Custom Properties Reference

28 December 2020

vRealize Automation 7.6

You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

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

Contents

Custom Properties Reference

Custom Properties Reference provides information about the custom properties, and their use, that are available when you use vRealize Automation.

This documentation is intended to be used with the vRealize Automation product documentation available from the vRealize Automation product documentation at https://docs.vmware.com/en/vRealize-Automation/index.html.

Intended Audience

This information is intended for IaaS administrators, fabric administrators, and business group managers of vRealize Automation. This content is written for experienced Windows or Linux system administrators who are familiar with virtualization technology and the basic concepts described in *Foundations and Concepts*.

Updated Information

This *Custom Properties Reference* is updated with each release of the product or when necessary.

This table provides the update history of the *Custom Properties Reference*.

Revision	Description
14-FEB-2019	Minor updates.
09-SEP-2019	Minor updates.
18 JUL 2019	Updated `VirtualMachine.Software.Execute`, `VirtualMachine.Admin.UseGuestAgent`, `VirtualMachine.Customize.WaitComplete`, and `VirtualMachine.Admin.CustomizeGuestOSDelay` in Custom Properties V.
01 MAR 2019	Updated Custom Properties V.
25 JAN 2019	Updated Custom Properties Underscore (_) .
13 NOV 2018	Minor updates.
04 OCT 2018	Minor updates.
20 SEP 2018	Initial release.

Custom Properties and the Property Dictionary

1

You can use supplied vRealize Automation custom properties to control various aspects of machine provisioning. You can also use the property dictionary to create new property definitions and property groups that are tailored to your specific needs.

You can use properties to add values or override existing or default values for configuring network, platform, and guest agent settings and many other deployment-related parameters.

This chapter includes the following topics:

Using Custom Properties

You can use vRealize Automation custom properties to add values or override existing or default values for configuring settings for network, platform, guest agent, and many other deployment parameters.

Some properties are determined by standard settings that you must specify for all machines. For example, memory and disk size values are required for all blueprints. You can specify additional properties individually or in property groups in blueprints and in reservations.

When you add a property to a blueprint or a property group, you can mark it as a required property. When a property is specified as required, the user must provide a value for that property when they request a machine, such as in the following examples.

- Require information about multiple disks sharing the machine's allocated storage.
- Require information about users or groups to be added to a local group on the machine.
- Require the host name of the machine.

The Windows guest agent records property values on the provisioned machine in the `%SystemDrive%\VRMGuestAgent\site\workitem.xml` file.

The Linux guest agent records property values on the provisioned machine in the `/usr/share/gugent/site/workitem.xml` file.

Creating and Adding Custom Properties and Property Groups

You can use custom properties to control machine provisioning. You can add supplied custom properties and also create and add your own properties and property groups.

You can add properties and property groups to overall blueprints, components in a blueprint, reservations and other vRealize Automation items, including some endpoint types. You can also create new custom properties and property groups.

You can add properties and property groups when you create a blueprint, or later when the blueprint is in the draft or published state. Alternatively you can add custom properties and property groups to individual components in the blueprint.

Blueprint-level custom properties take precedence over custom properties that are configured at the component level. For information about custom property precedence, see Understanding Custom Properties Precedence.

You can edit blueprint-level properties by using the blueprint properties page.

A custom property can optionally require that the user specify a property value when they create a machine request.

- Custom property names and values are typically case-sensitive. For example, a custom property expressed as `hostname` and another custom property expressed as `HOSTNAME` are considered different custom properties.
- Custom property names cannot contain spaces. When creating and using custom properties, do not include a space in the property name.
- Some custom property names are reserved and cannot be used as names when you create new custom properties. For example the property name `Encrypted` and `encrypted` is reserved.

For more information about creating new custom properties and property groups, see Using the Property Dictionary.

Using Properties in Machine Provisioning

Custom properties are vRealize Automation-supplied properties. You can also define your own properties. Properties are name-value pairs used to specify attributes of a machine or to override default specifications.

You can use custom properties to control different provisioning methods, types of machines, and machine options as in the following examples:

- Specify a particular type of guest OS.

- Enable WIM-based provisioning, in which a Windows Imaging File Format (WIM) image of a reference machine is used to provision new machines.
- Customize the behavior of Remote Desktop Protocol when connecting to a machine.
- Register a virtual machine with a XenDesktop Desktop Delivery Controller (DDC) server.
- Customize a virtual machine's system specifications, such as adding multiple disk drives.
- Customize the guest OS for a machine, for instance, by including specified users in selected local groups.
- Specify network and security settings.
- Add additional control options such as drop-down menus to make input and selection options available to the consumer at request time.

When you add a property to a blueprint, reservation, or other form you can specify if the property is to be encrypted and also if the user must be prompted to specify a value when provisioning. These options cannot be overridden when provisioning.

For an example of how to additional control options to dynamically set a custom property based on a consumer's selection from a list of predefined options, see the Adding a Network Selection Drop-Down in vRA 7 blog post.

A property specified in a blueprint overrides the same property specified in a property group. This enables a blueprint to use most of the properties in a property group while differing from the property group in some limited way. For example, a blueprint that incorporates a standard developer workstation property group might override the US English settings in the group with UK English settings.

You can apply properties in reservations and business groups to many machines. Their use is typically limited to purposes related to their sources, such as resource management. Specifying the characteristics of the machine to be provisioned is generally done by adding properties to blueprints and property groups.

Understanding Custom Properties Precedence

Properly authorized users can specify custom properties for blueprints, endpoints, business groups, and reservations. When the same property exists in more than one source, vRealize Automation follows a specific order of precedence when applying properties to the machine.

You can add custom properties that apply to provisioned machines to the following elements:

- A reservation, to apply the custom properties to all machines provisioned from that reservation.
- A business group, to apply the custom properties to the deployment and to all machines provisioned by business group members.
- A blueprint, to apply the custom properties to all machines provisioned from the blueprint.
- Property groups, which can be included in a blueprint, to apply all the custom properties in the group to all machines provisioned from the blueprint.

A blueprint can contain one or more property groups.

- A machine request to apply the custom properties to the machine being provisioned.
- An approval policy, if advanced approval support is enabled, to require approvers to provide values for the machine being approved.

The following list shows the order of precedence for custom properties. Property values specified in a source that appears later in the list override values for the same property specified in sources that appear earlier in the list.

If a conflict exists between a vRealize Automation-supplied custom property name and a user-defined property name, the vRealize Automation-supplied custom property name takes precedence.

1. Property group
2. Blueprint
3. Business group
4. Compute resource
5. Reservations
6. Endpoint
7. Runtime

Property group, blueprint, and business group custom properties are assigned at request time, while other compute resource, reservation, and endpoint properties are assigned during provisioning.

This order is further clarified as follows:

1. Custom properties and groups at the overall blueprint level
2. Custom properties and groups at the component level
3. Custom properties for the business group
4. Custom properties for the compute resource
5. Custom properties for the reservation
6. Custom properties for the endpoint
7. Custom properties at the nested blueprint request level
8. Custom properties at the component request level

In most situations, a runtime property takes precedence over other properties. A runtime property meets the following conditions:

- The custom property option to prompt the user is selected, which specifies that the user must supply a value for the property when they request machine provisioning.

- A business group manager is requesting machine provisioning and the property appears in the custom properties list on the machine request confirmation page.

There are exceptions to the precedence rules. For example, you add the `VMware.VirtualCenter.Folder` custom property to a business group, provide a property value, and do not select the option to show the property in the request. You add the same custom property to a blueprint and specify that the property be shown in the request. When your designated users request provisioning from the catalog, the property does not appear in the catalog request form because the property applies to reservation information that is only available after provisioning begins, and not when you request provisioning.

Custom Property Types

You can use vRealize Automation external and updated property types for cloned machines. You cannot use Internal and read-only property types for cloned machines.

The following vRealize Automation custom property types are available.

- Internal

  The specified value is maintained in the database only. For example, the email address of the manager who approved a machine request is recorded in the `VirtualMachine.Admin.Approver` property but the property has no effect on the machine.

- Read-only

  The specified value is implemented on the machine and cannot be changed. For example, `VirtualMachine.Admin.UUID` specifies the UUID of the machine, which cannot be changed.

- External

  A machine’s external properties are determined when the virtualization platform creates the machine or during the WinPE phase of the build process. To set these properties, their values must be provided to the proxy agent, which passes them on to the virtualization platform, or to the guest agent , which implements them in the WinPE phase.

  The specified value is implemented on the machine but is never updated. For example, if the property `VirtualMachine.Admin.AddOwnerToAdmins` is set to true, the owner of the machine is added to its local administrators group. If the owner is later removed from this group, the property is not updated to false.

- Updated

  The specified value is implemented on the machine and is updated through data collection. For example, if the compute resource of a machine is changed, a proxy agent updates the value of the machine’s `VirtualMachine.Admin.Hostname` property.

Internal and read-only property types set attributes that the template determines.

You can use the vRealize Automation machine menu to change all reserved custom properties except the read-only properties `VirtualMachine.Admin.AgentID`, `VirtualMachine.Admin.UUID`, and `VirtualMachine.Admin.Name`.

Custom Properties Grouped by Function

You can use custom properties to provide additional vRealize Automation controls.

Custom properties have been grouped here by function. To explore custom properties grouped by name, see Custom Properties Grouped by Name.

- Custom Properties for Deployments

  vRealize Automation provides several custom properties that are applicable to most deployments.

- Custom Properties for Naming and Analyzing Deployments

  If provisioning fails, vRealize Automation rolls back all resources included in the catalog item. For deployments that contain multiple components, you can use a custom property to override that default and receive information to debug the failure. These properties are best used when applied to the overall blueprint.

- Custom Properties for OpenStack Endpoints

  vRealize Automation includes custom properties you might want to use when you configure your OpenStack endpoints in vRealize Automation.

- Custom Properties for Clone Blueprints

  vRealize Automation includes custom properties that you can use to provide additional controls for clone blueprints.

- Custom Properties for Linked Clone Blueprints

  vRealize Automation includes custom properties that you can use to provide additional controls for linked clone blueprints.

- Custom Properties for FlexClone Blueprints

  vRealize Automation includes custom properties that you can use to provide additional controls for FlexClone blueprints.

- Custom Properties for Basic Workflow Blueprints

  vRealize Automation includes custom properties that you can use to provide additional controls for basic workflow blueprints.

