Virtual Design Master Season 2

Challenge One Submission: ShipDepot Architecture

Challenge 1 – Design and Orchestration

Byron Schaller
7-15-2014

Contents

1 Overview

1.1 Executive Summary

The zombie invasion has proven too much for humanity and as such we must move on to the moon and eventually to Mars. In preparation for this trip space shuttles must be constructed at depots around the world. The WGTG Corporation has been formed to accomplish this task.

The depots require an integrated system to run the manufacturing process. To meet this need the ShipDepot application stack has been developed. ShipDepot is a three-tier application designed to run in Docker containers to provide the upmost resiliency as any interruption in service will no doubt cost human lives.

The systems that supports the WGTG manufacturing facilities have designed with following considerations:

1. Each site will be supported by an identical self-contained system.
2. The system will consist of management pod and an application pod.
3. The management pod will be resilient and highly available.
4. The application pod will be failure tolerant.

1.1.1 Management Pod

The Management Pod will contain all systems that are not the part of the core line of business application. This includes directory and network services, management and monitoring, provisioning services, and configuration management.

1.1.2 Application Pod

The Application Pod will contain all systems that are part of the core line of business application. All services will be provisioned on demand and be highly resilient.

1.2 Requirements

Table 1 details the requirements gathered from the Virtual Design Master team.

Table 1

Reference	Requirement
R01	Solution must be highly reliable.
R02	Solution must be easy to deploy.
R03	Solution must be deployed in the least possible amount of time.

R04	Solution must include all components for infrastructure and the application.
R05	Solution must be scalable.
R06	Application must include a client facing web layer.
R07	Application must include a message queue middle tier.
R08	Application must include a database backend.

1.3 Constraints

Reference	Constraint
C01	The only Storage vendor to survive the zombies was Pure Storage due to the amazing fighting prowess of Vaughn Stewart, so Pure Storage must be used.

1.4 Assumptions

Table 2 details assumptions made in this design.

Table 2

Reference	Assumption
A01	Budget and product availability are not a concern.
A02	Workload profile of application is unknown.
A03	Data volume of application is unknown.
A04	Data retention and protection not a primary concern.

1.5 Contributors

Byron Schaller – Primary Author

Josh Coen – Reviewer

Adam Eckerle - Reviewer

2 vSphere vCenter Datacenter Design

2.1 Management Pod

2.1.1 Virtual Machine Sizing

Table 3 lists all required virtual machines in the management pod and their resource requirements.

Note: Functions and configuration of the virtual machines is detailed in sections 5,6 and 7.

Table 3

Virtual Machine	vCPU	vRAM	Storage	Network
Identity Virtual Appliance	1	2	10	1

vPostgres Virtual Appliance	2	4	20	1
vCloud Automation Center Virtual Appliance 1	4	16	30	1
vCloud Automation Center Virtual Appliance 2	4	16	30	1
Infrastructure Web/Manager Server 1	2	8	40	1
Infrastructure Web/Manager Server 2	2	8	40	1
Infrastructure DEM Server 1	2	6	40	1
Infrastructure DEM Server 2	2	6	40	1
Infrastructure Agent Server 1	2	4	40	1
Infrastructure Agent Server 2	2	4	40	1
MSSQL Database Server 1	8	16	200	1
MSSQL Database Server 2	8	16	200	1
vCloud Automation Center VA Load Balancer	1	4	10	1
Infrastructure Web Load Balancer	1	4	10	1
Infrastructure Manager Service Load Balancer	1	4	10	1
VCO Server 1	2	8	40	1
VCO Server 2	2	8	40	1
Active Directory Domain Controller 1	2	8	40	1
Active Directory Domain Controller 2	2	8	40	1
NTP Server 1	1	4	16	1
NTP Server 2	1	4	16	1
Management Cluster vCenter Server	2	12	40	1
Application Cluster vCenter Server	2	12	40	1
vCenter Operations Manager UI Appliance	2	7	40	1
vCenter Operations Manager Analytics Appliance	2	9	900	1
Virtual Infrastrucutre Navigator Appliance	2	4	16	1
Hyperic HQ Appliance	4	8	20	1
vPostgress Appliance for Hyperic	4	6	50	1
LogInsight Server	4	8	100	1
SaltStack Master Server	2	8	16	1
Software Package Repo Server	2	8	20	1
Total	**78**	**240**	**2194**	**31**

2.1.2 Cluster

Availability is of the upmost importance for the Managment Pod. If it is unavailable new application machines will not be able to be provisioned and that could impact production of the ships. To accomplish a 99.9% uptime even when one host is down for maintenance, it is necessary to size the cluster at the level of N +2.

The Management Pod Cluster will be comprised of 4 hosts for N+2 redundancy.

Table 4

Configuration Setting	Value

Number of Hosts	4
DRS	Enabled
HA	Enabled

2.1.2.1 vSphere Cluster High Availability

The vSphere Cluster HA configuration for the Application Pod cluster is detailed in Table 5.

Table 5

Configuration Setting	Value
Host Monitoring	On
Admission Control	Prevent
Admission Control Policy	CPU: 50%, Memory: 50%
Default Virtual Machine Restart Priority	Medium
Host Isolation Response	Leave Powered On
Virtual Machine Monitoring	Enabled
VM Monitoring Sensitivity	Medium

2.1.2.2 Distributed Resource Scheduler

The DRS configuration for the Managment Pod cluster is detailed in Table 6.

