VMware vCloud® Architecture Toolkit™
for Service Providers

Automated VMware vRealize® Automation™ Deployments for VMware Cloud Providers™

Version 2.9

January 2018

Harold Simon

and

Daniel Borenstein

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

Contents

List of Tables

List of Figures

Introduction and Solution Overview

Service providers are interested in implementing a streamlined method to deliver the self-service portal capabilities of VMware vRealize® Automation™ to their end customers on a per-tenant basis. The purpose of this document is to discuss a process to achieve this end goal using features of vRealize Automation (7.1) as well as other supported tools for automation. This document also details the common deployment models of vRealize Automation with service providers and provides recommendations for successful deployments.

1.1 Service Provider Deployment Models

vRealize Automation is typically implemented in Private Cloud − Enterprise environments. But service providers still have an interest in providing services based on vRealize Automation for customers on a per-tenant basis (hosted or virtual private cloud), and also in the management of the internal infrastructure.

Some of the common deployment models that service providers use for vRealize Automation are:

- **Internal Operations** – In this model, a single tenant instance of vRealize Automation is deployed by the service provider for internal operations users. All management and endpoint consumption in this model are intended for service provider related functions such as tenant onboarding, DevOps, management platform deployment automation, and other important service provider functions.
- **Dedicated Customer Private Cloud** – This entails a single tenant deployment of vRealize Automation with the optional use of multiple business groups. In this model, the customer is given access to manage most aspects of vRealize Automation, including fabric management, blueprint design, and catalog management, without the complication of multi-tenancy.
- **Fully Managed Service Offering** – In this model, the service offering leverages multiple business groups and is managed fully by the VMware Cloud Provider™ on behalf of the customer.

At a platform level, each of these models enables the consumption of single and multiple data centers provided by the service provider, while the dedicated private cloud and the managed service offering provide customers the capability to consume on-premises compute resources.

The solution discussed in this document focuses on the method by which VMware Cloud Providers can automate the deployment of vRealize Automation to increase the time to value of hosted or virtual private cloud to their customers. This method of deployment would be beneficial in the dedicated customer private cloud and the fully managed service offering. Though each environment is unique, this document outlines methods and considerations, used for this use case, that can provide guidance for any service providers interested in delivering a similar solution for their customers.

1.2 Business Drivers

For the customers of VMware Cloud Providers, the key business value of deploying vRealize Automation is an expedited time to value while also being able to offload the maintenance and management overhead of the private cloud infrastructure to a trusted VMware Cloud Provider of their choice.

With respect to VMware Cloud Providers, a few key business drivers for adopting this use case are as follows:

- Provide value-add to the traditional hosting platforms based on VMware vSphere®
- Dramatically decrease the deployment time of vRealize Automation for per-customer consumption
- Enable the provider to move up the stack to offer applications to their end-customer's conceptual design/architecture
- Automated deployment of vRealize Automation in an "as a Service" model within a VMware vCloud Director® OrgVDC (reuse automation on other platforms where possible)

- Connect the vRealize Automation platform to a vCloud Director OrgVDC for IaaS provisioning
- Reduce staff effort required to deploy the solution while also minimizing human error

Solution Design

2.1.1 vRealize Automation

vRealize Automation is a core component for the provisioning of workloads in private hosted cloud deployments. Leveraging the vRealize Automation self-service portal, tenants can deploy and maintain the lifecycle of workloads with their own dedicated endpoints hosted by the cloud service provider.

vRealize Automation consists of several sub-components and services that can run one system or be distributed for a large scale of users and workloads. The details of some of these major components are discussed here.

2.1.1.1 vRealize Automation Appliance

The vRealize Automation appliance is the front-end portal for self-service provisioning and management of vRealize Automation workloads. This appliance also hosts:

- Single sign-on capabilities that are used for authenticating user access to the vRealize Automation portal.
- Postgres Database for vRealize Automation.
- VMware vRealize Orchestrator™ for workflow-driven automation of workloads and extensibility to third-party services and XaaS capabilities. See section 2.1.1.6 for more information on Orchestrator.

2.1.1.2 vRealize Automation IaaS Web Service

The vRealize Automation IaaS Web Servers consist of Microsoft Windows servers that host model manager services for vRealize Automation. These servers are responsible for providing access to the MSSQL database associated with vRealize Automation for the rest of the vRealize Automation services.

2.1.1.3 vRealize Automation IaaS Manager Service

The IaaS Manager Servers host the vRealize Automation Manager Services, which provides the overall coordination of events within vRealize Automation. The IaaS Manager Servers can be deployed in a hot standby configuration for redundancy.

2.1.1.4 vRealize Automation DEM Workers and Agents

The distributed execution manager (DEM) role is installed on one or more Windows servers. The DEM Workers perform the execution of automation workflows against cloud and virtual endpoints that are managed by the vRealize Automation instance.

2.1.1.5 vRealize Automation vRealize Automation Agents

The vRealize Automation agent is a proxy service that is used to run workflows executed by the DEM Workers against virtual endpoints such as vSphere. These agents can be installed on the IaaS Server, or they can be installed on a separate server.

The vRealize Automation agents are deployed as close to the endpoint as possible. Two or more agents can be deployed per virtual endpoint for redundancy.

2.1.1.6 vRealize Orchestrator

vRealize Automation has many out-of-the-box features that allow for easy customization of workload deployment. However, there are some automation tasks that require greater extensibility of the platform. vRealize Orchestrator provides the means to extend the customization of workload deployments and also of Day-2 operations.

In addition to being able to orchestrate tasks against VMware's suite of products, vRealize Orchestrator can also leverage plug-ins for third-party tools that help decrease the time required to add integration with external solutions, such as load balancers, ticketing systems, and CMDBs.

2.1.2 Deployment Sizing

For the purpose of the use case, this document discusses the small size deployment model of vRealize Automation as outlined in the vRealize Automation Reference Architecture documentation. This deployment size can support up to 10,000 managed VMs, 500 catalog items, and 10 concurrent provisioning operations, and can accommodate the usage patterns of many prospective consumers.