- Custom Properties for Linux Kickstart Blueprints

  vRealize Automation includes custom properties that you can use to provide additional controls for Linux Kickstart blueprints.

- Custom Properties for SCCM Blueprints

  vRealize Automation includes custom properties that you can use to provide additional controls for SCCM blueprints.

- Custom Properties for WIM Blueprints

  vRealize Automation includes custom properties that provide additional controls for WIM blueprints.

- Custom Properties for vCloud Air and vCloud Director Blueprints

  You can add certain custom properties to a vCloud Air or vCloud Director machine component definition in a blueprint.

- Custom Properties for Networking and Security

  The vRealize Automation custom properties for networking specify configuration for a specific network device on a machine.

- Custom Properties and Property Groups for Containers

  You can add predefined property groups to a containers component in a vRealize Automation blueprint. When machines are provisioned by using a blueprint that contain these properties, the provisioned machine is registered as a Docker Container host machine.

- Custom Properties for PXE Provisioning

  PXE is the only provisioning method supported for Cisco UCS Manager. You can use the network bootstrap program with vRealize Automation custom properties to initiate WIM, SCCM, or Linux Kickstart provisioning. You can also use custom properties to call your own PowerShell scripts. Linux Kickstart provisioning does not require custom properties.

- Custom Properties for OVF Import

  When you import an OVF to a blueprint, you can import and configure several settings as custom properties.

- Custom Properties for vRealize Automation Guest Agent

  If you have installed the vRealize Automation guest agent in your templates for cloning or in your WinPE, you can use custom properties to run custom scripts within the guest operating system of a provisioned machine after the machine is fully deployed.

- Custom Properties for BMC BladeLogic Configuration Manager Integration

  vRealize Automation includes custom properties that you can use to provide additional controls for BMC BladeLogic Configuration Manager integration.

- Custom Properties for HP Server Automation Integration

  vRealize Automation includes custom properties that you can use to provide additional controls for HP Server Automation integration. Some custom properties are required for HP Server Automation integration. Other custom properties are optional.

Custom Properties for Deployments

vRealize Automation provides several custom properties that are applicable to most deployments.

Table 1-1. Custom Properties for Blueprints and Deployments

Custom Property	Description
`_debug_deployment`	Except for scale operations which allow partially successful deployments, the default behavior is to destroy the entire deployment if any of the individual resources fail to provision. You can override the default behavior by setting the `_debug_deployment` custom property value to true. If provisioning fails, the debugging custom property stops the resources from being rolled back so you can identify which of the components failed to provision successfully. In other words, by setting `_debug_deployment` to true, you can more easily debug customization and first-boot (for example, agent) issues because the setting ensures that machines are not destroyed after a provisioning failure. Otherwise, the setting doesn't directly change anything about the provisioning process, or affect guest agent or customization (for example, settings our outcomes relative to a vCenter customization spec). Note: A failed catalog item is normally inaccessible because it is immediately rolled back on failure. But when `_debug_deployment` is set to true, vRealize Automation treats the otherwise failed deployment as partially successful, which enables its accessibility. To apply the custom property to a blueprint, add `_debug_deployment` to the **Blueprint Properties** page using the **Properties** tab when you create or edit a blueprint. The `_debug_deployment` property is consumed at the software provisioning level, not the guest agent or machine provisioning level. You can also configure vRealize Automation to not delete virtual machines after deployment failure by using settings in the `VRMAgent.exe.config` file.
`_deploymentName`	When added to a blueprint, this property allows you to specify a custom name for the deployment by setting the value of `_deploymentName` to your custom string. If more than one instance of this deployment is provisioned in a single request, your custom name becomes a prefix. If you want users to specify their own deployment names, set this custom property to allow override. The following two caveats are required for usage: ■ You must add this property at the blueprint level, not at the component level. For example, when creating or editing a blueprint, click the **Properties** tab and then select **Custom Properties > New** to add the `_deploymentName` property to the blueprint. Do not add the property to a machine or other component in the blueprint. ■ You must add this property as a separate property and not as a member of a property group.

Custom Properties for Naming and Analyzing Deployments

If provisioning fails, vRealize Automation rolls back all resources included in the catalog item. For deployments that contain multiple components, you can use a custom property to override that default and receive information to debug the failure. These properties are best used when applied to the overall blueprint.

Table 1-2. Custom Properties for Analyzing Deployments

Custom Property	Description
`_debug_deployment`	Except for scale operations which allow partially successful deployments, the default behavior is to destroy the entire deployment if any of the individual resources fail to provision. You can override the default behavior by setting the `_debug_deployment` custom property value to true. If provisioning fails, the debugging custom property stops the resources from being rolled back so you can identify which of the components failed to provision successfully. Note: A failed catalog item is normally inaccessible because it is immediately rolled back on failure. But when `_debug_deployment` is set to true, vRealize Automation treats the otherwise failed deployment as partially successful, which enables its accessibility. In other words, by setting `_debug_deployment` to true, you can more easily debug customization and first-boot (for example, agent) issues because the setting ensures that machines are not destroyed after a provisioning failure. Otherwise, the setting doesn't directly change anything about the provisioning process, or affect guest agent or customization (for example, settings our outcomes relative to a vCenter customization spec). To apply the custom property to a blueprint, add `_debug_deployment` to the **Blueprint Properties** page using the **Properties** tab when you create or edit a blueprint. The `_debug_deployment` property is consumed at the software provisioning level, not the guest agent or machine provisioning level. You can also configure vRealize Automation to not delete virtual machines after deployment failure by using settings in the `VRMAgent.exe.config` file.
`_deploymentName`	When added to a blueprint, this property allows you to specify a custom name for the deployment by setting the value of `_deploymentName` to your custom string. If more than one instance of this deployment is provisioned in a single request, your custom name becomes a prefix. If you want users to specify their own deployment names, set this custom property to allow override. The following two caveats are required for usage: ■ You must add this property at the blueprint level, not at the component level. For example, when creating or editing a blueprint, click the **Properties** tab and then select **Custom Properties > New** to add the `_deploymentName` property to the blueprint. Do not add the property to a machine or other component in the blueprint. ■ You must add this property as a separate property and not as a member of a property group.

Custom Properties for OpenStack Endpoints

vRealize Automation includes custom properties you might want to use when you configure your OpenStack endpoints in vRealize Automation.

Table 1-3. Custom Properties for Openstack Endpoints

Custom Property	Description
`VirtualMachine.Admin.ConnectAddress.Regex`	Used by a vRealize Automation administrator to define a regular expression to match an IP address for terminal connections, such as an RDP connection. If matched, the IP address is saved under the `VirtualMachine.Admin.ConnectAddress` custom property. Otherwise, the first available IP address is designated. For example, setting the property value to `10.10.0.` allows selection of an IP address from a 10.10.0.* subnet that is assigned to the virtual machine. If the subnet has not been assigned, the property is ignored. This property is available for use with OpenStack.
`VirtualMachine.NetworkN.AdditionAddress`*M*	Defines additional *M* IP address allocated for an OpenStack instance for network *N*, excluding the IP address set specified by the `VirtualMachine.Network`*N*`.Address`. property. More addresses are displayed on the Network tab in the Additional Addresses column. This property is used by OpenStack machine state data collection. While this property is only data-collected by the OpenStack endpoint, it is not specific to OpenStack and can be used for lifecycle extensibility by other endpoint types. This property is not supported for on-demand NAT or on-demand routed networks.
`VMware.Endpoint.Openstack.IdentityProvider.Domain.Name`	Allows vRealize Automation to support required Keystone V3 domain-name authentication. If Keystone V3 is in effect, you can use the property to designate a specific domain for the OpenStack endpoint to authenticate with a Keystone V3 OpenStack identity provider. ■ For new endpoints, add the custom property to designate a specific domain. ■ For upgraded or migrated endpoints, add the custom property only if data collection fails after upgrade or migration.
`VMware.Endpoint.Openstack.IdentityProvider.Version`	Specifies the version of OpenStack identity provider (Keystone) to use when authenticating an OpenStack endpoint. Configure a value of `3` to authenticate with Keystone V3 OpenStack identity provider. If you use any other value, or do not use this custom property, authentication defaults to Keystone V2.

Custom Properties for Clone Blueprints

vRealize Automation includes custom properties that you can use to provide additional controls for clone blueprints.

Table 1-4. Custom Properties for Clone Blueprints

Custom Property	**Description**
`VirtualMachine.DiskN.StorageReservationPolicy`	Specifies the storage reservation policy to use to find storage for disk *N*. Also assigns the named storage reservation policy to a volume. To use this property, substitute the volume number for *N* in the property name and specify a storage reservation policy name as the value. This property is equivalent to the storage reservation policy name specified on the blueprint. Disk numbering must be sequential. This property is valid for all Virtual and vCloud reservations. This property is not valid for Physical, Amazon, or OpenStack reservations.
`VirtualMachine.NetworkN.NetworkProfileName`	Specifies the name of a network profile from which to assign a static IP address to network device *N* or from which to obtain the range of static IP addresses that can be assigned to network device *N* of a cloned machine, where *N*=0 for the first device, 1 for the second, and so on. The network profile that the property points to is used to allocate an IP address. The property determines the network that the machine attaches to, based on the reservation. Changing this property value after the network is assigned has no effect on the expected IP address values for the designated machines. With WIM-based provisioning for virtual machines, you can use this property to specify a network profile and network interface or you can use the Network section of the Virtual Reservation page. The following attributes of the network profile are available to enable static IP assignment in a cloning blueprint: ■ `VirtualMachine.NetworkN.SubnetMask` ■ `VirtualMachine.NetworkN.Gateway` ■ `VirtualMachine.NetworkN.PrimaryDns` ■ `VirtualMachine.NetworkN.SecondaryDns` ■ `VirtualMachine.NetworkN.PrimaryWins` ■ `VirtualMachine.NetworkN.SecondaryWins` ■ `VirtualMachine.NetworkN.DnsSuffix` ■ `VirtualMachine.NetworkN.DnsSearchSuffixes` `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation.