Table 6

Configuration Setting	Value
DRS	Enabled
Automation Level	Fully Automated
Migration Threshold	Moderate
DPM	Disabled
Enhanced vMotion Compatibility	Enabled
Swap File Location	Same Directory as VM

2.1.3 Storage

2.1.3.1 Datastores

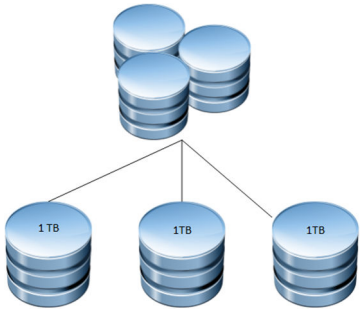

Figure 1

The total storage required for the management cluster is 2TB. The datastores also need to meet a N+1 level of availability. N+2 is deemed unnecessary due to the availability and redundancy built into the storage array.

The Management Pod Cluster will contain a Datastore Cluster backed by 3 1TB Datastores for N+1 availability.

Single Initiator zoning will be used to eliminate unnecessary Registered State Change Notifications.

The Storage DRS settings, Datastore Cluster settings, and Host configuration settings are detailed in Tables 7, 8, and 9.

Table 7

Configuration Setting	Value
Storage DRS	Enabled

Table 8

DataStore Cluster	I/O Metric	Automation Level	Datastores
DS-Cluster01	Enabled	Manual	3

Note: Path Selection Policy and IO Operation Limit are configured to use the documented best practices of Pure Storage for using the FA-405 with VMware Esxi.

Table 9

Host Configuration Setting	Value
Boot Partiation Location	SAN
Storage Array Type	VMW_SATP_ALUA
Path Selection Policy	Round Robin
IO Operation Limit	1
Zoning	Single Initiator

2.1.4 Network

2.1.4.1 vNICs

Due to the design of the Cisco UCS C460, each host will have 4 1GbT Ethernet and 2 10Gbt Ethernet vNICs connected to a vDS Uplink Group.

2.1.4.2 Physical Switches

There will be 2 physical switches capable of both 1GbT and 10GbT. Switch will be connected to 2 1GbT vNICs and 1 10GbT vNIC on each host. VLANs will be used to create separate broadcast domains.

2.1.4.3 Virtual Switch

There will be one distributed switch with three port groups.

The configuration of dvSwitch0 is detailed in Table 10.

Table 10

Virtual Switch	Number of Ports	Physical NICs	Port Group (VLAN ID)
dvSwitch0	6	6	Management (10) vMotion (20) VM Network (30)

2.1.4.4 Port Groups

The port group configurations are detailed in Table 11.

The failover configuration and load balancing settings were chosen based on the expected bandwidth requirements for each of the networks.

Table 11

Virtual Switch	DVPortGroup	Network Ports	Load Balancing
dvSwitch0	Management	dvUplink0 (active) dvUplink1 (standby)	Route based on virtual port Id
dvSwitch0	vMotion	dvUplink2 (active) dvUplink3 (standby)	Route based on virtual port ID
dvSwitch0	VM Network	dvUplink4 (active) dvUplink5 (active)	Route based on physical network adapter load

Figure 2 details the linkages between all components of the distributed switch and the physical network.

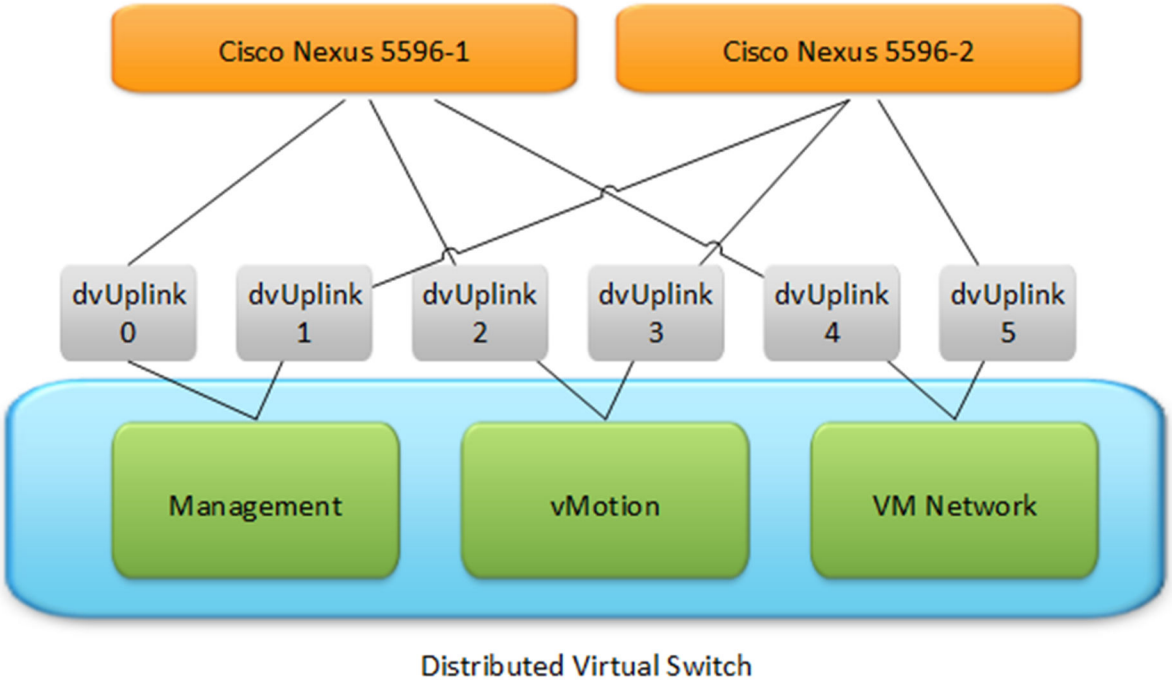

Figure 2

2.2 Application Pod

2.2.1 Virtual Machine Sizing

The application pod runs between 4 and 480 workloads all cloned from a single template. This template is configured with 2 vCPUs, 32GB of vRAM, and a 100GB Hard Drive. All IPs are assigned by DHCP.

The workloads will be thin provisioned. 25 percent of the workloads will consume all 100 GB provisioned. The other 75 percent will consume a maximum of 10GB.