Table 1. vRealize Automation Sizing

Deployment Size	Managed Machines	Catalog Items	Concurrent Provisions
Small	10,000	500	10
Medium	30,000	1,000	50
Large	50,000	2,500	100

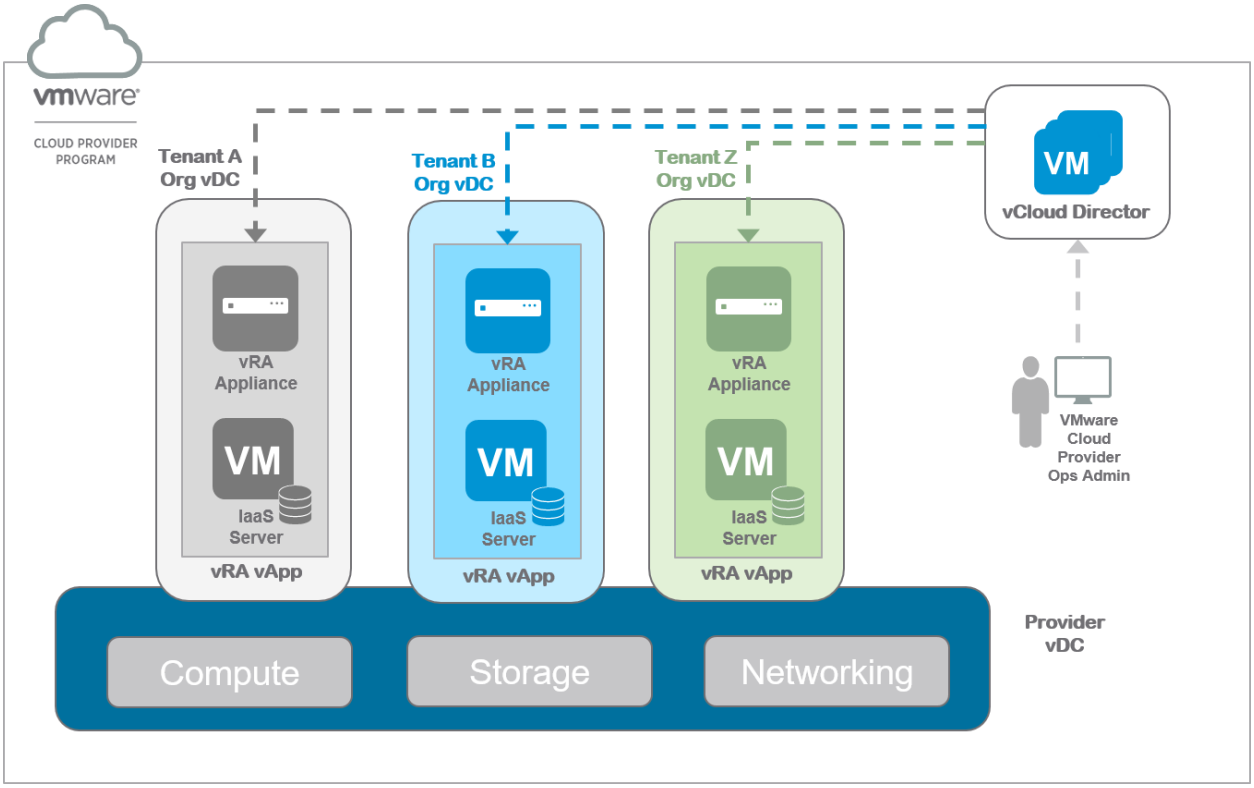
2.1.3 Hosting Environment Management Infrastructure

This automated deployment of vRealize Automation can be provisioned to service providers leveraging vCloud Director or vSphere.

2.1.3.1 vCloud Director

In cloud providers that offer vCloud Director as the IaaS platform, instances of vRealize Automation can be deployed from a VMware vSphere vApp™ template that has the necessary components, such as the vRealize Automation appliance and a Windows VM with MS SQL installed. The service provider can then deploy the vApp, and leverage the vRealize Automation unattended installer and their own custom configuration scripts to deploy and configure the instance of vRealize Automation.

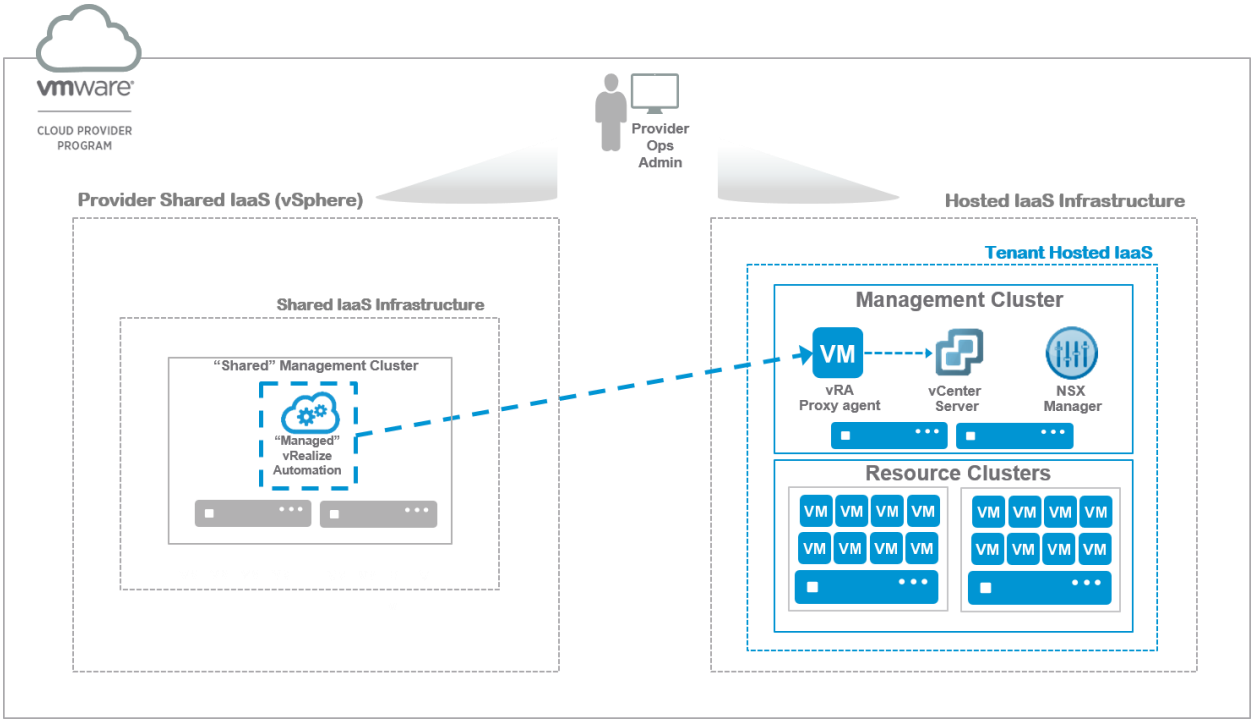
Figure 1. vApp Deployment of vRealize Automation in vCloud Director



