Learning VMware vRealize Automation

With the growing interest in Software Defined Data Centers (SDDC), vRealize Automation offers data center users an organized service catalog and governance for administrators. This way, end users gain autonomy while the IT department stays in control, making sure security and compliance requirements are met. Learning what each component does and how they dovetail with each other will bolster your understanding of vRealize Automation.

The book starts off with an introduction to the distributed architecture that has been tested and installed in large scale deployments. Implementing and configuring distributed architecture with custom certificates is unarguably a demanding task, and it will be covered next. After this, we will progress with the installation. A vRealize Automation blueprint can be prepared in multiple ways; we will focus solely on vSphere endpoint blueprint. After this, we will discuss the high availability configuration via NSX loadbalancer for vRealize Orchestrator. Finally, we end with Advanced Service Designer, which provides service architects with the ability to create advanced services and publish them as catalog items.

Who this book is written for

This book is for anyone who wants to start their journey with vRealize. It is your one-stop instruction guide to installing and configuring a distributed architecture using NSX load balancer. Regardless of whether or not you have used vRealize Automation before, following the steps provided in each chapter will get you started with the product.

What you will learn from this book

- Understand the basic building blocks of vRealize Automation before moving on to installation
- Familiarize yourself with the requirements and the steps that need to be performed during the first phase of the distributed installation
- Carry out a functional validation of the first phase of installation before completing the installation
- Build a blueprint for a vSphere endpoint, an essential step for the successful deployment of a service catalog
- Create, configure, and deploy tenants, endpoints, blueprints, and the service catalog
- Get to grips with the failover process for all the components of vRealize Automation
- Learn to configure the NSX loadbalancer for vRealize Orchestrator for high availability
- Leverage Advanced Service Designer to develop XaaS (Anything as a Service) in vRealize Automation
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 7 'vRealize Orchestrator in High Availability via the NSX Load Balancer'
- A synopsis of the book’s content
- More information on Learning VMware vRealize Automation
Sriram Rajendran is a member of the CTO Ambassador program at VMware. He has been a veteran of the IT industry, for more than 12 years, and a focused technologist with expertise in cloud computing, networking, storage, and server virtualization technologies.

Sriram wears multiple hats at VMware. As a solution architect, he provides technical leadership and expertise to design, deploy, and scale the VMware SDDC stack for its Fortune 500 customers. His primary focus for VMware SSDC are automation, operations, and third-party integration.

As a senior escalations manager, he is the go-to person for handling critical executive escalations that have out blown traditional GSS escalation processes. His focus here is not just managing escalations through various internal VMware organizations but also external partner organizations and their processes and extranet multivendor support processes like TSANET.

As a CTO Ambassador, he is responsible for connecting the research and development team with customers, partners, and field as the global VMware evangelist. His focus is on defining and communicating VMware's vision and strategy, and acting as an advisor for VMware's vRealize Automation solutions, product roadmap, and portfolio.

Previously, as a staff escalation engineer, he worked on customer escalations and prioritizing the requests for the team. He was also the lead on recruitment and talent management for the support and escalations team. He also worked closely with various engineering teams within VMware to help provide early feedback on the design and architecture of products based on escalations and his other field interactions.
Prior to joining VMware, he worked at Slash Support and HP in their support organizations in technical leadership roles.

Sriram has devoted much of his professional career to the design, implementation, and maintenance of large physical and virtual networks, storage and servers, and cloud architectures based on VMware, Microsoft, and other leading enterprise technologies.
DynamicOps originated at Credit Suisse. Its software was initially developed at Credit Suisse's Global Research and Development Group in 2005 to help the company address the operational and governance challenges of rolling out virtualization technology. In 2008, after having deployed and used the software to manage thousands of its virtual machines, Credit Suisse decided to form a company based on the technology to form a new company—DynamicOps. Operations Virtualization is a foundational technology for DynamicOps' cloud offerings. Operations Virtualization is an abstraction layer between the multiple management systems that make up a cloud infrastructure and their consumers. It allows IT staff to apply management to the layers below without the layers above needing to know how or why. Later in July 2012, DynamicOps was acquired by VMware and the product was renamed to vCenter Automation Center (vCAC). With version 6.2 of vCAC, the product has been renamed to vRealize Automation (vRA) to align with their new strategies.