2.2.2 Cluster

The Application Pod Cluster is configured with 4 hosts each configured with 2 15 core CPUS and 512 GB of RAM. This will support 45 instances of the workload, due to N+1 redundancy. Each addition server added to the cluster will add support for an additional 15 workloads. This scales to 465 workloads in a 32 host cluster.

The Management Pod Cluster will be comprised of 4 hosts for N+1 redundancy. The Admission Control Policy needs to be adjusted for every new host that is added to the cluster to ensure that no more than N+1 is maintained. See Appendix A for a complete breakdown of Admission Control Policy percentages per number of hosts.

Table 12

Configuration Setting	Value
Number of Hosts	4
DRS	Enabled
HA	Enabled

2.2.2.1 vSphere Cluster High Availability

The vSphere Cluster HA configuration for the Application Pod cluster is detailed in Table 13.

Table 13

Configuration Setting	Value
Host Monitoring	On
Admission Control	Prevent
Admission Control Policy	CPU: 25%, Memory: 25%
Default Virtual Machine Restart Priority	Medium
Host Isolation Response	Leave Powered On
Virtual Machine Monitoring	Enabled
VM Monitoring Sensitivity	Medium

2.2.2.2 Distributed Resource Scheduler

The DRS configuration for the Application Pod cluster is detailed in Table 14.

Table 14

Configuration Setting	Value
DRS	Enabled
Automation Level	Fully Automated
Migration Threshold	Moderate
DPM	Disabled
Enhanced vMotion Compatibility	Enabled
Swap File Location	Same Directory as VM

2.2.3 Storage

On average 500 GB of space is required for each host. With 4 hosts at the initial configuration 2TB will be required to accommodate for expected growth.

2.2.3.1 Datastores

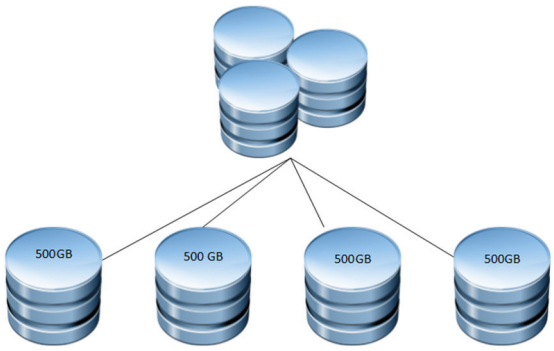

Figure 3

Datastores will be created in increments of 500 GB. All datastores will be presented as one datastore cluster.

The initial cluster will be created with 4 500GB datastores in the datastore cluster.

Single Initiator zoning will be used to eliminate unnecessary Registered State Change Notifications.

The Storage DRS settings, Datastore Cluster settings, and Host configuration settings are detailed in Tables 15, 16, and 17.

Table 15

Configuration Setting	Value
Storage DRS	Enabled

Table 16

DataStore Cluster	I/O Metric	Automation Level	Datastores
DS-Cluster01	Enabled	Manual	4

Note: Path Selection Policy and IO Operation Limit are configured to use the documented best practices of Pure Storage for using the FA-405 with VMware Esxi.

Table 17

Host Configuration Setting	Value
Boot Partiation Location	SAN
Storage Array Type	VMW_SATP_ALUA
Path Selection Policy	Round Robin
IO Operation Limit	1
Zoning	Single Initiator

2.2.4 Network

2.2.4.1 vNICs

Due to the configuration of the Cisco UCS hosts, each host will have 4 1GbT Ethernet and 2 10Gbt Ethernet vNICs connected to a vDS Uplink Group.

2.2.4.2 Physical Switches

There will be 2 physical switches capable of both 1GbT and 10GbT. Switch will be connected to 2 1GbT vNICs and 1 10GbT vNIC on each host. VLANs will be used to create separate broadcast domains.

2.2.4.3 Virtual Switch

There will be one distributed switch with three port groups.

The configuration of dvSwitch0 is detailed in Table 18.

Table 18

Virtual Switch	Number of Ports	Physical NICs	Port Group (VLAN ID)
dvSwitch0	6	6	Management (10) vMotion (20) VM Network (30)

2.2.4.4 Port Groups

The failover configuration and load balancing settings were chosen based on the expected bandwidth requirements for each of the networks.

The port group configurations are detailed in Table 19.

Table 19

Virtual Switch	DVPortGroup	Network Ports	Load Balancing
dvSwitch0	Management	dvUplink0 (active) dvUplink1 (standby)	Route based on virtual port Id
dvSwitch0	vMotion	dvUplink2 (active) dvUplink3 (standby)	Route based on virtual port ID
dvSwitch0	VM Network	dvUplink4 (active) dvUplink5 (active)	Route based on physical network adapter load

Figure 4 details the linkages between all components of the distributed switch and the physical network.

