Virtual Design Master

Challenge 3 – Stacking the Odds

Timothy J. Patterson

ABSTRACT

Many island nations fared better than expected during the outbreak. Their isolation could be very valuable if we face a third round of infection before the Earth has been evacuated. As such, we need to get them back online as soon as possible.

Table of Contents

1. Purpose and Overview

1.1. Executive Summary

Many island nations fared better than expected during the outbreak. Their isolation could be very valuable if we face a third round of infection before the Earth has been evacuated. As such, we need to get them back online as soon as possible. Since travel is dangerous and difficult, the infrastructure should be first created in a lab to ensure it will be able to be quickly deployed by a local team. Once successful, satellite links will be established to deploy the software to the remote locations by local teams.

1.2. Summary Analysis

The intent of this design is to utilize blank hardware to build a readily deployable private cloud environment that is capable of running on top of different hypervisors. The design should be easy to implement, and be proven in a lab environment.

1.3. Intended Audience

This document is aimed at the last few technical humans that are surviving on the planet Earth. They are the key stakeholders and will be implementing this design to preserve their future.

1.4. Requirements

OpenStack Horizon must be used to provision all of the instances (virtual machines). The proof-of-concept should contain one Linux and one Windows image per hypervisor.

Any kind of OpenStack networking may be used.

1.4.1. Availability

Since the application that will run on the infrastructure stack is key to continued survival, the production design must be available 100% of the time.

R001	**Production systems require a 100% availability SLO.**

1.4.2. Maintainability

Maintainability must be ensured. Since we are dealing with whatever blackbox hardware is available, we must rely on our local technical teams to keep the hardware running at all times.

R002	**Must be able to quickly diagnose and repair the infrastructure.**
R003	**Design must be documented properly.**

1.4.3 Integrity

System integrity ensures that adds / moves / changes are not done on production systems. This design will first be validated in a lab environment. Once the proof-of-concept has been established, and the production systems have been deployed, no production changes will be allowed without following a strict change control board process.

R004	**A proof-of-concept for this design must be proven in a lab environment.**
R005	**A change control board must be established and production changes must be approved.**

1.4.4 Reliability

Reliability of the infrastructure must be guaranteed. The weight of meeting this goal falls onto the infrastructure engineers and the local implementation teams. These people must take all of the necessary precautions to ensure the infrastructure runs without errors.

R006	**A peer-based design review process needs to be put into place.**
R007	**Infrastructure component configurations must be audited before placing into production.**

1.4.5 Safety

Even though we are dealing with island nations, zombies are relentless creatures and have started to learn how to drive boats. As such, we need to keep our physical security at a top-notch level. Lucky for us, there are weapon depots on each island nation.

R008	**The datacenter must implement physical security.**
R009	**Employees must carry weapons.**

1.4.6 Scalability

This design must scale horizontally to support any number of compute nodes. As extra hardware is found, we must allow it to be used. As such, the design should implement multiple hypervisors for full compatibility across the board. Both Linux and Windows are in use powering the target applications.

R010	**The design must support the addition of compute nodes for horizontal scaling.**
R011	**Multiple hypervisors must be supported. One must be ESXi / vSphere.**
R012	**Shared storage, vSphere HA, and DRS must be used.**
R013	**The system must support Linux and Windows on all hypervisors used.**

1.4.7 Automation

In order to meet the needs of all other requirements, automation and orchestration is required. For this task, we are to make use of OpenStack Horizon's dashboard. All instances must be provisioned from its web portal.

R014	**The system must use OpenStack Horizon's web portal for provisioning instances.**

1.5. Constraints

C001	**The infrastructure cannot tolerate any serious malfunctions in the production application environment.** This is critical, as human lives are at stake!
C002	**The infrastructure can only use OpenStack Horizon for provisioning.** This is both a requirement and a constraint.
C003	**The target hardware is unknown.** The design must be made generic so that it can function on a multitude of hardware platforms.
C004	**Two different hypervisors must be used, one must be ESXi / vSphere.** This is both a requirement and a constraint.
C005	**DevStack cannot be used.**

1.6. Risks

I001	**The technical skills of the local implementation staff are unknown.** This would impact our deployment capability.

I002	**The satellite links may not be reliable or fast enough to support remote deployment.** This would impact our deployment capability.