If there's one thing people should know about vRA, it's that it enables customers of any knowledge level to consume the cloud resources you give them access to. At the end of the day, customers don't care where a machine gets spun up as long as it's fast and it will do what they want. That means there's an approval in the request process, but then it goes off to one of the many hypervisor or cloud vendors we support. Imagine not having to put your cloud admins to work to build VMs daily, while at the same time they are getting deprovisioned automatically so that you don't have to buy hardware as often—that's the goal: ease of use for the customer, cost savings for the organization.
Today, the main value that vRA adds is the ability to manage and automate multiple cloud management tools (vSphere, RHEL KVM, AWS, and so on) as well as provision to physical hardware (through UCS, iDRAC, and iLO) to build manageable hybrid cloud, private cloud, virtual desktop, and platform as a service environments. That's a pretty large feat in itself, and you can bet that there are plans to add even more value to this product as it further integrates into the VMware suite of products.

What this book covers

Chapter 1, *vRealize Automation and the Deconstruction of Components*, intends to refresh your understanding with a succinct introduction to the vRealize automation architecture, and it depicts the high level details of every component involved.

Chapter 2, *Distributed Installation Using Custom Certificates*, implements and configures distributed architecture with custom certificates, which is a formidable task. While many blogs and official documentation talk about default installation, this chapter has the step-by-step illustrative recipe that will make it easy to follow and help you install and configure vRealize automation quickly and with a much better understanding.

Chapter 3, *Functional Validation – Phase 1 and Installing Secondary Nodes*, continues to install the remaining components in the distributed architecture; it will be worthwhile only if the installed components function out of the box. Once the setup is corroborated to be functional, we will advance and complete the installation.

Chapter 4, *Configuring a Guest OS for vRealize Automation vSphere Blueprints*, explains that the vRA blueprint can be created for different endpoints; this chapter will focus on the blueprint for the vSphere endpoint. Before we configure a blueprint for the vSphere endpoint, the vCenter-based templates need to go through a few configuration procedures. This is important for a successful deployment of the catalog items.

Chapter 5, *Functional Validation – Phase 2 and Zero to VM Provisioning*, spends time checking whether the setup is working as expected. While we deploy a service catalog item from the self-service user portal, we will discover the several stages of catalog deployment.

Chapter 6, *Testing Failover Scenarios for vRealize Automation Components*, explains that the job is not yet done once the installation and functional verification are successful. We'll spend time checking the failover scenarios for various components in this chapter.
Chapter 7, *vRealize Orchestrator in High Availability via the NSX Load Balancer*, focuses on the central topic of discussion in this chapter, which is the high availability configuration via NSX load balancer for vRealize Orchestrator. The Orchestrator cluster provides not only high availability, but also load balancing when configured with NSX or other third-party load balancer. We will delve into this in depth.

Chapter 8, *The Power of Advanced Service Designer (ASD)*, provides the ability for service architects to create advanced services and publish them as catalog items. This provides the ability to create XaaS or *Anything as a Service* using VMware vRealize Orchestrator.
vRealize Orchestrator in High Availability via the NSX Load Balancer

While there are countless public documents that talk about the goodness of vRealize Orchestrator, we wanted to pivot our discussion on high availability configuration via the NSX load balancer for vRealize Orchestrator in this chapter. VMware vRealize Orchestrator (formerly known as VMware vCenter Orchestrator) can be configured to work in two server modes: standalone and cluster. To increase the availability of the VMware vRealize Orchestrator services both in standalone and cluster mode, you can put the Orchestrator behind a load balancer.

Starting with vRO 5.5, clustering has been included as an out-of-box option that enables greater availability for the Orchestrator engine. If an active Orchestrator server becomes unavailable midway through a workflow run, another active or the standby Orchestrator node will take over and complete the workflow without any service interruptions. However, we have two known limitations while configuring the Orchestrator cluster mode:

- Cannot be configured with an embedded database
- Cannot use the embedded directory service

In this chapter, we will talk about the following:

- Types of Orchestrator configuration
- Certificate creation process
vRealize Orchestrator in High Availability via the NSX Load Balancer

- Orchestrator cluster configuration
  - Preparing the database
  - Configuring the first Orchestrator server
  - Configuring the second and third Orchestrator server nodes
  - Configuring the NSX load balancer

Types of Orchestrator configuration
Clustered Orchestrator servers guarantee high availability and load balancing to protect production installations, such as vRealize Automation.

