Deploying an OpenStack infrastructure on a single server with Ansible

James Freeman

April 2022

Who am I?

- Technical Account Manager at AWS
- Expowerment Coach
- Reiki Master Teacher
- Autism Dad
- Inventor
- GTD Enthusiast
- Published Author (Ansible)

https://github.com/jamesfreeman959

linkedin.com/in/jamesfreeman959

twitter.com/jamesfreeman959

A horse walks into a bar…

- Why the long title?
  - It didn't fit on the title slide!
- "Digital Twinning with Ansible and cloud-init to support destructive infrastructure testing"
- Let me tell you a story

In a prior role, the CEO says to me…

- Build me OpenStack
- A simple enough request in itself(?)
- All-in-one nodes are easy and quick to build
  - Simple
  - Effective for learning the basics
  - Bear no relation to real life
- Oh by the way…
  - Here's one single blade to build it on
  - With mechanical storage

Digital Twinning

- First used at NASA
  - Full scale mock-ups of space capsules
  - Paved the way for digital simulations
- Can be used in any context
- My mission:
  - To build an OpenStack environment with an architecture and network representative of something you might deploy in the real world

…with Ansible and cloud-init…

- Hypothesis:
  - I'm not going to get this right first time
  - Rebuilds are going to be a necessary corrective action
    - Think Docker-like behaviour
  - I don't want to waste hours performing basic tasks
    - Building/deploying blank operating system images
    - Setting IP addresses and initial passwords
    - Deploying software and configuration

Why Ansible?

- Yes, existing familiarity, but also…
- Ansible overtook Chef and Puppet last year as top cloud configuration management tool[1]
- Agentless
  - Uses SSH to communicate with nodes
  - Works well with switches
- Idempotent
- Self-documenting code
- Vault for securing sensitive data
- See also - the openstack-ansible project

1. Source: https://www.techrepublic.com/article/ansible-overtakes-chef-and-puppet-as-the-top-cloud-configuration-management-tool/

Why cloud-init?

- Ansible is great if:
  1. Your operating system image has an IP address and appropriate network configuration
  2. You can authenticate with your operating system image when it boots
  3. (Linux nodes): Python is installed
- Simple to achieve, yet tedious if performed at scale and repetitively
- The solution: Couple publicly available cloud Ubuntu images with cloud-init
  - cloud-init is baked in
  - Each node has its own ISO image
  - Configures bonding, IP addressing, routing, initial password, SSH keys
  - Even installs Python!

I have but one blade to give…

- Fairly clear that I'm going to need a hypervisor
- Requirements:
  - Free
  - Fast
  - Flexible
  - Support nested virtualization (CPU support is present)
- Linux KVM seemed like the obvious choice

What about the switch layer?

- Must run on Linux KVM
- Preferably modern, and most importantly, free!
- Cumulus Networks (now NVIDIA)
  - Open standards based
  - Linux!
  - Runs on over 130 hardware platforms
    - including libvirt!
  - Capable of building modern web-scale network architectures

