OpenShift 3.11 on VMware Cloud Assembly installation guide

Alpha-1-v1.2

Sajal Debnath

Staff Architect – Advanced Customer Engagements

vmware

Table of contents

Version History

VERSION HISTORY				
DATE	REV	AUTHOR	DESCRIPTION	REVIEWERS
07-24-2019	1.0	Sajal Debnath	First Edition – Alpha-1-v1.0 Release	Rafael Brito, Emad Benjamin
10-31-2019	1.1	Michael Patton	Second Edition – Alpha-1-v1.1 Release	Jon Schulz, Sajal Debnath, Rafael Brito
11-1-2019	1.2	Rafael Brito	Minor changes	Michael Patton

VMware, Inc.
3401 Hillview Ave
Palo Alto, CA 94304
www.vmware.com

Purpose

With the increased adoption of container technologies, it has become a business priority to streamline the installation and configuration of enterprise container orchestrations such as *VMware Enterprise PKS* and Red Hat OpenShift.

This document provides the details of a mechanism to deploy an enterprise production grade OpenShift 3.11 cluster via VMware Cloud Assembly Services with minimal effort and in a repeatable, fully automated fashion.

Major Decisions

Before we describe the details of the installation it is important to explain the basic decisions made in selecting the platform and other major choices made during the initial stages. Few other choices will be explained as and when we encounter them during the document.

Virtualization Platform

The main purpose of the project is to use virtualization platform as the infrastructure platform. This provides all the flexibility and advantages of virtualization namely snapshots, templates, automated deployment, live migration of the workloads, in-built HA to name a few.

Options:

- VMware vSphere Environment
- Microsoft Hyper-V
- Red Hat Enterprise Virtualization (RHEV)

Decision:

We selected VMware as the virtualization platform as this is the leading virtualization platform available in the market. Which provides most stable platform and wide range of features.

Selection: VMware vSphere Virtualization

Automation Solution

One of the major requirements is to make the deployment process repeatable and error free. Also, it should complete with minimal interaction. Hence, we needed to automate the end to end deployment of the solution. The end users should be able to provide minimum inputs at the runtime and the solution should be deployed as is (with the inputs). In VMware environment we have following two options for automation.

Options:

- vRealize Automation (vRA)
- vRealize Automation Cloud (vRAC)

Decisions:

We wanted to select a platform with "build once use many times" capability. We decided to select vRealize Automation Cloud (vRAC) as it provides stability and flexibility for deploying VM's to various endpoints including multiple cloud endpoints (VMware Cloud on AWS, AWS, Azure and Google Cloud Platform (GCP)).

By selecting vRAC, we ensure compatibility with upcoming vRealize Automation (vRA) 8.0 release.

Also, by selecting vRAC we ensure to test and deploy the workloads anywhere and consume the services without the extra overhead of managing the underlying solution. This gives us a lot of flexibility and time to focus on the development of the solution itself.

Selection: vRealize Automation Cloud (vRAC)

Deployment Platform

For this project version 1.1, as infrastructure platform, we validated both options.

Options:

- On-prem VMware vSphere Platform
- VMware Cloud on AWS (VMC on AWS)

Decision:

We selected VMware Cloud on AWS (VMC on AWS) as an endpoint as it is quick to set up and easier to manage. Our main aim was to automate OpenShift installation using existing technologies and did not want to spend much time in configuring the underlaying platform. VMC on AWS provides that functionality where without much installation and configuration we can get a full working vSphere environment.

This also gives us the flexibility to develop and test a solution which will work not only on the on prem vSphere environment but also on the cloud platforms as well.

We selected On-prem VMware vSphere Platform as an endpoint as it represents many enterprise customers. Our main aim was to automate OpenShift installation using foundational vSphere technologies. On-prem VMware vSphere Platform provides configuration familiarity in a full working vSphere environment.

This extends our flexibility to develop and test a solution which will work not only on the On-prem vSphere environment but also on VMware Cloud on AWS (VMC on AWS).

Selection: VMware Cloud on AWS (VMC on AWS)

Selection: On-prem VMware vSphere Platform

Extensibility Platform

The solution requires a lot of customization and custom scripts. We needed to select an extensibility platform.

Options:

- CloudInit
- ABX Actions
- vRealize Orchestrator (vRO) Workflows

Decisions:

Since Red Hat Enterprise Linux is the platform, so, we can use CloudInit to customize and do the installation. This will make the solution "endpoint platform agnostic" as well (Azure, GCP or AWS). But it also has certain limitations, such as, we will never know when the script in guest finishes running. We used Phone home option and found it to be not robust or accurate consistently. This is more valid in case of OpenShift installer which takes a lot of time.

For ABX actions, we need access to either AWS or Azure Function as a Service (FAAS) services. This increases the dependency.

So, finally we decided to use vRealize Orchestrator as an extensibility endpoint. This will provide most flexibility and works really well in vSphere environments.

Selection: vRealize Orchestrator (vRO)

Architecture

Deployment is a two-part process, while the first part deals with the infrastructure deployment and automation. The second part deals with the actual OpenShift installation process. The deployment layers are depicted in the following picture.

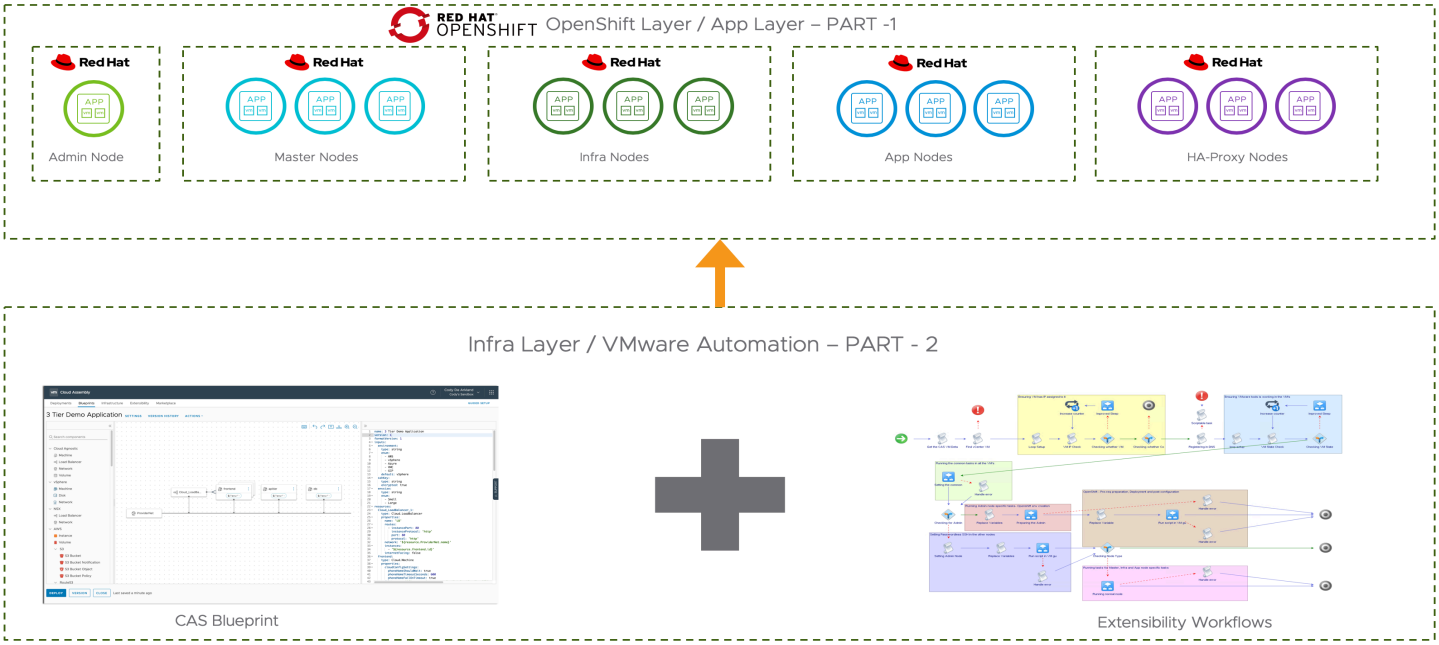

FIGURE 1: OpenShift deployment layers

The next picture shows the overall deployment flow for end to end request and solution deployment.

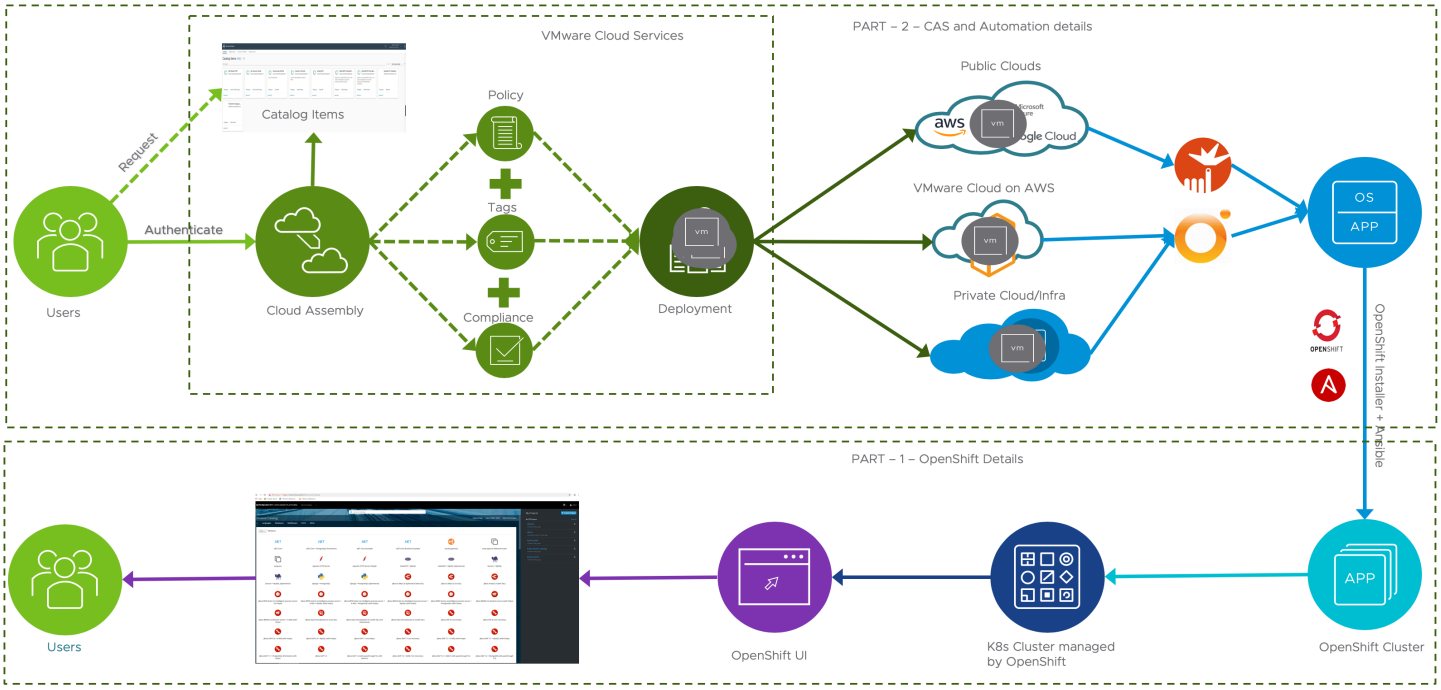

FIGURE 2: Deployment Flow.

OpenShift 3.11 Components

At minimum an enterprise grade production level OpenShift installation should have the following separate components.

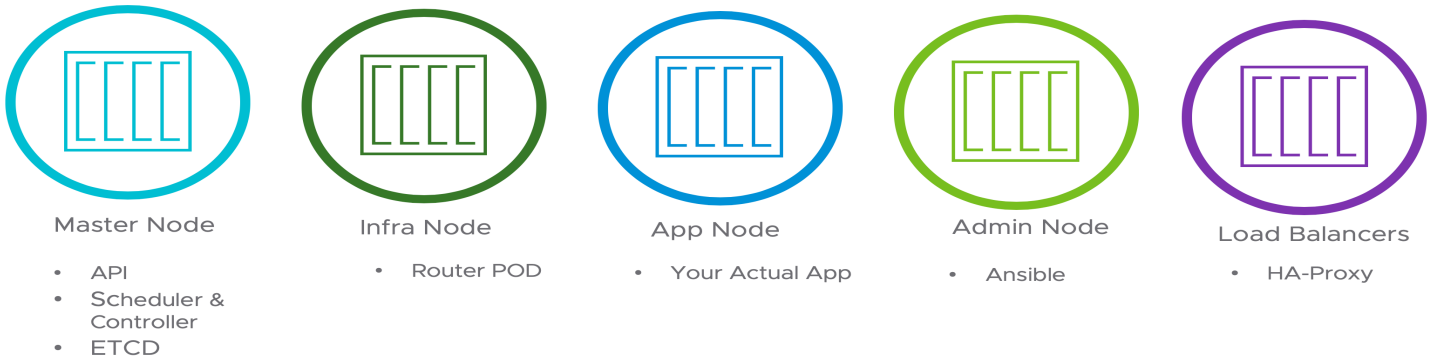

FIGURE 3: OpenShift Components

Note: In place of HA-Proxy nodes, we can use external load balancers as well. For this deployment, we used HA-Proxy nodes as load balancers.

Keeping above in mind, we used the following. The OpenShift cluster components interactions are provided in Figure 3.

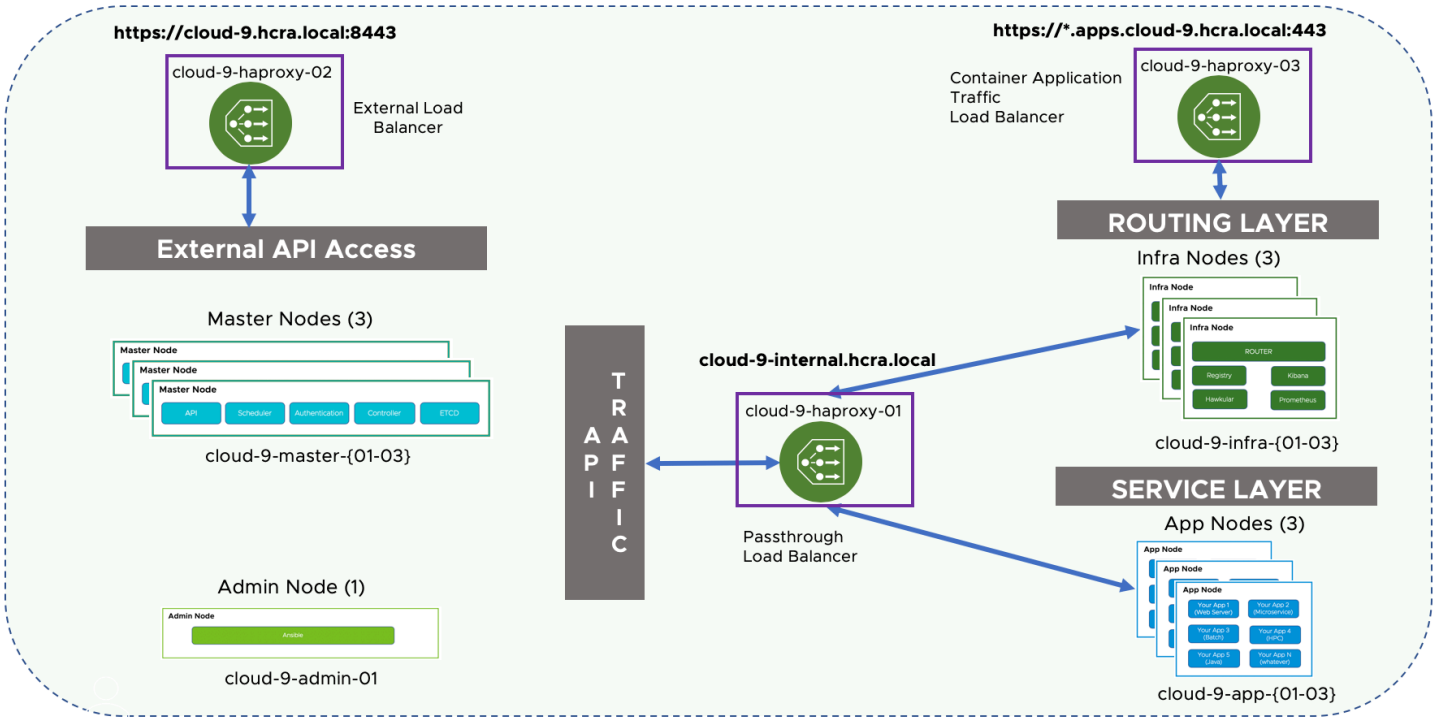

FIGURE 4: OpenShift components and their interactions

Note, the name cloud-9 is used as an example. We ask for the cluster name as input and based on that, generate the names for all the components. We add -admin-01, -master-{01-03}, -infra-{01-03} and -app-{01-03} to generate the respective component names.