2.1.3.2 vSphere

In environments that have vSphere as the main IaaS platform, the service provider can use VM templates. The service provider can then use a similar unattended install and configuration method as described for vCloud Director in Section 2.1.3.1, Hosting Environment Management Infrastructure.

Figure 2. Hosted vSphere Example



2.1.4 Development and Test Environments

During the lifetime of a vRealize Automation installation, new use cases arise that require the implementation of new service catalog items and features. Customers might make additions to the virtual machine lifecycle workflow or have a need for new Day-2 operations to be exposed to users for a better user experience. The testing of new extensibility features does not impact the immediate or long-term usability of a production instance of vRealize Automation. Therefore, VMware recommends that a development instance of vRealize Automation be implemented to develop new service catalog content and features.

2.1.4.1 Service Provider Development Instance

VMware recommends that service providers deploy a development or non-production instance of vRealize Automation to test the deployment of new blueprints or extensibility workflows.

When the development and validation of new blueprints or extensibility workflows is complete, the artefacts can be migrated to a test instance (if this is the standard operating procedure) or migrated to the customer production instance of vRealize Automation.

After successful validation of the new updates in production, the blueprint or Day-2 Operation can be added as an entitlement to the requisite business groups and users. vRealize CloudClient can be used in this process to export and import content between environments, or a tool such as VMware vRealize Code Stream™ Management Pack for IT DevOps can be considered.

2.1.4.2 Customer Development Instance

In a dedicated customer private cloud scenario, the service provider can deploy a development instance of vRealize Automation for the customer. The customer can use this instance of vRealize Automation to develop the necessary extensibility required for users in the production environment. As with the service provider development instance scenario, extensibility updates are developed and validated in the

development environment, and then moved to a test instance (if this is standard operating procedure) or to the production instance of vRealize Automation.

2.2 Infrastructure Prerequisites

2.2.1 Active Directory

For increased ease of user management, the use case assumes that an Active Directory (AD) infrastructure is available for user authentication and group management. VMware recommends that this Active Directory instance be a central instance that the service provider controls. Using centrally controlled instance of Active Directory also simplifies the configuration of managing hosted vSphere environments that are consumed as an endpoint in vRealize Automation.

VMware recommends that a naming convention for the corresponding AD groups be created and maintained. This allows for more consistent implementation of the prescribed automation and reusability of the configuration script. This also reduces the number of inputs required for usage of the configuration script. The following are examples of AD group naming conventions:

- Tenant Admin Group − <*tenant prefix*>-tenantadmin
- IaaS Admin Group − < *tenant prefix*>-iaasadmin
- Fabric Group Admin Group − < *tenant prefix*>-fgadmin
- Business Group Admin Group − < *tenant prefix*>-bgadmin

2.2.2 Domain Name Service (DNS)

vRealize Automation components must be referenced by Fully Qualified Domain Name (FQDN). For external access to the vRealize Automation instance from remote location or over the Internet, the FQDN of the vRealize Automation appliance can be configured to point to a NATed or load-balanced IP address of the vRealize Automation appliance.

2.2.3 Certificates

To provide secure communications between components, as well as for external access to the vRealize Automation instance by external users, each instance of vRealize Automation must be configured with CA signed certificates that are placed on the vRealize Automation appliance and IaaS server.

In the case that there are Internet-facing components of the solution, namely the vRealize Automation appliance, VMware recommends that the appliance (or load balancer) be configured with a public CA signed certificate. Using public CA signed certificates at this tier enhances security between vRealize Automation service tiers.

For the rest of the components, internal CA signed certificates can be issued for the secure communication of vRealize Automation components. For these internal components, VMware recommends using a signed SAN certificate to help decrease the complexity of certificate placement, especially with distributed deployments of vRealize Automation.

2.2.4 Network Connectivity

When considering the network connectivity for vRealize Automation in a service provider context, it is important that the vRealize Automation appliance node or load balancer has the proper connectivity for external access if required. If the vRealize Automation instance is accessed from outside the cloud service provider's environment, a load balancer, associated public IP address, and DNS must be configured with appropriate security measures.

2.2.4.1 Connectivity with vCloud Director

Edge Gateway

In this example, the Organization VDC is configured with an Edge Gateway that is connected to an external network.

Organization VDC Networks

The Organization VDC is configured with two Organization VDC Networks. The number of Org Networks listed here is an example, and the actual networks implemented are dependent on the connectivity needs within the cloud environment.