Table 1-4. Custom Properties for Clone Blueprints (continued)

Custom Property	Description
`Linux.ExternalScript.Name`	Specifies the name of an optional customization script, for example `config.sh`, that the Linux guest agent runs after the operating system is installed. This property is available for Linux machines cloned from templates on which the Linux agent is installed. If you specify an external script, you must also define its location by using the `Linux.ExternalScript.LocationType` and `Linux.ExternalScript.Path` properties.
`Linux.ExternalScript.LocationType`	Specifies the location type of the customization script named in the `Linux.ExternalScript.Name` property. This can be either local or nfs. You must also specify the script location using the `Linux.ExternalScript.Path` property. If the location type is nfs, also use the `Linux.ExternalScript.Server` property.
`Linux.ExternalScript.Server`	Specifies the name of the NFS server, for example lab-ad.lab.local, on which the Linux external customization script named in `Linux.ExternalScript.Name` is located.
`Linux.ExternalScript.Path`	Specifies the local path to the Linux customization script or the export path to the Linux customization on the NFS server. The value must begin with a forward slash and not include the file name, for example `/scripts/linux/config.sh`.

If your administrators installed the guest agent to run scripts that accept custom properties and customize provisioned machines, you can use custom properties to further customize cloned machines that use the guest agent.

Table 1-5. Custom Properties for Customizing Cloned Machines with a Guest Agent

Custom Property	Description
`VirtualMachine.Admin.AllowLogin`	Set to True (default) to add the machine owner to the local remote desktop users group, as specified by the `VirtualMachine.Admin.Owner` property.
`VirtualMachine.Admin.UseGuestAgent`	If the guest agent is installed as a service on a template for cloning, set to True on the machine blueprint to enable the guest agent service on machines cloned from that template. When the machine is started, the guest agent service is started. Set to False to deactivate the guest agent. If set to False, the enhanced clone workfow will not use the guest agent for guest operating system tasks, reducing its functionality to `VMwareCloneWorkflow`. If not specified or set to anything other than False, the enhanced clone workflow sends work items to the guest agent.
`VirtualMachine.DiskN.Active`	Set to True (default) to specify that the machine's disk *N* is active. Set to False to specify that the machine's disk *N* is not active.

Table 1-5. Custom Properties for Customizing Cloned Machines with a Guest Agent (continued)

Custom Property	Description
`VirtualMachine.DiskN.Label`	Specifies the label for a machine's disk *N*. The disk label maximum is 32 characters. Disk numbering must be sequential. When used with a guest agent, specifies the label of a machine's disk *N* inside the guest operating system.
`VirtualMachine.DiskN.Letter`	Specifies the drive letter or mount point of a machine's disk *N*. The default is C. For example, to specify the letter D for Disk 1, define the custom property as `VirtualMachine.Disk1.Letter` and enter the value D. Disk numbering must be sequential. When used in conjunction with a guest agent, this value specifies the drive letter or mount point under which an additional disk *N* is mounted by the guest agent in the guest operating system.
`VirtualMachine.Admin.CustomizeGuestOSDelay`	Specifies the time to wait after customization is complete and before starting the guest operating system customization. The value must be in HH:MM:SS format. If the value is not set, the default value is one minute (00:01:00). If you choose not to include this custom property, provisioning can fail if the virtual machine reboots before guest agent work items are completed, causing provisioning to fail.
`VirtualMachine.Customize.WaitComplete`	Set to True to prevent the provisioning workflow from sending work items to the guest agent until all customizations arecomplete. Set to False to allow work items to be created before customization is complete.
`VirtualMachine.SoftwareN.Name`	Specifies the descriptive name of a software application *N* or script to install or run during provisioning. This is an optional and information-only property. It serves no real function for the enhanced clone workflow or the guest agent but it is useful for a custom software selection in a user interface or for software use reporting.
`VirtualMachine.SoftwareN.ScriptPath`	Specifies the full path to an application's install script. The path must be a valid absolute path as seen by the guest operating system and must include the name of the script filename. You can pass custom property values as parameters to the script by inserting {*CustomPropertyName*} in the path string. For example, if you have a custom property named `ActivationKey` whose value is 1234, the script path is `D:\InstallApp.bat -key {ActivationKey}`. The guest agent runs the command `D:\InstallApp.bat -key 1234`. Your script file can then be programmed to accept and use this value.

Table 1-5. Custom Properties for Customizing Cloned Machines with a Guest Agent (continued)

Custom Property	Description
`VirtualMachine.SoftwareN.ISOName`	Specifies the path and filename of the ISO file relative to the datastore root. The format is `/folder_name/subfolder_name/file_name.iso`. If a value is not specified, the ISO is not mounted.
`VirtualMachine.SoftwareN.ISOLocation`	Specifies the storage path that contains the ISO image file to be used by the application or script. Format the path as it appears on the host reservation, for example `netapp-1:it_nfs_1`. If a value is not specified, the ISO is not mounted.

Custom Properties for Linked Clone Blueprints

vRealize Automation includes custom properties that you can use to provide additional controls for linked clone blueprints.

Certain vRealize Automation custom properties are required to use with linked clone blueprints.

Table 1-6. Custom Properties for Linked Clone Blueprints

Custom Property	Description
`VirtualMachine.DiskN.Storage`	Specifies the datastore on which to place the machine disk *N*, for example DATASTORE01. This property is also used to add a single datastore to a linked clone blueprint. *N* is the index (starting at 0) of the volume to assign. Enter the name of the datastore to assign to the volume. This is the datastore name as it appears in the Storage Path on the Edit Compute Resource page. Disk numbering must be sequential.
`VirtualMachine.DiskN.StorageReservationPolicy`	Specifies the storage reservation policy to use to find storage for disk *N*. Also assigns the named storage reservation policy to a volume. To use this property, substitute the volume number for *N* in the property name and specify a storage reservation policy name as the value. This property is equivalent to the storage reservation policy name specified on the blueprint. Disk numbering must be sequential. This property is valid for all Virtual and vCloud reservations. This property is not valid for Physical, Amazon, or OpenStack reservations.
`VirtualMachine.DiskN.Label`	Specifies the label for a machine’s disk *N*. The disk label maximum is 32 characters. Disk numbering must be sequential. When used with a guest agent, specifies the label of a machine's disk *N* inside the guest operating system.

Table 1-6. Custom Properties for Linked Clone Blueprints (continued)

Custom Property	Description
`VirtualMachine.DiskN.Letter`	Specifies the drive letter or mount point of a machine's disk *N*. The default is `C`. For example, to specify the letter D for Disk 1, define the custom property as `VirtualMachine.Disk1.Letter` and enter the value D. Disk numbering must be sequential. When used in conjunction with a guest agent, this value specifies the drive letter or mount point under which an additional disk *N* is mounted by the guest agent in the guest operating system.
`MaximumProvisionedMachines`	Specifies the maximum number of linked clones for one machine snapshot. The default is unlimited.
`Linux.ExternalScript.Name`	Specifies the name of an optional customization script, for example `config.sh`, that the Linux guest agent runs after the operating system is installed. This property is available for Linux machines cloned from templates on which the Linux agent is installed. If you specify an external script, you must also define its location by using the `Linux.ExternalScript.LocationType` and `Linux.ExternalScript.Path` properties.
`Linux.ExternalScript.LocationType`	Specifies the location type of the customization script named in the `Linux.ExternalScript.Name` property. This can be either local or nfs. You must also specify the script location using the `Linux.ExternalScript.Path` property. If the location type is nfs, also use the `Linux.ExternalScript.Server` property.
`Linux.ExternalScript.Server`	Specifies the name of the NFS server, for example lab-ad.lab.local, on which the Linux external customization script named in `Linux.ExternalScript.Name` is located.
`Linux.ExternalScript.Path`	Specifies the local path to the Linux customization script or the export path to the Linux customization on the NFS server. The value must begin with a forward slash and not include the file name, for example `/scripts/linux/config.sh`.

If you installed the guest agent to customize cloned machines, you use some custom properties more often than others.

Table 1-7. Custom Properties for Customizing Cloned Machines with a Guest Agent

Custom Property	Description
`VirtualMachine.Admin.UseGuestAgent`	If the guest agent is installed as a service on a template for cloning, set to True on the machine blueprint to enable the guest agent service on machines cloned from that template. When the machine is started, the guest agent service is started. Set to False to deactivate the guest agent. If set to False, the enhanced clone workfow will not use the guest agent for guest operating system tasks, reducing its functionality to `VMwareCloneWorkflow`. If not specified or set to anything other than False, the enhanced clone workflow sends work items to the guest agent.
`VirtualMachine.Admin.CustomizeGuestOSDelay`	Specifies the time to wait after customization is complete and before starting the guest operating system customization. The value must be in HH:MM:SS format. If the value is not set, the default value is one minute (00:01:00). If you choose not to include this custom property, provisioning can fail if the virtual machine reboots before guest agent work items are completed, causing provisioning to fail.
`VirtualMachine.Customize.WaitComplete`	Set to True to prevent the provisioning workflow from sending work items to the guest agent until all customizations arecomplete. Set to False to allow work items to be created before customization is complete.
`VirtualMachine.SoftwareN.ScriptPath`	Specifies the full path to an application's install script. The path must be a valid absolute path as seen by the guest operating system and must include the name of the script filename. You can pass custom property values as parameters to the script by inserting {*CustomPropertyName*} in the path string. For example, if you have a custom property named `ActivationKey` whose value is 1234, the script path is `D:\InstallApp.bat –key {ActivationKey}`. The guest agent runs the command `D:\InstallApp.bat –key 1234`. Your script file can then be programmed to accept and use this value.

Custom Properties for FlexClone Blueprints

vRealize Automation includes custom properties that you can use to provide additional controls for FlexClone blueprints.