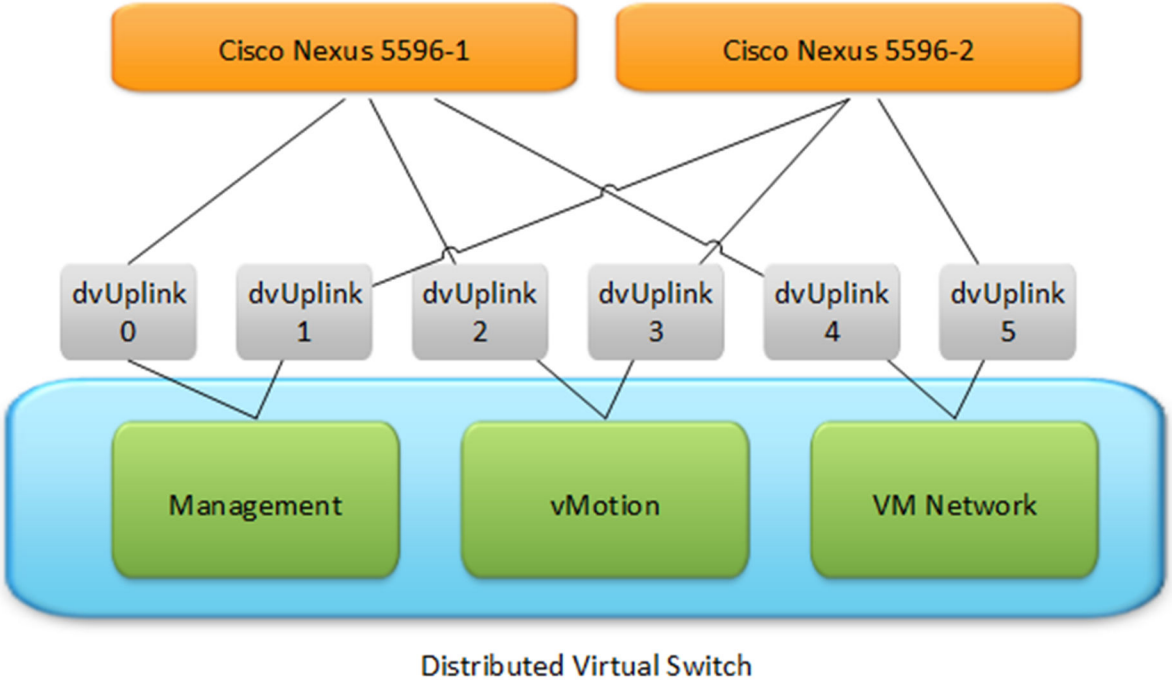

Figure 4

3 vSphere Physical Design

Note: The following configuration is valid for an Application Pod Cluster of 4 hosts. As ore hosts are added the network connectivity and backend storage will need to be grown as well.

3.1 Management Pod Host Servers

The Management Pod will be comprised of 4 Cisco UCS C460 servers connected to a pair of Cisco Nexus 5596 switch via Gigabit Ethernet, 10GB Ethernet, and 8GB Fibre Channel.

C-Series rack servers were chosen to eliminate the Blade Chassis as a point of failure.

Table 20 lists the build specifications for each host in the Management Pod.

Table 20

Make and Model	Cisco UCS C460
CPU	4 x 3.2 Ghz 8 core Intel Xeon
Memory	16 x 32GB DRAM Mirrored
Internal Hard Drives	2 x 256 GB SSD Mirrored
HBA	2x Emulex Dual Port 8GB FC
Gigabit Ethernet	4 x On Board Intel NIC
10 Gigabit Ethernet	2x Cisco 10GB Ethernet PCI NIC

3.2 Application Pod Host Servers

The Application Pod will be comprised of 4 – 32 Cisco UCS C240 servers connected to a pair of Cisco Nexus 5596 switch via Gigabit Ethernet, 10GB Ethernet, and 8GB Fibre Channel.

C-Series rack servers were chosen to eliminate the Blade Chassis as a point of failure.

Each host has the potential to run hundreds of Docker containers. To minimize the size of the failure domain 2 socket hosts were chosen in lieu of 4 socket hosts.

Table 21 lists the build specifications for each host in the Management Pod.

Table 21

Make and Model	Cisco UCS C240 M3
CPU	2 x 3.2 Ghz 15 core Intel Xeon
Memory	16 x 32GB DRAM
Internal Hard Drives	2 x 256 GB SSD Mirrored
HBA	2x Emulex Dual Port 8GB FC
Gigabit Ethernet	4 x On Board Intel NIC
10 Gigabit Ethernet	2x Cisco 10GB Ethernet PCI NIC

3.3 Storage

Dual Pure Storage FA-405s are configured as the SAN. They are connected to each other with 56GB Infinband. They are configured in an Active/Active cluster. ALUA is supported. They are connected to the Cisco Nexus 5596 via 8GB FC. Both controllers have 2 dual port FC HBAs.

The Management Pod and the Application Pod share the Nexus 5596s as well as the FA-405s.

Figure 5 shows the connectivity of the Storage Network. One Managment Pod cluster host and one Application Pod cluster host are shown for simplicity. Each additional host can be assumed to have the same connectivity.

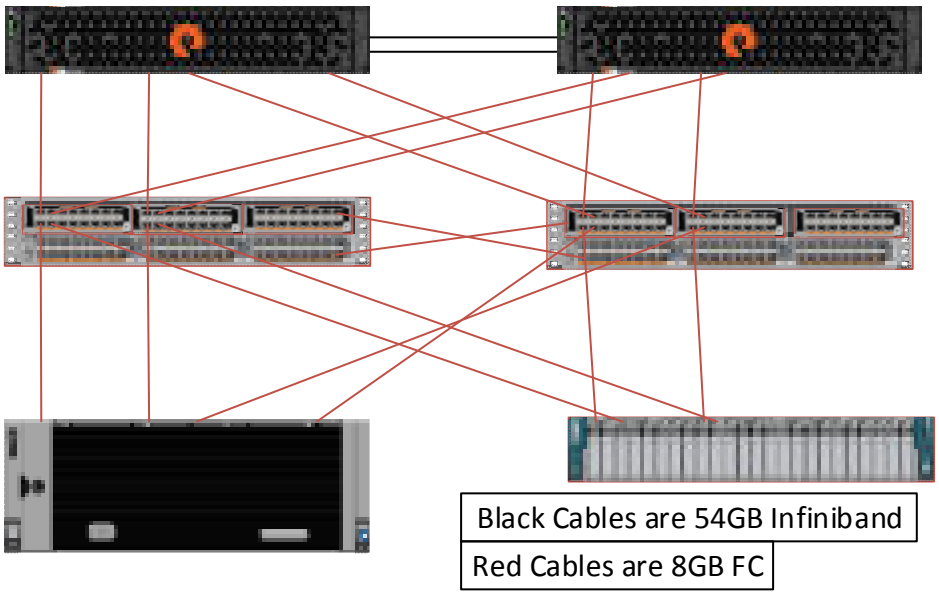

Figure 5

3.3.1 VAAI

vSphere Storage API for Array Integration is supported by the FA-405. This enables the use of block storage primatives. The Pure Storage vCenter will also be installed.