Figure 3. Example of OrgVDC Network Connectivity

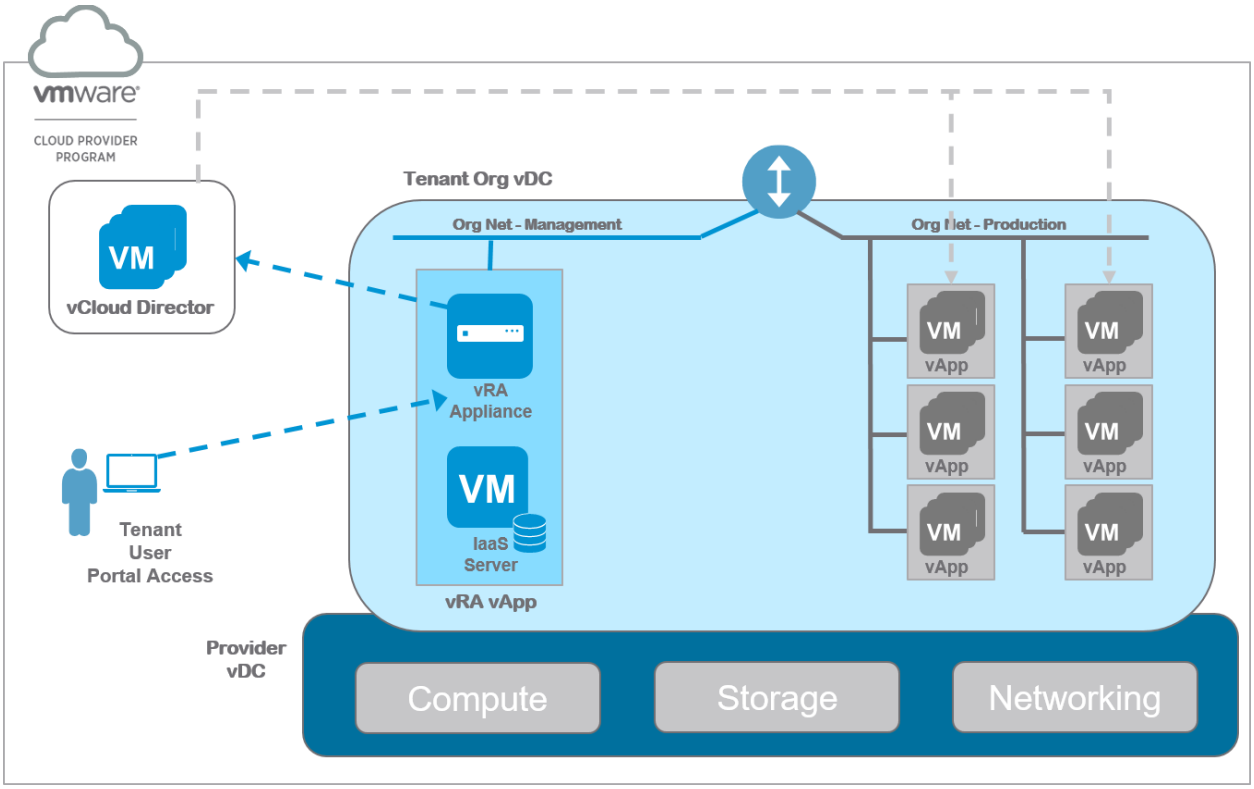


Table 2. Example Org Networks

Org Network	Purpose
Management	Used for management-related functions such as AD, DNS, and other cloud service or functions
Customer	Used for connectivity of workloads deployed from vRealize Automation

2.3 Use Case Software Components and Considerations

2.3.1 vRealize Automation

For the purposes of this use case, vRealize Automation is deployed in a small size deployment. This decision was made to simplify installation and management of the vRealize Automation instance. The vRealize Automation instance consists of the following servers:

- One (1) vRealize Automation appliance
- One (1) Windows 2012 R2 server with MSSQL 2012 installed

The Windows server hosts all IaaS functions and must be sized appropriately based on the vRealize Automation Reference Architecture documentation. The vRealize Automation appliance provides the vRealize Automation portal and vRealize Orchestrator services.

2.3.2 vRealize CloudClient

vRealize CloudClient is a CLI utility with a unified interface across vRealize Automation APIs. CloudClient is used to configure each single-tenant instance of vRealize Automation. CloudClient must be configured to leverage the "*cloudclient.properties*" and *cloudclient.config* files when being leveraged by scripts. In a later section, examples are provided on how to update the parameters provided by these files during the execution of a PowerShell script.

Cloud Client is a Java application and requires the Java Runtime Environment (JRE) to be installed. Refer to the documentation for the recommended version of Java.

2.3.3 Converged Blueprint JSON Files (Optional)

These files can be leveraged with a PowerShell script and CloudClient to automate the configuration of Blueprints into vRealize Automation. This script can then be configured to leverage CloudClient to associate the blueprint with the necessary service and entitlements, and to add the blueprint as a consumable catalog item.

Blueprint	Path
Windows	C:\Blueprints

The listed Blueprints are exported from an existing vRealize Automation instance that has vCloud Director configured as an endpoint. This way, the pertinent details of the vApp templates leveraged for the blueprints are captured.

2.3.4 Execution with vRealize Orchestrator

A vRealize Orchestrator wrapper workflow can be leveraged to remotely run a PowerShell configuration script. This allows the script to be executed as an XaaS workflow in a management instance of vRealize Automation.

2.4 vRealize Automation Component Configuration

2.4.1 vRealize Automation vApp

The vRealize Automation vApp template that is configured connects both VMs configured to the Management Org Network. The VMs in the vApp have dependencies on the existing AD and DNS services provided by the service provider.

2.5 Silent Installation

vRealize Automation 7.1 introduced silent installation functionality, which utilizes an answer file (ha.properties) at install time.