Table 1-8. Custom Properties for FlexClone Blueprints

Custom Property	Description
`VirtualMachine.NetworkN.NetworkProfileName`	Specifies the name of a network profile from which to assign a static IP address to network device *N* or from which to obtain the range of static IP addresses that can be assigned to network device *N* of a cloned machine, where *N*=0 for the first device, 1 for the second, and so on. The network profile that the property points to is used to allocate an IP address. The property determines the network that the machine attaches to, based on the reservation. Changing this property value after the network is assigned has no effect on the expected IP address values for the designated machines. With WIM-based provisioning for virtual machines, you can use this property to specify a network profile and network interface or you can use the Network section of the Virtual Reservation page. The following attributes of the network profile are available to enable static IP assignment in a cloning blueprint: ■ `VirtualMachine.NetworkN.SubnetMask` ■ `VirtualMachine.NetworkN.Gateway` ■ `VirtualMachine.NetworkN.PrimaryDns` ■ `VirtualMachine.NetworkN.SecondaryDns` ■ `VirtualMachine.NetworkN.PrimaryWins` ■ `VirtualMachine.NetworkN.SecondaryWins` ■ `VirtualMachine.NetworkN.DnsSuffix` ■ `VirtualMachine.NetworkN.DnsSearchSuffixes` `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation.
`Linux.ExternalScript.Name`	Specifies the name of an optional customization script, for example `config.sh`, that the Linux guest agent runs after the operating system is installed. This property is available for Linux machines cloned from templates on which the Linux agent is installed. If you specify an external script, you must also define its location by using the `Linux.ExternalScript.LocationType` and `Linux.ExternalScript.Path` properties.
`Linux.ExternalScript.LocationType`	Specifies the location type of the customization script named in the `Linux.ExternalScript.Name` property. This can be either local or nfs. You must also specify the script location using the `Linux.ExternalScript.Path` property. If the location type is nfs, also use the `Linux.ExternalScript.Server` property.

Table 1-8. Custom Properties for FlexClone Blueprints (continued)

Custom Property	Description
`Linux.ExternalScript.Server`	Specifies the name of the NFS server, for example lab-ad.lab.local, on which the Linux external customization script named in `Linux.ExternalScript.Name` is located.
`Linux.ExternalScript.Path`	Specifies the local path to the Linux customization script or the export path to the Linux customization on the NFS server. The value must begin with a forward slash and not include the file name, for example `/scripts/linux/config.sh`.

If you installed the guest agent to customize cloned machines, the Custom Properties for Customizing FlexClone Machines with a Guest Agent table describes the most commonly used custom properties for your situation.

Table 1-9. Custom Properties for Customizing FlexClone Machines with a Guest Agent

Custom Property	Description
`VirtualMachine.Admin.UseGuestAgent`	If the guest agent is installed as a service on a template for cloning, set to True on the machine blueprint to enable the guest agent service on machines cloned from that template. When the machine is started, the guest agent service is started. Set to False to deactivate the guest agent. If set to False, the enhanced clone workfow will not use the guest agent for guest operating system tasks, reducing its functionality to `VMwareCloneWorkflow`. If not specified or set to anything other than False, the enhanced clone workflow sends work items to the guest agent.
`VirtualMachine.DiskN.Label`	Specifies the label for a machine's disk *N*. The disk label maximum is 32 characters. Disk numbering must be sequential. When used with a guest agent, specifies the label of a machine's disk *N* inside the guest operating system.
`VirtualMachine.DiskN.Letter`	Specifies the drive letter or mount point of a machine's disk *N*. The default is C. For example, to specify the letter D for Disk 1, define the custom property as `VirtualMachine.Disk1.Letter` and enter the value D. Disk numbering must be sequential. When used in conjunction with a guest agent, this value specifies the drive letter or mount point under which an additional disk *N* is mounted by the guest agent in the guest operating system.
`VirtualMachine.Admin.CustomizeGuestOSDelay`	Specifies the time to wait after customization is complete and before starting the guest operating system customization. The value must be in HH:MM:SS format. If the value is not set, the default value is one minute (00:01:00). If you choose not to include this custom property, provisioning can fail if the virtual machine reboots before guest agent work items are completed, causing provisioning to fail.

Table 1-9. Custom Properties for Customizing FlexClone Machines with a Guest Agent (continued)

Custom Property	Description
`VirtualMachine.Customize.WaitComplete`	Set to True to prevent the provisioning workflow from sending work items to the guest agent until all customizations arecomplete. Set to False to allow work items to be created before customization is complete.
`VirtualMachine.SoftwareN.ScriptPath`	Specifies the full path to an application's install script. The path must be a valid absolute path as seen by the guest operating system and must include the name of the script filename. You can pass custom property values as parameters to the script by inserting {*CustomPropertyName*} in the path string. For example, if you have a custom property named `ActivationKey` whose value is 1234, the script path is `D:\InstallApp.bat –key {ActivationKey}`. The guest agent runs the command `D:\InstallApp.bat –key 1234`. Your script file can then be programmed to accept and use this value.

Custom Properties for Basic Workflow Blueprints

vRealize Automation includes custom properties that you can use to provide additional controls for basic workflow blueprints.

Table 1-10. Custom Properties for Basic Workflow Blueprints

Custom Property	Description
`VirtualMachine.CDROM.Attach`	Set to False to provision the machine without a CD-ROM device. The default is True.
`VirtualMachine.Admin.ThinProvision`	Determines whether thin provisioning is used on ESX compute resources. Disk provisioning is abstracted from the underlying storage. Set to True to use thin provisioning. Set to False to use standard provisioning. This property is for virtual provisioning.
`VirtualMachine.DiskN.StorageReservationPolicy`	Specifies the storage reservation policy to use to find storage for disk *N*. Also assigns the named storage reservation policy to a volume. To use this property, substitute the volume number for *N* in the property name and specify a storage reservation policy name as the value. This property is equivalent to the storage reservation policy name specified on the blueprint. Disk numbering must be sequential. This property is valid for all Virtual and vCloud reservations. This property is not valid for Physical, Amazon, or OpenStack reservations.

Table 1-10. Custom Properties for Basic Workflow Blueprints (continued)

Custom Property	Description
`VirtualMachine.Storage.AllocationType`	Stores collected groups to a single datastore. A distributed environment stores disks round-robin style. Specify one of the following values: ■ Collected Keep all disks together. ■ Distributed Allow disks to be placed on any datastore or datastore cluster that is available in the reservation. For an example of how to use the `VirtualMachine.Storage.AllocationType` property to create datastore clusters, see the Keeping Multiple Disks Together blog post.
`VirtualMachine.Storage.Name`	Identifies the storage path on which the machine resides. The default is the value specified in the reservation that was used to provision the machine.
`VirtualMachine.Storage.ReserveMemory`	Set to True to manage vSwap storage allocation to ensure availability and set allocation in the reservation. vSwap allocation is considered when you create or reconfigure a virtual machine. vSwap allocation checking is only available for vSphere endpoints. **Note** If you do not specify the `VirtualMachine.Storage.ReserveMemory` custom property when you create or provision the machine from vRealize Automation, swap space availability is not ensured. If you add the property for an already provisioned machine, and the allocated reservation is full, the storage allocated in the reservation might exceed the actual allocated storage.
`VMware.Hardware.Version`	Specifies the VM hardware version to be used for vSphere settings. Supported values are currently vmx-04, vmx-07, vmx-08, vmx-09 and vmx-10. This property is applicable for VM Create and VM Update workflows and is available only for basic workflow blueprints.

Custom Properties for Linux Kickstart Blueprints

vRealize Automation includes custom properties that you can use to provide additional controls for Linux Kickstart blueprints.

Certain vRealize Automation custom properties are required to use with Linux Kickstart blueprints.

Table 1-11. Required Custom Properties for Linux Kickstart Blueprints

Custom Property	Description
`VMware.VirtualCenter.OperatingSystem`	Specifies the vCenter Server guest operating system version (`VirtualMachineGuestOsIdentifier`) with which vCenter Server creates the machine. This operating system version must match the operating system version to be installed on the provisioned machine. Administrators can create property groups using one of several property sets, for example, `VMware[OS_Version]Properties`, that are predefined to include the correct `VMware.VirtualCenter.OperatingSystem` values. This property is for virtual provisioning. For related information, see the enumeration type `VirtualMachineGuestOsIdentifier` in vSphere API/SDK Documentation. For a list of currently accepted values, see the vCenter Server documentation.
`Image.ISO.Location`	Values for this property are case sensitive. Specifies the location of the ISO image from which to boot. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning.
`Image.ISO.Name`	Values for this property are case sensitive. Specifies the name of the ISO image from which to boot, for example /ISO/Microsoft/WinPE.iso. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning.
`Image.ISO.UserName`	Specifies the user name to access the CIFS share in the format *username@domain*. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.
`Image.ISO.Password`	Specifies the password associated with the `Image.ISO.UserName` property. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.

Optional custom properties are available.

Table 1-12. Optional Custom Properties for Linux Kickstart Blueprints

Custom Property	Description
`VirtualMachine.Admin.ThinProvision`	Determines whether thin provisioning is used on ESX compute resources. Disk provisioning is abstracted from the underlying storage. Set to True to use thin provisioning. Set to False to use standard provisioning. This property is for virtual provisioning.
`Machine.SSH`	Set to True to enable the **Connect Using SSH** option on the vRealize Automation Items page for Linux machines provisioned from this blueprint. If set to True and the **Connect using RDP or SSH** machine operation is enabled in the blueprint, all Linux machines that are provisioned from the blueprint display the **Connect Using SSH** option to entitled users. The **Connect Using SSH** option requires that your browser has a plug-in that supports SSH, for example the FireSSH SSH terminal client for Mozilla Firefox and Google Chrome. When the plug-in is present, selecting **Connect Using SSH** displays an SSH console and prompts for your administrator credentials.

Custom Properties for SCCM Blueprints

vRealize Automation includes custom properties that you can use to provide additional controls for SCCM blueprints.

Certain custom properties are required to use with SCCM blueprints.

Table 1-13. Required Custom Properties for SCCM Blueprints

Custom Property	Description
`Image.ISO.Location`	Values for this property are case sensitive. Specifies the location of the ISO image from which to boot. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning.
`Image.ISO.Name`	Values for this property are case sensitive. Specifies the name of the ISO image from which to boot, for example /ISO/Microsoft/WinPE.iso. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning.
`Image.ISO.UserName`	Specifies the user name to access the CIFS share in the format *username@domain*. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.

Table 1-13. Required Custom Properties for SCCM Blueprints (continued)

Custom Property	Description
`Image.ISO.Password`	Specifies the password associated with the `Image.ISO.UserName` property. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.
`SCCM.Collection.Name`	Specifies the name of the SCCM collection that contains the operating system deployment task sequence.
`SCCM.Server.Name`	Specifies the fully qualified domain name of the SCCM server on which the collection resides, for example lab-sccm.lab.local.
`SCCM.Server.SiteCode`	Specifies the site code of the SCCM server.
`SCCM.Server.UserName`	Specifies a user name with administrator-level access to the SCCM server.
`SCCM.Server.Password`	Specifies the password associated with the `SCCM.Server.UserName` property.