Resource Requirements

Below table provides details of resource requirement for different types of nodes.

RESOURCE REQUIREMENT FOR COMPONENTS				
NODE TYPE	NO. OF NODES	VCPU / PER VM	VRAM / PER VM	DISK (MINIMUM)
Admin	1	2	8	1 disk:100 GB (/root filesystem)
Master	3	8	32	Total 4 disks: • 50 GB (/root filesystem) • 50 GB (/var) • 50 GB (docker storage) • 40GB (/var/lib/etcd)
Infra	3	4	16	Total 3 disks: • 50 GB (root filesystem) • 50 GB (/var) • 50 GB (docker storage)
Apps	3	4 (minimum) **Maximum:** The optimum number of vCPU within a NUMA boundary.	64 GB (minimum) **Maximum:** The optimum number within a NUMA boundary. OCP supports maximum 250 PODs/Node. One needs to calculate by number of PODs X average POD memory size	Total 3 disks: • 50 GB (/root filesystem) • 100 GB (/var) • 150 GB (docker storage)
HA-Proxy	3	2	16	1 disk: 50 GB

TABLE 1: Resource Requirement for Components

Note: Because of the separate disk requirements for separate node types, we used dedicated templates for each type of nodes. We could have used a single template and then defined storage details in the blueprint but that would have complicated the solution more and would need more time. So, to make the solution simpler, we used five separate templates for different node types.

Pre-Requisites

Pre-requisites for the solution are provided below.

- Operating System: Red Hat Enterprise 7.5 and/or Red Hat Enterprise 7.7
- Active Red Hat Subscription (to get the packages)
- OpenShift Version: 3.11
- An account in VMware Cloud on AWS or Private/Public vSphere environment (this solution was tested on vSphere 6.7)
- An account in VMware Cloud Services with access to vRealize Automation Cloud (vRAC): Cloud Assembly. It is assumed that rest of the configuration is already done in vRealize Automation Cloud (vRAC) to do a successful VM deployment
- SaaS enabled vRealize Orchestrator (version 7.6)
- A Windows 2016 server working as PowerShell Host (Windows PowerShell version 5.1.14393.3053)
- Windows 2016 Server (AD, DNS and NTP server roles)
- Don't use network segments 10.87.0.0/15 and 10.144.0.0/16. These are used by OpenShift internal network.

Special point to note:

- All the components depend on VM and alias FQDN name resolution. We had complete access to AD/DNS server, hence included the workflows which can automatically manage DNS records.
  If access to DNS server is not available, then the workflow needs to be modified and the Object marked as "Registering in DNS" should be deleted.
  Also, in the blueprint a minor modification can be done to ask the end users for static IP's and set the IP's in VM's accordingly. All the IP's should be pre-registered in DNS and both forward and reverse name resolutions should work.
- For VMC on AWS, enable ESXi ICMP and Https rules in Gateway firewall otherwise Guest Script Manager will not run

Installations Instructions

Provided below are the details on how to import the blueprint and workflows and to setup the environment.

There are two aspects of the installation and configuration. The first part is to setup the infrastructure (Templates, Images, PowerShell Host etc.) and the second part is to setup the variables and scripting environment.

Infrastructure Setup

The infrastructure setup is divided into multiple sub-categories a list of which is provided below. Note they are mentioned in the sequence they should be performed.

Under Cloud Assembly Services

A. Creating the Templates

B. Creating Image Mappings

C. Creating Flavor Mappings

D. Importing the Blueprint

Under vRealize Orchestrator

E. Setting up the vRO environment

F. Importing the workflow

G. Importing the PowerShell scripts

H. Creating Subscriptions in VRAC (this will be done in Cloud Assembly Services but depends on completion of step F)

Setting the Variables

A. Setting the variables and values in vRealize Orchestrator

Details of the above options are provided in the below sections.

Infrastructure Setup

We will first go through the infrastructure options.

A. Creating the Templates

For the templates we used Red Hat Enterprise Linux 7.5 / 7.7. For installation "minimum server deployment" was selected. Note, for Red Hat Enterprise Linux 7.7 template and use with vRAC – before leveraging customization specs during provisioning, it is recommended to install Perl or Cloud-Init as script configurations looks for either of these packages as a dependency.

For the resource requirements please follow the guideline provided back in Table 1.

Note:

1. Docker needs to be installed on all the different type of nodes. Only Master, Infra and App nodes need to be configured with docker storage. By default, Infra and App nodes has 3 disks attached to them. Hence, inside OS, their naming convention typically is sda, sdb and sdc.

   In a later script I have used sdc as the docker storage. This is to standardize the installation process. So, for all the nodes dev sdc needs to be free.

   For master nodes, there are 4 disks attached to it (sda, sdb, sdc and sdd). Though docker storage device is marked for last. In master nodes, I created /var/lib/etcd on sdd. That leaves sdc for docker storage.

   Please configure accordingly otherwise docker storage configuration in master nodes will fail at a later stage.

2. I used LVM for disk creation but will leave it up for the user to decide format. Once the installation was done, I used the script provided in the next page to prepare the templates.

3. I used five different VM Templates for different type of node requirements. This is to keep the installation simple (storage requirements are different).

4. Also note, in Linux environment when you partition the disks a SWAP partition will automatically be created. The size of the partition depends on the Memory size of the VM. Since the VM's has large requirement for memory hence SWAP size will typically be of 8 GB to 16GB. So, to compensate that, please allocate more storage for root partition.

Finally, I created following five templates for five different types of nodes.

TEMPLATE TO NODE TYPE MAPPING	
TEMPLATE NAME	NODE TYPE
openshift-admin-template	Admin Node
openshift-master-template	Master Nodes
openshift-infra-template	Infra Nodes
openshift-app-template	App Nodes
openshift-haproxy-template	HA-Proxy Nodes

TABLE 2: Template to Node Type Mapping

Note: You can name the templates differently. Just remember to size them accordingly.

```
#!/bin/bash
#stop logging services
/sbin/service rsyslog stop
/sbin/service auditd stop
#remove old kernels
/bin/package-cleanup --oldkernels --count=1
# clean yum cache
/usr/bin/yum clean all
#force logrotate to shrink logspace and remove old logs as well as truncate logs
/usr/sbin/logrotate -f /etc/logrotate.conf
/bin/rm -f /var/log/*-???????? /var/log/*.gz
/bin/rm -f var/log/dmesg.old
/bin/rm -f /var/log/anaconda/*
/bin/cat /dev/null > /var/log/audit/audit.log
/bin/cat /dev/null > /var/log/wtmp
/bin/cat /dev/null > /var/log/lastlog
/bin/cat /dev/null > /var/log/grubby
#remove udev hardware rules
/bin/rm -f /etc/udev/rules.d/70*
#remove uuid from ifcfg scripts
/bin/sed -i '/UUID/d' /etc/sysconfig/network-scripts/ifcfg-e*
/bin/sed -i '/^(HWADDR|UUID)=/d' /etc/sysconfig/network-scripts/ifcfg-e*
#remove SSH host keys
/bin/rm -f /etc/ssh/*key*
#remove root users shell history
/bin/rm -f ~root/.bash_history
unset HISTFILE
#remove root users SSH history
/bin/rm -rf ~root/.ssh/
#disable ipv6
echo "net.ipv6.conf.all.disable_ipv6 = 1" >> /etc/sysctl.conf
echo "net.ipv6.conf.default.disable_ipv6 = 1 = 1" >> /etc/sysctl.conf sysctl -package-cleanup
sysctl -p
echo "AddressFamily inet" >> /etc/ssh/sshd_config
```

FIGURE 5: Template preparation script

B. Creating Image Mappings

Next step is to create the Image Mappings in vRAC. Steps for that are provided below.

First login to vRAC and go to Infrastructure → Image Mappings Tab. Then create five different new Image Mappings to map with each five different types of Templates. The mapping is provided below:

TEMPLATE TO IMAGE MAPPING	
TEMPLATE NAME	**IMAGE NAME**
openshift-admin-template	openshift-admin
openshift-master-template	openshift-master
openshift-infra-template	openshift-infra
openshift-app-template	openshift-app
openshift-haproxy-template	openshift-haproxy

TABLE 3: Template to Image Mapping

Note: You can change the Image names, but in that case, you need to modify the Blueprints as well. So, for simplicity, you can keep them as is or change as per your requirement.

The final mappings look like the picture given below.

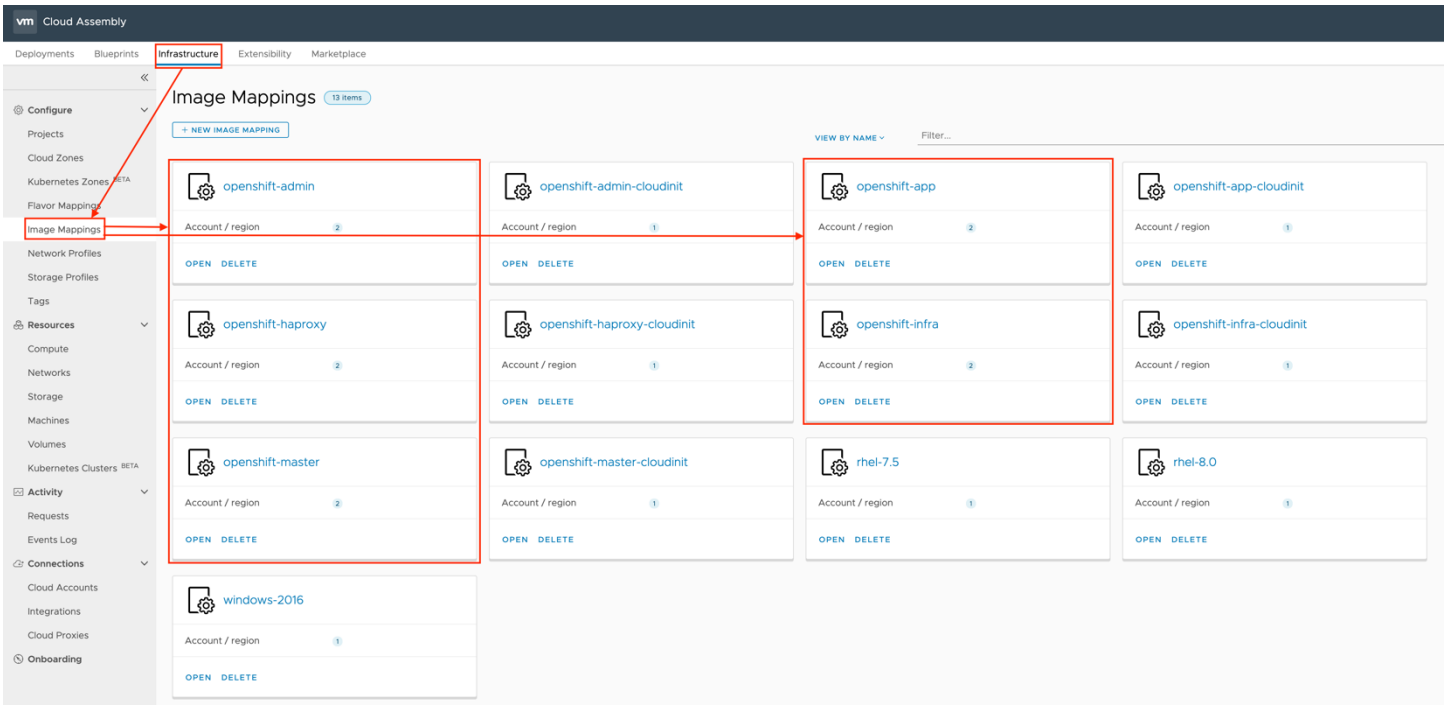

FIGURE 6: Image Mappings

More read: *Learn more about image mappings in Cloud Assembly*

C. Creating Flavor Mappings

Next step is to configure the flavor mappings.

First login to vRAC and go to Infrastructure → Flavor Mappings Tab. Then create five different new flavor mappings to map with each five different types of Templates. The mapping is provided below

TEMPLATE TO IMAGE MAPPING	
TEMPLATE NAME	**IMAGE NAME**
openshift-admin-template	openshift-admin
openshift-master-template	openshift-master
openshift-infra-template	openshift-infra
openshift-app-template	openshift-app
openshift-haproxy-template	openshift-haproxy

TABLE 4: Flavor Mappings

Note: You can change the Image names, but in that case, you need to modify the Blueprints as well. So, for simplicity, you can keep them as is or change as per your requirement.

The final mappings look like the picture given below.

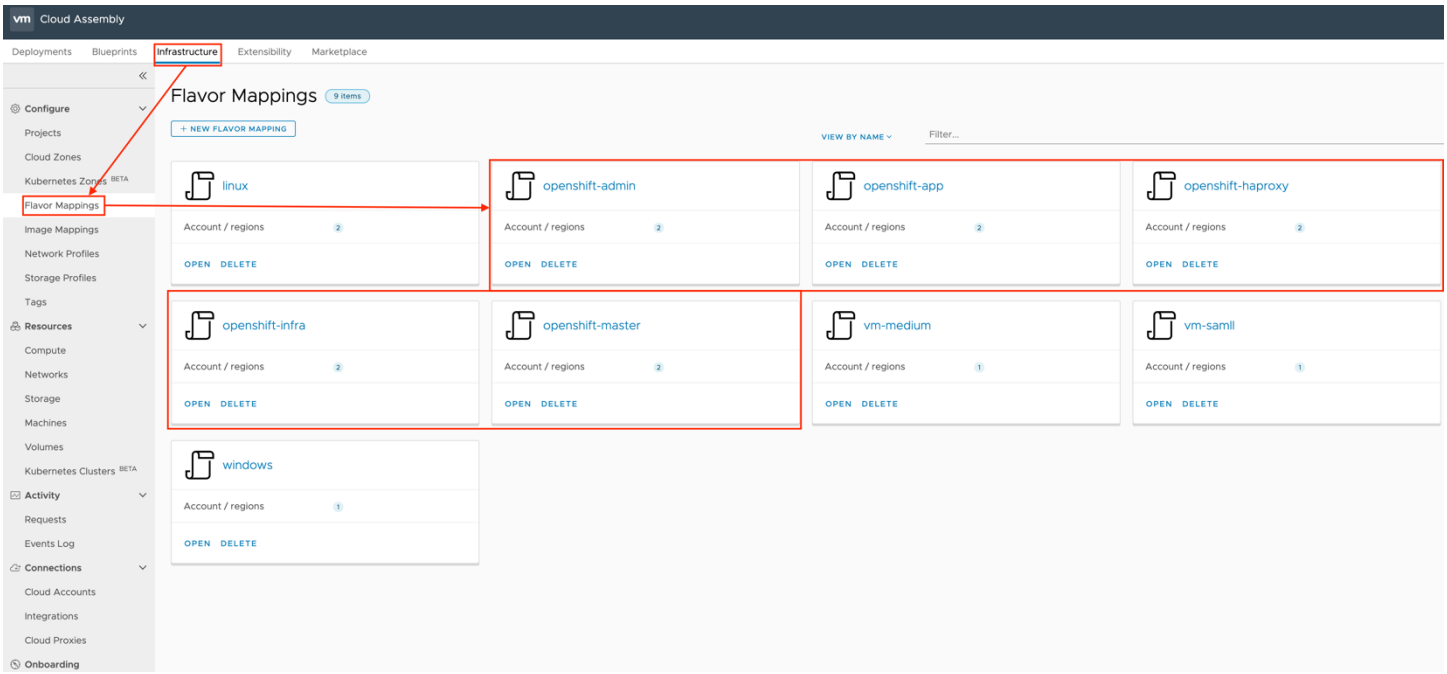

FIGURE 7: Flavor Mappings

More read: *WordPress use case: add flavor mappings*

D. Importing the Blueprint

Next, we need to import the Blueprint in vRAC. Download the "OpenShift-on-VRAC-Blueprint-Alpha-1-v1.0-1.2.yaml" file. This is the blueprint file. We need to import this file in vRAC. Steps for the import are provided below.

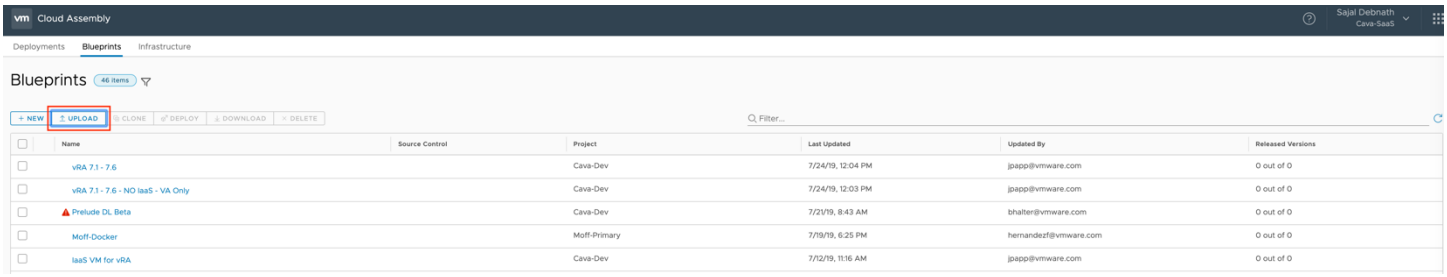

FIGURE 8: Importing Blueprint - 1

You can name the blueprint any way you want.