At the time of deployment, the vApp or VM components are provisioned, and then the vRealize Automation silent installation is performed. By utilizing the **ha.properties** answer file on the vRealize Automation appliance, a unique, single-tenant instance of vRealize Automation is configured for the associated customer.

```
#All values should be entered using single quotes:''
#If a value contains the single quote symbol:', it must be escaped in the value as '\''
#for example "value's" must be entered in the following way:
#param='value'\''s'

#Accept EULA. Change to true, if EULA is accepted. It can be found in license.txt.
#The installation will not proceed if EULA is not accepted.
AcceptEULA='False'

#Certificate Generation data
## CERT_SIGN - Certificate Signature Algorithm
## VALID OPTIONS: sha1; sha256; sha384; sha512
CERT_SIGN='sha256'

#KEY_LEN='PRIVATE KEY LENGTH'
## VALID OPTIONS: 1024; 2048; 2046
KEY_LEN='4096'

## KEY_TYPE: VALID OPTIONS: RSA; NO_RSA
KEY_TYPE='RSA'
DAYS_VALID='365'

#vRA License
vRA_LICENSE=''

#Timeout in seconds to wait for vRA services to start
SLEEP_MAX='1200'

#IaaS service user credentials.
#If USE_SINGLE_IAAS_CREDENTIALS is set to true all IaaS services will be installed with this user credentials and any o
#SINGLE_IAAS_USER must be specified with the domain prefix (e.g "domain\user").
#if the user is local the hostname cropped to the 15-th character must be provided as domain.
USE_SINGLE_IAAS_CREDENTIALS='true'
SINGLE_IAAS_USER='<iaas_domain>\ADMINISTRATOR'
SINGLE_IAAS_PASSWORD=''
```

It is important to note that the parameter values in the ha.properties file have to be updated programmatically in order to complete the silent installation procedure. One method of achieving this is to generate a version of the ha.properties files through a scripted procedure with the necessary values, and then copy this newly generated file to the vRealize Automation appliance prior to execution of the silent installation.

2.6 vRealize CloudClient Configuration

vRealize CloudClient can be installed on a utility server or other server that is dedicated for scripting and orchestration tasks. Because the use case described by this document leverages the most recent version of PowerShell, this server must be a Windows 2012 R2 VM and must be accessed and managed by the service provider.

2.6.1 CloudClient, User Roles, and Environment Variables

One item to be aware of when using CloudClient for scripting is that certain commandlets require the correct roles in vRealize Automation, and for some commandlets specific credentials (such as the *administrator@vsphere.local* account) are required for successful execution. Due to the requirements of different credentials for certain CloudClient commands, environment variables are used for the successful execution of CloudClient commands. Details of how to update environment variables are covered in a later section of this document.

2.7 PowerShell Script Configuration

For the demonstration of scripting using CloudClient, this use case focuses on PowerShell. PowerShell is a powerful and widely used scripting utility, and can be easily used to execute the necessary CloudClient commands for the configuration post installation.

To increase the flexibility of the script in this example use case, it has been implemented to use input parameters. These parameters allow for execution of the script by the user from the command line. They also provide the additional benefit of allowing the script to be executed from vRealize Orchestrator if necessary, for more robust orchestration of the deployments in a service provider environment.

2.7.1 Input Parameters

This section covers an example of some of the parameters that are necessary for the configuration of a vRealize Automation instance from CloudClient. The necessary parameters vary depending on the specifics of the deployment. Some of the common parameters related to the infrastructure involved are items such as:

- Active Directory Domain Name
- Base Distinguished Name of AD User and Groups
- Distinguished Name for Identity Store Login User and Password
- Active Directory/LDAP Server URL

The PowerShell script can be configured to accept parameters that are used to configure the necessary tenant objects in vRealize Automation. The following is an example of potential parameters to be captured as inputs of the PowerShell script:

Parameter	Details
$vraApplHostname	vRealize Automation appliance hostname
$vraApplIpAddr	vRealize Automation appliance IP address
$vraApplAdminUsername	vRealize Automation appliance administrator username
$vraApplAdminPassword	vRealize Automation appliance administrator password
$vraIaasHostname	vRealize Automation IaaS server hostname

Parameter	Details
$vraIaasIpAddr	vRealize Automation IaaS server IP address
$vraIaasAdminUsername	vRealize Automation IaaS server administrator username
$vraIaasAdminPassword	vRealize Automation IaaS server administrator password
$idStoreDomain	AD Domain Name
$idStoreBaseDn	Base DN used for search AD users and groups
$idStoreLoginUserDn	DN for identity store login user (typically the vRealize Automation Windows service account)
$idStoreDcUrl	URL of the AD/LDAP identity source
$tenantName	vRealize Automation tenant name (must be vsphere.local)
$customerPrefix	Prefix used to create unique naming for vRealize Automation groups and objects (**Note**: Because this parameter is also used for the creation of the machine names generated during requests, the $customerPrefix + Numeral suffix cannot exceed 15 characters.)
$credsUsersname	Username for target endpoint
$credsPassword	Password for target endpoint
$ComputerResourceName	Compute resource name (name of OrgVDC that is consumed)

The configuration of parameters also enables the script to be called from vRealize Orchestration without any changes. To help reduce the need for additional input variables required for the script, such as the business group, fabric group and entitlement naming, the CustomerPrefix value is prepended to the established naming convention of the script similar to the method described in section 3.2.1 for AD group naming.

2.7.2 Environment Variables

Because the successful configuration of vRealize Automation uses different CloudClient credentials for different tasks, these settings must be updated during the execution of the script for some commands.

One example of when updating the environment variables is necessary is using CloudClient for the addition of an identity source:

```
vra tenant identitystore add
```

or updating the infrastructure admin role membership:

```
vra tenant admin update
```

These CloudClient commands require the *administrator@vpshere.local* account. To set the update environment variables of the PowerShell script with the correct values, the following lines can be inserted in to the script to update the credentials used with CloudClient:

```
$env:CLOUDCLIENT_SESSION_KEY="administrator"
$env:vra_server="vra01.corp.local"
$env:vra_username="administrator@vsphere.local"
$env:vra_tenant="vsphere.local"
$env:vra_password="VMware1!"
```

With the environment variables updated to the *administrator@vsphere.local* credentials, the above-mentioned CloudClient commands can be successfully executed within the PowerShell script.