3.4 Network

The Cisco Nexus 5596 is populated with unified ports. Both GE network adapters and 10GE connect to the 5596s. The management ports of the Pure Storage FA-405s also connect to GE ports on the 5596s.

Figure 6 shows the connectivity of the Ethernet Network. One Managment Pod cluster host and one Application Pod cluster host are shown for simplicity. Each additional host can be assumed to have the same connectivity.

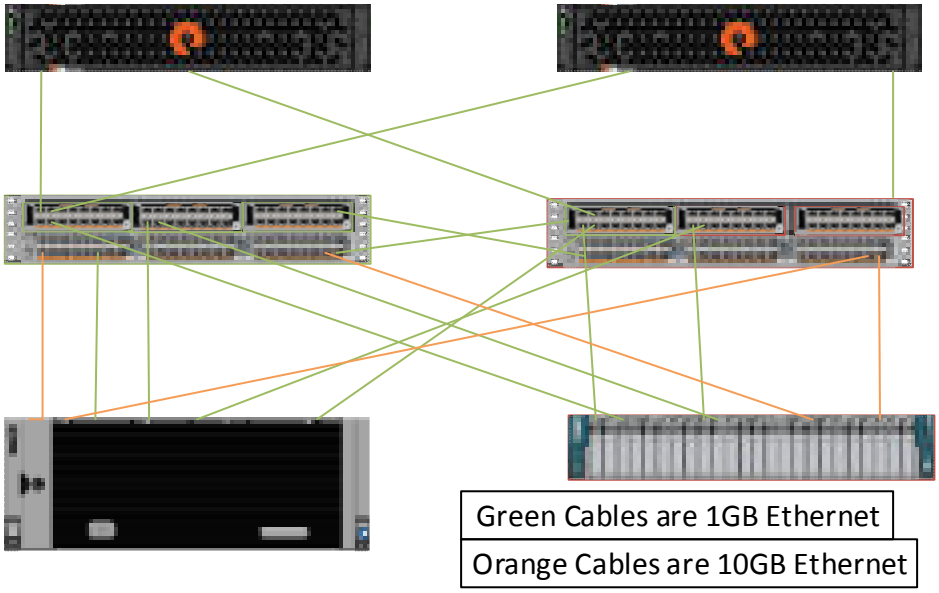

Figure 6

3.5 Business Continuity

Each ShipDepot site consists of a multiple buildings in a campus configuration. All buildings are well connected with 1GB Ethernet and <10ms latency. The ShipDepot Management and Application Pods will be located in the main campus building.

In case of any interruption in service, up to and including the complete loss of the main campus building, the following business continuity plan is in place.

A redundant piece of all hardware at the main campus building will be stored in a second building as far from the main campus building as possible. All hardware will be powered off except for the Cisco 5596 Nexus switches and the Pure FA-405 storage controllers.

All data will be replicated synchronously from the main campus site to the backup site using Pure Storage Purity Flash Recover software.

When a business continuity event is initiated, all hosts will be powered on and booted from the secondary SAN. Once fully powered on, services will be resumed.

A Note about DR: The ShipDepot system and all associated hardware is designed to drive the manufacturing capabilities of one site. If a site suffers a total loss, DR is moot point. A new site would need to constructed a new clusters built.

3.6 Security

3.6.1 Access Control

All access to systems will be controlled through Role Based Access Control with ShipDepot.local being the authoritative directory.

There will be three access roles:

Administrator: Full Privileges on entire system

Developer: Read/ Write Privileges

Operator: Read and Execute Privileges

3.6.2 Network Security

All ports for all services not being used hosts will be blocked via the local host firewall.

3.6.3 Auditing

Auditing of Access Control will be handled by the security team using Log Insight to scan the correlated logs.

4 Application Architecture

In the days before the zombies there was a service called Netflix.

Netflix pioneered a style of distributed applications that could withstand any one node failing without effecting the application performance or system as a whole. In some circles this style became known as “Chaos Monkey’. It is in the spirit of those crazy primates that ShipDepot was designed.

The ShipDepot application is a mission critical line of business application that is responsible for supporting the manufacturing of space ships. It is a three-tier application consisting of an HTTP presentation layer, message queue middleware, and a database backend. There is also an HTTP load balancer layer that sits between the application and the end users for high availability.

Each instance of a service has been designed to run in its own Docker container. The application has been designed to be failure tolerant. Any container can be destroyed at any time and have no impact on the applications ability to service transactions. This is accomplished by have a stateless presentation and middleware layer and a distributed database layer that natively stores data in replicas.

Figure 7 details the flow of the ShipDepot application.