Certain custom properties are used most often with SCCM blueprints.

Table 1-14. Common Custom Properties for SCCM Blueprints

Custom Property	Description
`SCCM.CustomVariable.`*Name*	Specifies the value of a custom variable, where *Name* is the name of any custom variable to be made available to the SCCM task sequence after the provisioned machine is registered with the SCCM collection. The value is determined by your choice of custom variable. If your integration requires it, you can use `SCCM.RemoveCustomVariablePrefix` to remove the `SCCM.CustomVariable.` prefix from your custom variable.
`SCCM.RemoveCustomVariablePrefix`	Set to *true* to remove the prefix `SCCM.CustomVariable.` from SCCM custom variables you created by using the custom property `SCCM.CustomVariable.`*Name*.

Custom Properties for WIM Blueprints

vRealize Automation includes custom properties that provide additional controls for WIM blueprints.

Certain vRealize Automation custom properties are required for WIM blueprints.

Table 1-15. Required Custom Properties for WIM Blueprints

Custom Property	Description
`Image.ISO.Location`	Values for this property are case sensitive. Specifies the location of the ISO image from which to boot. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning.
`Image.ISO.Name`	Values for this property are case sensitive. Specifies the name of the ISO image from which to boot, for example /ISO/Microsoft/WinPE.iso. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning.
`Image.ISO.UserName`	Specifies the user name to access the CIFS share in the format *username@domain*. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.
`Image.ISO.Password`	Specifies the password associated with the `Image.ISO.UserName` property. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.
`Image.Network.Letter`	Specifies the drive letter to which the WIM image path is mapped on the provisioned machine. The default value is K.
`Image.WIM.Path`	Specifies the UNC path to the WIM file from which an image is extracted during WIM-based provisioning. The path format is \\*server**share$* format, for example `\\lab-ad\dfs$`.
`Image.WIM.Name`	Specifies the name of the WIM file as located by the `Image.WIM.Path` property.
`Image.WIM.Index`	Specifies the index used to extract the correct image from the WIM file.
`Image.Network.User`	Specifies the user name with which to map the WIM image path (`Image.WIM.Path`) to a network drive on the provisioned machine. This is typically a domain account with access to the network share.
`Image.Network.Password`	Specifies the password associated with the `Image.Network.User` property.

Table 1-15. Required Custom Properties for WIM Blueprints (continued)

Custom Property	Description
`VirtualMachine.Admin.Owner`	Specifies the user name of the machine owner.
`VMware.VirtualCenter.OperatingSystem`	Specifies the vCenter Server guest operating system version (`VirtualMachineGuestOsIdentifier`) with which vCenter Server creates the machine. This operating system version must match the operating system version to be installed on the provisioned machine. Administrators can create property groups using one of several property sets, for example, `VMware[OS_Version]Properties`, that are predefined to include the correct `VMware.VirtualCenter.OperatingSystem` values. This property is for virtual provisioning. For related information, see the enumeration type `VirtualMachineGuestOsIdentifier` in vSphere API/SDK Documentation. For a list of currently accepted values, see the vCenter Server documentation.

Optional custom properties are also available for WIM blueprints.

Table 1-16. Common Custom Properties for WIM Blueprints

Custom Property	**Description**
`SysPrep.`*`Section.Key`* - `SysPrep.GuiUnattended.AdminPassword` - `SysPrep.GuiUnattended.EncryptedAdminPassword` - `SysPrep.GuiUnattended.TimeZone`	Specifies information to be added to the SysPrep answer file on machines during the WinPE stage of provisioning. Information that already exists in the SysPrep answer file is overwritten by these custom properties. *Section* represents the name of the section of the SysPrep answer file, for example GuiUnattended or UserData. *Key* represents a key name in the section. For example, to set the time zone of a provisioned machine to West Pacific Standard Time, define the custom property `GuiUnattended.UserData.TimeZone` and set the value to 275. For a full list of sections, keys, and accepted values, see the System Preparation Utility for Windows documentation. The following *Section.Key* combinations can be specified for WIM-based provisioning: - GuiUnattended - AdminPassword - EncryptedAdminPassword - TimeZone - UserData - ProductKey - FullName - ComputerName - OrgName - Identification - DomainAdmin - DomainAdminPassword - JoinDomain - JoinWorkgroup
`Sysprep.Identification.DomainAdmin`	Specifies a user name with administrator-level access to the target domain in Active Directory. Do not include the user domain in the credentials that you send to vCloud Director or vCloud Air.
`Sysprep.Identification.DomainAdminPassword`	Specifies the password to associate with the `Sysprep.Identification.DomainAdmin` property.
`Sysprep.Identification.JoinDomain`	Specifies the name of the domain to join in Active Directory.
`Sysprep.Identification.JoinWorkgroup`	Specifies the name of the workgroup to join if not using a domain.
`SysPrep.UserData.ComputerName`	Specifies a machine name, for example lab-client005.
`SysPrep.UserData.FullName`	Specifies the full name of a user.
`SysPrep.UserData.OrgName`	Specifies the organization name of the user.

Table 1-16. Common Custom Properties for WIM Blueprints (continued)

Custom Property	Description
`SysPrep.UserData.ProductKey`	Specifies the Windows product key.
`VirtualMachine.Admin.ThinProvision`	Determines whether thin provisioning is used on ESX compute resources. Disk provisioning is abstracted from the underlying storage. Set to True to use thin provisioning. Set to False to use standard provisioning. This property is for virtual provisioning.

Custom Properties for vCloud Air and vCloud Director Blueprints

You can add certain custom properties to a vCloud Air or vCloud Director machine component definition in a blueprint.

For vSphere machine components with associated NSX, use network, security, and load balancing setting in the user interface. For machine components that do not have a **Network** or **Security** tab, you can add network and security custom properties, such as `VirtualMachine.Network0.Name`, to their **Properties** tab in the design canvas. NSX network, security, and load balancer properties are only applicable to vSphere machines.

Table 1-17. Custom Properties for vCloud Air and vCloud Director Machine Components in the Design Canvas

Custom Property	Description
`Sysprep.Identification.DomainAdmin`	Specifies a user name with administrator-level access to the target domain in Active Directory. Do not include the user domain in the credentials that you send to vCloud Director or vCloud Air.
`Sysprep.Identification.DomainAdminPassword`	Specifies the password to associate with the `Sysprep.Identification.DomainAdmin` property.
`Sysprep.Identification.JoinDomain`	Specifies the name of the domain to join in Active Directory.
`VirtualMachine.DiskN.IsFixed`	Deactivates the editing of a specific disk when reconfiguring a machine. Set to True to deactivate display of the edit capacity option for a specific volume. The True value is case-sensitive. The *N* value is the 0-based index of the disk. Alternatively, you can set the `VirtualMachine.Disk`*N*`.IsFixed` custom property to True in the VirtualMachineProperties table in the database or use the Repository API to specify a URI value such as `.../Repository/Data/ManagementModelEntities.svc/VirtualMachines(guid'60D93A8A-F541-4CE0-A6C6-78973AC0F1D2')/VirtualMachineProperties`.

Table 1-17. Custom Properties for vCloud Air and vCloud Director Machine Components in the Design Canvas (continued)

Custom Property	Description
`VirtualMachine.DiskN.StorageReservationPolicy`	Specifies the storage reservation policy to use to find storage for disk *N*. Also assigns the named storage reservation policy to a volume. To use this property, substitute the volume number for *N* in the property name and specify a storage reservation policy name as the value. This property is equivalent to the storage reservation policy name specified on the blueprint. Disk numbering must be sequential. This property is valid for all Virtual and vCloud reservations. This property is not valid for Physical, Amazon, or OpenStack reservations.
`VirtualMachine.EULA.AcceptAll`	Set to true to specify that all the EULAs for the VM templates of the vCloud Air or vCloud Director endpoints are accepted during provisioning.
`VirtualMachine.NetworkN.Name`	Specifies the name of the network to connect to, for example the network device *N* to which a machine is attached. This is equivalent to a network interface card (NIC). By default, a network is assigned from the network paths available on the reservation on which the machine is provisioned. Also see `VirtualMachine.NetworkN.AddressType`. You can ensure that a network device is connected to a specific network by setting the value of this property to the name of a network on an available reservation. For example, if you give properties for N= 0 and 1, you get 2 NICs and their assigned value, provided the network is selected in the associated reservation. `VirtualMachine.Network`*N* custom properties are specific to blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks. For an example of how to use this custom property to dynamically set `VirtualMachine.Network0.Name` based on a consumer's selection from a list of predefined available networks, see the Adding a Network Selection Drop-Down in vRA 7 blog post. For related information, see Custom Properties for Networking and Security.

Table 1-17. Custom Properties for vCloud Air and vCloud Director Machine Components in the Design Canvas (continued)

Custom Property	Description
`VirtualMachine.NetworkN.AddressType`	Specifies how IP address allocation is supplied to the network provider, where Network*N* is the network number, starting with 0. The following values are available: - DHCP - Static - MANUAL (available for vCloud Air and vCloud Director only) This property is available for configuring vCloud Air, vCloud Director, and vSphere machine components in the blueprint. Also see `VirtualMachine.NetworkN.Name`. This property is not supported for on-demand NAT or on-demand routed networks.
`VirtualMachine.Reconfigure.DisableHotCpu`	Set to true to specify that the reconfigure machine action restarts the specified machine. By default, the reconfigure machine action does not restart the machine. Performing a hot add of CPU, memory, or storage causes the reconfigure machine action to fail and not to restart the machine unless the `Hot Add` setting is enabled in vSphere for the machine or template. You can add `VirtualMachine.Reconfigure.DisableHotCpu=true` to a machine component in a vRealize Automation blueprint to deactivate the `Hot Add` setting and force the machine to restart regardless of the vSphere `Hot Add` setting. The custom property is only available for machine types that support hardware reconfiguration, which are vSphere, vCloud Air, and vCloud Director.
`VCloud.Lease.Sync.TimeBufferMins`	Specifies a threshold integer value for a compute resource such that lease synchronization between vCloud Director and vRealize Automation only occur for vCloud Director or vCloud Air-provisioned machines that are set to expire in vCloud Director or vCloud Air in that time period. If a conflict is found, the lease value is synchronized to match the lease length defined in vRealize Automation. The default `VCloud.Lease.Sync.TimeBufferMins` value is 720 minutes, which is 12 hours. If `VCloud.Lease.Sync.TimeBufferMins` is not present, the default value is used. For example, if the default values are used, vRealize Automation runs the lease synchronization check workflow every 45 minutes, which is the workflow default, and only the leases of machines that are set to expire within 12 hours are changed to match the lease length defined in vRealize Automation.
`VCloud.Owner.UseEndpointAccount`	Set to true to assign the endpoint account as the vCloud Air or vCloud Director machine owner for provisioning and import operations. For change ownership operations, the owner is not changed on the endpoint. If not specified or set to false, the vRealize Automation owner is the machine owner.