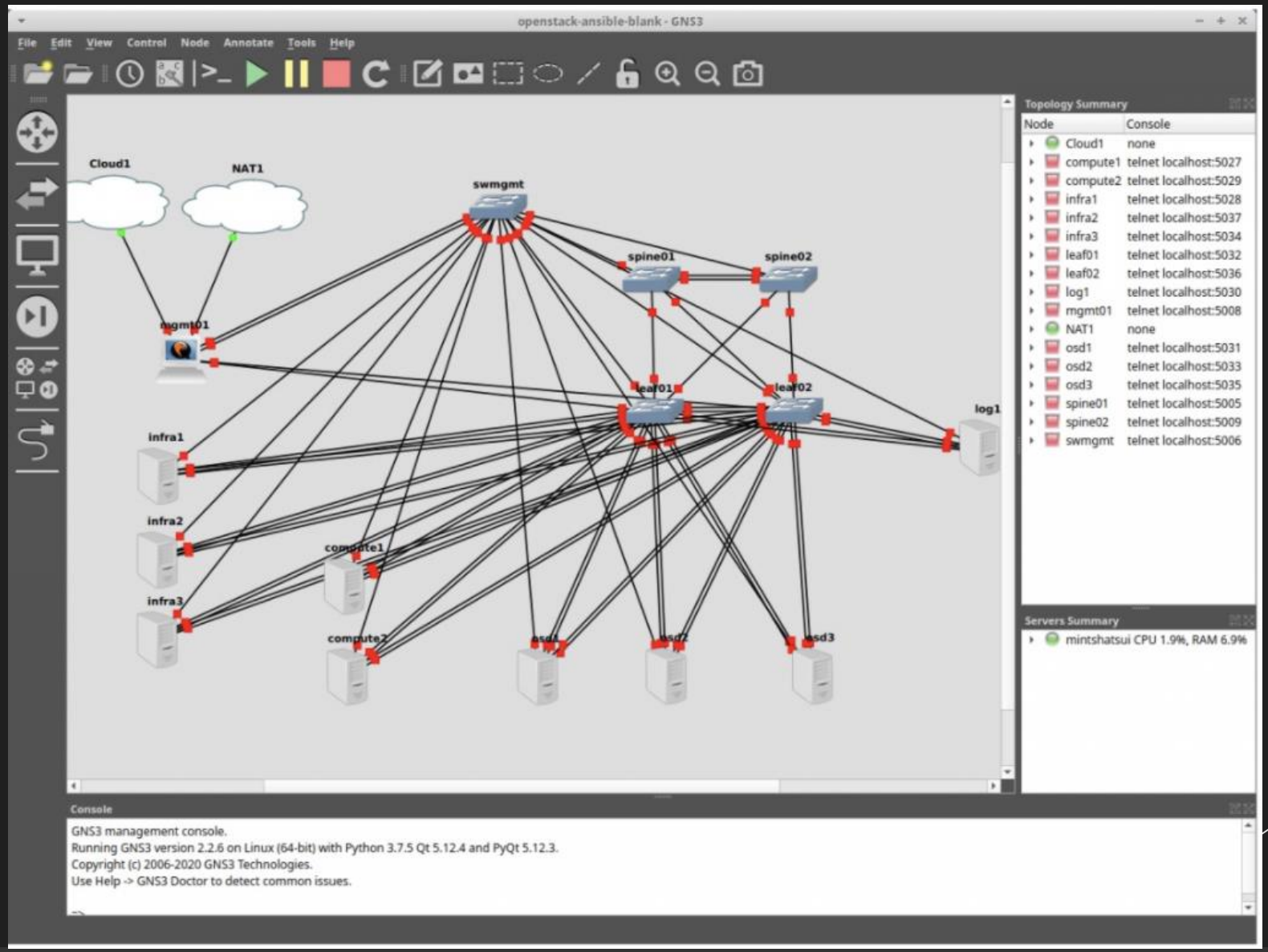
My brain is melting

- Imagine a highly available deployment of OpenStack with Ceph for the backend storage
  - 10 Linux nodes, 5 switches
- Separate management, tunnel and storage networks
  - Using two different physical networks
  - VLAN tagging
  - VxLAN for the tunnel network
  - All ports bonded
  - Leaf and spine switch architecture for redundancy
- Now describe that in XML for libvirt

Heresy in the form of GNS3

- I want a GUI to map out the network
- “I wonder if GNS3 would work…?”
- It turns out that GNS3:
  - Uses libvirt for to create and run virtual machines
  - Is designed to consume cloud images
  - Can directly support the use of ISO files for cloud-init
  - Supports extensive customization of VM’s (e.g. adding nested virtualization)
  - Stores VM data in standard QCOW2 images that you can manipulate with qemu-img
- Oh, and it has a GUI

The end result



The end result

- It works! (after much tweaking)
- A representation of a possible real world OpenStack architecture
- All deployed with Ansible and cloud-init

What does this actually mean?

- The playbooks and cloud-init data used to build the environment could be re-used, as is, to produce a real-world OpenStack infrastructure just like this
  - A real digital twin!
- The GNS3 network is isolated, so you could test with real world IP addresses and MAC addresses
- This technique can be applied to building just about any technology
  - Very little runs on just one server in production any more
- Perfect as a training tool!
  - Combines visual appeal with real-world like software deployments

What else can we do...?

- Let's break something...

Destructive testing ideas

- Network link failure
- Switch failure
- Node failure
- Denial of service attack
- Brute force attack
- Data corruption
- Backup and recovery techniques
- Business Continuity Planning

Yet...

- You can back up and restore your entire digital twin with just a few commands or clicks
- GNS3 even supports snapshots of the entire environment!
- You get to blow it up more than once!

A worked example

- In this test setup, we have 3 OSD nodes
  - All with a dedicated data disk
- Let's pull the plug on a node - does the cluster keep running as expected?
- Ok - now whilst that node is shut down, let's corrupt the disk
  - Idea - let's use dd from copy from /dev/random to the QCOW2 image backing the data
- Now power up the OSD node - what happens?
- Can we recover the node according to the standard recovery procedures?

Limitations

- This is a big step forwards
- Moves us away from test environments not being representative of production
- But…
  - It’s not fast compared to real hardware
    - Especially nested virtualization
  - No hardware specific testing
    - e.g. RAID controller, low level Ethernet setup, FibreChannel
  - You won’t do this on your laptop

Things I learned from this experience

- The Linux community is amazing, and those staffing the OpenStack Ansible IRC channel are so helpful
- Do throw hardware at this
  - I purchased a second hand workstation to run this on
  - 2 x Xeon (16c/32t), 128GB ECC RAM, 1TB SSD
- Do use virtio for disks and network cards wherever possible
- Don’t attempt this setup with GNS3 on anything other than Linux
  - Windows = nested virtualization inside a VM

Moving Forwards

- I now have a whole suite of lab environments built in this manner
- Configurations have evolved and become more streamlined
- Client demos are performed using this
  - People like the visual nature of the architecture
- I use this to train myself on new technologies
- Perhaps we can all learn more using these techniques?

Questions?

Thank you!

- For more information, please reach out to me:

 https://github.com/jamesfreeman959  linkedin.com/in/jamesfreeman959  twitter.com/jamesfreeman959

- Code is available here:
  - https://github.com/jamesfreeman959/gns3-cumulus-openstack-ansible