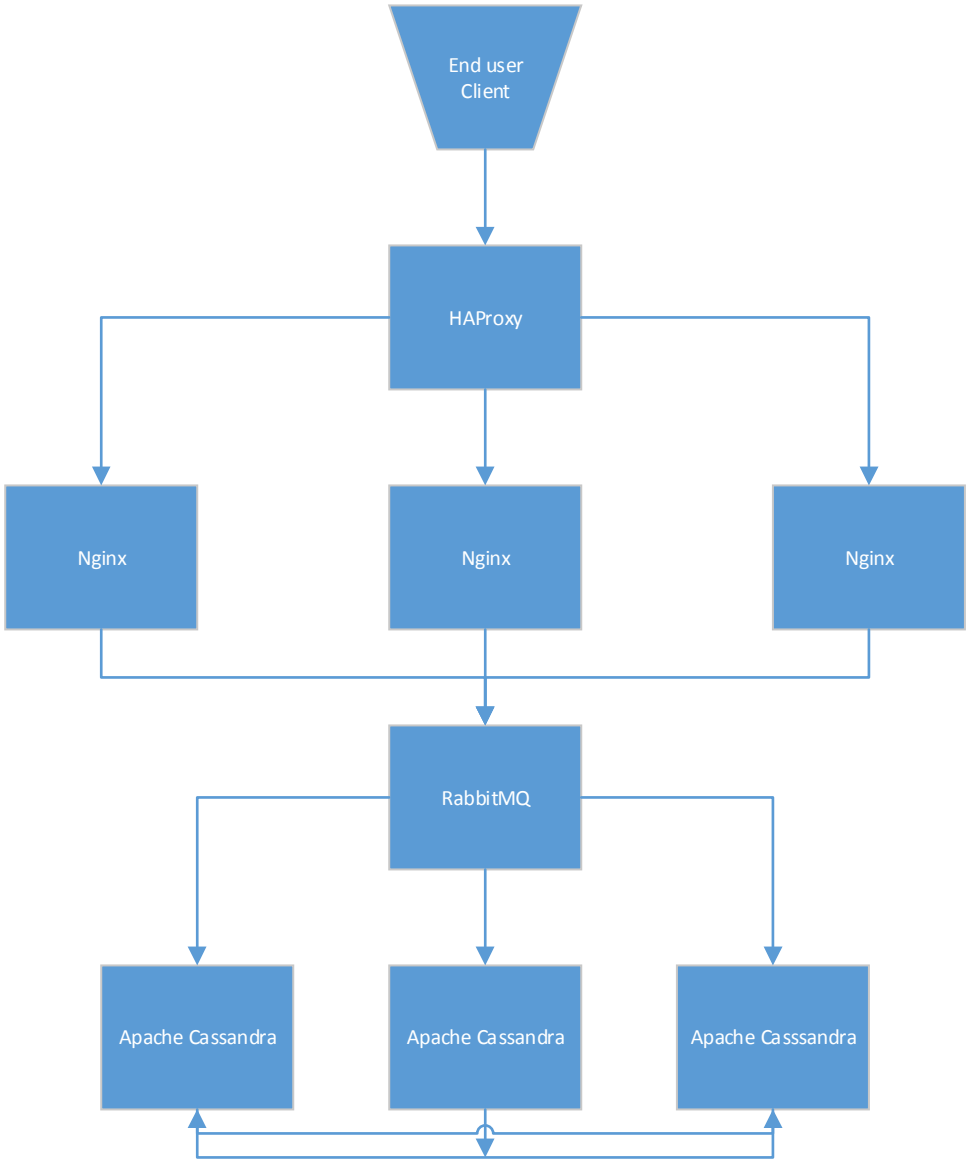

Figure 7

4.1 Load Balancer Layer

HAProxy has been chosen as the HTTP Load Balancer to front the ShipDepot application. It was chosen for its speed, small footprint, and ability to massively scale.

4.2 Presentation Layer

NginX has been chosen as the Presentation layer of ShipDepot. It is lightweight, resilient, and easy to massively scale. The Presentation Layer is also stateless and communicates to the message queue using non-blocking notifications.

4.3 Middleware Layer

RabbitMQ has been chosen as the middleware message queue layer. RabbitMQ was chosen because of its speed and its ability to present several instances of the service as one logical set of exchanges and queues. This layer is also stateless.

4.4 Database Layer

Apache Cassandra has been chosen as the database layer. It is lightweight, is massively scalable and replicates all data to multiple nodes upon commit.

5 Management and Monitoring

The Management and Monitoring Virtual Machines all reside in the management pod. They are detailed blow:

5.1 VMware vCenter 5.5

There are two vCenter virtual machines. Both run Windows Server 2008 R2 and connect to 2 different vCenter databases running on the Microsoft SQL Server 2012 availability group. vCenter1 manages the Management Pod Cluster. vCenter2 manages the Application Pod Cluster.

Note: SQL 2012 AGs are not an officially supported as a backend for vCenter. However most of VMware support was killed by zombies, rendering this point moot.

Changes should not be made directly to any virtual machine that is manages by vCenter2 as it is the compute resource for a vCAC Fabric Group.

5.2 Microsoft Active Directory Domain Controllers

There are two Windows Server 2012 R2 domain controllers running the shipdepot.local domain. The AD domain is also the directory server for both vCenter servers as well as vCAC.

The servers also run DNS and DHCP services for all machines on the network.

5.2.1 DHCP

The DCHP scope for the network is 172.168.20.0 /23.

172.168.20.1 – 172.168.20.50 are reserved for the management pod virtual machines.

5.3 Microsoft SQL Server 2012 AlwaysOn Availability Group

There is a two node SQL Server 2012 AOAG. Databases for the following services are hosted here:

- Both vCenter Servers
- vCAC IaaS

5.4 NTP Servers

There are two CentOS 7 ntpd servers for network time sync.

5.5 VMware vCenter Operations Manager 5.8.1

Operations Manager is the central point for monitoring and performance analytics. The following systems are connected to vC Ops:

- Both vCenter Servers
- vCenter Infrastructure Navigator
- Log Insight

5.5.1 Adapters

The following adapters are installed:

- Management Pack for Storage Devices
- Hyperic

5.6 VMware vCenter Infrastructure Navigator

VIN maps inter-application dependencies and allows for greater insight with vCenter Operations Manager. VMware Tools is required to be installed on every guest for VIN to properly map the services running in the workload.

