if not File.exists?(disk)
      # create the controller on the first disk
      if port == 1
        vbox.customize ['storagectl', :id, '--name', nsdiskcontroller, '--add', 'sata', '--portcount', disks.leng
      end
      vbox.customize ['createhd', '--filename', disk, '--variant', 'Fixed', '--size', 5 * 1024]
      vbox.customize ['storageattach', :id,  '--storagectl', nsdiskcontroller, '--port', port, '--device', 0, '--
    end
    port = port + 1
  end
end
```

# Install a base box so you don't have to pull updates



```
#!/bin/bash

#set -x
OS=centos7.4
NAME=scalebaseos

read -e -p "Box Name: " -i "${OS}_$(date +%F)" BOXNAME

vagrant destroy -f
vagrant box update
vagrant up
vagrant halt
if [ -d /var/lib/libvirt/images/ ]; then
    if [ -f /var/lib/libvirt/images/scale_centos7base_scalebaseos.img ]; then
        sudo chmod a+r /var/lib/libvirt/images/scale_centos7base_scalebaseos.img
    fi
fi
vagrant package --output $BOXNAME
vagrant box add $BOXNAME $BOXNAME
vagrant destroy -f
rm -fr $BOXNAME
```

# Provision Scripts

Currently calling a shell script that uses the install toolkit

Points to a SCALESOURCE tree and extracts data

# Let’s demo

# Let’s demo

Coming soon GIT tree public

- vagrantbuild – sample Vagrant files for Scale
- cssdeployenv – install toolkit and runbooks

Integrate with Ansible from others

# Thank You.

IBM Storage & SDI