The Orchestrator can be configured in two modes:

- Active-Active
- Active-Standby

Active-Active
Active nodes are the Orchestrator server instances that run workflows and respond to client requests. If an active Orchestrator node stops responding, it is replaced by one of the inactive Orchestrator server instances:

- All nodes in the cluster are active and provide concurrent connections to the sessions
- There is no service interruption because if one node fails, then the other active nodes keep the client session active
- Workflows should be first created in standalone mode and then imported after configuring the Orchestrator in active–active cluster mode
- The maximum number of nodes in active state is five

Active-Standby
In active-standby configuration, at least one node will be in standby state:

- If the standby node does not receive the heartbeat from the active node for the configured timeout, then it becomes active
- If the Orchestrator client logs into the standby node, then it will throw an error and you have to log in to the active node and the workflow will run through that node
• The maximum number of nodes in active state is five
• The maximum number of nodes in standby state is three

Planning and preparing
Here are some of the key tasks before we start the actual implementation:

• **Software**: Download VMware vRealize Orchestrator OVA to a file share within the target datacenter.

• **Hostnames and IP address planning**: Based on your enterprise naming convention, list the hostname and IP address for vRO nodes including the virtual IP in the load balancer.

• **SSL certificate generation**: Signed or self-signed certificates should be created to contain the Orchestrator virtual IP and the Orchestrator node's hostnames in the **SubjectAltNames** section. This allows traffic to be served by the load balancer without throwing SSL errors. We will leverage a certification generation tool for this task; refer to— kb.vmware.com/kb/2107816.

• **Create DNS entries**: FQDNs will be used throughout our installation. Manually create a record (forward lookup) and a **PTR** record (reverse lookup) for Linux-based VMware virtual appliances and load balancer virtual addresses.

• **Load balancer configuration**: Commonly used one-armed load balancer topology will be configured. Create node entries, virtual addresses (VIPs).

• **Microsoft SQL Server**: We will be leveraging the MSSQL database node to create the Orchestrator database. Windows Clustering is desirable in order to meet availability objectives at the database level and is not covered here as it's beyond the scope of this book.

Infrastructure details
In our setup, we will be configuring three vRealize Orchestrator nodes—two active nodes and one standby node:

<table>
<thead>
<tr>
<th>Hostname</th>
<th>IP address</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>vRO.PKCT.LOCAL</td>
<td>10.112.103.127</td>
<td>The virtual IP for the Orchestrator in load balancer</td>
</tr>
<tr>
<td>vRO1.PKCT.LOCAL</td>
<td>10.112.103.128</td>
<td>The first vRO node</td>
</tr>
<tr>
<td>vRO2.PKCT.LOCAL</td>
<td>10.112.103.129</td>
<td>The second vRO node</td>
</tr>
</tbody>
</table>
vRealize Orchestrator in High Availability via the NSX Load Balancer

Bill of materials
Here is the summary of the Orchestrator versions and build numbers used in this chapter:

- vRealize Orchestrator 6.0.3 build 3000579
- Existing vRealize Automation infrastructure including the NSX load balancer service

Generating Orchestrator certificates
Please refer to the Creating certificates section in Chapter 2, Distributed Installation Using Custom Certificates to generate the certificates using Certgen tools (http://kb.vmware.com/kb/2107816 for vRealize Orchestrator).

1. Create a file, servers.txt, and add the list of servers for which certificates need to be created:

```
VIM1:/Certificates/vRO_Certificate # cat servers.txt
VBO
vRO1
vRO2
vRO3
VIM1:/Certificates/vRO_Certificate #
```

2. Follow the instructions in step 2 and 3 as mentioned in Chapter 2, Distributed Installation Using Custom Certificates under the Creating certificates section.

The password used is changeme.

Once the script finishes successfully, it will load the screen with the instructions on how to upload the certificates for all the components. However, at this time, we are interested only in vRO—the following screenshot is pertinent to the Orchestrator:

JKS for vRealize Orchestrator (VCO)
-----------------------------
File: /root/Certificates/vRO_Certificate/jssecacerts
Installation Method:
Copy jssecacerts to /etc/vco/app-server/security/ on each VCO appliance and run the following commands:
chown vco:vco /etc/vco/app-server/security/jssecacerts
cwac-config vco-configure (embedded vCAC version only)
service vco-configurator restart
service vco-server restart
Copy the jssecacerts file to the /tmp directory of all the Orchestrator appliances.

This concludes the steps involved in generating the certificate for the Orchestrator. Let's move forward to install and configure the vRO appliance.