Table 1-17. Custom Properties for vCloud Air and vCloud Director Machine Components in the Design Canvas (continued)

Custom Property	Description
`VCloud.Template.MakeIdenticalCopy`	Set to true to clone an identical copy of the vCloud Air or vCloud Director template for machine provisioning. The machine is provisioned as an identical copy of the template. Settings specified in the template, including storage path, supersede settings specified in the blueprint. The only changes from the template are the names of the cloned machines, which are generated from the machine prefix specified in the blueprint. vCloud Air or vCloud Director machines that are provisioned as identical copies can use networks and storage profiles that are not available in the vRealize Automation reservation. To avoid having unaccounted reservation allocations, verify that the storage profile or network specified in the template is available in the reservation.
`VMware.SCSI.Sharing`	Specifies the sharing mode of the machine's VMware SCSI bus. Possible values are based on the `VirtualSCSISharing` ENUM value and include noSharing, physicalSharing, and virtualSharing. The `VMware.SCSI.Sharing` property is not available for use with the CloneWorkflow provisioning workflow. If you specify the CloneWorkflow provisioning workflow when configuring your machine component in the blueprint design canvas, you cannot use the `VMware.SCSI.Sharing` property.
`VMware.SCSI.Type`	For vCloud Air, vCloud Director, or vSphere machine components in blueprints, specifies the SCSI machine type using one of the following case-sensitive values: ■ buslogic Use BusLogic emulation for the virtual disk. ■ lsilogic Use LSILogic emulation for the virtual disk (default). ■ lsilogicsas Use LSILogic SAS 1068 emulation for the virtual disk. ■ pvscsi Use para-virtualization emulation for the virtual disk. ■ none Use if a SCSI controller does not exist for this machine. The `VMware.SCSI.Type` property is not available for use with the CloneWorkflow provisioning workflow. If you specify the CloneWorkflow provisioning workflow when configuring your machine component in the blueprint design canvas, you cannot use the `VMware.SCSI.Type` property.

Custom Properties for Networking and Security

The vRealize Automation custom properties for networking specify configuration for a specific network device on a machine.

For vSphere machine components with associated NSX, use network, security, and load balancing setting in the user interface. For machine components that do not have a **Network** or **Security** tab, you can add network and security custom properties, such as `VirtualMachine.Network0.Name`, to their **Properties** tab in the design canvas. NSX network, security, and load balancer properties are only applicable to vSphere machines.

Network assignments are performed during machine allocation. vRealize Automation retrieves network information from the blueprint. If you want to assign more than one network, use the `VirtualMachine.NetworkN.Name` custom property on your machine blueprint. If you do not provide custom properties, allocation only assigns one network which is picked using a round robin method in conjunction with the selected reservation.

Note With the exception of the following properties, the properties in the table do not apply to Amazon Web Services:

- `agent.download.url`
- `software.agent.service.url`
- `software.ebs.url`

Note Network-specific custom properties that point to on-demand networks are not supported. For example, you cannot use network custom properties, such as `VirtualMachine.Network0.NetworkProfileName`, for on-demand NAT and on-demand routed network components.

By default, a machine has one network device configured with the `VirtualMachine.Network0.Name` property. You can configure additional network devices by using the `VirtualMachine.NetworkN.Name` custom property, where *N* is the network number.

The numbering of network properties must be sequential, starting with 0. For example, if you specify custom properties for only `VirtualMachine.Network0` and `VirtualMachine.Network2`, the properties for `VirtualMachine.Network2` are ignored, because the preceding network, `VirtualMachine.Network1`, was not specified.

Although general support for vCloud Networking and Security has ended, the VCNS custom properties continue to be valid for NSX purposes. See the Knowledge Base article 2144733.

Table 1-18. Custom Properties for Networking Configuration

Custom Property	Description
`agent.download.url`	When using port forwarding, specifies the private IP address of your Amazon AWS tunnel machine and port for your software agent file, for example **`https://`*`Private_IP:1443`*`/software-service/resources/nobel-agent.jar`**. Specifies the URL for the VPN agent on your deployment. The URL format is `https://` *`Private_IP`*`:1443/software-service//resources/noble-agent.jar` You can add this property, in combination with `software.agent.service.url` and `software.ebs.url`, to a reservation or the compute resource endpoint. You can also use this property to specify a private address and port when using PAT or NAT translation and port forwarding.
`NSX.Edge.ApplianceSize`	Specifies the allowed NSX edge appliance size types for the provisioned machine or deployment. The options are: - `compact` For small deployments, POCs, and single service use. - CPU = 1 - RAM = 512 MB - Disk = 512 MB - `large` For small to medium or multi-tenant deployments. - CPU = 2 - RAM = 1 GB - Disk = 512 MB - `quadlarge` For high throughput equal-cost multi-path routing (ECMP) or high performance firewall deployments. - CPU = 4 - RAM = 1 GB - Disk = 512 MB - `xlarge` For L7 load balancing and dedicated core deployments. - CPU = 6 - RAM = 8 GB - Disk = 4.5GB (4GB Swap) For related information, see System Requirements for NSX.

Table 1-18. Custom Properties for Networking Configuration (continued)

Custom Property	Description
`NSX.Edge.HighAvailability`	When set to true (`NSX.Edge.HighAvailability`=true), enables high availability (HA) mode on the NSX edge machine that is deployed from the blueprint. When used with `NSX.Edge.HighAvailability.PortGroup`=*port_group_name*, this property allows you to configure an NSX edge during blueprint authoring. You can add this property to an NSX load balancer component in the vRealize Automation blueprint or to the vRealize Automation blueprint itself. Must be used in conjunction with `NSX.Edge.HighAvailability.PortGroup`= *port_group_name*.
`NSX.Edge.HighAvailability.PortGroup`	Creates an internal interface or internal vNIC attached to the specified port group name, for example `NSX.Edge.HighAvailability.PortGroup`=`VM Network` where `VM Network` is an HA (high availability) distributed (vLAN-backed) or NSX logical switch port group. NSX HA mode requires at least one internal network interface, or vNIC. When used with `NSX.Edge.HighAvailability`=true, this property allows you to configure high availability (HA) an NSX edge during blueprint authoring. When using one arm load balancer with HA enabled, you must specify a separate port group for the HA. **Note** The specified port group network cannot be a member of the reservation pool, as the property's use of the port group conflicts with the normal deployment's use of the port group, resulting in the following error: `Portgroup must be unique within an Edge...` Must be used in conjunction with `NSX.Edge.HighAvailability`=true.

Table 1-18. Custom Properties for Networking Configuration (continued)

Custom Property	Description
`NSX.Validation.Disable.Single.Edge.Uplink`	When set to true, the NSX validation that checks for the following conditions is disabled: - All on-demand NAT networks on the blueprint source the same external network. - All on-demand routed networks on the blueprint that use the load balancer VIP source the same external network. - All on-demand load balancer components on the blueprint have VIPs on the same external network or on-demand networks backed by the same external network. Disabling this validation check can result in a deployment that succeeds but in which some network components might be inaccessible. If not present or if set to false, the validation check is enabled (default). A single NSX edge can only support one external network as its uplink network. Multiple IPs from the same external network are supported. While a blueprint can contain any number of external or on-demand network components, NSX only supports one external network as the uplink network. This property can only be specified at the blueprint level. It cannot be specified on a component in the blueprint canvas.
`NSX.Validation.Disable.Blueprint.NSXT`	When set to true, all NSX-T validation is disabled for the blueprint **Finish** action. If not present or if set to false, the NSX-T validation check is enabled (default). For example, if you have overlapping subnets in the blueprint, an error message appears when you click **Finish** in the blueprint and the overlap prevents you from finishing the blueprint, although you can save it. If you want to finish the blueprint, you can add `NSX.Validation.Disable.Blueprint.NSXT` by using the **Blueprint Properties** page and then finish the blueprint. The property only disables NSX-T validations for the blueprint **Finish** action.
`software.agent.service.url`	When using port forwarding, specifies the private IP address of your Amazon AWS tunnel machine and port for the vRealize Automation software service API, for example **`https://`*`Private_IP:1443`*`/software-service/api`**. You can add this property, in combination with `software.ebs.url` and `agent.download.url`, to a reservation or the compute resource endpoint. You can also use this property to specify a private address and port when using PAT or NAT and port forwarding.

Table 1-18. Custom Properties for Networking Configuration (continued)

Custom Property	Description
`software.ebs.url`	When using port forwarding, specifies the private IP address of your Amazon AWS tunnel machine and port for the vRealize Automation event broker service, for example **`https://`*`Private_IP:1443`*`/event-broker-service/api`**. You can add this property, in combination with `software.agent.service.url` and `agent.download.url`, to a reservation or the compute resource endpoint. You can also use this property to specify a private address and port when using PAT or NAT and port forwarding.
`VirtualMachine.NetworkN.Address`	Specifies the IP address of network device *N* in a machine provisioned with a static IP address. For Amazon, see `Amazon.elasticIpAddress.ipAddress` .
`VirtualMachine.NetworkN.MacAddressType`	Indicates whether the MAC address of network device *N* is generated or user-defined (static). This property is available for cloning. The default value is generated. If the value is static, you must also use `VirtualMachine.NetworkN.MacAddress` to specify the MAC address. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks.
`VirtualMachine.NetworkN.MacAddress`	Specifies the MAC address of a network device *N*. This property is available for cloning. If the value of `VirtualMachine.NetworkN.MacAddressType` is generated, this property contains the generated address. If the value of `VirtualMachine.NetworkN.MacAddressType` is static, this property specifies the MAC address. For virtual machines provisioned on ESX server hosts, the address must be in the range specified by VMware. For details, see vSphere documentation. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks.