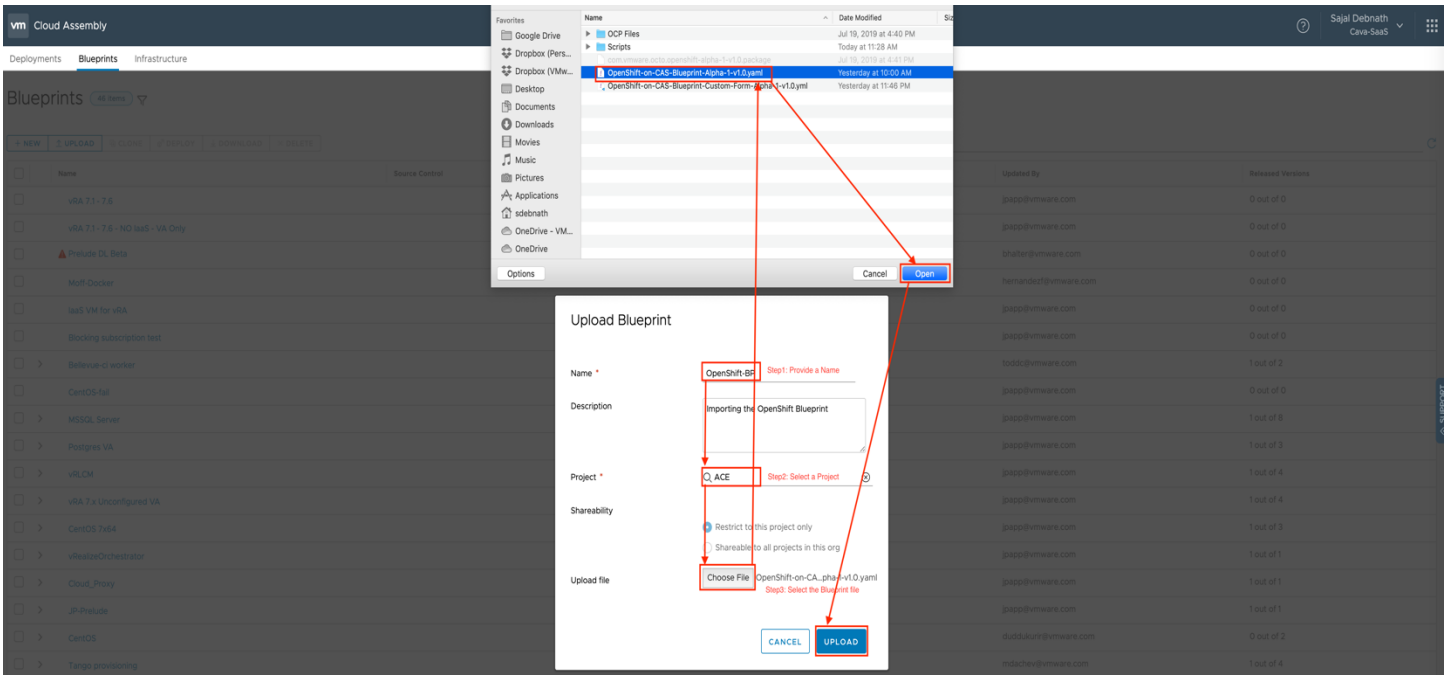

FIGURE 9: Importing Blueprint - 2

E. Setting up the vRO environment

Next is setting up the vRO environment. Before we can even configure the environment, we need to download and deploy the SaaS enabled vRO server from vRAC site. Please check the official documentation on *Configure vRealize Orchestrator integration in Cloud Assembly*.

Note: Just after the vRO deployment you need to provide an authentication provider for further configuring vRO. You have two options to use as authentication provider, first being a vCenter server and second being a vRealize Automation Server.

In VMC on AWS environment, we do not have enough permission in vCenter server to add it as an authentication provider in vRO server. So, in order to get an authentication provider, I deployed a separate standalone vRealize Automation Server and used it as authentication provider. You will not face this issue in on-prem or otherwise any other vCenter Server system.

Once the vRO server is deployed and the integration is configured, we need to configure the following things in the vRO server.

a) Add vCenter server endpoint

b) Add the PowerShell host endpoint

c) Add the VRAC endpoint

Provided below are the details of the steps required to configure the endpoints:

a) Add vCenter Server

Use the following steps to add the vCenter Server as an endpoint. First go to Library → vCenter → Configuration → Add a vCenter Server Instance. Right Click on the workflow and select "Start Workflow".

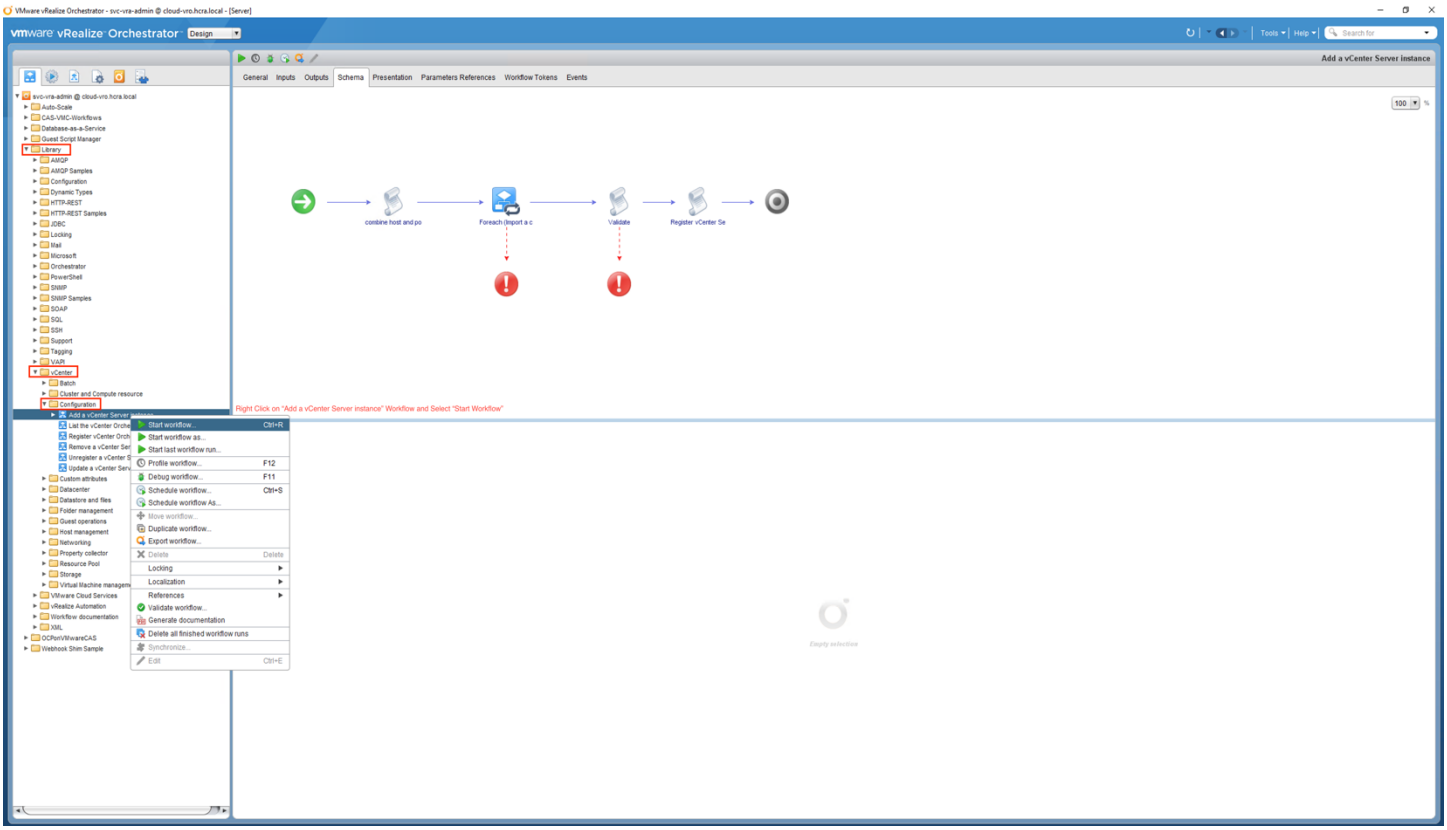

FIGURE 10: Running the "Add a vCenter Server Instance" Workflow

Next, in the new popped up window provide all the required details. Please note the options selected below.