**Configuring the vRealize Orchestrator cluster**

The Orchestrator cluster provides not only high availability, but also load balancing when configured with NSX or another third-party load balancer. Orchestrator clustering is a zero-touch configuration, which means that the Orchestrator cluster is managing itself. The maximum number of active nodes that you define in the configuration dictates how many nodes are switched from standby to running. For example:

- You define the number of active nodes as two; however, you configure three Orchestrator installations in this cluster and power them all on. This would result in two nodes being active and one being in standby mode.
- If you now proceed to power off one of the active nodes, the standby node will become active. You could test this by setting the number of active nodes to 1 in the setup we are about to build.

There are certain drawbacks you should be aware of. It is not recommended to use the Orchestrator client to connect to the nodes running in a cluster. This is done on purpose to make sure that changes to workflows don't occur.

**Configuring the setup**

We will be configuring two vRealize Orchestrator appliances with both the nodes in active configuration. Since the configuration includes cluster mode, using an external database and authentication (SSO) server is mandatory. The VMware Identity Manager appliance will be used as our SSO server and MSSQL server will be used for the database.

Configuring the Orchestrator to work with an external authentication enables AD users to log in to the Orchestrator client. As SSO is now a highly integrated part of vSphere, using the Orchestrator with AD (or LDAP) isn't really a good solution any longer. SSO can proxy multiple AD and/or LDAP domains and lets you integrate the Orchestrator directly into vCenter as well as other corner pieces of VMware software offerings, making SSO integration a better choice for the future.
Before we explore further, let's add our PKCT.LOCAL domain with the default vRA Tenant (vsphere.local):

1. Log in as administrator@vsphere.local to the vRA default tenant page—https://CAFE.PKCT.LOCAL/vcac and click on the vsphere.local tenant:

   ![Image of vRA default tenant page]

   - Navigate to the Identity Stores tab, click Add Identity Stores, and fill the field as applicable to your setup; click Test Connection, click Add, and click Update. Please use this screenshot as reference:

   ![Image of Add Identity Stores]

2. Navigate to the Identity Stores tab, click Add Identity Stores, and fill the field as applicable to your setup; click Test Connection, click Add, and click Update. Please use this screenshot as reference:

   ![Image of Edit Identity Store: PKCT Domain]

This concludes the step of adding an AD domain (PKCT.LOCAL) to the default tenant.
Prerequisites
Before we set out to begin our implementation, gathering and completing the prerequisites are an important task. Let's review each item:

- Allocate the hostname, IP address, and create DNS records (A and PTR) for the vRO virtual IP in the load balancer and two vRO nodes (vRO1, vRO2)

Creating NSX load balancer configurations for CAFÉ
1. Freshly deploy three VMware vRealize Orchestrator appliances without any configurations. (I'm not going to walk through the OVA deployment in this book.)
2. Power on and perform the following checks in every Orchestrator appliance.
3. Connect to the VAMI page of the virtual appliance—https://<IP_address-or_FQDN-of-vRO(1,2)-appliance>:5480.
4. Navigate to Network | Address:
   - Check whether the hostname and DNS entries are accurate.
   - IP settings should be set to static.
5. Navigate to Admin | Admin and check whether the SSH is enabled and Status shows Running.
6. Navigate to Admin | Time Settings and ensure that the NTP settings are pointing to your infrastructure NTP server.
7. SSH into the virtual appliance as root user and check whether you are able to ping the virtual IP of vRO in the load balancer and the infrastructure management virtual machines (AD, DNS) using FQDN.
8. If all the preceding steps are true for both the vRO appliances, move on to the next step.

How to do it
While we have multiple sections, here is the flow of steps:

1. Preparing the database
2. Configuring the first Orchestrator server
3. Configuring the second and third Orchestrator server nodes
4. Configuring the NSX load balancer
Preparing the database

Create a database, `vCOdb`, in the SQL Server and execute the following SQL query to set the following two settings on the Orchestrator database:

In this example, the Orchestrator database is called `vCOdb`:

```
ALTER DATABASE vCOdb SET READ_COMMITTED_SNAPSHOT ON
ALTER DATABASE vCOdb SET ALLOW_SNAPSHOT_ISOLATION ON
```

Configuring the first Orchestrator server

In this section, we will configure the first Orchestrator appliance and prepare it for cluster configuration.

**The Orchestrator configuration page**

Log in to the Orchestrator configuration page (https://vRO1.PKCT.LOCAL:8283) as the `vmware` user:

1. Key in the password that was provided while deploying the Orchestrator appliance.
2. Locate **Network**, set **IP address** from the drop-down menu, and click **Apply changes**.