Remember, other CloudClient commands need Tenant Admin, Infrastructure Admin, or Infrastructure Architect roles. To change the environment variables to an account with the appropriately applied roles, add a section to the script updated with the new values:

```
$env:CLOUDCLIENT_SESSION_KEY="configurationadmin"
$env:vra_server="vra01.corp.local"
$env:vra_username="configurationadmin@vsphere.local"
$env:vra_tenant="vsphere.local"
$env:vra_password="VMware1!"
```

In this example, the environment variables have been updated with the credential values for the *configrationadmin@vsphere.local* account.

2.7.3 vRealize Automation Groups and Object Naming Creation

To reduce user error during script execution, the script has been configured to create the required group and object names automatically using the value of the $customerPrefix parameter/variable and $idStoreDomain (where required).

2.7.4 Leveraging Cloud Client with PowerShell

After the credentials for the *administrator@vsphere.local* account have been set, proceed to execute the CloudClient commands to add the required accounts to the tenant administrator role and IaaS administrator role:

```
###-------------------------------------------------------------------------
### 'vra tenant identitystore add' Section - Add Identity Store to
###  vsphere.local Tenant
###-------------------------------------------------------------------------


## Construct 'vra identitystore add' Command
$idStoreAddCommand = $CMD + " vra tenant identitystore add --tenantname " +
$tenantName + " --name "+ $idStoreDomain + " --domain " +
$idStoreDomain + " --groupbasedn " + $idStoreBaseDn + " --userdn " +
$idStoreLoginUserDn + " --password " + $credsPassword +
" --type AD --url " + $idStoreDcUrl + " --userbasedn " + $idStoreBaseDn

## Print 'vra identitystore add' Command to screen and then execute
Write-Host $idStoreAddCommand
Invoke-Expression $idStoreAddCommand


###-------------------------------------------------------------------------
### 'vra tenant admin update' Section - Update Infrastructure Admin role for
###  vsphere.local Tenant
###-------------------------------------------------------------------------

## Declare IaaS Admin Group
$iaasGroup = $customerPrefix + "-iaasadmin@" + $idStoreDomain

## Construct 'vra tenant admin update' Command
$tenantUpdateCommand = $CMD + " vra tenant admin update --tenantname " +
$tenantName + " --role IAAS_ADMIN --action ADD --users " + $iaasGroup

## Print 'vra tenant admin update' Command to screen and then execute
Write-Host $tenantUpdateCommand
Invoke-Expression $tenantUpdateCommand
```

In this example, variables are declared to construct the CloudClient commands "`vra tenant identity store add`" and "`vra tenant admin update`" with the desired parameters, of which the former requires the *administrator@vsphere.local* credentials. We then use the "Invoke-Expression" PowerShell commandlet to run the resulting CloudClient commands.

Once we have completed the necessary commands to update the tenant and IaaS administrator roles, we can update the environment credential variables for proper execution of vRealize Automation tenant constructs.

At this point, additional scripting can be created to continue the customer configuration of the tenant, such as:

- Creation of the customer's vCloud Director OrgVDC as an endpoint
- Fabric group creation
- Machine prefix
- Business group creation

Additionally, services, entitlements, and the required actions can be created for the consumption of pre-created converged blueprints backed by standard templates offered by the VMware Cloud Provider. After the necessary scripted tasks have been completed, reservations are created manually and the vRealize Automation instance can be turned over to the customer.

2.7.5 vRealize Automation API

There are some circumstances where it is necessary to use direct API commands against the vRealize Automation instance for configuration tasks. One area in which this is required is the update of role assignments. As of CloudClient 4.2, the support role updates consist of IAAS administrator, tenant administrator, service architect and approval administrator. For management of certain tenant items such

as blueprints, users require the infrastructure architect role, which for the purposes of this document, is enabled with the API.

By using the "**Invoke-RestMethod**" PowerShell commandlet, the script can be configured to update the infrastructure architect role (listed as the COMPOSITE_SERVICE_INFRASTRUCTURE ARCHITECT role in the API) to add the new user or group.

Interacting directly with the vRealize Automation API requires authentication to obtain a bearer token. This token is used to provide that operations executed through the API are authorized.

The following code snippet demonstrates how to use PowerShell to obtain a bearer token and execute a REST API call.

```
###-------------------------------------------------------------------------------------------------------------
### Use REST API natively to assign the 'Infrastructure Architect' role to configurationadmin@corp.local
###-------------------------------------------------------------------------------------------------------------

# API Credentials
$restAPICreds = @{
    username = $vraAdmin
    password = $vraPasswd
    tenant   = $tenantName
}

# Convert to JSON object
$json = $restAPICreds | ConvertTo-Json

# Build  URLs
$tenantRole      = 'COMPOSITION_SERVICE_INFRASTRUCTURE_ARCHITECT'
$tenantUpdateURL = 'https://' + $vraServer + '/identity/api/authorization/tenants/' + $tenantName + '/principals/' `
    + $configAdmin + '/roles/' + $tenantRole
$bearerTokenURL  = 'https://' + $vraServer + '/identity/api/tokens'

# Accept self signed certificates
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = { $True }

# Get bearer (auth) token
Write-Host "Getting API Bearer token" -ForegroundColor Magenta
Write-Host "Invoke-RestMethod -Uri $bearerTokenURL -Method Post -Body $json -ContentType 'application/json'"
$response = Invoke-RestMethod -Uri $bearerTokenURL -Method Post -Body $json -ContentType 'application/json'

# Embed token in header object
$headerParams = @{
    Authorization = "Bearer " + $response.id
}

# Assign the role
Write-Host 'Invoke-RestMethod -Uri $tenantUpdateURL -Headers $headerParams -Method Put' -ForegroundColor Magenta
Invoke-RestMethod -Uri $tenantUpdateURL -Headers $headerParams -Method Put
```