5.7 VMware Hyperic HQ

5.7.1 Plugins

The following Plugins are installed:

- VMware vCenter
- Management Pack
- Microsoft IIS Monitoring
- Active Directory Monitoring
- Microsoft SQL

5.8 VMware Log Insight

5.8.1 Content Packs

The following Content Packs are installed:

- vSphere Content Pack
- vCAC Content Pack
- vCenter
- Microsoft Windows
- Microsoft Active Directory

6 Provisioning Services Automation

6.1 vCloud Automation Center Architecture

The vCAC architecture has been designed to allow for provisioning 50 workloads simultaneously. Resilience and availability are also a primary concern. The system enables the rapid massive scaling of the ShipDepot system.

Figure 8 details the relationship of all virtual machines in the vCloud Automation Center system.

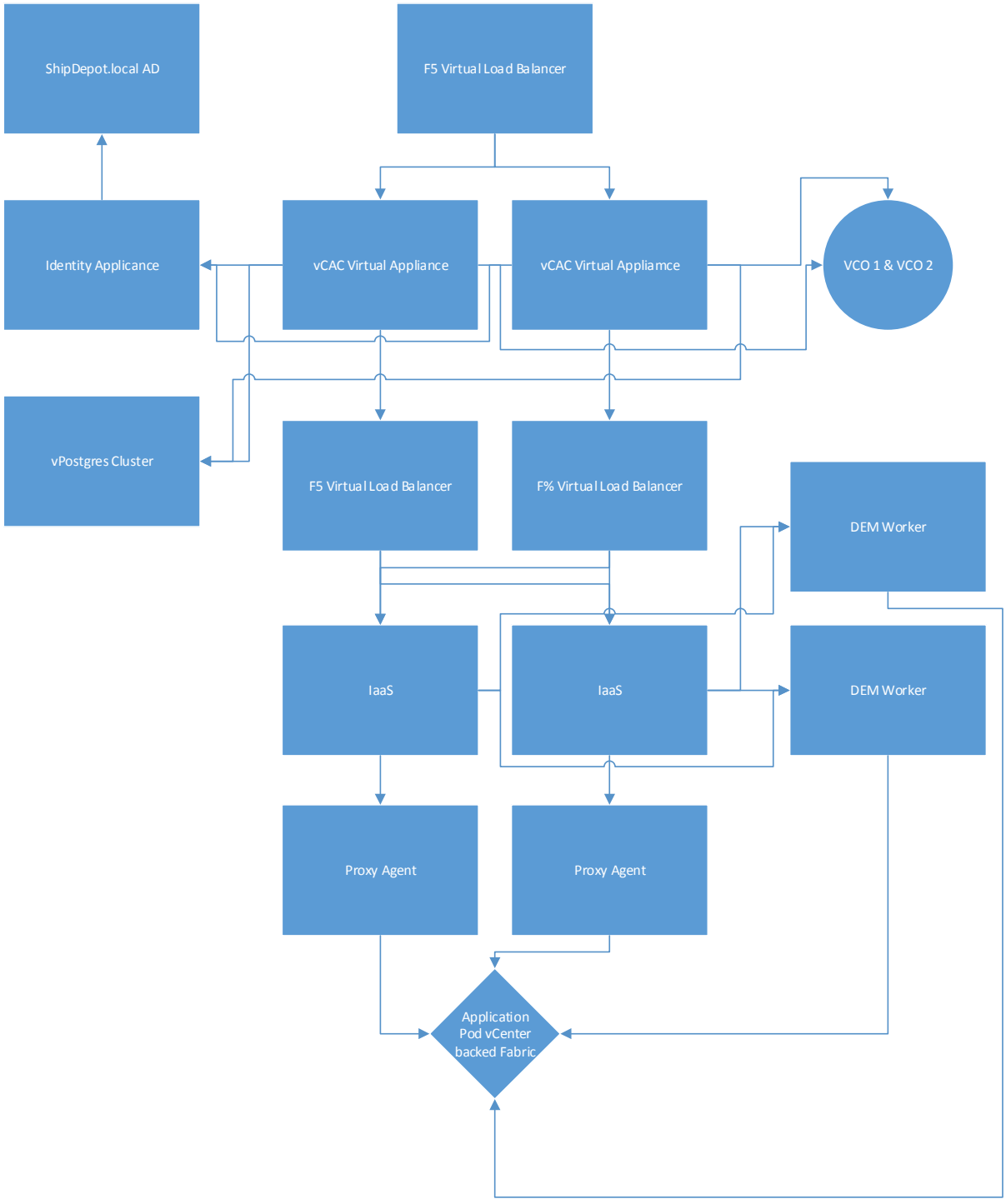

Figure 8

An F5 BIG IP Virtual load balancer presents a Virtual IP balancing between vCAC Virtual Appliance 1 and 2. The vApps connect to a vProgres cluster and two vCenter Orchestrator servers.

Note: F5 was chosen as the load balancer here because it is the only one that works consistently with vCAC 6.0.1.

Authentication is handled by the Identity Virtual Appliance connected to the ShipDepot.local Active Directory domain.

Two Windows 2008 R2 IaaS servers are configured in an Active/Passive configuration fronted by 2 F5 BIG IP Virtual Load Balancers. The IaaS Database is hosted on the shared Microsoft SQL 2012 AlwaysOn Availability Group.

2 Infrastructure Proxy Agent Servers and 2 Infrastructure DEM Worker servers control the Fabric Group backed by the Application Pod Cluster vCenter.

6.2 Fabric Groups

There is one fabric group. This group has the Application Pod vCenter configured as the compute resource.

6.3 Business Groups

There is one business group, the IT Operations Group. They are entitled to all services and blueprints.

6.4 Services

There is one Service, ShipDepot. All Blueprints are part of this service.

6.5 Blueprints

There are 5 Blueprints. 1 to deploy the CoreOS base template and 5 to deploy one of the needed types of Docker containers.

