Getting Started With Infrastructure as Code

Tim Quinlan

Ohio LinuxFest 2019

Warning

This talk covers building a **virtual** on-premise lab on one largish machine using open source tools. The following content is for learning purposes only. It is not a traditional virtualization talk, rather it details using VMs to simulate -- rather than replace -- expensive hardware.

Things like nested virtualization will never ever perform well in production. Don’t do it.

Yes, I work for Red Hat, but this is a community presentation. What is covered here is not supported by Red Hat.

WTH is IaC?

Infrastructure as code (**IaC**) is the process of managing and provisioning computer data centers through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools.[1] The IT infrastructure managed by this comprises both physical equipment such as bare-metal servers as well as virtual machines and associated configuration resources. The definitions may be in a version control system. It can use either scripts or declarative definitions, rather than manual processes, but the term is more often used to promote declarative approaches.

IaC approaches are promoted for cloud computing, which is sometimes marketed as infrastructure as a service (IaaS). IaC supports IaaS, but should not be confused with it.

https://en.wikipedia.org/wiki/Infrastructure_as_code

WTH is IaC?

Infrastructure as code (**IaC**) is the process of managing and provisioning computer data centers through **machine-readable definition files,** rather than physical hardware configuration or interactive configuration tools.[1] The IT infrastructure managed by this comprises both physical equipment such as **bare-metal servers as well as virtual machines** and associated configuration resources. The definitions may be in a version control system**.** It can use either scripts or **declarative definitions, rather than manual processes**, but the term is more often used to promote declarative approaches.

IaC approaches are promoted for cloud computing, which is sometimes marketed as infrastructure as a service (IaaS). IaC supports IaaS, but should not be confused with it.

https://en.wikipedia.org/wiki/Infrastructure_as_code

WTH is IaC?

Infrastructure as code (**IaC**) is the process of managing and provisioning computer data centers through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools.[1] The IT infrastructure managed by this comprises both physical equipment such as bare-metal servers as well as virtual machines and associated configuration resources. The definitions may be in a version control system. It can use either scripts or declarative definitions, rather than manual processes, but the term is more often used to promote declarative approaches.

IaC approaches are **promoted for cloud computing**, which is sometimes marketed as infrastructure as a service (IaaS). **IaC supports IaaS, but should not be confused with it.**

https://en.wikipedia.org/wiki/Infrastructure_as_code

Yay! Cloud Computing*

* IaC is also great for HPC!

IaC vs IaaS vs BMaaS

Infrastructure as Code - Defining and controlling physical infrastructure with declarative templates

Infrastructure as a Service - Running cloud services on top of said physical servers, defining/controlling/scheduling cloud resources such as networks, containers and vms through APIs and declarative templates

Bare Metal as a Service - Controlling physical servers as user schedulable resources (We'll talk about this at the end)

Why IaC?

Flexibility

Scalability

Density

Commodity hardware

Lab Requirements

KVM Hypervisor(s*) with plenty of RAM!

One non-dhcp virtual network

A way to use IPMI to power on/off VMs - virsh commands won’t cut it

*We’ll discuss how to combine more than one hypervisor at the end if there is time

Controlling Hardware

Out of Band Management

IPMI

Hardware BMC (iDrac, ILO) - common in datacenter computing

Controlling Hardware

Out of Band Management

IPMI

Hardware BMC (iDrac, ILO) - common in datacenter computing

VirtualBMC - uses libvirt to control virtual machines using IPMI commands

We only need two features:

- Power control
- Adjust boot order

Installing and Starting Virtual BMC

Fedora - python-devel libvirt-devel python-libvirt python-subprocess32

Debian - python-dev libvirt-dev python-libvirt python-pip

$ pip install virtualbmc

$ vbmc add vmname --port 6011 --username admin --password password --libvirt-uri qemu+ssh://root@192.168.122.1/system

$ vbmc start vmname

** the control node should have root ssh keys to the hypervisor

We can power on/off, now what?

TripleO - OpenStack on OpenStack - program aimed at installing, upgrading and operating OpenStack clouds using OpenStack's own cloud facilities as the foundations

Ironic - program which aims to provision bare metal machines instead of virtual machines. Uses PXE and IPMI to provision and turn on/off machines

Red Hat OpenStack Platform Director

Why OpenStack for this Lab?

It’s heavy, but….

It’s what I know

Red Hat OpenStack Director has everything needed for infrastructure level hardware provisioning built in

Why OpenStack for this Lab?

It’s heavy, but….

It’s what I know

Red Hat OpenStack Director has everything needed for infrastructure level hardware provisioning built in

If you are familiar with some other cloud platform, HPC provisioning or good old DHCP/PXE then you can install Virtual BMC on your hypervisors and roll with it

Keepin’ it real (small)

node_info.yaml

```
parameter_defaults:
  OvercloudControllerFlavor: control
  OvercloudComputeFlavor: compute
  ControllerCount: 1
  ComputeCount: 1
```

Provisioning Network Considerations

Separate VLAN for management only

No native DHCP on the provisioning network!

Allow the Director node to serve DHCP, TFTP, PXE images

DHCP config for nodes can change at any time depending on the state of said node

Virtual Server Config

Mind the networking

Primary NIC on the provisioning network

Primary NIC bootable

Node Definition JSON


```
{
   "nodes": [
      {
         "name":"osp13-p50-a-controller",
         "pm_type":"ipmi",
         "mac":[
            "52:54:00:d1:3b:b0"
         ],
         "pm_user":"admin",
         "pm_password":"password",
         "pm_addr":"192.168.122.1",
         "pm_port":"6011",
         "capabilities": "profile:control,boot_option:local"
      }
}
```

Node Introspection

Tests IPMI

Tests PXE booting

Tests hardware

Inventories hardware

Introspection and Deployment Demo

$ openstack overcloud node import ~/instackenv.json

$ openstack baremetal node list

$ openstack overcloud node introspect --all-manageable --provide

$ openstack overcloud deploy --stack newname --templates -e /home/stack/templates/node-info.yaml

Base Image Considerations

Minimal customizations to the image itself

- Set root password
- Modify only things that will break boot

Rely on post-install tools for most of the configuration

Root Disk Hints

Get root disk WWN from introspection data

Set as root_device

Prevents deployment from using/wiping the wrong disk during install

NIC Config Templates

Production servers will have multiple NICs

NICs will do different functions and connect to different VLANs

Templates tell Director which NICs to use for which function

BMaaS Overview

Service on top of IaaS

Bare metal nodes that are cloud resources, not cloud infrastructure

Available for scheduling by users

Ironic installs an image

Network, storage, etc controlled by cloud, just like a VM

Multiple Lab Hypervisors

Physical provisioning network should not have DHCP

Each hypervisor should have dedicated NIC for provisioning

Connect the VMs' provisioning NIC directly to the provisioning network with a Bridge interface

Control node cannot be a hypervisor

Control node can be a VM on one of the hypervisors though

Thank You

Tim Quinlan

tquinlan@redhat.com

@trquacker

https://github.com/timquinlan