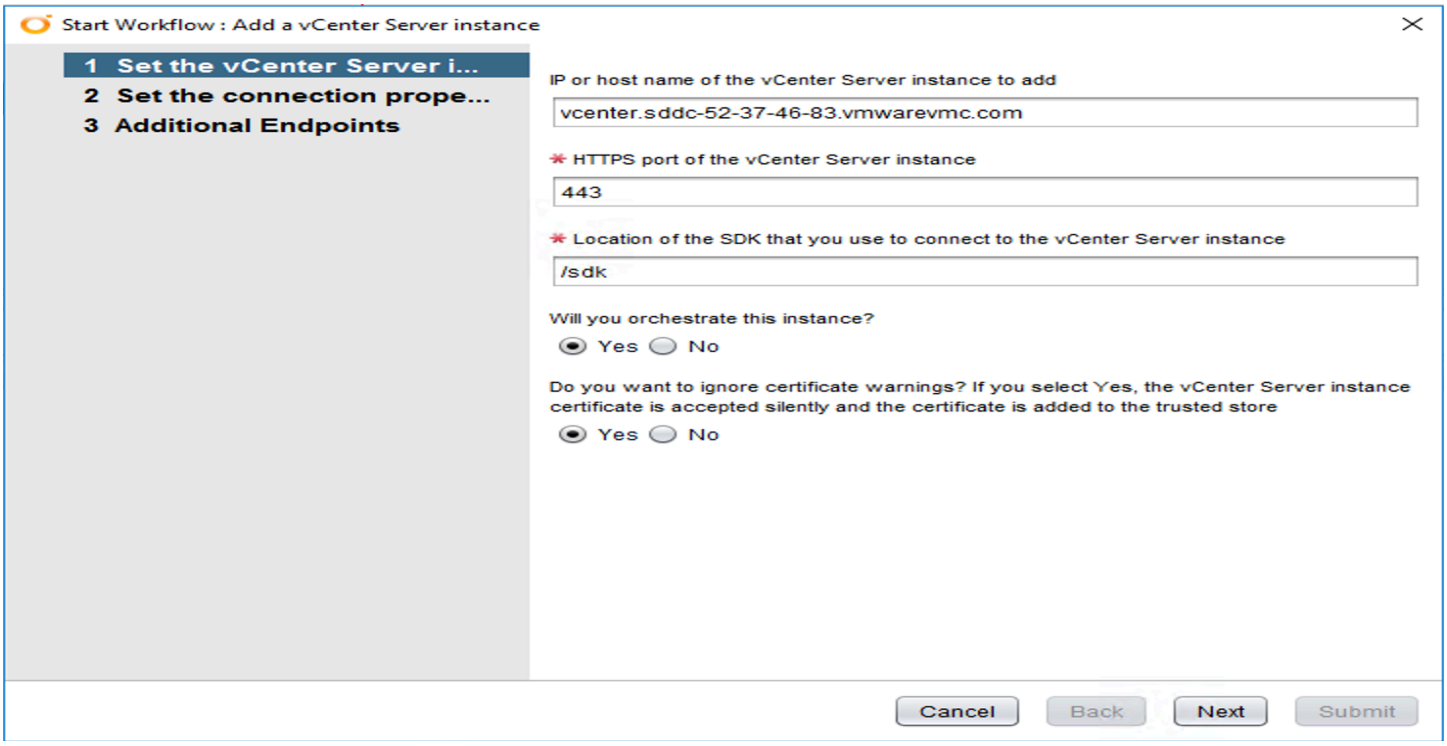

FIGURE 11: Input Page 1

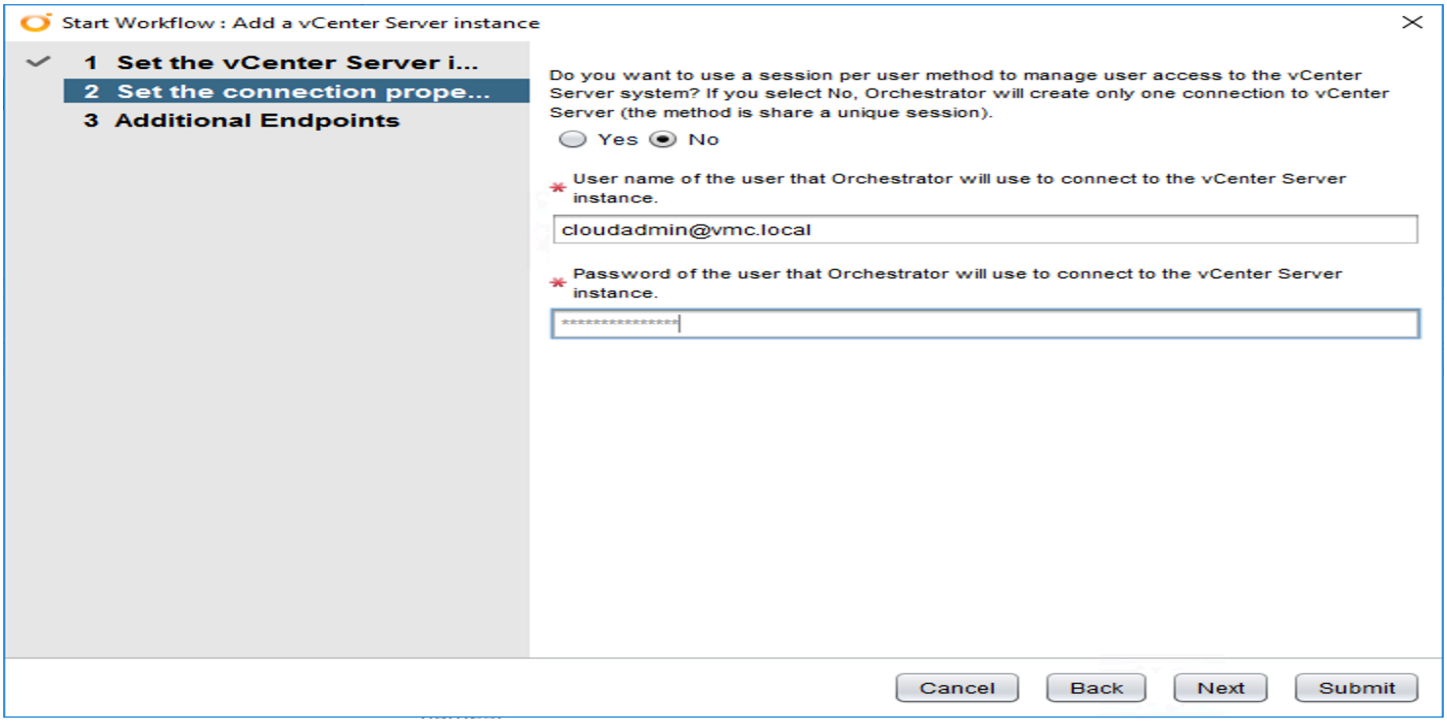

FIGURE 12: Input Page 2

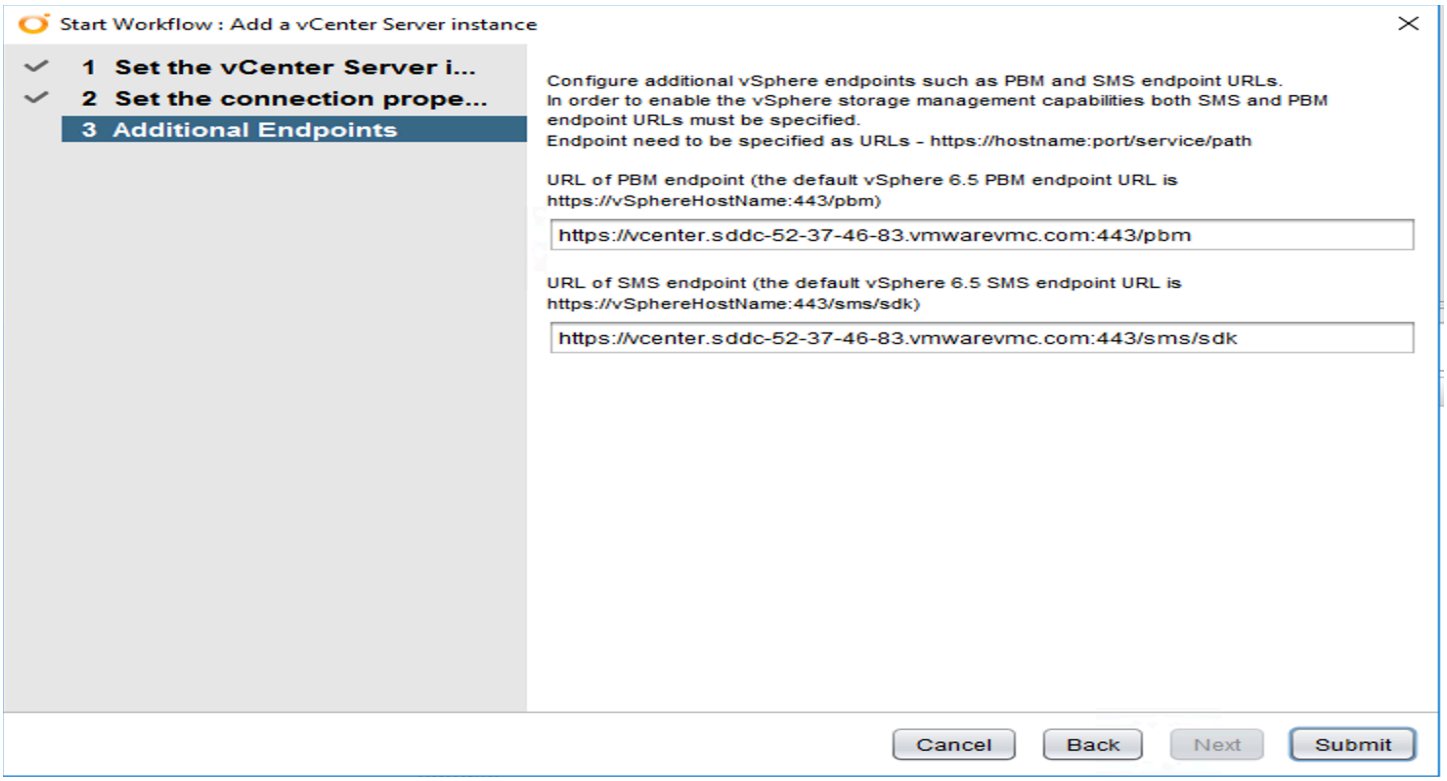

FIGURE 13: Submit the workflow

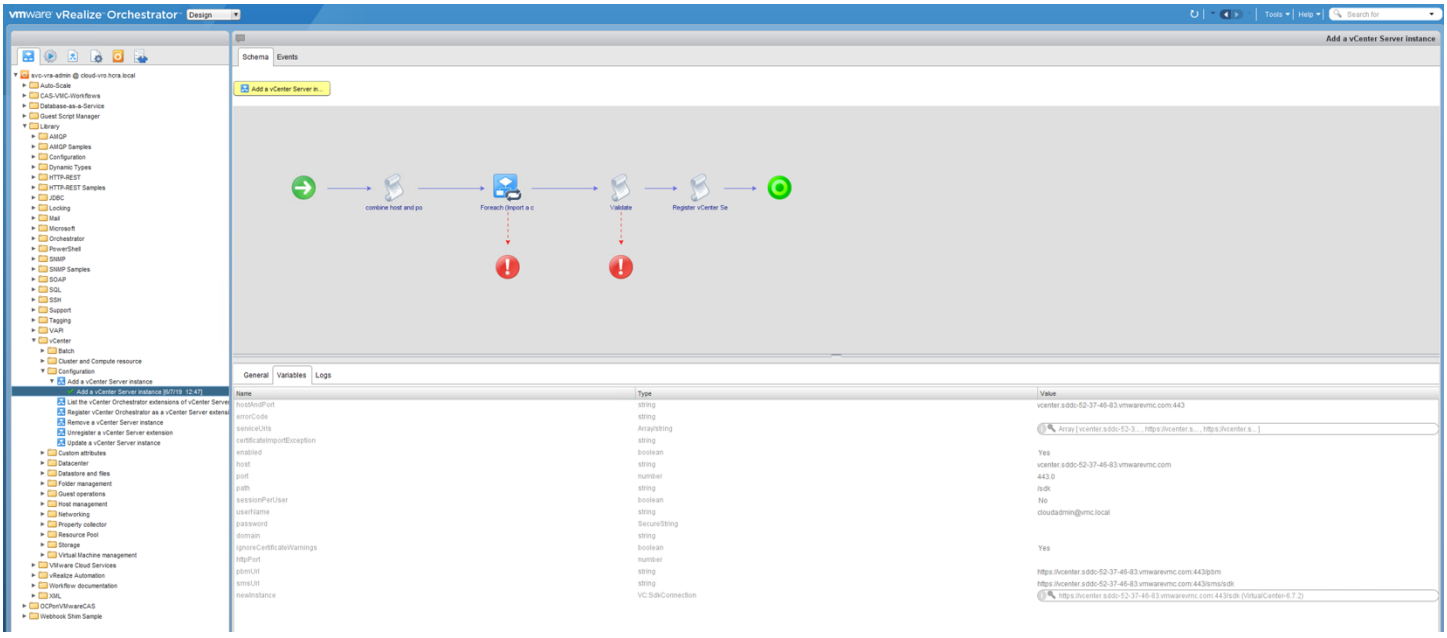

FIGURE 14: Successful completion of the workflow

b) Add the PowerShell host endpoint

Please follow the detailed guide *Introduction to the VMware vRealize Orchestrator PowerShell Plug-In* to configure PowerShell Host endpoint in vRO. The document has detailed instructions on how to configure a PowerShell host and how to use it to run PowerShell Hosts. In configuring the PowerShell Plug-in, it is recommended to use the User Principle Name (e.g. *username@domain.com*) vs. domain\user

The link below provides configuration assistance related to winrm and krb5.conf.

URL: *https://docs.vmware.com/en/vRealize-Automation/7.5/com.vmware.vrealize.orchestrator-use-plugins.doc/GUID-1A829CCF-A147-422E-80B3-0819F8D6E6AE.html*

We used the PowerShell host to run the PowerShell scripts required for managing DNS servers. Details of the scripts will be provided in a later section.

Note: For the PowerShell host configuration we used Kerberos authentication.

```
C:\Windows\system32>winrm get winrm/config/service
Service
    RootSDDL = O:NSG:BAD:P(A;;GA;;;BA)(A;;GR;;;IU)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)
    MaxConcurrentOperations = 4294967295
    MaxConcurrentOperationsPerUser = 1500
    EnumerationTimeoutms = 240000
    MaxConnections = 300
    MaxPacketRetrievalTimeSeconds = 120
    AllowUnencrypted = true
    Auth
        Basic = false
        Kerberos = true
        Negotiate = true
        Certificate = false
        CredSSP = false
        CbtHardeningLevel = Relaxed
    DefaultPorts
        HTTP = 5985
        HTTPS = 5986
    IPv4Filter = *
    IPv6Filter = *
    EnableCompatibilityHttpListener = false
    EnableCompatibilityHttpsListener = false
    CertificateThumbprint
    AllowRemoteAccess = true
```

FIGURE 15: Status of the WinRM configuration in PowerShell host

For this configuration I have put all the scripts under "C:\scripts" directory. The following scripts are used for this environment:

- registerCname.ps1 - Script to register alias names to DNS server
- registerDns.ps1 - Script to register host names to DNS server
- unregisterCnames.ps1 - Script to un-register alias names from DNS server
- unregisterDns.ps1 - Script to un-register alias names from DNS server

Add the vRAC endpoint

The last step in the configuration is to add vRAC as an endpoint. For the addition we need the following information.

Name: Name of the endpoint. You can use name of your choice (e.g. vRAC-VMC-HCRA)

cspUri: *https://console.cloud.vmware.com*

cloudApiUri: *https://api.mgmt.cloud.vmware.com/iaas/login*

refreshToken: Something like "wTTxxxxxxxxxxsybXNPfBxxxxxxxxxxw5riBlWC7rw2dG940BQ2EROKxxxxxxxxxxx"

Follow the document on how to get the refresh Token: *Generating a Cloud Services Platform API Token for Cloud Automation Services*.

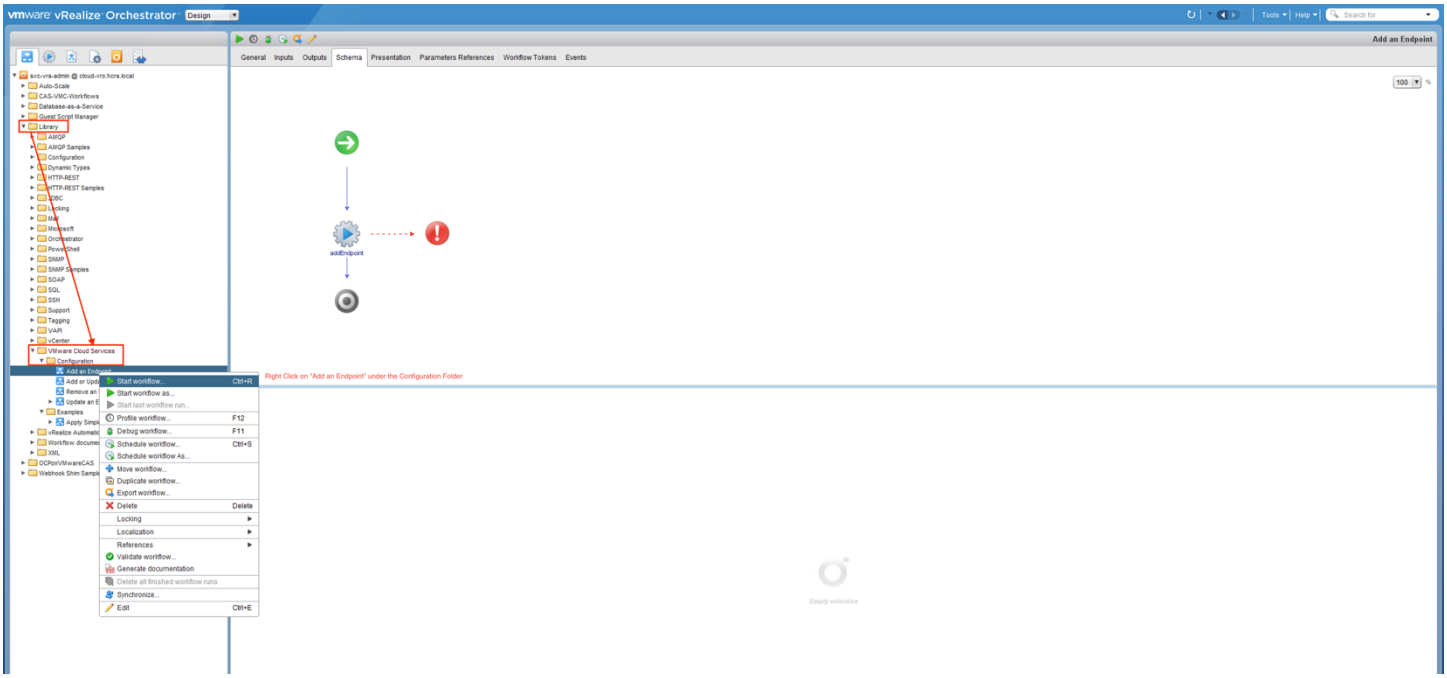

FIGURE 16: Addition of vRAC Endpoint workflow – 1

Provide the collected information in the new window and click on the submit window

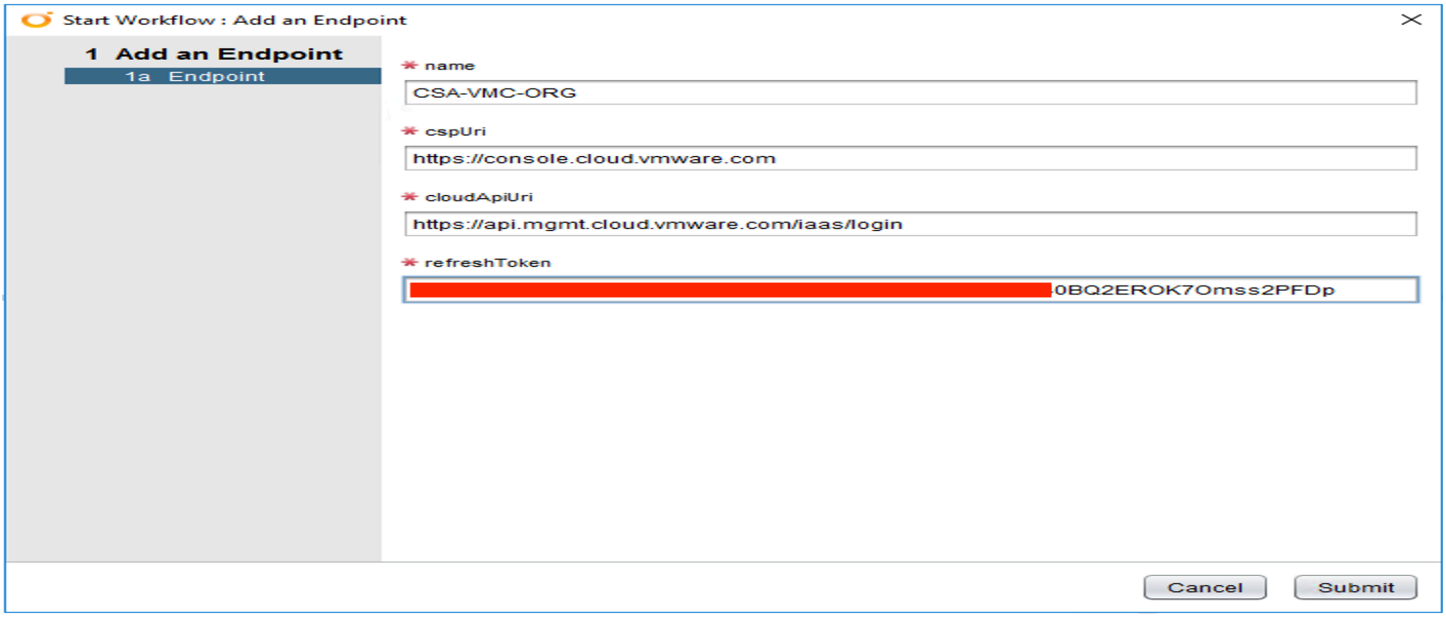

FIGURE 17: Addition of VRAC Endpoint workflow – 2

F. Importing the workflow

Next step in the setup process is to import the workflow in vRO server. Download the "com.vmware.octo.openshift-alpha-1-v1.1.package" file in the local system. This is the vRO package file that we need to import in the environment. The process is shown in the given image below.

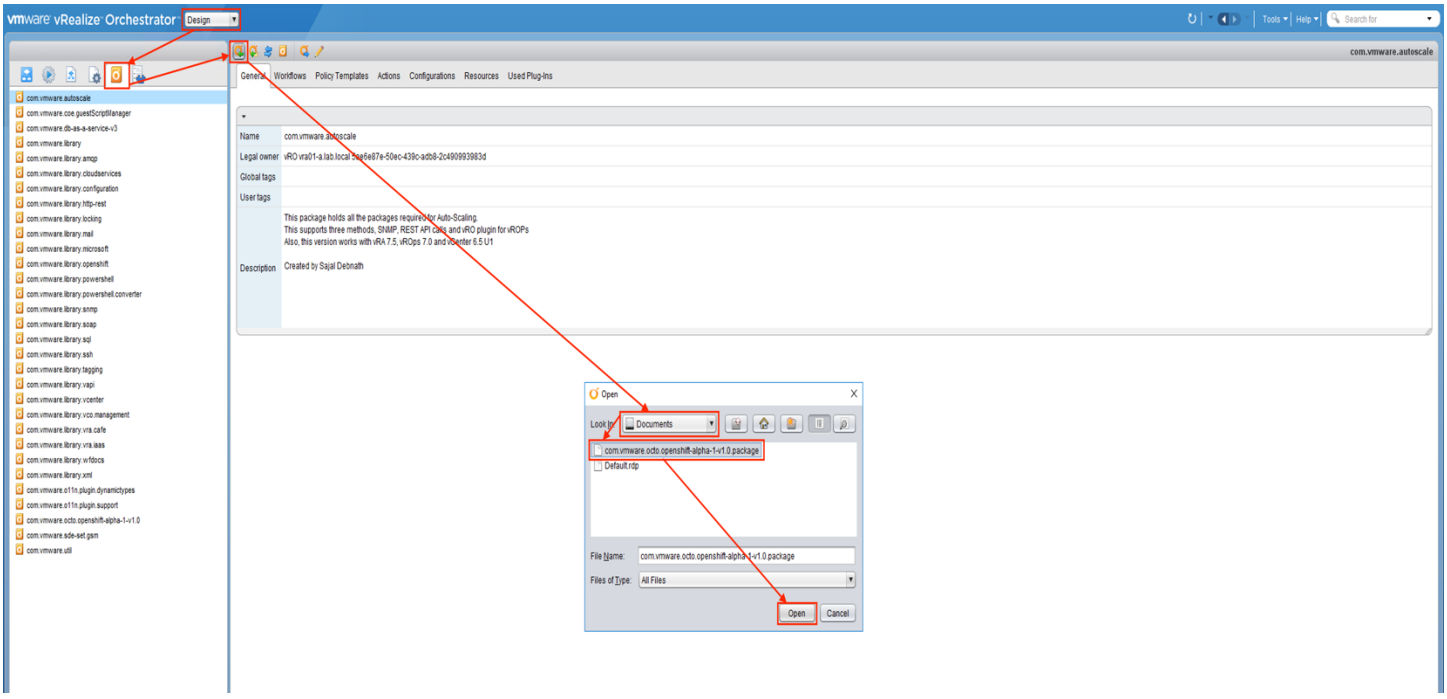

FIGURE 18: Import the Workflow – 1

Once the package is imported you will see a folder named “OCPonVMwarevRAC” and the following three workflows under the folder.