Configuring the database

In this section, we will be configuring the Orchestrator appliance with an external database:

1. Navigate to **Database** and fill in the details as applicable to your setup (the following screenshot can be used as reference) and click **Apply changes**.
2. Once you get the message, **Database configuration saved successfully.** Create the database tables by clicking the following link, click the **Create the database tables** link:

<table>
<thead>
<tr>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install</td>
</tr>
<tr>
<td>Database Type</td>
</tr>
<tr>
<td>Select the database type:</td>
</tr>
<tr>
<td>Connection Parameters</td>
</tr>
<tr>
<td>For more details about installing this database, see</td>
</tr>
<tr>
<td>User name:</td>
</tr>
<tr>
<td>Password (if any):</td>
</tr>
<tr>
<td>Use SSL:</td>
</tr>
<tr>
<td>Database server IP address or DNS name:</td>
</tr>
<tr>
<td>Port:</td>
</tr>
<tr>
<td>Database name:</td>
</tr>
<tr>
<td>Instance name (if any):</td>
</tr>
<tr>
<td>Domain:</td>
</tr>
<tr>
<td>Use Windows authentication mode (NTLMv2):</td>
</tr>
</tbody>
</table>

**Configuring certificates**

The certificates that were created via the Certgen tool in the beginning of this chapter will be used in this section. This is an important step, and I would recommend that you read the steps carefully before you start the process. As a disclaimer, I want to highlight that a simple mistake can push your appliance to a flawed state:

1. Using PuTTY, make an SSH connection to the Orchestrator appliance and authenticate using the root credentials.
2. Back up the exiting Java key store file using the following command:
   ```
mv /etc/vco/app-server/security/jssecacerts /etc/vco/app-server/security/jssecacerts-old
   ```
3. Move the `jssecacerts` file from the `/tmp` directory into the `/etc/vco/app-server/security` location using the following command:

   ```shell
   mv /tmp/jssecacerts /etc/vco/app-server/security
   ```

4. Execute the following commands to ensure the permission on this file is appropriate; otherwise, the appliance might not be able to start correctly:

   ```shell
   chmod 600 /etc/vco/app-server/security/jssecacerts
   chown vco:vco /etc/vco/app-server/security/jssecacerts
   ```

5. Reboot the vRealize Orchestrator appliance by running the following command:

   ```shell
   reboot
   ```

6. Once the Orchestrator appliance has restarted, confirm that the newly signed certificates are correctly installed. Go to the vRealize Orchestrator configuration page using its FQDN: `https://vRO1.PKCT.LOCAL:8283`.

7. Right-click the padlock on the top-left corner of the URL and click More Information:

   - Review the certificate and verify that it is signed by the CA authority used by your organization:
8. If you want to confirm the same from the Java keystore update in the appliance, execute the following command:

```
keytool -list -keystore /etc/vco/app-server/security/jssecacerts -v
```

If all goes well, the details in the last two screenshots should match.
Orchestrator server configuration continues

1. On the VMware vRealize Orchestrator configuration login page, log in with the vreal username and the password you specified when deploying the appliance.

2. Click the Network menu on the left-hand side and navigate to SSL Trust Manager, and import and accept the certificate of the SSO server before connecting to an external directory service (URL—https://vIDM.PKCT.LOCAL:7444; if you are using PSC, then append the 443 port instead of 7444).

   For each system you want the vRealize Orchestrator appliance to interact with, enter the target system and click Import in the URL section to import the certificate. Ensure to configure a vCenter Server if you want to export the license into Orchestrator.

3. Click the Authentication menu on the left-hand side and select SSO Authentication as Authentication mode. Type the SSO server host without the port number followed by its credentials and click Register Orchestrator.

4. Under SSO Configuration, set SSO domain and vRO Admin as applicable to your setup (the following screenshot can be used as reference) and click Accept Orchestrator Configuration.

   We were able to list the pkct.local domain in the SSO domain field since we added the domain to the default tenant in the earlier step.
It is recommended to create a user for vRO purposes in the PKCT.LOCAL directory services. I have created a user, vroadmin, in the PKCT.LOCAL directory services.