Table 1-18. Custom Properties for Networking Configuration (continued)

Custom Property	Description
`VirtualMachine.NetworkN.Name`	Specifies the name of the network to connect to, for example the network device *N* to which a machine is attached. This is equivalent to a network interface card (NIC). By default, a network is assigned from the network paths available on the reservation on which the machine is provisioned. Also see `VirtualMachine.NetworkN.AddressType`. You can ensure that a network device is connected to a specific network by setting the value of this property to the name of a network on an available reservation. For example, if you give properties for N= 0 and 1, you get 2 NICs and their assigned value, provided the network is selected in the associated reservation. `VirtualMachine.Network`*N* custom properties are specific to blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks. For an example of how to use this custom property to dynamically set `VirtualMachine.Network0.Name` based on a consumer's selection from a list of predefined available networks, see the Adding a Network Selection Drop-Down in vRA 7 blog post.
`VirtualMachine.NetworkN.PortID`	Specifies the port ID to use for network device *N* when using a dvPort group with a vSphere distributed switch. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks.

Table 1-18. Custom Properties for Networking Configuration (continued)

Custom Property	Description
`VirtualMachine.NetworkN.NetworkProfileName`	Specifies the name of a network profile from which to assign a static IP address to network device *N* or from which to obtain the range of static IP addresses that can be assigned to network device *N* of a cloned machine, where *N*=0 for the first device, 1 for the second, and so on. The network profile that the property points to is used to allocate an IP address. The property determines the network that the machine attaches to, based on the reservation. Changing this property value after the network is assigned has no effect on the expected IP address values for the designated machines. With WIM-based provisioning for virtual machines, you can use this property to specify a network profile and network interface or you can use the Network section of the Virtual Reservation page. The following attributes of the network profile are available to enable static IP assignment in a cloning blueprint: ■ `VirtualMachine.NetworkN.SubnetMask` ■ `VirtualMachine.NetworkN.Gateway` ■ `VirtualMachine.NetworkN.PrimaryDns` ■ `VirtualMachine.NetworkN.SecondaryDns` ■ `VirtualMachine.NetworkN.PrimaryWins` ■ `VirtualMachine.NetworkN.SecondaryWins` ■ `VirtualMachine.NetworkN.DnsSuffix` ■ `VirtualMachine.NetworkN.DnsSearchSuffixes` `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. You cannot use this custom property to define an on-demand NAT or on-demand routed network profile name. Because on-demand network profile names are generated at allocation time (during provisioning), their names are unknown when creating or editing the blueprint. To specify NSX on-demand network information, use the applicable network component in the blueprint design canvas for your vSphere machine components.

Table 1-18. Custom Properties for Networking Configuration (continued)

Custom Property	Description
■ `VirtualMachine.NetworkN.SubnetMask` ■ `VirtualMachine.NetworkN.Gateway` ■ `VirtualMachine.NetworkN.PrimaryDns` ■ `VirtualMachine.NetworkN.SecondaryDns` ■ `VirtualMachine.NetworkN.PrimaryWins` ■ `VirtualMachine.NetworkN.SecondaryWins` ■ `VirtualMachine.NetworkN.DnsSuffix` ■ `VirtualMachine.NetworkN.DnsSearchSuffixes`	Configures attributes of the network profile specified in `VirtualMachine.NetworkN.NetworkProfileName`. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation.
`VCNS.LoadBalancerEdgePool.Names.`*name*	Specifies the NSX load balancing pools to which the virtual machine is assigned during provisioning. The virtual machine is assigned to all service ports of all specified pools. The value is an *edge/pool* name or a list of *edge/pool* names separated by commas. Names are case-sensitive. **Note** You can add a machine IP address to an existing load balancer by using the `VCNS.LoadBalancerEdgePool.Names` custom property. vRealize Automation and NSX use the first member of the specified edge load balancer pool to determine the new member port and monitor port settings. However, NSX 6.2 does not require that the member port setting be specified. To avoid provisioning failure when using `VCNS.LoadBalancerEdgePool.Names` with NSX 6.2 to add a machine to an existing pool, you must specify a port value for the first member of the load balancer pool in NSX. Appending a name allows you to create multiple versions of a custom property. For example, the following properties might list load balancing pools set up for general use and machines with high, moderate, and low performance requirements: ■ `VCNS.LoadBalancerEdgePool.Names` ■ `VCNS.LoadBalancerEdgePool.Names.moderate` ■ `VCNS.LoadBalancerEdgePool.Names.high` ■ `VCNS.LoadBalancerEdgePool.Names.low`
`VCNS.SecurityGroup.Names.`*name*	Specifies the NSX security group or groups to which the virtual machine is assigned during provisioning. The value is a security group name or a list of names separated by commas. Names are case-sensitive. Appending a name allows you to create multiple versions of the property, which can be used separately or in combination. For example, the following properties can list security groups intended for general use, for the sales force, and for support: ■ `VCNS.SecurityGroup.Names` ■ `VCNS.SecurityGroup.Names.sales` ■ `VCNS.SecurityGroup.Names.support`

Table 1-18. Custom Properties for Networking Configuration (continued)

Custom Property	Description
`VCNS.SecurityTag.Names.`*name*	Specifies the NSX security tag or tags to which the virtual machine is associated during provisioning. The value is a security tag name or a list of names separated by commas. Names are case-sensitive. Appending a name allows you to create multiple versions of the property, which can be used separately or in combination. For example, the following properties can list security tags intended for general use, for the sales force, and for support: ■ `VCNS.SecurityTag.Names` ■ `VCNS.SecurityTag.Names.sales` ■ `VCNS.SecurityTag.Names.support`
`VMware.Endpoint.NSX.HideDiscoveredSecurityObjects`	Set to true to hide newly discovered security objects in the active tenant for the NSX endpoints to which the security objects are associated. Otherwise, all new security objects are available to all tenants after data collection, provided that the security object is for an endpoint in which you have a reservation. This option allows you to prevent users from accessing security objects when you want to assign those objects to a single tenant or to mask from all tenants. Set to false to toggle back to global, which enables all new security objects to be available to all tenants after data collection, provided that the security object is for an endpoint in which you have a reservation. To take effect, the fabric administrator adds the `VMware.Endpoint.NSX.HideDiscoveredSecurityObjects` custom property to the associated NSX endpoint that is associated to a vSphere endpoint. The setting applies to the next inventory data collection. Existing security objects remain unchanged. To change the tenancy setting of a security object that has already been data-collected, such as existing security objects after upgrading to the current vRealize Automation release, you can edit the security object's Tenant ID setting programmatically by using the vRealize Automation REST API or vRealize CloudClient. The available Tenant ID settings for the NSX endpoint are as follows: ■ "`<global>`" - the security object is available to all tenants. This is the default setting for existing security objects after upgrade to this release and for all new security objects that you create. ■ "`<unscoped>`" - the security object is not available to any tenants. Only the system administrator can access the security object. This is an ideal setting when defining security objects that are to eventually be assigned to a specific tenant. ■ "*`tenant_id_name`*" - the security object is only available to a single, named tenant.

Custom Properties and Property Groups for Containers

You can add predefined property groups to a containers component in a vRealize Automation blueprint. When machines are provisioned by using a blueprint that contain these properties, the provisioned machine is registered as a Docker Container host machine.

Containers for vRealize Automation supplied the following two property groups of container-specific custom properties. When you add a container component to a blueprint you can add these property groups to the container to register provisioned machines as container hosts.

- Container host properties with certificate authentication
- Container host properties with user/password authentication

These property groups are visible in vRealize Automation when you select **Administration > Property Dictionary > Property Groups**.

Because property groups are shared by all tenants, if you are working in a multi-tenant environment, consider cloning and customizing your properties. By uniquely naming property groups and properties in the groups, you can edit them to define custom values for use in a specific tenant.

The most commonly used properties are `Container.Auth.PublicKey` and `Container.Auth.PrivateKey` in which the container administrator provides the client certificate for authenticating with the container host.

Table 1-19. Containers Custom Properties

Property	Description
`containers.ipam.driver`	For use with containers only. Specifies the IPAM driver to be used when adding a Containers network component to a blueprint. The supported values depend on the drivers installed in the container host environment in which they are used. For example, a supported value might be `infoblox` or `calico` depending on the IPAM plug-ins that are installed on the container host.
`containers.network.driver`	For use with containers only. Specifies the network driver to be used when adding a Containers network component to a blueprint. The supported values depend on the drivers installed in the container host environment in which they are used. By default, Docker-supplied network drivers include bridge, overlay, and macvlan, while Virtual Container Host (VCH)-supplied network drivers include the bridge driver. Third-party network drivers such as `weave` and `calico` might also be available, depending on what network plug-ins are installed on the container host.
`Container`	For use with containers only. The default value is `App.Docker` and is required. Do not modify this property.
`Container.Auth.User`	For use with containers only. Specifies the user name for connecting to the Containers host.

Table 1-19. Containers Custom Properties (continued)

Property	Description
`Container.Auth.Password`	For use with containers only. Specifies either the password for the user name or the public or private key password to be used. Encrypted property value is supported.
`Container.Auth.PublicKey`	For use with containers only. Specifies the public key for connecting to the Containers host.
`Container.Auth.PrivateKey`	For use with containers only. Specifies private key for connecting to the Containers host. Encrypted property value is supported.
`Container.Connection.Protocol`	For use with containers only. Specifies the communication protocol. The default value is `API` and is required. Do not modify this property.
`Container.Connection.Scheme`	For use with containers only. Specifies the communication scheme. The default is `https`.
`Container.Connection.Port`	For use with containers only. Specifies the Containers connection port. The default is 2376.
`Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.MachineActivated`	For use with containers only. Specifies the event broker property to expose all Containers properties and is used for registering a provisioned host. The default value is `Container*` and is required. Do not modify this property.
`Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.Disposing`	For use with containers only. Specifies the event broker property to expose all Containersproperties above and is used for unregistering a provisioned host. The default value is `Container*` and is required. Do not modify this property.