- OpenShift-on-VMware-VRAC-Alpha-1-v1.1
- Set VM Name
- Un-Register-VM-from-DNS-RHN-Alpha-1-v1.1

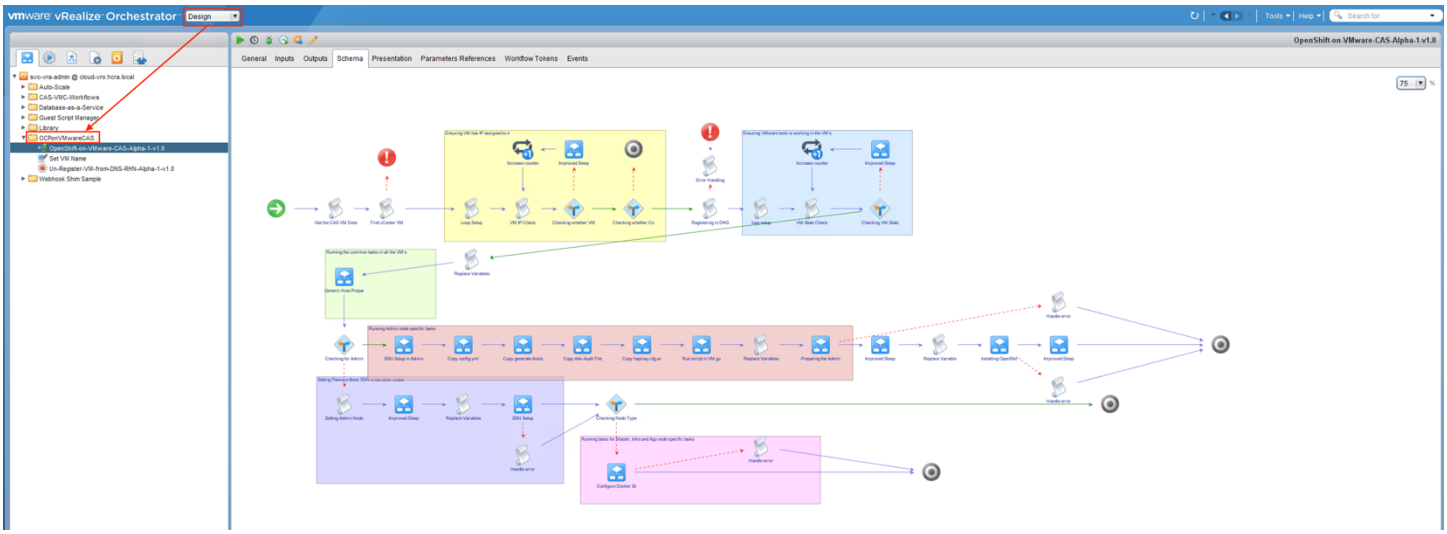

FIGURE 19: Import the Workflow – 2

G. Importing the PowerShell scripts

If you decided to use the DNS management scripts. Then download the package "PowerShell Scripts.zip". This zipped file further contains the following four PowerShell scripts.

- registerDns.ps1 – Registers the VM names to DNS server
- registerCname.ps1 – Registers the aliases to DNS server
- unregisterDns.ps1 – Unregisters the VM names from DNS server
- unregisterCname.ps1 – Unregisters the aliases from DNS server

For my environment the PowerShell scripts are places under "C:\scripts" directory on PowerShell host. You can select any other location in PowerShell host to put these files. But in that case update the scripts location in the relevant variable in vRO workflow.

H. Creating Subscriptions in vRAC

The last point in the infrastructure setup is setting up the subscriptions in vRAC. Remember this step needs to be done once the workflows are imported into vRO and one data collection completes in vRAC. Otherwise the workflows will not be visible in vRAC.

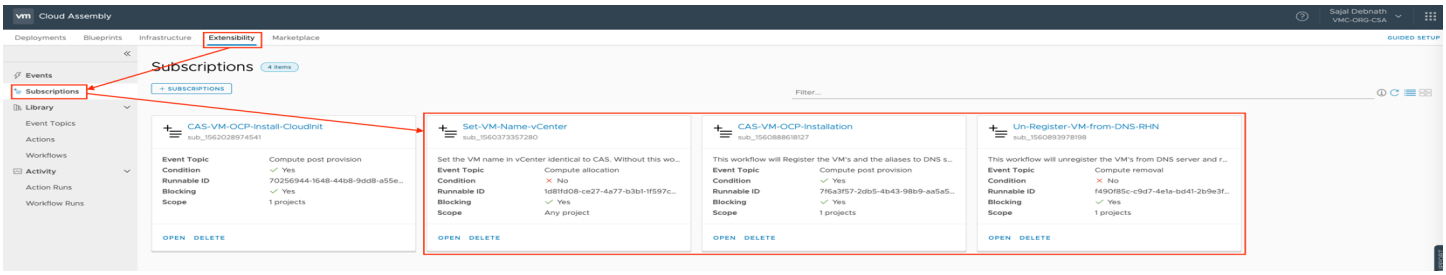

FIGURE 20: Subscriptions

We need to create three subscriptions. Details of the subscriptions are provided below. Remember you can set any name you want but the mapping to vRO workflow needs to be correct. For details on how to create subscriptions in VRAC please check the document *Create an extensibility subscription* and in *Define Workflow Subscription Details*.

First Subscription: Set-VM-Name-vCenter

Purpose of this workflow is to set the VM name in vCenter to match the name generated in blueprint (based on the user input). If this is not set, then the VM names in vCenter will have an UUID attached to the end of the VM name.

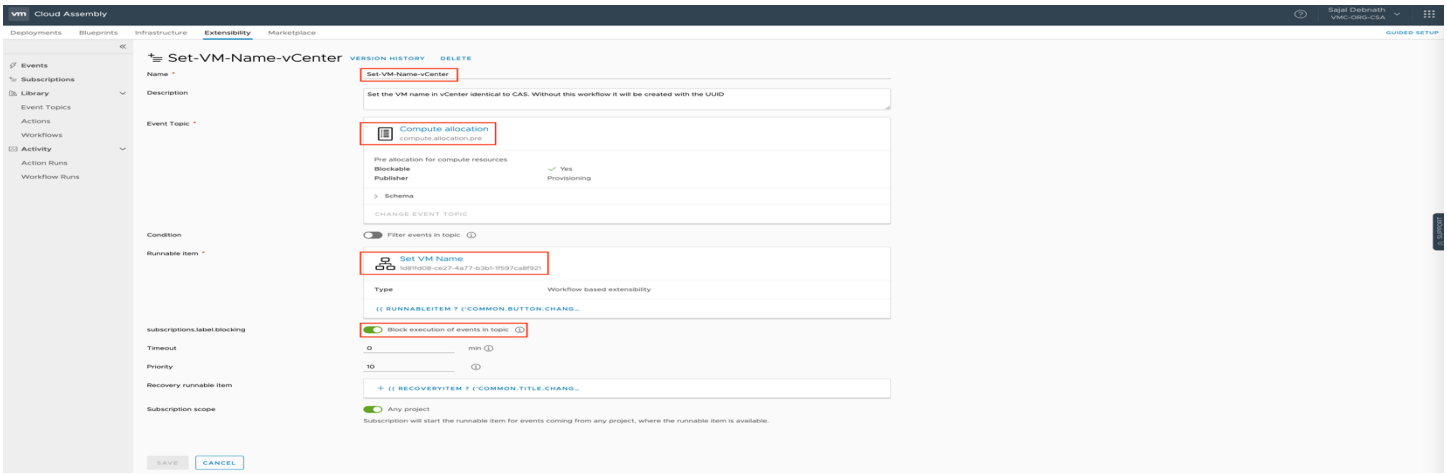

FIGURE 21: Subscriptions – 1

Note, the Subscription type. It is "Compute Allocation" which is of "compute.allocation.pre". So, this is for pre-allocation of resources. Next select the runnable item and select the "Set VM Name" workflow from vRO. Select "Block execution of events in topic" and select the subscription scope. I have selected "Any Project", so any request in any project will run this workflow.

Second Subscription: vRAC-VM-OCP-Installation

This is the master workflow. This will set the DNS names dynamically and install the OCP components.

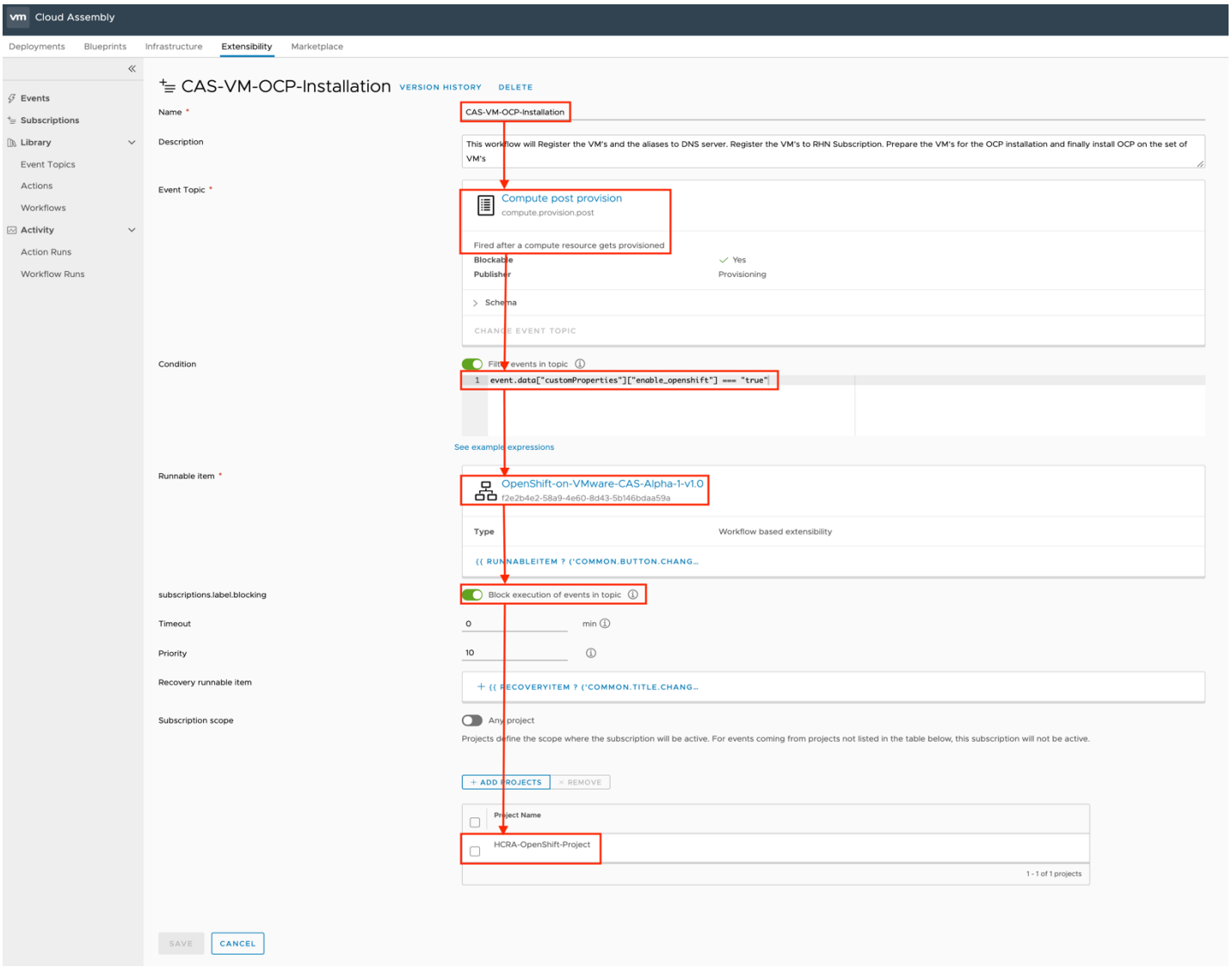

FIGURE 22: Subscriptions – 2

Note, the Subscription type. It is "Compute post provisions" which is of "compute.provision.post type". So, this is for post-allocation of resources. Next select the scope for the event. I do not want to run this workflow for every request. So, I put a condition

event.data["customProperties"]["enable_openshift"] === "true"

In the blueprint I put an extra parameter "enable_openshift". Here I am checking for that parameter. Next, select the runnable item and select the "OpenShift-on-VMware-VRAC-Alpha-1-v1.0" workflow from vRO. Select "Block execution of events in topic" and then select the subscription scope. I have selected a particular project as the scope as I expect the requests to come from a single project. Please change according to your environment.

Third Subscription: Un-Register-VM-from-DNS-RHN

This is the last subscription. This will unregister the DNS entries for the VM's and their aliases. It will also unregister the VM from RHN Subscription.

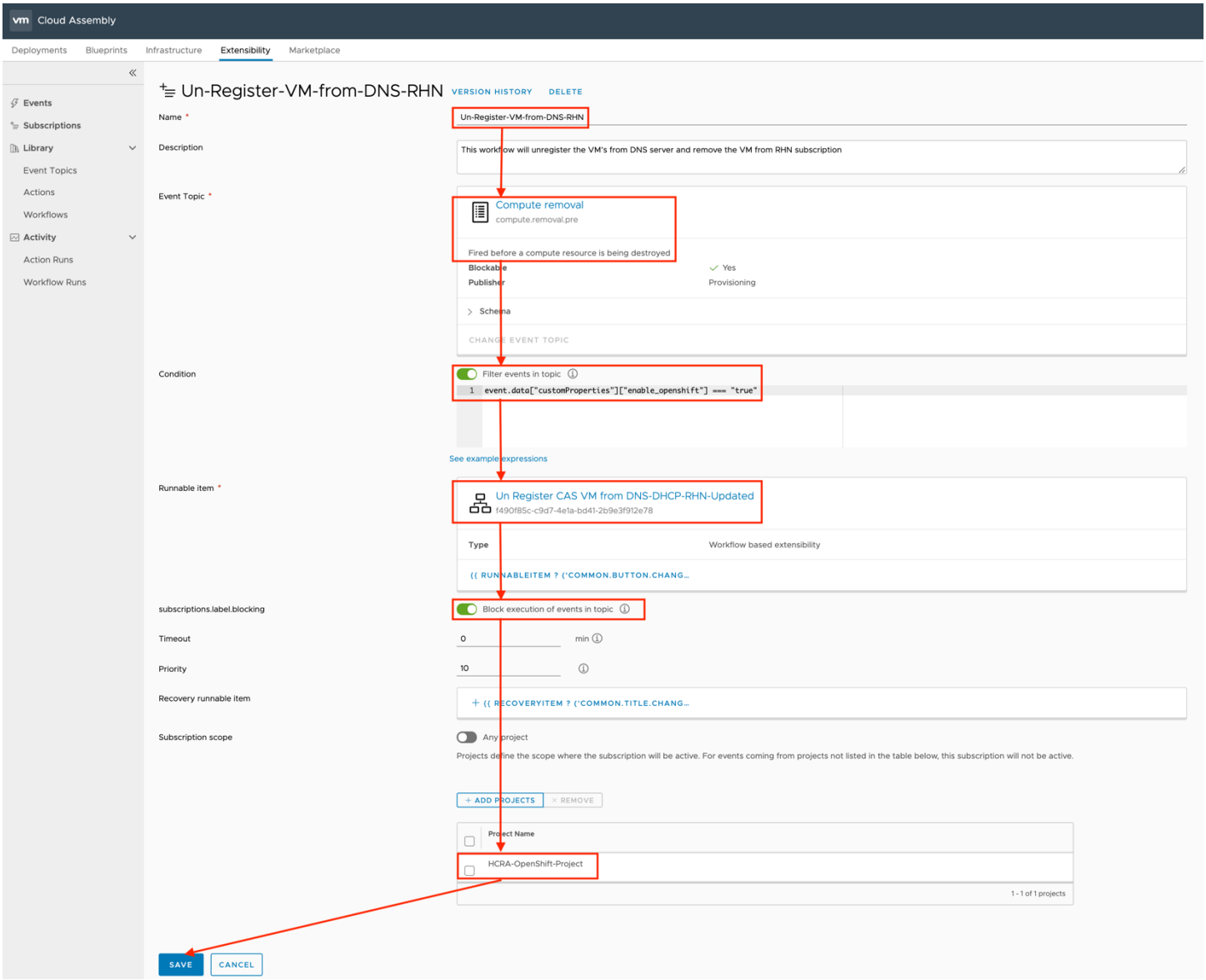

FIGURE 23: Subscriptions - 3

Note the Subscription type. It is "Compute removal" which is of "compute.remove.pre type". So, this is for pre removal of the items. Next select the scope for the event. I do not want to run this workflow for every request. So, I put a condition

event.data["customProperties"]["enable_openshift"] === "true"

I am using a custom property "enable_openshift" defined in the blueprint to filter out the items.