5. Navigate to the Test Login tab to validate the login. If for some reason you receive the error **The User is not allowed to log in**, and if you have validated all the steps, reboot the Orchestrator appliance and perform the same test—this happened in my setup and the reboot resolved it:
6. Navigate to the Licenses menu on the left-hand side and select Use vCenter Server license. Fill in the FQDN or IP address of the vCenter server in the Host textbox followed by the server credentials. Click Apply Changes for the configuration to take effect:

7. Click the Startup Options menu on the left-hand side and click Start service to restart the service:
Installing the Orchestrator client

The Orchestrator client interface is designed for the developers who have administrative rights and want to develop workflows, actions, and other custom elements. If you want to use the Orchestrator client to connect to the Orchestrator server not through the Java Web Start but to have the client installed on your local machine, you must download and install the Orchestrator client. In this section, we will take a look at how to install the Orchestrator client:

2. Either click Start Orchestrator Client (this requires Java to work) or Download Orchestrator Client Installable (full installer depending on the client OS version):
3. Once the Orchestrator client is installed, launch the client and use the vroadmin@pkct.local user to log in. Since you are logging in for the first time using the client, you should be presented with the **Certificate Warnings** screen. View, verify, and install any certificates that are shown to continue further:

![Login Screen](image)

4. Once you are able to successfully log in to the Orchestrator server using the client, go back to the Orchestrator configuration page. (https://vRO1.pckt.local:8283).

**Installing plugins**

Since vRealize Automation (vRA) is deemed to become the central cornerstone in the VMware automation effort, vRealize Orchestrator (vRO) is used by vRA to interact with and automate VMware and non-VMware products and infrastructure elements. This requires us to install the vRA plugin (oinplugin-vcac-6.2.0-2287231.vmoapp):

1. Click the **Plug-ins** menu and scroll to the bottom of the list.

2. Click the magnifying glass icon to browse to the vRA plugin file and click **Upload and install**:

   - Click **I accept the terms and of the license agreement**
   - Click **Apply changes**

3. Reboot the vRealize Orchestrator appliance and confirm that the installation is **OK** for the new plugins.
Configuring the cluster

To increase the availability of the Orchestrator services, you can configure a cluster of Orchestrator server instances. An Orchestrator cluster consists of at least two Orchestrator server instances that share one database.

To work properly in the cluster, all the Orchestrator server instances must be configured identically with each other and must have the same plugins installed. After you set up the Orchestrator cluster, do not change the configurations of its nodes:

1. Click the Server availability menu and click Cluster mode. Set Number of active nodes to 2 (as per your design) and click Apply change.
2. Click the Startup options menu on the left-hand side and click Restart service (under vRO Server) to restart the service.
3. Click the Server Availability menu. After a few minutes (this can range from 1 to 5 min), you should see the first node appear under Started cluster nodes in the RUNNING status:

<table>
<thead>
<tr>
<th>Server Availability Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server mode</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Cluster mode settings</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Started cluster nodes</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Export configuration

The Orchestrator configuration interface provides a mechanism to export the Orchestrator configuration settings to a local file. This mechanism allows you to take a snapshot of your system configuration at any moment and import this configuration into a new Orchestrator instance:

1. Click the General menu and click the Export Configuration tab and click Export.

2. A configuration file will be exported to the following folder: /var/lib/vco/, in the Orchestrator appliance with the file name, vmo_config_xxxxxxxxxxx.vmconfig:

3. Copy (using the WinSCP tool) the exported file from this Orchestrator node to a Windows machine where the Orchestrator client is installed.

This successfully concludes cluster node configuration in the first Orchestrator node. Proceed and configure the second node of the cluster.

Configuring the second Orchestrator server

Things are going to be easy from now on since you already have hands-on experience of configuring the first Orchestrator server. The following steps help you to configure the second node of the vRealize Orchestrator cluster:

1. Perform the steps detailed in the Configuring certificates section.

2. On the VMware vRealize Orchestrator Configuration login page, log in with the vmware username and the password that you specified when you deployed the appliance.

3. Click the General menu and then click the Import Configuration tab.
4. Click the magnifying glass icon, browse to the local file system location where the exported file was saved, and click **Import**.

5. Click the **Network** menu on the left-hand side and click the **Network** tab:
   - From the **IP address** drop-down menu, select the IP address of the second appliance since the DNS name will be set to the FQDN of the first node and click **Apply changes**.

6. Continue to follow the steps detailed in the *Installing the plugins* section:
   - It is important to ensure that the same version and plugins are installed on both nodes.