6.5.1 Provision CoreOS Workload

New instances of the CoreOS workload are provisioned via “Clone From Template”. IPs for new workloads are dynamically assigned by DHCP. Machine names are assigned in the form of “coreos-*xxxxx*” where *xxxxx* is a five digit number incremented by 1 starting at 00001.

The template contains the Linux Guest Agent and the SaltStack minion daemon.

Upon completion of provisioning the workload the Linux Guest Agent runs the following commands:

1. Add user SSL keys from the Software Repository Server via git
2. Configure etcd service discovery
3. Join the server to the established fleet cluster
4. Connect the SaltStack minion to the SaltStack master server

6.5.2 Provision HAProxy Container

This blueprint calls a VCO workflow that does the following:

1. Query a list of all virtual machines in the Application Pod Cluster
2. Save returned list to an array
3. For each VM in the array query the vCenter Operations Manager server via the HTTP/REST VCO plugin and return the Health KPI
4. Save all returned Health KPIs with the associated VM name in an array of key:value pairs
5. Sort the KPI\VM array by highest Health score. Return the name of the associated VM.
6. Connect to the SaltStack Master server salt-api via the HTTP/REST VCO plugin

7. Call the salt.modules.dockerio module to provision and start an instance of the HAProxy image stored on the Software Repository Server on the VM returned in step 5
8. Bootstrap the salt-minion inside the container to connect to the SaltStack master
9. Call the salt.modules.state module to configure the new instance of HAProxy

6.5.3 Provision nginx Container

This blueprint calls a VCO workflow that does the following:

1. Query a list of all virtual machines in the Application Pod Cluster
2. Save returned list to an array
3. For each VM in the array query the vCenter Operations Manager server via the HTTP/REST VCO plugin and return the Health KPI
4. Save all returned Health KPIs with the associated VM name in an array of key:value pairs
5. Sort the KPI\VM array by highest Health score. Return the name of the associated VM.
6. Connect to the SaltStack Master server salt-api via the HTTP/REST VCO plugin
7. Call the salt.modules.dockerio module to provision and start an instance of the nginx image stored on the Software Repository Server on the VM returned in step 5
8. Bootstrap the salt-minion inside the container to connect to the SaltStack master
9. Call the salt.modules.state module to configure the new instance of nginx

6.5.4 Provision RabbitMQ Container

This blueprint calls a VCO workflow that does the following:

1. Query a list of all virtual machines in the Application Pod Cluster
2. Save returned list to an array
3. For each VM in the array query the vCenter Operations Manager server via the HTTP/REST VCO plugin and return the Health KPI
4. Save all returned Health KPIs with the associated VM name in an array of key:value pairs
5. Sort the KPI\VM array by highest Health score. Return the name of the associated VM.
6. Connect to the SaltStack Master server salt-api via the HTTP/REST VCO plugin
7. Call the salt.modules.dockerio module to provision and start an instance of the RabbitMQ image stored on the Software Repository Server on the VM returned in step 5
8. Bootstrap the salt-minion inside the container to connect to the SaltStack master
9. Call the salt.modules.state module to configure the new instance of RabbitMQ

6.5.5 Provision Cassandra Container

This blueprint calls a VCO workflow that does the following:

1. Query a list of all virtual machines in the Application Pod Cluster
2. Save returned list to an array
3. For each VM in the array query the vCenter Operations Manager server via the HTTP/REST VCO plugin and return the Health KPI
4. Save all returned Health KPIs with the associated VM name in an array of key:value pairs
5. Sort the KPI\VM array by highest Health score. Return the name of the associated VM.
6. Connect to the SaltStack Master server salt-api via the HTTP/REST VCO plugin
7. Call the salt.modules.dockerio module to provision and start an instance of the Cassandra image stored on the Software Repository Server on the VM returned in step 5.

8. Bootstrap the salt-minion inside the container to connect to the SaltStack master
9. Call the salt.modules.state module to configure the new instance of Cassandra

7 Configuration Management Automation

7.1 SaltStack

SaltStack is the configuration management and remote execution engine of the ShipDepot solution. Salt was designed with two main considerations, massive scalability and speed. Give the requirements of the ShipDepot system, Salt is the right choice for the solution.

7.1.1 Master and Minions

Salt is comprised of two types of daemons, masters and minions. All minions register with the master when they first come online via a SSL secured channel. The master is then able to gather information about the minions and execute commands remotely on the minion systems.

The ShipDepot system has two levels of minions and it is important to make the distinction. There is a minion daemon installed both on the CoreOS virtual machines and also in each Docker container.

The minion on the CoreOS VM is used to spawn and kill the Docker containers.

The minion in the Dicker containers is used to configure and start the services provided by each of the four Docker images.

7.1.2 salt-api

The vCenter Orchestrator workflows configured in vCloud Automation Center connect to the Salt Master server and issue commands programmatically via the salt-api REST API.

The salt-api package is not part of the core salt install and will need to be installed on the master server when it is created.

7.1.3 Salt and Docker

Salt will execute the following tasks on the CoreOS systems via the dockerio module:

1. Pull all Docker images from the Software Repository Server
2. Power on containers
3. Control port forwarding and mapping
4. Power Off containers
5. Delete containers
6. Delete images

7.1.4 Salt States

States are YAML files that salt uses to apply configuration to minion systems. States will be stored for each tier of the application and applied inside the Docker container after the container is brought online.

7.2 Software Repository

The Software Repository server will serve an NFS share that contains the Docker images.

8 Appendix A – Admission Control Policy Percentages by Host Count

Table 22

Number of Hosts	Admission Control Policy %
5	20
6	17
7	14
8	12.5
9	11
10	10
11	9
12	8
13	7.5
14	7
15	6.5
16	6.25
17	5.8
18	5.5
19	5.26
20	5
21	4.7
22	4.5
23	4.3
24	4.1
25	4
26	3.8
27	3.7
28	3.5
29	3.4
30	3.3
31	3.2
32	3.125