Next, select the runnable item and select the "Un-Register-VM-from-DNS-RHN-Alpha-1-v1.1" workflow from vRO. Select "Block execution of events in topic" and then select the subscription scope. I have selected a particular project as the scope as I expect the requests to come from a single project. Please change according to your environment.

Note: The name of the workflow in the above screenshot is different from vRO. Make a special note of the workflow ID. In this case it is "f490f85c-c9d7-4e1a-bd41-2b9e3f912e78". This is a unique id and would be same everywhere. Till the time it is same, name difference is not a problem.

With the above steps we completed the infrastructure setup. Next, we need to setup the variables.

Setting the Variables

To proceed, we need to set few variables in vRO server workflows. We don't need to set any variable in the workflow "Set VM Name". The blueprint asks from the end users a name for the OpenShift Cluster and all the names of the VM's are generated based on that. "Set VM Name" workflow sets the VM names in vCenter accordingly.

A. OpenShift-on-VMware-VRAC-Alpha-1-v1.1

This is the master workflow and has a lot of components in it. Based on the tasks they perform; the objects are grouped and colored accordingly. This workflow runs post the deployment and customization of the VM's.

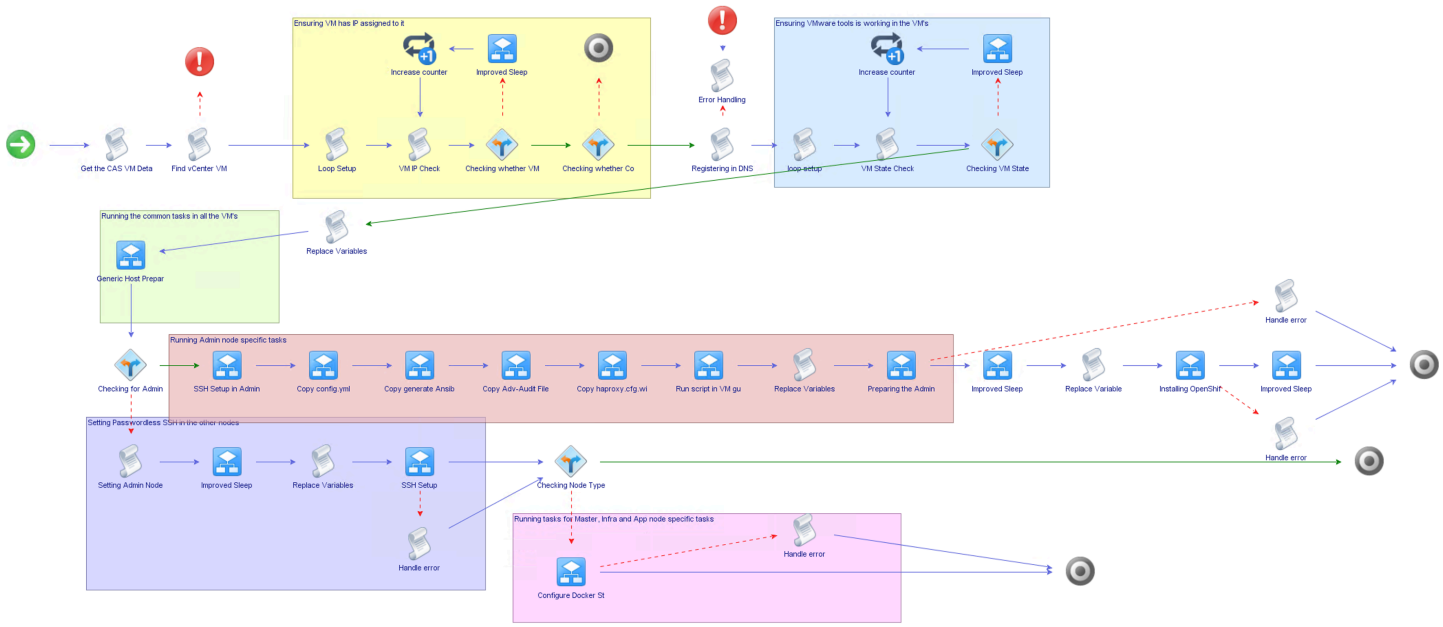

FIGURE 24: Master Workflow

As input this workflow takes only a single input "inputProperties". This comes from vRAC and has the EBS Event Data.

Details of attribute parameters which needs to be set are provided in the next table. Remember these needs to be set in vRO server.

OPENSHIFT-ON-VMWARE-VRAC-ALPHA-1-V1.0 - ATTRIBUTE PARAMETERS TO SET				
NAME	TYPE	TO BE SET	VALUE	DESCRIPTION
host	PowerShell: PowerShellHost	YES	Set it to the configured PowerShell Host	PowerShell Host
externalScript	string	YES	C:\scripts\registerDns.ps1 (change it as per env)	External Script to run inside the Powershell host
externalscriptAlias	string	YES	C:\scripts\registerCname.ps1 (change it as per env)	External Script to run inside the Powershell host
dnsHost	string	YES	Set it as per DNS server	DNS HostName
dnsZone	string	YES	Set it as per DNS Zone	DNS Zone name where to add the DNS entries
hostAccount	string	YES	Host account in the DNS server with which to run the PowerShell script to add the DNS entries	DNS Host account with which to run the PowerShell commands. Ideally administrative user account **
hostPassword	SecureString	YES	DNS host account password	DNS Host Account Password
poolID	string	YES	Set it as per Subscription values	Red Hat Subscription Pool ID which has OpenShift packages
vmUsername	string	YES	root (It should be set to root)	VM user name with which to run the guest script
vmPassword	SecureString	YES	root / user password	Password of the VM user

TABLE 5: Attribute parameters to set for master workflow

** Ensure DNS add/remove rights are properly assigned to user that will be performing registration and un-registration.

Note: For a detailed list of parameters check Table-7 under Appendix section.

B. Un-Register-VM-from-DNS-RHN-Alpha-1-v1.1

This workflow deletes the DNS entries for the VM's and then unregisters the VM's from Red Hat Network (RHN). It runs before the VM's are destroyed.

As input this workflow takes only a single input "inputProperties". This comes from vRAC and has the EBS Event Data. Details of attribute parameters which needs to be set are provided in the next table. Remember these needs to be set in vRO server.

UN-REGISTER-VM-FROM-DNS-RHN-ALPHA-1-V1.0 - ATTRIBUTE PARAMETERS TO SET				
NAME	TYPE	TO BE SET	VALUE	DESCRIPTION
host	PowerShell: PowerShellHost	YES	Set it to the configured PowerShell Host	PowerShell Host
externalScript	string	YES	C:\scripts\unregisterDns.ps1 (change it as per env)	External Script to run inside the Powershell host
externalscriptAlias	string	YES	C:\scripts\unregisterCname.ps1 (change it as per env)	External Script to run inside the Powershell host
dnsHost	string	YES	Set it as per DNS server	DNS HostName
dnsZone	string	YES	Set it as per DNS Zone	DNS Zone name where to add the DNS entries
hostAccount	string	YES	Host account in the DNS server with which to run the PowerShell script to add the DNS entries	DNS Host account with which to run the PowerShell commands. Ideally administrative user account
hostPassword	SecureString	YES	DNS host account password	DNS Host Account Password
poolID	string	YES	Set it as per Subscription values	Red Hat Subscription Pool ID which has OpenShift packages
vmUsername	string	YES	root (It should be set to root)	VM user name with which to run the guest script
vmPassword	SecureString	YES	root / user password	Password of the VM user

TABLE 6: Attribute parameters to set for Un-Register-VM-from-DNS-RHN-Alpha-1-v1.0 workflow

Note: For a detailed list of parameters check Table-8 under Appendix section.

Provided below are the screenshots detailing on how you can modify the variable parameters in vRO.

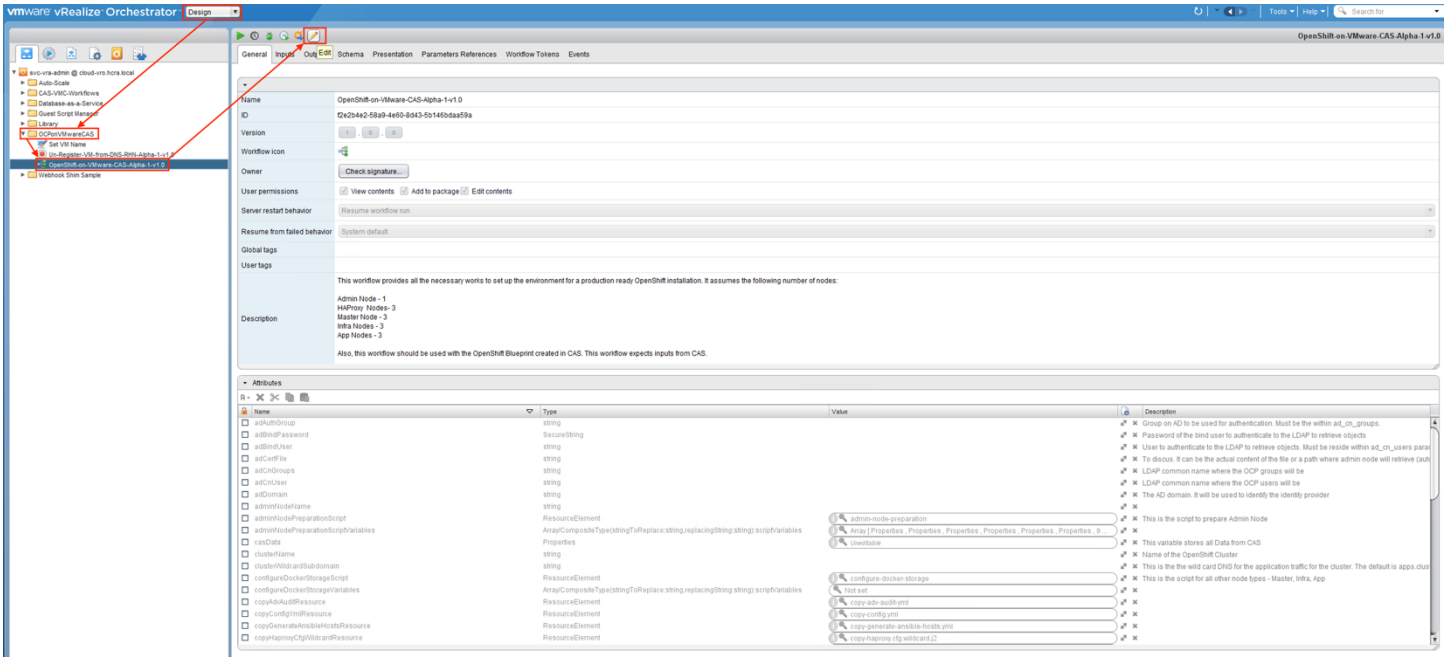

FIGURE 25: Variable Replace – 1

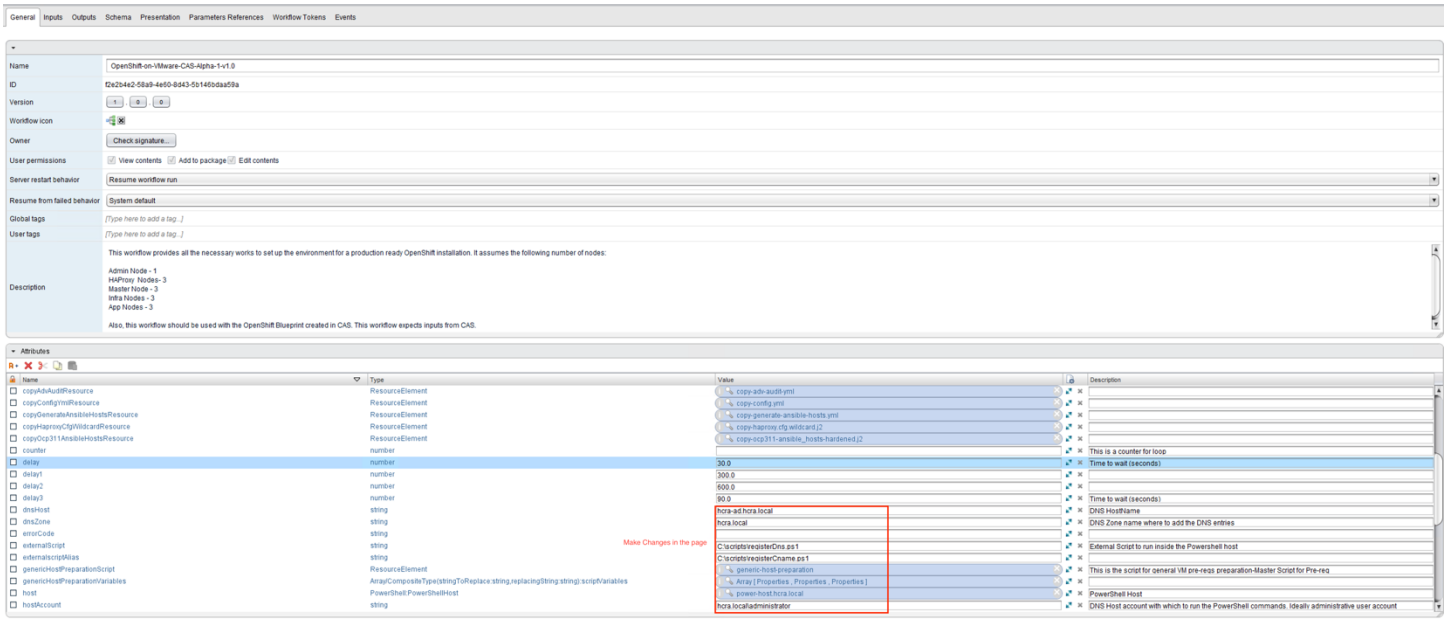

FIGURE 26: Variable Replace – 2

Deployment

Once everything is installed and configured, it is time for a deployment. There are two ways we can initiate a deployment.

First and preferable method is through "VMware Service Broker". The blueprint needs to be versioned and released to Service Broker. In the Service Broker it needs to be shared to the Catalog items. For details check the document *Setting up Service Broker for your organization* . Once all these are done, we can make a request from the Catalog items.

Second method is to make a request directly from the blueprint from VMware Cloud Assembly.

For this document I have taken the first route and using the Catalog item in Service Broker. I have also created a custom form for the catalog item. If you want to use the same customer form, then download and import the custom form OpenShift-on-VRAC-Blueprint-Custom-Form-Alpha-1-v1.1.yml. Remember this step is optional. You can directly make a request from Cloud Assembly Blueprints Page. For details check the document *Customize a Service Broker request form*. Once the form is imported and Catalog item is set. Make a request for the Catalog item to make a deployment.

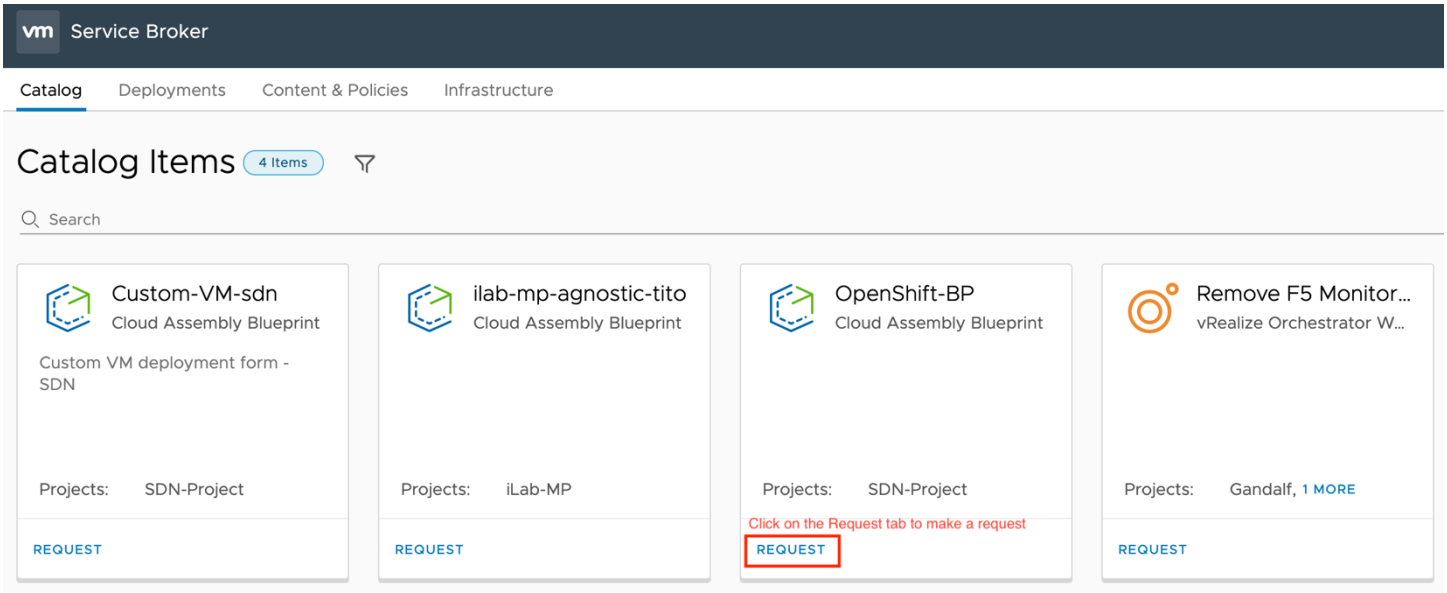

FIGURE 27: Catalog Item Request - 1

Required information is divided into three categories.