1.7. Assumptions

A001	**Hardware is available and assumed to be on the VMware HCL.** Hardware is available for use, albeit of an unknown type.
A002	**All equipment in this design is new and validated to function properly.** The equipment has run for 1 week. All tests were passed without issue.
A003	**All component firmware is at the identical revision.** This is ensures interoperability.
A004	**Infrastructure team will not do software development.** A separate team exists to maintain the application stack and deploy it to the provided infrastructure.

2. Lab Architecture Design

Within this section the proof-of-concept lab will be described.

2.1. Physical Design

The physical design for the proof-of-concept lab is very simple. The lab itself is built upon a single HP ProLiant DL360 G7 1U rackmount server with the following specs:

- 2 x Intel Xeon X5670 CPUs
    - Running at 2.93GHz
    - 6 cores per socket
    - 24 logical processors (with hyperthreading)
- 96GB RAM
- 1TB RAID 10 local storage
- 2 x 1Gbps NICs
    - One connected to a “private” backend VLAN
    - One connected to a “public” NAT VLAN for Internet connectivity
- 1 x Synology DS412+ NAS for NFS shared storage

2.2. Logical Design

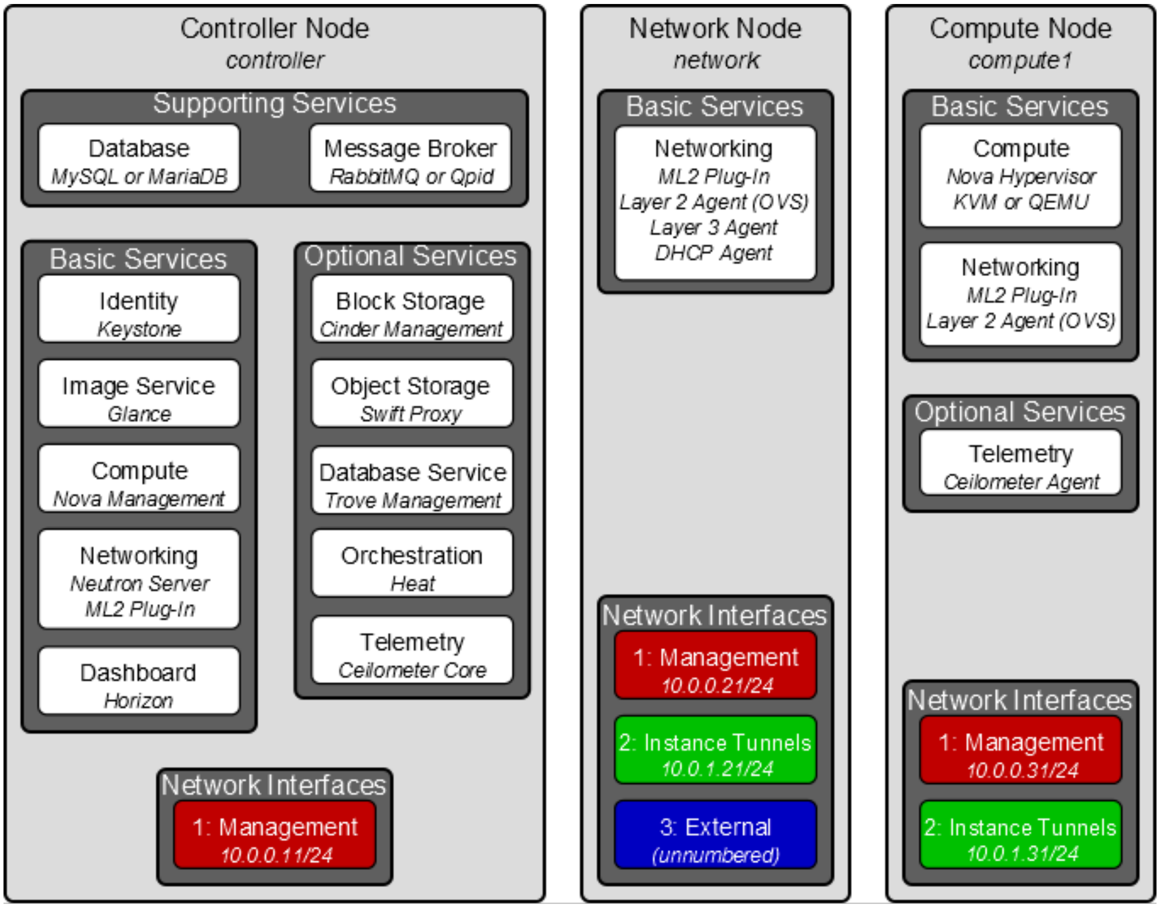
2.2.1 Virtual Machines Layout

The lab server is running ESXi 5.5 installed onto its local disk. To enable use of the nested ESXi hosts, the appropriate CPU flags are allowed to pass into the guests. Within this environment the following virtual machines exist:

- **vcenter.pcpro.local**
  - vCenter Server Appliance
  - This provides a point of management for all ESXi hosts and the vSphere infrastructure.
- **controller.pcpro.local**
  - OpenStack Controller Node
  - This VM runs the core services for OpenStack (Keystone, Glance, Nova Management, Neutron Server, and the Horizon dashboard)
  - CentOS 6.5 64-bit
- **networking.pcpro.local**
  - OpenStack Network Controller Node
  - This VM provides SDN services (Neutron) via OVS and supporting plug-ins.
  - CentOS 6.5 64-bit
- **compute-kvm.pcpro.local**
  - OpenStack Compute Node
  - This VM provides compute power to the OpenStack environment. It utilizes QEMU/KVM for its hypervisor. The Neutron client services also exist on this VM.
  - CentOS 6.5 64-bit
- **compute-vmware.pcpro.local**
  - OpenStack Compute Node
  - This VM provides a conduit between the OpenStack environment and vSphere. It translates OpenStack API calls into their vSphere equivalents and handles image transfer, etc.
- **esxi1.pcpro.local**
  - Nested ESXi 5.5 host.
  - This node will be used as part of the OpenStack vSphere target cluster.
- **esxi2.pcpro.local**
  - Nested ESXi 5.5 host.
  - This node will be used as part of the OpenStack vSphere target cluster.
- **esxi3.pcpro.local**
  - Nested ESXi 5.5 host.
  - This node will be used as part of the OpenStack vSphere target cluster.

2.2.2 OpenStack Architecture

The OpenStack architecture used in the lab environment is based off of a 3 node design. The layout of services follows this example (provided by the OpenStack installation guide):



Additionally, one more compute node is used as the conduit between OpenStack and vSphere. The VM, compute-vsphere.pcpro.local is used for this purpose.

All OpenStack components were installed manually via the steps found in the OpenStack "IceHouse" installation guide. All services have been deployed.

2.2.3 vSphere Design

To enable use within an OpenStack environment, a vSphere cluster must meet strict requirements. This is due to how OpenStack views vSphere. From an OpenStack perspective, an entire cluster is presented as compute power to the Nova service. As such, the OpenStack environment has no idea as to the layout of your hosts, etc. The most relevant requirements are as follows:

- **DRS must be turned on** in Fully Automatic mode.

- **Shared storage must be used.** Storage DRS is used to present an entire datastore cluster to OpenStack. This allows Storage DRS to balance the placement of new VMs across the datastores.
- **Clusters and datastores should not be used for other purposes**, otherwise OpenStack will not report the correct usage information.
- **Firewall settings must be open** to allow VNC connections to the VM consoles.
- **Ephemeral disks are NOT supported** in vSphere.

In this POC design, a vSphere cluster has been established in vCenter named 'OpenStack'. Since all of the resources in a vSphere cluster are essentially one entity in OpenStack's eyes we can still take advantage of the enterprise-class features vSphere has to offer (such as vMotion, etc.)

The vSphere cluster is configured as follows:

- Cluster name: OpenStack
- HA enabled
  - Admission control enabled
  - Set to reserve 34% of all resources
  - Set to enforce limits
- DRS enabled / fully automatic
- vMotion enabled on each host
- A Storage DRS cluster (OpenStack-Storage) has been created with 2 datastores:
  - OpenStack1: NFS to Synology – 1TB
  - OpenStack2: NFS to Synology – 100GB
- 3 x nested ESXi hosts joined as members of this cluster
- A single vDS with the following portgroups:
  - Public-vlan1: Allows VMs to access NAT network
  - Private-vlan2: Management network
  - OVS-vlan3: OpenStack integration bridge
  - Ext-vlan4: "External" network for OpenStack tenants

2.2.4 OpenStack Configuration

All configuration of the OpenStack components came directly from the steps outlined in the OpenStack Installation Guide for CentOS, found here: http://docs.openstack.org/icehouse/install-guide/install/yum/content/index.html

By following the installation guide you are left with a working setup that includes a compute node with the QEMU/KVM hypervisor, and a fully functional dashboard running on the controller node.