Deploying the Solution

3.1 Executing vRealize Automation Instance Deployment

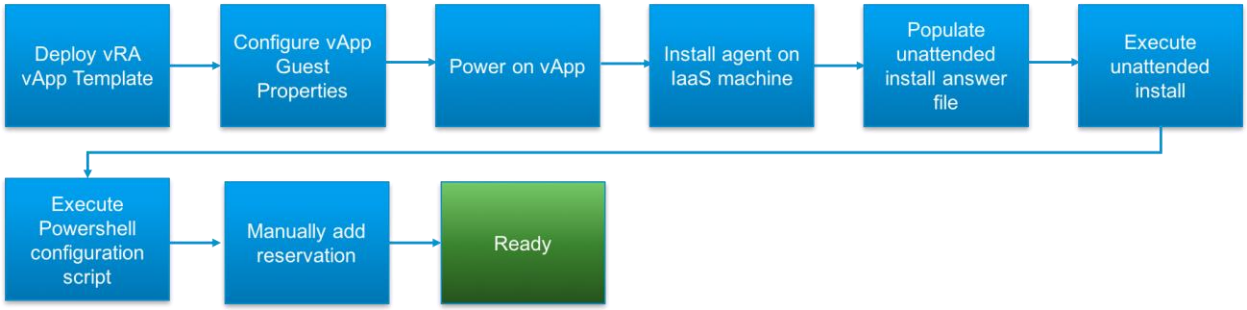
In this section, we outline the high-level workflow as an example of how vRealize Automation can be deployed programmatically for multiple and consistent deployments. Production implementations of this type of provisioning vary from use case to use case, as well as between the environments that vRealize Automation is being deployed in.

1. Deploy vRealize Automation vApp template into the OrgVDC of the customer from the provider's vApp catalog
2. Configure Guest Properties for each VM
3. Power On vApp
4. Install the vRealize Automation installation agent on the IaaS Windows VM
5. Populate the vRealize Automation ha.properties answer file
6. Execute the silent installation of vRealize Automation via a script
7. When the silent installation of vRealize Automation has been completed, the PowerShell script design for vRealize Automation configuration can be executed to:
    a. Configure the tenant identity store
    b. Add necessary users and groups for management of the vRealize Automation instance
    c. Configure vRealize Automation roles access
    d. Configure endpoint compute resource and perform data collection
    e. Configure fabric group and default machine prefix
    f. Configure business group and assign business group roles
    g. Import any pre-configure converged blueprints (optional)
    h. Create necessary services and entitlements
8. When the configuration script has successfully executed, an operations administrator must log into the instance and create the necessary reservations for the business group (if using vSphere Compute Resources, this can be configured via CloudClient during the PowerShell script).
9. Login as a test user and verify installation and configuration of vRealize Automation instance

The steps described are included in the PowerShell script and can then be configured for execution through vRealize Orchestrator. Using the Invoke an external script workflow included with the PowerShell plugin for vRealize Orchestrator, a workflow can be configured that accepts the parameters for the script and then executes the script by connecting to the designated PowerShell host. Additionally, it is possible to register this workflow to be executed from a "Master" vRealize Automation instance managed by the cloud service provider

The following figure illustrates the deployment process.

Figure 4. Deployment Process Diagram



Operational Considerations

4.1 Tenant Roles and Responsibilities

This section details some of the recommendations for key vRealize Automation role assignment as it relates to usage in a dedicated customer private cloud or fully managed service offering deployment. The actual role assignment that is used may vary based on the specific offering. However, these guidelines can provide clarity on the most feasible alignment for users and administrators that access the vRealize Automation instances. For more details on the permissions and descriptions of these roles, see the vRealize Automation Foundations and Concepts document.

4.1.1 Tenant Admin

The Tenant Admin role typically contains Active Directory groups for cloud service provider administrators that manage the Tenant configuration of the vRealize Automation instance.

4.1.2 Infrastructure Admin

The tenant admin role must contain groups from the cloud service provider administrators that manage Infrastructure configuration of the vRealize Automation instance.

4.1.3 Fabric Group Admin

The fabric group admin role must contain groups from the cloud service provider administrators that manage the compute resources.

4.1.4 Infrastructure Architect

The infrastructure architect role is used for creating and managing blueprints. This role is also necessary for the import and export of blueprints from vRealize Automation. In a fully managed offering of vRealize Automation, cloud service provider administrators are assigned to this role.

4.1.5 XaaS Architect

This role has the permissions to create XaaS blueprints and are typically assigned to the cloud service provider administrators. In a fully dedicated private cloud, this role can be assigned to the appropriate customer personnel responsible for creating XaaS blueprints.

4.1.6 Software Architect

Assigned to cloud service provider administrators who create and manage software blueprint components in a fully managed offering. Can be extended to the appropriate customer personnel in a Dedicated Private Cloud offering.

4.1.7 Catalog Administrator

Role that is designated for the creation and management of service catalogs and catalog items. Typically assigned to cloud service provider administrators in a fully managed offering.

4.1.8 Business Group Manager

The business group manager role must contain groups from the cloud service provider administrators that are responsible for adding and removing users to the business group and for overall management of machines in the business group.

In situations that require the customer to manage business group users or create blueprints based on existing templates on the underlying endpoints, select end users from the customer can be added to the

business group manager role. VMware recommends that customer assignment to the business group manager role be used sparingly in a fully managed offering.

4.1.9 Business Group Support User

The business group support user role must be used to allow users within a business group to see all of the machines provisioned in the business group. This is ideal for customers that require all of the users in a business group to have access to all of the systems created in the business group, for example when a team is working on the development of an application. If this is the only role that a user has been granted, then this is only able to request machines on behalf of other users and not for themselves.

4.1.10 Business Group User

The business group user role allows users to request workloads. if this is the only role that the user has within the business group, then the users are able to manage only the workloads that they have requested.

4.1.11 Service Provider vs. Customer Role Demarcation

The following tables show recommended role assignments between the service provider and a customer for the different deployment models.