- Deployment Information – Required information for the deployment
- Domain Information – DNS and cluster domain and related information
- User Information – User related information

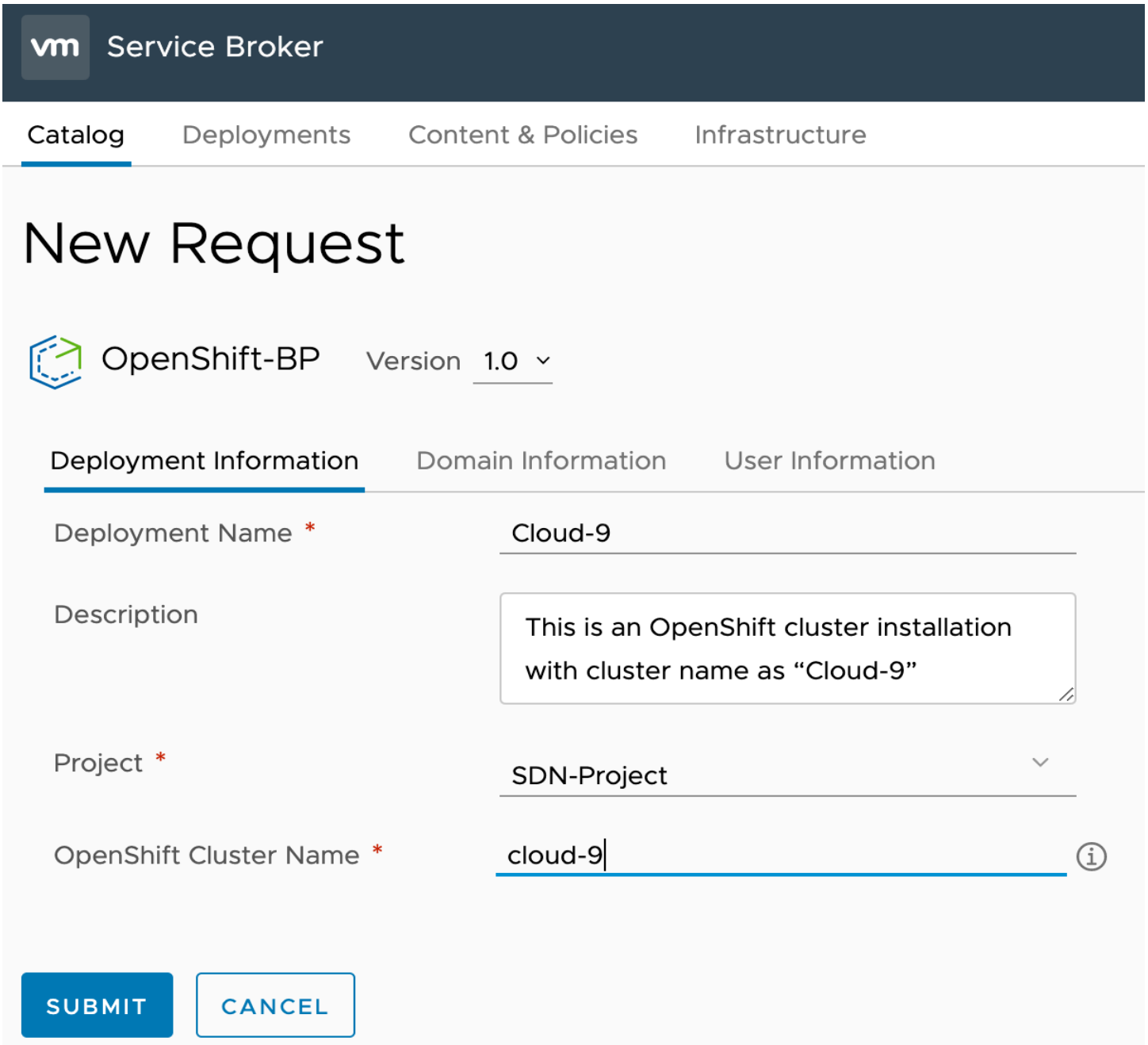

FIGURE 28: Catalog Item Request - 2

Provide the details in the “Deployment Information” tab.

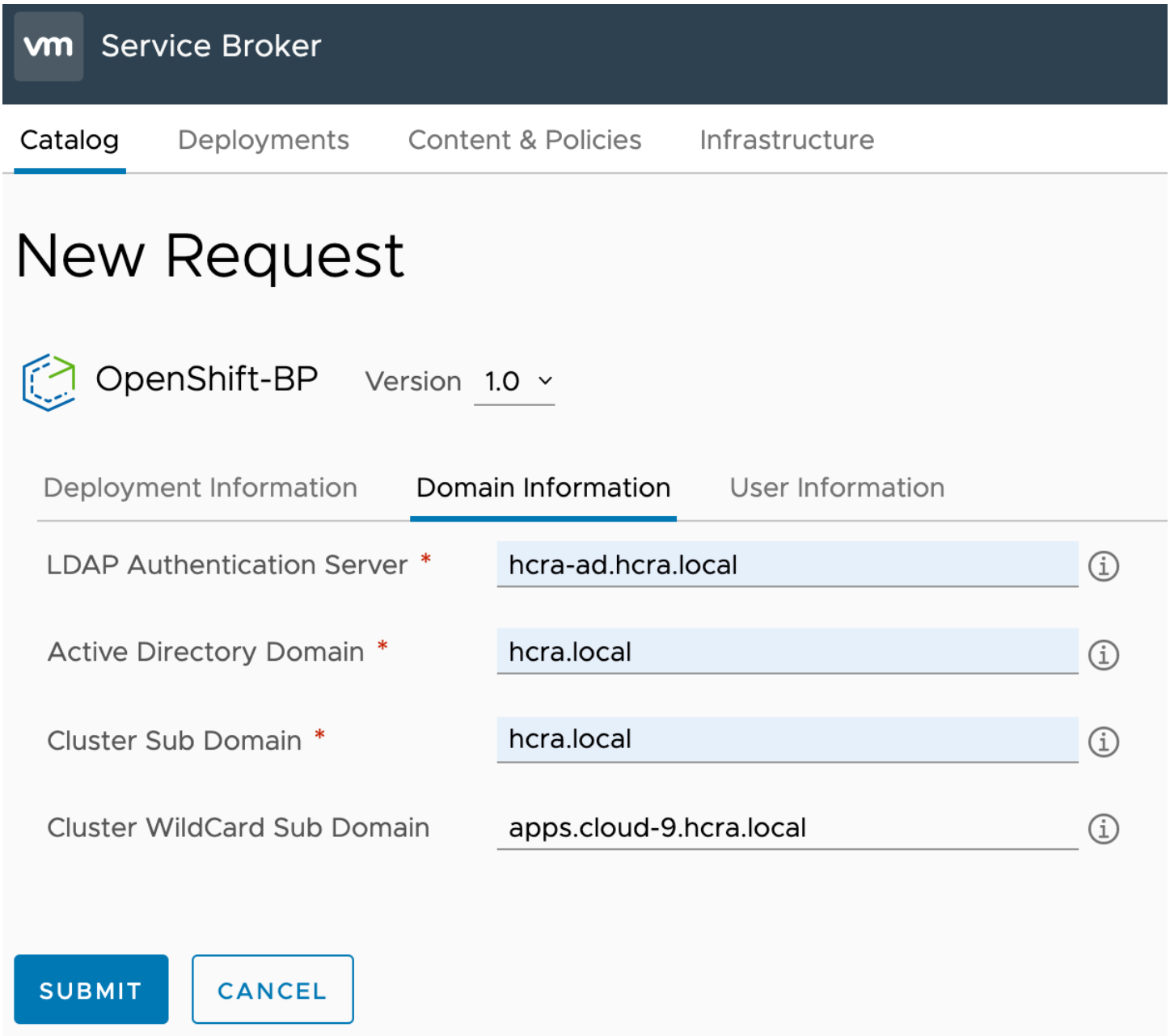

FIGURE 29: Catalog Item Request - 3

Provide the details in the "Domain Information" tab.

Note, in general Cluster Sub Domain is same as Active Directory Domain.

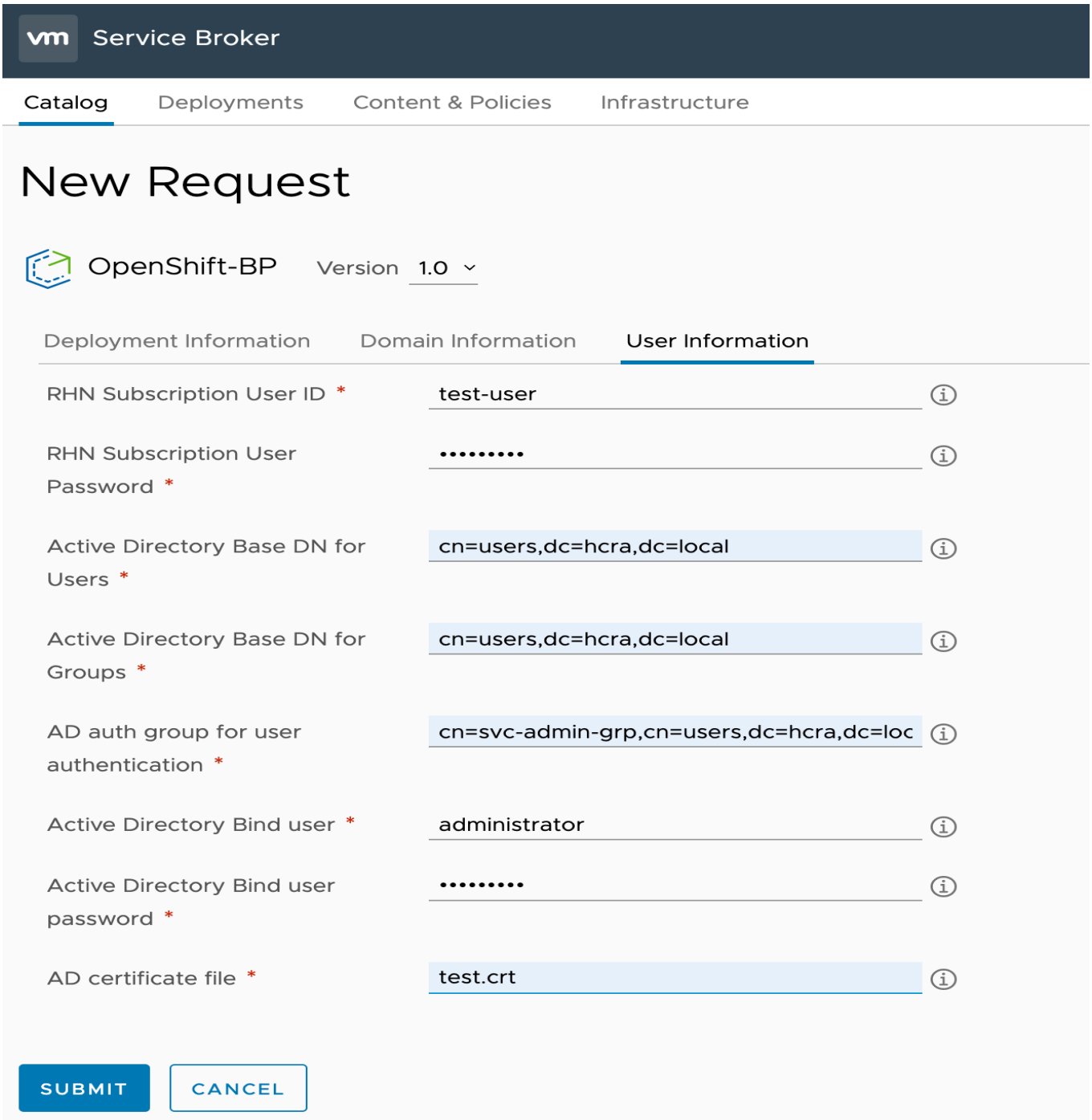

FIGURE 30: Catalog Item Request - 5

Provide the details in the "User Information" tab.

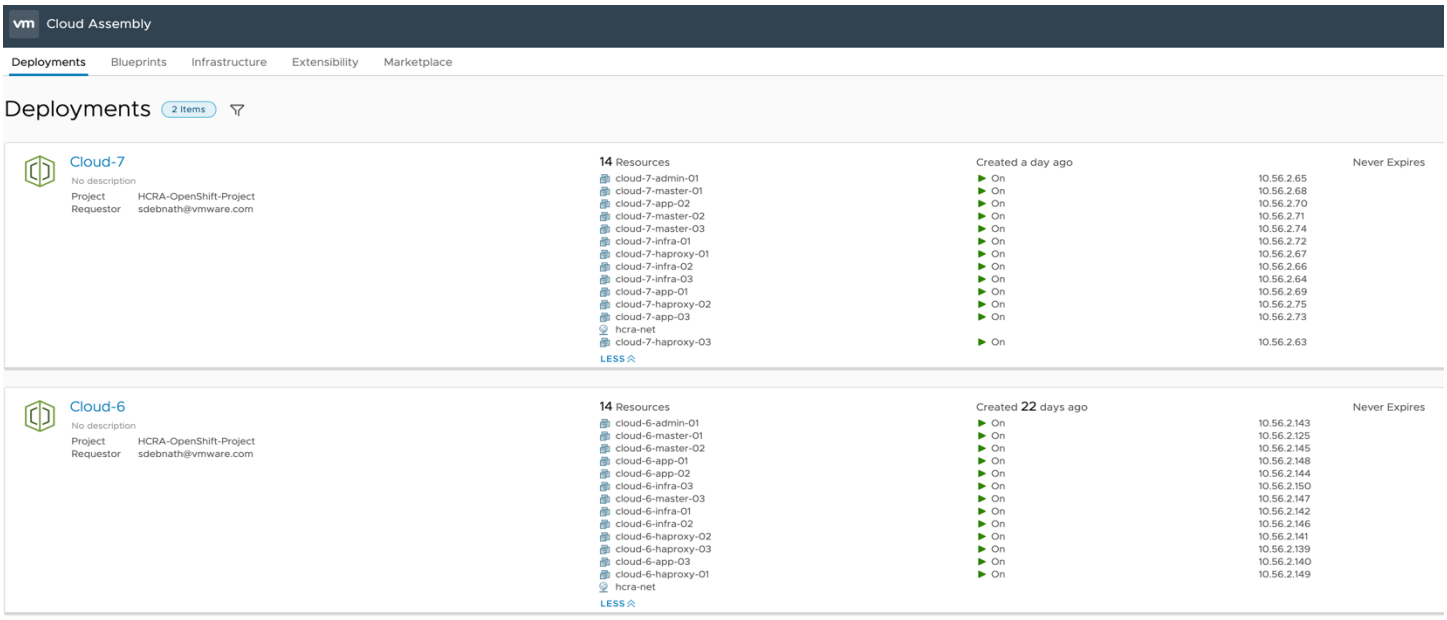

FIGURE 31: Catalog Item Request - 6

Above picture is a sample of a successfully competed deployment in vRAC.

Note: It takes about an hour to complete a successful deployment. For failures, please check the Troubleshooting section.