Custom configuration was done to the compute-vmware.pcpro.local node to connect it to vSphere. This node was set up identically to the compute-kvm.pcpro.local node, but had the following changes applied:

- Edit /etc/nova/nova.conf. Add the following line under the [DEFAULT] section:

  ```
  compute_driver = vmwareapi.VMwareVCDriver
  ```

- Edit /etc/nova/nova.conf. Change the [vmware] section to match this:

  ```
  [vmware]

  # Hostname or IP address for connection to VMware ESX/VC host.
  # (string value)
  host_ip=10.0.0.103

  # Username for connection to VMware ESX/VC host. (string
  # value)
  host_username=Administrator@vsphere.local

  # Password for connection to VMware ESX/VC host. (string
  # value)
  host_password=stackpw

  # Name of a VMware Cluster ComputeResource. Used only if
  # compute_driver is vmwareapi.VMwareVCDriver. (multi valued)
  cluster_name=OpenStack

  # Regex to match the name of a datastore. (string value)
  datastore_regex=OpenStack*

  # VNC starting port (integer value)
  #vnc_port=5900

  # Total number of VNC ports (integer value)
  #vnc_port_total=10000

  # Whether to use linked clone (boolean value)
  #use_linked_clone=true

  # Physical ethernet adapter name for vlan networking (string
  # value)
  vlan_interface=vmnic0
  ```

```
        # Name of Integration Bridge (string value)
        integration_bridge=br-int
```

Once the above configuration changes were implemented, restart the OpenStack services on the compute node.

2.2.5 Instance Images

In order to test both Linux and Windows in the OpenStack environment, images must be created for each OS and each environment.

To test Linux functionality, a cirros-0.3.2-x86_64 image was downloaded in the QEMU native QCOW format. This image was copied to VMDK format via the qemu-img tool. Additionally, a CentOS 6.5 image was created in VMware natively, then exported into OpenStack by copying the flat VMDK file to the environment.

To test Windows functionality, I am using a pre-packaged Windows Server 2012 R2 Standard image that was prepared by CloudBase. This image is provided in QCOW format. This image was also converted to VMDK via the qemu-img tool.

These images were uploaded into the OpenStack glance service and fully tested in their respective environments.

The lab setup described in the previous sections produced a fully working POC. Based upon lessons learned in the lab, recommendations can be extrapolated for a full-scale production design.

Both Linux and Windows were running flawlessly on KVM and vSphere via OpenStack.