Table 3. Fully Managed Tenant Role Recommendations

	Tenant Admin	Infrastructure Admin	Fabric Group Admin	Infrastructure Architect	XaaS Architect	Software Architect	Catalog Admin
Cloud Provider Admins	✓	✓	✓	✓	✓	✓	✓
Customer Admins	✗	✗	✗	✗	✗	✗	✗

Table 4. Dedicated Private Cloud Tenant Role Recommendations

	Tenant Admin	Infrastructure Admin	Fabric Group Admin	Infrastructure Architect	XaaS Architect	Software Architect	Catalog Admin
Cloud Provider Admins	✓	✓	✓	✗	✗	✗	✗
Customer Admins	✓	✓	✓	✓	✓	✓	✓

Table 5. Business Group Recommendations

	Business Group Manager	Business Group Support User	Business Group User
Cloud Provider Admins	✓	✗	✗
Customer Admins/Line of Business Manager	✓	✗	✗
Customer Users	✗	✓	✓

4.1.12 AD Groups Recommendations

As mentioned in an earlier section of this document, leverage AD groups where possible for the assignment of roles within vRealize Automation. For more granular assignment of users to roles, create one AD group to correspond with each vRealize Automation role.

4.1.13 Blueprints

Blueprints are the core representation of virtual machines in vRealize Automation. Blueprints contain the compute, storage, and networking configurations of machines that are deployed from vRealize Automation. In addition to this, Blueprints contain the custom properties that are used during any extensibility activities required during the life cycle of the deployed virtual machine. One of the key components of the blueprint is the VM/vApp template that is created in the endpoint in which resources are consumed.

4.1.13.1 Templates

Before successfully completing the configuration of blueprints, create a template in the endpoint in which resources will be provisioned. One key goal when creating this template is to make the VM template as reusable as possible. This allows the one template to be used by many blueprints in vRealize Automation. This helps ease the management overhead of blueprints, since there are fewer templates to manage. Reducing the amount of software installed on the template also reduces the number of updates that need to be made to the template.

For implementations in which vRealize Automation is deployed in vCloud Director, and for managing the OrgVDC in which it is contained, cloud service provider administrators create the vApp templates and publish them in the public catalog for consumption.

In fully managed offerings in which vSphere is used as the compute resource, the required VM templates have to be created by the cloud service provider administrators in each respective hosted vSphere instance. For dedicated private cloud offerings, the customer's internal administrators are responsible for creating the necessary VM templates for use with vRealize Automation blueprints.

4.2 Upgrading and Patching vRealize Automation

As of vRealize Automation 7.2, upgrades and patching can be performed programmatically with the API provided by the virtual appliance management interface (VAMI).

Automated upgrades are out of scope for this document, but much like the deployment process previously described, automated upgrades can be driven by PowerShell REST-method invocation commandlets.

4.3 Backups

Given the "'black box" nature of the vRealize Automation appliance, VMware recommends taking a full image backup using a vADP-compliant or vCloud Director-aware backup solution, depending on where vRealize Automation is deployed. Back up customer vRealize Automation appliances and the IaaS Windows servers together as full image backups.

4.4 Content Management

If multiple customers are likely to consume common IaaS blueprints, then a content management process enables a service provider to create one blueprint that can be pushed to customer vRealize Automation instances for consumption.

Content management is out of scope for this document, but VMware recommends the use of vRealize Code Stream Management Pack for IT DevOps to efficiently deploy vRealize Automation-based content to multiple customer vRealize Automation instances

Conclusion

Deploying vRealize Automation with a VMware Cloud Provider offers a method for deployment of a private cloud for customers that need the quick deployment of the Cloud Management Platform with limited day-to-day management of the solution. VMware Cloud Providers can help customers realize an expedited time to value with vRealize Automation on top of their VMware validated infrastructure. This provides increased uptime and an environment that is similar to the one their customers run in their on-premises data centers today. For more details on this and other VMware Cloud Provider Program solutions, visit the VMware vCloud Architecture Toolkit™ site.

Appendix A: Acronyms and Terms

Term	Description
AD	Active Directory
API	Application Programming Interface
CA	Certificate Authority
CMDB	Change Management Database
Day-2	Operations that occur after provisioning and before decommissioning of a VM
DEM	Distributed Execution Manager
DN	Distinguished Name
DNS	Domain Name Service
FQDN	Fully Qualified Domain Name
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
LDAP	Lightweight Directory Access Protocol
NAT	Network Address Translation
Org	Organization
OrgVDC	Organization Virtual Data Center
REST	Representational State Transfer
SDDC	Software-Defined Data Center
UI	User Interface
URL	Uniform Resource Locator
vApp	VMware vCloud Director® VM Container
VM	Virtual Machine
XaaS	X as a Service (Anything as a Service)

Appendix B: References

The following items provides additional information pertinent to this document and its topics.

Document Title	Link or URL
Foundations and Concepts	http://pubs.vmware.com/vrealize-automation-71/topic/com.vmware.ICbase/PDF/vrealize-automation-71-foundations-and-concepts.pdf
Installing vRealize Automation 7.1	http://pubs.vmware.com/vrealize-automation-71/topic/com.vmware.ICbase/PDF/vrealize-automation-71-installation-and-configuration.pdf
Configuring vRealize Automation	http://pubs.vmware.com/vrealize-automation-71/topic/com.vmware.ICbase/PDF/vrealize-automation-71-configuration.pdf
Managing vRealize Automation	http://pubs.vmware.com/vrealize-automation-71/topic/com.vmware.ICbase/PDF/vrealize-automation-71-management.pdf
Programming Guide	http://pubs.vmware.com/vrealize-automation-71/topic/com.vmware.ICbase/PDF/vrealize-automation-71-programming-guide.pdf
REST API Reference	http://pubs.vmware.com/vrealize-automation-71/topic/com.vmware.vra.restapi.doc/docs/index.html
CloudClient 4.2 Documentation	https://my.vmware.com/group/vmware/get-download?downloadGroup=CLOUDCLIENT_420&productId=601