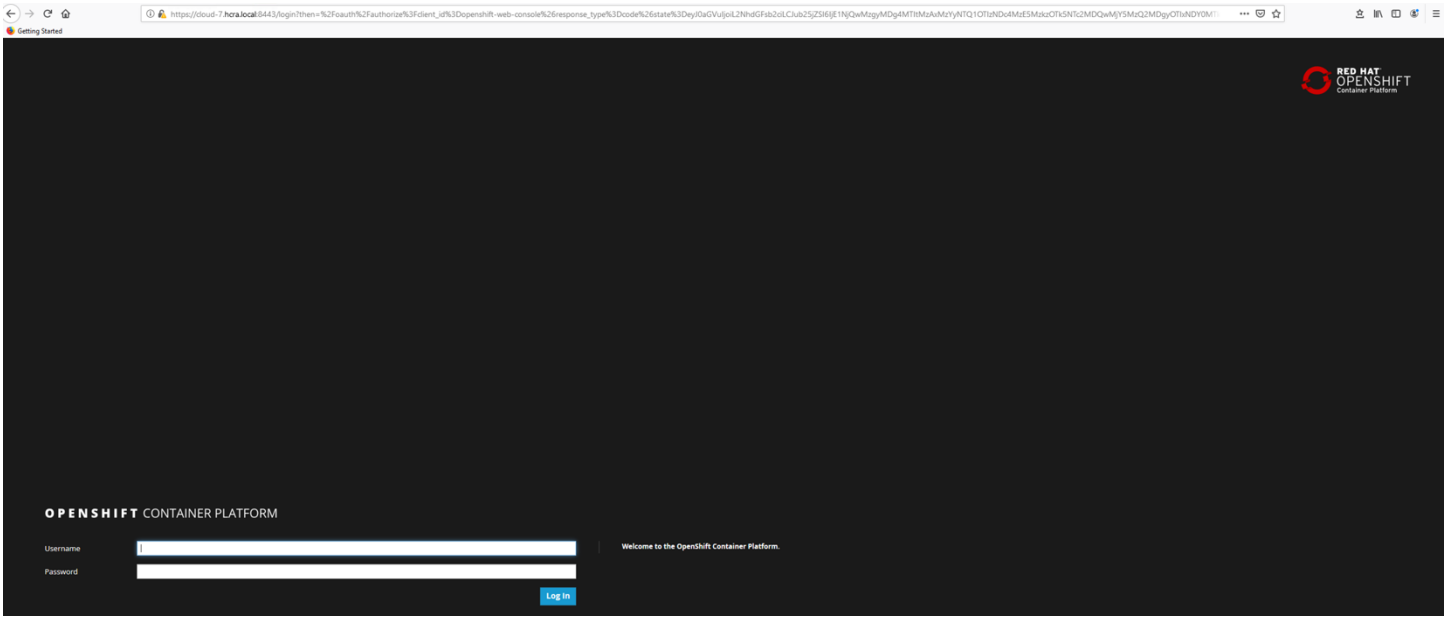

FIGURE 32: Catalog Item Request - 7

Once deployed, OpenShift console can be reached at *https://<cluster-name>.<subdomain>:8443*

For example, *https://cloud-7.hcra.local:8443*

Login user id: Admin

Login user password: 12345

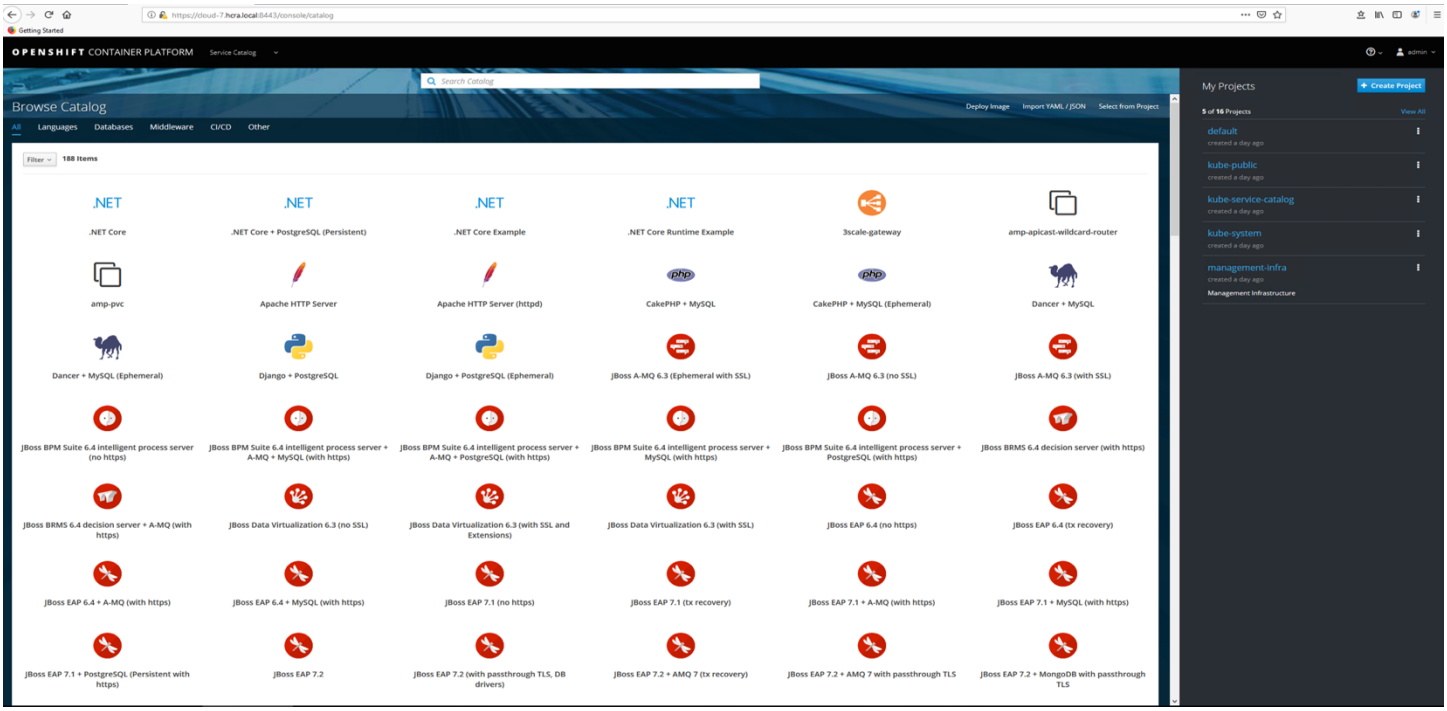

FIGURE 33: Catalog Item Request - 8

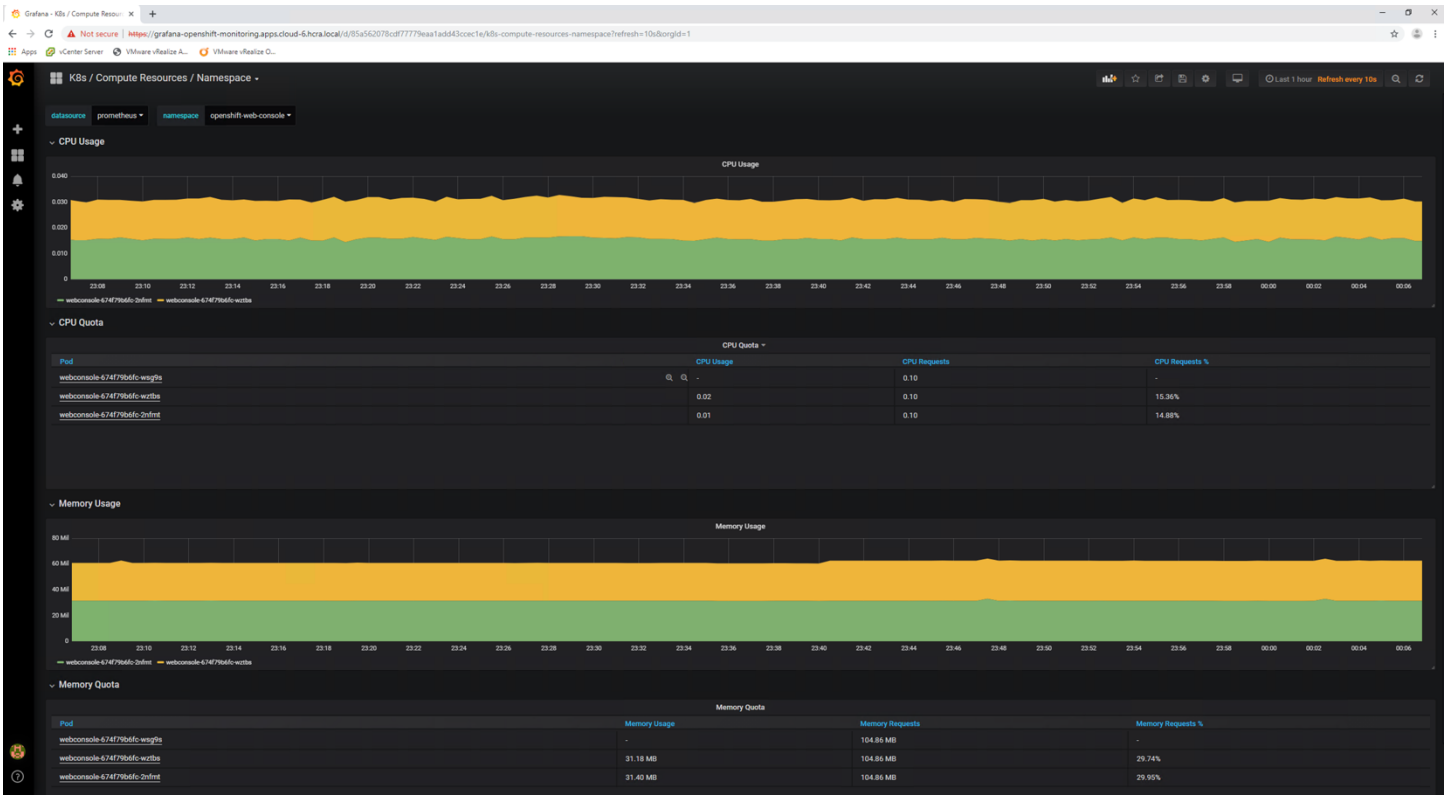

FIGURE 34: Catalog Item Request – 9 – Grafana page. Uses OpenShift login information.

Limitations

Please note, this is Alpha version, hence has some limitations. These are provided below.

There are non-parameterized values on ansible inventory file on <cluster-name>-admin-01:/root/<cluster-name>/ansible-hosts.

These values include:

- OCP admin password (as htpasswd_auth) as "12345"
- AD Authentication line is commented out
- RH Subscription Pool ID was previously a fixed value, now it is passed as a Blue Print parameter. The pool id depends of the entitlement of the Red Hat subscription.
- SDN Networks are 10.87.0.0/15 and 10.144.0.0/16. If your actual VMC network vLAN falls into these addresses, the installation will fail.

Work to do

- Parameterize all the values everywhere.
- Integrate AD Authentication in OpenShift installation
- Give an option for not selecting DNS registration/un-registration and workflow should adopt accordingly
- A better logging options

Troubleshooting

- In case of the OpenShift cluster installation does not succeed after a period of time and after multiple attempts, user might want to login on the admin node to troubleshoot for a possible root cause.

  The OpenShift installation log is on <cluster-name>-admin-01:/root/openshift-ansible.log. Check for the last messages.
- For vRO related issues and failures please check the workflow run logs in vRO. It has detailed information about the workflow run.

Help

Reach out to *octo-openshift@vmware.com* for any related help.

Appendix

A. Detailed attribute list for OpenShift-on-VMware-VRAC-Alpha-1-v1.0 workflow

OPENSHIFT-ON-VMWARE-VRAC-ALPHA-1-V1.0 - ALL ATTRIBUTE PARAMETERS		
NAME	TYPE	DESCRIPTION
adAuthGroup	string	Group on AD to be used for authentication. Must be the within ad_cn_groups.
adBindPassword	SecureString	Password of the bind user to authenticate to the LDAP to retrieve objects
adBindUser	string	User to authenticate to the LDAP to retrieve objects. Must be reside within ad_cn_users parameter
adCertFile	string	To discuss. It can be the actual content of the file or a path where admin node will retrieve (automation will copy to the admin node at /root/)
adCnGroups	string	LDAP common name where the OCP groups will be
adCnUser	string	LDAP common name where the OCP users will be
adDomain	string	The AD domain. It will be used to identify the identity provider
adminNodeName	string	FQDN of the Admin Node
adminNodePreparationScript	ResourceElement	This is the script to prepare Admin Nodev
adminNodePreparationScriptVariables	Array/CompositeType (stringToReplace:string, replacingString:string):scriptVariables	Replacement variables for Admin Node Script
casData	Properties	This variable stores all Data from VRAC
clusterName	string	Name of the OpenShift Cluster
clusterWildcardSubdomain	string	This is the the wild card DNS for the application traffic for the cluster. The default is apps.cluster_name.subdomain
configureDockerStorageScript	ResourceElement	This is the script for all other node types - Master, Infra, App
configureDockerStorageVariables	Array/CompositeType (stringToReplace:string, replacingString:string):scriptVariables	Replacement variables for Docker config Script (this is setup everywhere except Admin and HA-Proxy nodes)
copyAdvAuditResource	ResourceElement	Script to copy adv-audit.yaml file
copyConfigYmlResource	ResourceElement	Script to copy config.yaml file
copyGenerateAnsibleHostsResource	ResourceElement	Script to copy generate-ansible-hosts.yaml file
copyHaproxyCfgWildcardResource	ResourceElement	Script to copy haproxy.cfg.wildcard.j2 file

OPENSHIFT-ON-VMWARE-VRAC-ALPHA-1-V1.0 - ALL ATTRIBUTE PARAMETERS		
NAME	TYPE	DESCRIPTION
copyOcp311AnsibleHostsResource	ResourceElement	Script to copy ocp311-ansible_hosts-hardened.j2 file
counter	number	This is a counter for loop
delay	number	Time to wait (seconds)
delay1	number	Time to wait (seconds)
delay2	number	Time to wait (seconds)
delay3	number	Time to wait (seconds)
dnsHost	string	DNS HostName
dnsZone	string	DNS Zone name where to add the DNS entries
errorCode	string	Errorcode
externalScript	string	External Script to run inside the Powershell host
externalscriptAlias	string	External Script to run inside the Powershell host
genericHostPreparationScript	ResourceElement	This is the script for general VM pre-reqs preparation-Master Script for Pre-req
genericHostPreparationVariables	Array/CompositeType (stringToReplace:string, replacingString:string):scriptVariables	Replacement variables for generic host preparation script
host	PowerShell:PowerShellHost	PowerShell Host
hostAccount	string	DNS Host account with which to run the PowerShell commands. Ideally administrative user account
hostPassword	SecureString	DNS Host Account Password
ldapServer	string	The URL of the ldap server that OCP masters will connect.
openshiftInstallationScript	ResourceElement	Script for OpenShift installation
openShiftInstallVariables	Array/CompositeType (stringToReplace:string, replacingString:string):scriptVariables	Replacement variables for OpenShift installation script
poolID	string	RHN Subscription pool id
regPass	SecureString	OpenShift registry user password
regUser	string	OpenShift registry User
rhnPass	SecureString	RHN user password
rhnUser	string	RHN user id

OPENSHIFT-ON-VMWARE-VRAC-ALPHA-1-V1.0 - ALL ATTRIBUTE PARAMETERS		
NAME	TYPE	DESCRIPTION
scriptVariables	Array/CompositeType (stringToReplace:string, replacingString:string):scriptVariables	Replacement variables for scripts
sshEquivServerResource	ResourceElement	Script for sshEquiv Server preparation
sshScriptVariables	Array/CompositeType (stringToReplace:string, replacingString:string):scriptVariables	Replacement variables for sshEquivServer script
sshSetupScript	ResourceElement	Set to SSH script
subDomain	string	DNS domain name where the OCP cluster will be
vm	VC:VirtualMachine	vCenter Virtual Machine
vmIP	string	IP of the VM
vmName	string	VM Name
vmPassword	SecureString	Password of the VM user
vmState	boolean	Checking the VM state for VMware Tools readiness
vmUsername	string	VM user name with which to run the guest script

TABLE 7: Detailed attribute list for OpenShift-on-VMware-VRAC-Alpha-1-v1.0 workflow

B. Detailed attribute list for Un-Register-VM-from-DNS-RHN-Alpha-1-v1.0 workflow

UN-REGISTER-VM-FROM-DNS-RHN-ALPHA-1-V1.0 - ATTRIBUTE PARAMETERS TO SET		
NAME	TYPE	DESCRIPTION
clusterName	string	Name of the OpenShift Cluster
dnsHost	string	DNS HostName
dnsZone	string	DNS Zone name where to add the DNS entries
externalScript	string	External Script to run inside the Powershell host
errorCode	string	Errocode
externalScriptCnames	string	External Script to run inside the Powershell host
host	PowerShell:PowerShellHost	PowerShell Host
hostAccount	string	DNS Host account with which to run the PowerShell commands. Ideally administrative user account
hostPassword	SecureString	DNS Host Account Password
poolID	string	Red Hat Subscription Pool ID which has OpenShift packages
scriptConfigurationResource	ResourceElement	Unregister-VM-from-DNS-RHEL-Subscription script
subDomain	string	DNS domain name where the OCP cluster will be
vmUsername	string	VM user name with which to run the guest script
vmPassword	SecureString	Password of the VM user
vmName	string	
vmIP	string	IP of the VM
vm	VC:VirtualMachine	vCenter Virtual Machine

TABLE 8: Detailed attribute list for Un-Register-VM-from-DNS-RHN-Alpha-1-v1.0 workflow

Glossary

Term	Meaning
vRAC	vRealize Automation Cloud / Cloud Assembly
LDAP	Lightweight Directory Access Protocol
DNS	Domain Name Services
RHN	Red Hat Network
vRA	vRealize Automation
vRO	vRealize Orchestrator

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com.