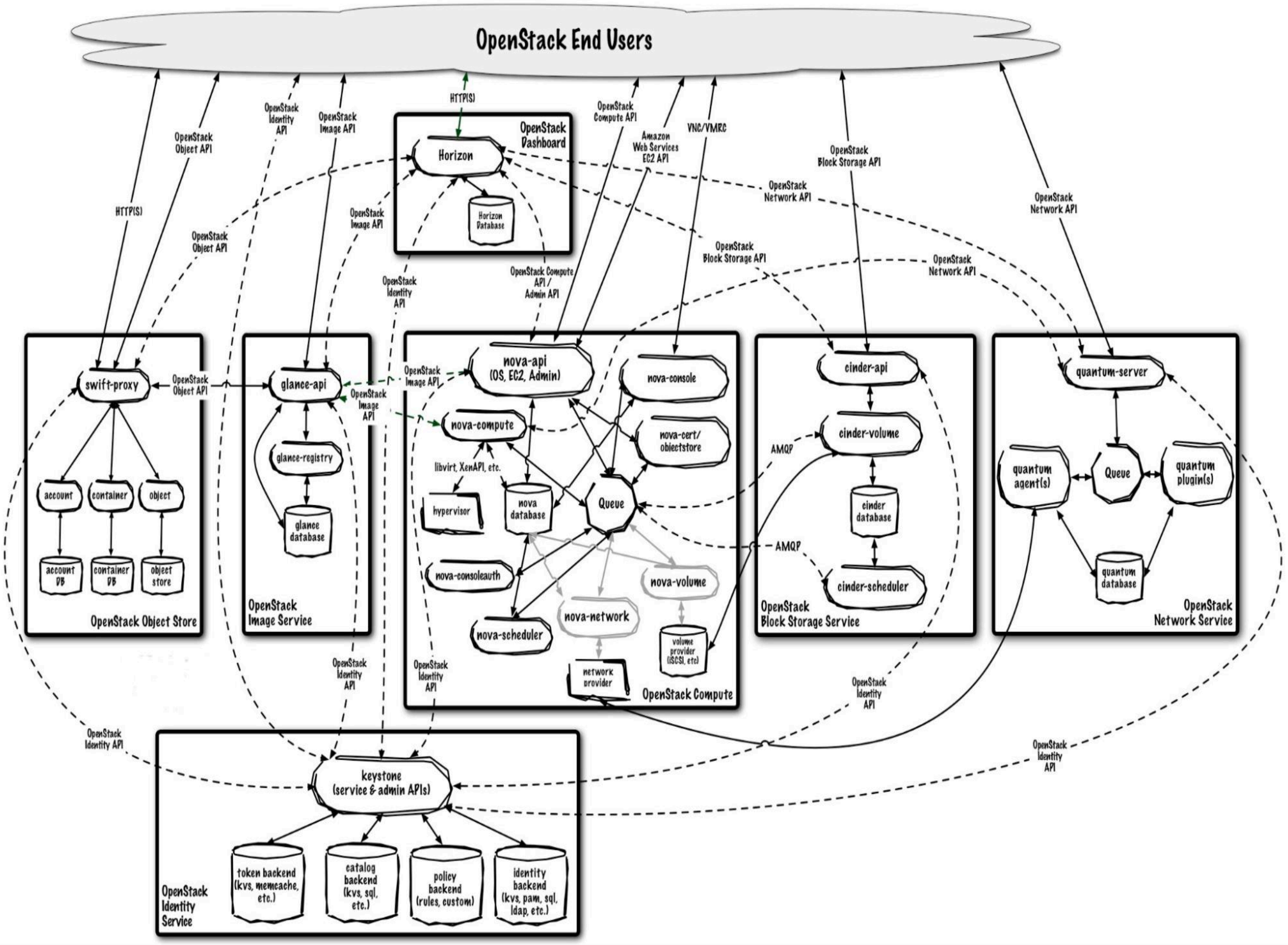
3. Production Deployment Guidelines

Since our lab environment successfully displayed the power inherited via the adoption of OpenStack, it is time to look towards the future. What will it take to scale this design out into a real world deployment?

3.1. Production Architecture

While the components to production architecture were all represented in the lab environment, the resiliency was not. In a production deployment, single points of failure cannot exist. Additionally, separation of services should be present to allow for limitless scalability.

The following illustration shows a full-blown OpenStack deployment. It illustrates how each of the services interacts with one another and gives a nice visual representation of how the separation of services should be achieved.



No matter which hypervisors get implemented in a production environment, it is recommended to have single purpose hosts performing a particular role. These hosts should be deployed in batches so that their underlying service will continue to function without error.

It has been said that in a large enough environment failure will always be present. If you design for failure nothing will ever fail.

3.2. Managing a Production Deployment

It is evident from looking at the large-scale deployment diagram that a complete OpenStack installation is massive. As such, a configuration management system needs to accompany an installation of this size.

Puppet, for example, has all of the tools necessary to help you deploy the host OS's, enforce their configuration, and when combined with a code repository such as git you can version control the entire infrastructure.

When the entire environment starts to live in code you will find yourself in an advantageous position. Inherently you gain the agility to quickly scale based upon demand, as well as re-usability when you need to deploy this infrastructure to a new location.

3.3. Scalability

The services in OpenStack are all designed to scale in both directions. For compute, you can add larger nodes and/or more nodes. This same truth is in effect for an OpenStack deployment based on vSphere as well. By adding more nodes to a vSphere cluster, OpenStack will make use of the cluster (again, seeing it as a single entity).

The production deployment should be able to scale all OpenStack services as needed.

3.4. Implementation

The production implementation will rely on a solid installation and implementation guide. Unfortunately due to the time constraints around this lab POC design, a full-blown guide is not yet available. A step-by-step document should be prepared that walks our on-site engineers through each phase of deployment in such a way that anyone with even a minor technical background could figure it out.

3.5. An Eye Towards the Future

As humans continue to look for a new home to escape the zombie apocalypse one thing is certain… Our fearless leader Melissa will come up with a new set of constraints and challenge us to remain agile and on our toes. Implement this infrastructure carefully and always keep the future in the back of your mind.

4. References

OpenStack Installation Guide for Red Hat Enterprise Linux, CentOS, and Fedora
http://docs.openstack.org/icehouse/install-guide/install/yum/content/index.html

OpenStack Configuration Reference Guide – VMware
http://docs.openstack.org/trunk/config-reference/content/vmware.html

Getting Started With OpenStack and VMware vSphere
http://www.vmware.com/files/pdf/techpaper/VMWare-Getting-Started-with-OpenStack-and-vSphere.pdf