7. Click the **Server Availability** menu. After a few minutes, you should see the second node appear under the **Started cluster nodes RUNNING** status:

<table>
<thead>
<tr>
<th>Server Availability Options</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server mode</strong></td>
</tr>
<tr>
<td>○ Standalone mode</td>
</tr>
<tr>
<td>○ Cluster mode</td>
</tr>
<tr>
<td><strong>Cluster mode settings</strong></td>
</tr>
<tr>
<td>Number of active nodes: 2</td>
</tr>
<tr>
<td>Heartbeat interval in milliseconds: 5000</td>
</tr>
<tr>
<td>Number of failover heartbeats: 12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Started cluster nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server host address</td>
</tr>
<tr>
<td>10.112.103.129</td>
</tr>
<tr>
<td>10.112.103.128</td>
</tr>
</tbody>
</table>

**Configuring the NSX load balancer**

This section only describes the load balancing aspect of the NSX product configuration, assuming that NSX has already been configured and validated to work properly on the target environment or networks:

1. Log in to vCenter Server where NSX has been configured.

2. Navigate to **Home** | **Networking & Security** | **NSX Edges** and select the **Edge appliance** deployed for the use of a distributed vRealize Automation installation.
3. Navigate to Manage | Settings and select the Interfaces menu on the left-hand side.
4. Select the first vNIC and click the Edit button. This will be your load balancer virtual appliance.
5. Click the Add button to assign a static IP address (virtual IP for Orchestrator) to the virtual interface.

Orchestrator NSX load balancer configurations
In this section, we will be configuring an application profile, service monitoring, pool and virtual server settings in the NSX load balancer server for the Orchestrator cluster to work behind a load balancer.

Configuring an application profile
1. Log in to vCenter Server where NSX has been set up.
3. On the Load Balancer tab select the Application Profiles menu.
4. Click the Add button to create a new profile and complete the form according to the following table:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Enable SSL passthrough</th>
<th>Persistence</th>
</tr>
</thead>
<tbody>
<tr>
<td>vRO</td>
<td>HTTPS</td>
<td>Checked</td>
<td>None</td>
</tr>
</tbody>
</table>

Configuring service monitors
1. Log in to vCenter Server where NSX has been set up.
3. In the Load Balancer tab, select the Service Monitoring menu on the left-hand side.
4. Click the **Add** button to create a new monitor and complete the form according to the following table:

<table>
<thead>
<tr>
<th>Name</th>
<th>Interval</th>
<th>Timeout</th>
<th>Retries</th>
<th>Type</th>
<th>Method</th>
<th>URL</th>
<th>Receive</th>
</tr>
</thead>
<tbody>
<tr>
<td>vRO_SM_8281</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>HTTPS</td>
<td>GET</td>
<td>/vco/api/docs/index.html</td>
<td>200</td>
</tr>
</tbody>
</table>

**Configuring pools**

1. Log in to vCenter Server where NSX has been set up.
2. Navigate to **Home | Networking & Security | NSX Edges** and select your previously created NSX edge.
3. In the **Load Balancer** tab, select **Pools**.
4. Click on the **Add** button to create a new pool and complete the form according to the following table:

<table>
<thead>
<tr>
<th>Pool name</th>
<th>Algorithm</th>
<th>Monitors</th>
<th>Member name</th>
<th>Example IP address</th>
<th>Port</th>
<th>Monitor port</th>
</tr>
</thead>
<tbody>
<tr>
<td>vRO_POOL</td>
<td>Round Robin</td>
<td>vRO_SM_8281</td>
<td>vRO1</td>
<td>10.112.103.128</td>
<td>8281</td>
<td>8281</td>
</tr>
<tr>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>vRO2</td>
<td>10.112.103.129</td>
<td>8281</td>
<td>8281</td>
</tr>
</tbody>
</table>

Before configuring the **Virtual Servers**, ensure both the members are enabled and are in the **UP** status under **Show Pool Statistics**.
Configuring virtual servers

1. Log in to vCenter Server where NSX has been set up.
3. In the Load Balancer tab, select Virtual Servers.
4. Click on the Add button to create a new virtual server and complete the form according to the following table:

<table>
<thead>
<tr>
<th>Application profile</th>
<th>Name</th>
<th>IP address</th>
<th>Protocol</th>
<th>Port</th>
<th>Default pool</th>
</tr>
</thead>
<tbody>
<tr>
<td>vRO</td>
<td>vRO_LB_8281</td>
<td>10.112.103.126</td>
<td>HTTPS</td>
<td>8281</td>
<td>vRO_POOL</td>
</tr>
</tbody>
</table>

We need to perform one last step before we conclude that the vRO load balancer configuration is complete. Open IE browser and connect to the vRO service monitoring URL—https://vRO.PKCT.LOCAL:8281/vco/api/docs/index.html and check whether you see the Orchestrator API page.

The FQDN, vRO.PKCT.LOCAL, is the Orchestrator virtual IP in the load balancer.

**vRealize Orchestrator API**

**REST Resources**

This API supports a Representational State Transfer (REST) model for accessing a set of resources through a fixed set of operations. The following resources are accessible through the RESTful model:

- Actions Service
- Catalog Service
- Category Service
- Configuration Service
- Content Service
- Inventory Service
- Packages Service
- Plugin Service
- Resource Service
- Server Configuration Service
- Service Descriptor Service
- Tagging Service
- Task Service
- User Interaction Service
This concludes the NSX load balancer configuration.

At this stage, connect to Orchestrator using the Orchestrator client with the load balancer virtual IP:

![vRealize Orchestrator Login](image)

In case you don’t see all the folders under Library once you log in, log out and browse to the first Orchestrator node—`https://vRO1.PKCT.LOCAL:8283`

1. Navigate to **Startup Options** and click the **Stop** service.
2. Navigate to **Troubleshooting** and click the **Reset current version** link.
3. Once the preceding step is successful, navigate to **Startup Options** and click **Start service**.

Repeat the preceding three steps in the second Orchestrator node. This should help solve the issue.
vRealize Orchestrator high availability mechanism

VMware vRealize Orchestrator nodes update their heartbeat in the database every 5 seconds. The default number of missed heartbeats that indicates a problem on a node is 3 heartbeats.

If a vRealize Orchestrator node has a problem and stops the heartbeat, vRealize Orchestrator is aware that its heartbeat has stopped. When the heartbeat entry in the database is not updated, other vRealize Orchestrator nodes in the cluster will know that the heartbeat of the node with a problem has stopped.

When the timeout is reached, the following happens:

- The vRealize Orchestrator node with the problem disappears from the Started Cluster Nodes section under the Server Availability option in the Orchestrator configuration page
- The other vRealize Orchestrator nodes in the cluster determine that it is nonresponsive and take over the workflows
- Once the server is ready to be brought online after fixing the issue, it's recommended to restart the impacted Orchestrator appliance or at least, restart the vco-server and vco-configurator services

Workflow continues from the last point of execution when the vRealize Orchestrator node is started. This mechanism provides the vRealize Orchestrator workflow HA.

If the default heartbeat timeout is considered too sensitive, you can increase the heartbeat timeout:

1. Stop all the vRealize Orchestrator server nodes.
2. In the cluster configuration tab of the Orchestrator configuration interface, change the value of the number of failover heartbeats from 3 to a higher number for every vRealize Orchestrator server node. The values should be the same for all the nodes in the cluster.
3. Start all the vRealize Orchestrator server nodes one by one.

While writing the workflows, ensure that in the event of a failover, the building block of that workflow (for example, scriptable) task should rerun when the workflow resumes on the failover vRealize Orchestrator node.
Configuring vRO at CAFÉ or IaaS makes a difference!

Before we conclude this chapter, I want to talk about how vRO can be leveraged depending on the use case:

- **Use case 1:** For instance, if you want to automate any post provisioning operations, then you may want to add vRO at the CAFÉ layer. This enables to create Anything as a Service (XaaS) by exposing the service blueprint to the customers utilizing the power of vRealize Orchestrator (vRO) called Advanced Service Designer. You will see more about ASD in the next chapter.

- **Use case 2:** The Orchestrator can be configured as an endpoint at the IaaS level if you would like to provision or decommission a machine, especially for mission-critical systems that typically require interactions with a number of different management systems including DNS servers, load balancers, CMDBs, IP address management, and other systems.


**Summary**

The primary purpose of this chapter was to explain the detailed steps involved in creating a vRealize Orchestrator cluster behind an NSX load balancer. Infrastructure as a Service (IaaS) represents the deployment and life cycle management of server workloads—whether they are a traditional vSphere virtual machine in an organization’s internal private cloud, a cloud workload in VMware vCloud Air, Microsoft Azure, Amazon Workspace Services, or another provider, or even physical servers. Anything as a Service (XaaS) represents virtually anything else that isn’t IaaS. vRO enables XaaS inside vRA. The relationship between the two works like this—vRA is the request and approval portal, vRO is the orchestration engine.

In the next chapter, you will read about the power of Advanced Service Designer and how it can be leveraged to create XaaS.
Where to buy this book

You can buy Learning VMware vRealize Automation from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.