Custom Properties for PXE Provisioning

PXE is the only provisioning method supported for Cisco UCS Manager. You can use the network bootstrap program with vRealize Automation custom properties to initiate WIM, SCCM, or Linux Kickstart provisioning. You can also use custom properties to call your own PowerShell scripts. Linux Kickstart provisioning does not require custom properties.

Custom Properties for Provisioning With PowerShell Scripts

You can use these properties for calling PowerShell scripts.

Table 1-20. Custom Properties for Calling PowerShell Scripts

Custom Property	Description
`Pxe.Setup.ScriptName`	Specifies a custom EPI PowerShell script to run on the machine before it is started by using the PXE network boot program. The value is the name assigned to the script when it is uploaded to the model manager, for example `setup.ps1`.
`Pxe.Clean.ScriptName`	Specifies the name of an EPI PowerShell script installed in the vRealize Automation Model Manager, to run on the machine after it is provisioned. The value is the name assigned to the script when it is uploaded to the Model Manager, for example `clean.ps1`.

Custom Properties For PXE and SCCM Provisioning

You can use these properties for PXE and SCCM provisioning.

Table 1-21. Custom Properties for PXE and SCCM Provisioning

Custom Property	Description
`SCCM.Collection.Name`	Specifies the name of the SCCM collection that contains the operating system deployment task sequence.
`SCCM.Server.Name`	Specifies the fully qualified domain name of the SCCM server on which the collection resides, for example lab-sccm.lab.local.
`SCCM.Server.SiteCode`	Specifies the site code of the SCCM server.
`SCCM.Server.UserName`	Specifies a user name with administrator-level access to the SCCM server.
`SCCM.Server.Password`	Specifies the password associated with the `SCCM.Server.UserName` property.
`SCCM.CustomVariable.`	Specifies the value of a custom variable, where *Name* is the name of any custom variable to be made available to the SCCM task sequence after the provisioned machine is registered with the SCCM collection. The value is determined by your choice of custom variable. If your integration requires it, you can use `SCCM.RemoveCustomVariablePrefix` to remove the `SCCM.CustomVariable.` prefix from your custom variable.

Custom Properties For PXE and WIM Provisioning

You can use these properties for PXE and WIM provisioning.

Table 1-22. Custom Properties for PXE and WIM Provisioning

Custom Property	**Description**
`Image.Network.Letter`	Specifies the drive letter to which the WIM image path is mapped on the provisioned machine. The default value is K.
`Image.WIM.Path`	Specifies the UNC path to the WIM file from which an image is extracted during WIM-based provisioning. The path format is \\server\share$ format, for example `\\lab-ad\dfs$`.
`Image.WIM.Name`	Specifies the name of the WIM file as located by the `Image.WIM.Path` property.
`Image.WIM.Index`	Specifies the index used to extract the correct image from the WIM file.
`Image.Network.User`	Specifies the user name with which to map the WIM image path (`Image.WIM.Path`) to a network drive on the provisioned machine. This is typically a domain account with access to the network share.
`Image.Network.Password`	Specifies the password associated with the `Image.Network.User` property.
`SysPrep.`*Section.Key* ■ `SysPrep.GuiUnattended.AdminPassword` ■ `SysPrep.GuiUnattended.EncryptedAdminPassword` ■ `SysPrep.GuiUnattended.TimeZone`	Specifies information to be added to the SysPrep answer file on machines during the WinPE stage of provisioning. Information that already exists in the SysPrep answer file is overwritten by these custom properties. *Section* represents the name of the section of the SysPrep answer file, for example GuiUnattended or UserData. *Key* represents a key name in the section. For example, to set the time zone of a provisioned machine to West Pacific Standard Time, define the custom property `GuiUnattended.UserData.TimeZone` and set the value to 275. For a full list of sections, keys, and accepted values, see the System Preparation Utility for Windows documentation. The following *Section.Key* combinations can be specified for WIM-based provisioning: ■ GuiUnattended ■ AdminPassword ■ EncryptedAdminPassword ■ TimeZone ■ UserData ■ ProductKey ■ FullName ■ ComputerName ■ OrgName ■ Identification ■ DomainAdmin ■ DomainAdminPassword ■ JoinDomain ■ JoinWorkgroup

Table 1-22. Custom Properties for PXE and WIM Provisioning (continued)

Custom Property	Description
`Sysprep.Identification.DomainAdmin`	Specifies a user name with administrator-level access to the target domain in Active Directory. Do not include the user domain in the credentials that you send to vCloud Director or vCloud Air.
`Sysprep.Identification.DomainAdminPassword`	Specifies the password to associate with the `Sysprep.Identification.DomainAdmin` property.
`Sysprep.Identification.JoinDomain`	Specifies the name of the domain to join in Active Directory.
`Sysprep.Identification.JoinWorkgroup`	Specifies the name of the workgroup to join if not using a domain.
`SysPrep.UserData.ComputerName`	Specifies a machine name, for example lab-client005.
`SysPrep.UserData.FullName`	Specifies the full name of a user.
`SysPrep.UserData.OrgName`	Specifies the organization name of the user.
`SysPrep.UserData.ProductKey`	Specifies the Windows product key.

Custom Properties for OVF Import

When you import an OVF to a blueprint, you can import and configure several settings as custom properties.

Table 1-23. Custom Properties for Blueprints In Which An OVF Is Imported

Custom Property	Description
`VMware.Ovf.Thumbprint`	If the OVF resides on an HTTPS server that has a certificate, this property stores the value of that certificate's thumbprint and is used to validate that certificate. It has no relevance when the OVF is hosted on an HTTP server. The property is automatically created when you import an OVF by using the `ImportOvfWorkflow` provisioning workflow in the blueprint component's user interface. If you create the blueprint programmatically with vRealize Automation REST APIs or vRealize CloudClient, you must manually create the property. **Note** The thumbprint can be stored in a comma-separated format to support a certificate chain. When the `VMware.Ovf.TrustAllCertificates` is present and set to true, the `VMware.Ovf.Thumbprint` property is ignored.
`VMware.Ovf.TrustAllCertificates`	When this property is present and set to true, the `Vmware.Ovf.Thumbprint` property is ignored and no certificate validation is performed when you import an OVF by using the `ImportOvfWorkflow` provisioning workflow.
`VMware.Ovf.Configuration.`*X*	An OVF can contain user-configurable properties, for example a property that sets the root password of a VM provisioned from the OVF. When you import an OVF into a blueprint, the user-configurable properties that are defined in the OVF are parsed and converted into custom properties of the form `Vmware.Ovf.Configuration.`*X*, where *X* is the name of the user-configurable property from the OVF.

Custom Properties for vRealize Automation Guest Agent

If you have installed the vRealize Automation guest agent in your templates for cloning or in your WinPE, you can use custom properties to run custom scripts within the guest operating system of a provisioned machine after the machine is fully deployed.

Table 1-24. Custom Properties for Customizing Provisioned Machines with a Guest Agent

Custom Property	Description
`VirtualMachine.Admin.AddOwnerToAdmins`	Set to True (default) to add the machine’s owner, as specified by the `VirtualMachine.Admin.Owner` property, to the local administrators group on the machine. This property is not available for provisioning by cloning.
`VirtualMachine.Admin.AllowLogin`	Set to True (default) to add the machine owner to the local remote desktop users group, as specified by the `VirtualMachine.Admin.Owner` property.

Table 1-24. Custom Properties for Customizing Provisioned Machines with a Guest Agent (continued)

Custom Property	Description
`VirtualMachine.Admin.UseGuestAgent`	If the guest agent is installed as a service on a template for cloning, set to True on the machine blueprint to enable the guest agent service on machines cloned from that template. When the machine is started, the guest agent service is started. Set to False to deactivate the guest agent. If set to False, the enhanced clone workfow will not use the guest agent for guest operating system tasks, reducing its functionality to `VMwareCloneWorkflow`. If not specified or set to anything other than False, the enhanced clone workflow sends work items to the guest agent.
`VirtualMachine.DiskN.Active`	Set to True (default) to specify that the machine's disk *N* is active. Set to False to specify that the machine's disk *N* is not active.
`VirtualMachine.DiskN.Label`	Specifies the label for a machine's disk *N*. The disk label maximum is 32 characters. Disk numbering must be sequential. When used with a guest agent, specifies the label of a machine's disk *N* inside the guest operating system.
`VirtualMachine.DiskN.Letter`	Specifies the drive letter or mount point of a machine's disk *N*. The default is C. For example, to specify the letter D for Disk 1, define the custom property as `VirtualMachine.Disk1.Letter` and enter the value D. Disk numbering must be sequential. When used in conjunction with a guest agent, this value specifies the drive letter or mount point under which an additional disk *N* is mounted by the guest agent in the guest operating system.
`VirtualMachine.DiskN.FS`	For use with Windows guest agent (gugent). Specifies the file system of the machine's disk *N*. The options are NTFS (default), FAT and FAT32. For example usage, see the `10_setupdisks.bat` Windows agent script.
`VirtualMachine.DiskN.FileSystem`	For use with Linux guest agent (gugent). Specifies the file system of the machine's disk *N*. The options are ext3, ext4, and XFS. For example usage, see the `30_DiskSetup.sh` Linux agent script.
`VirtualMachine.Admin.CustomizeGuestOSDelay`	Specifies the time to wait after customization is complete and before starting the guest operating system customization. The value must be in HH:MM:SS format. If the value is not set, the default value is one minute (00:01:00). If you choose not to include this custom property, provisioning can fail if the virtual machine reboots before guest agent work items are completed, causing provisioning to fail.
`VirtualMachine.Customize.WaitComplete`	Set to True to prevent the provisioning workflow from sending work items to the guest agent until all customizations arecomplete. Set to False to allow work items to be created before customization is complete.

Table 1-24. Custom Properties for Customizing Provisioned Machines with a Guest Agent (continued)

Custom Property	**Description**
`VirtualMachine.SoftwareN.Name`	Specifies the descriptive name of a software application *N* or script to install or run during provisioning. This is an optional and information-only property. It serves no real function for the enhanced clone workflow or the guest agent but it is useful for a custom software selection in a user interface or for software use reporting.
`VirtualMachine.SoftwareN.ScriptPath`	Specifies the full path to an application's install script. The path must be a valid absolute path as seen by the guest operating system and must include the name of the script filename. You can pass custom property values as parameters to the script by inserting {*CustomPropertyName*} in the path string. For example, if you have a custom property named `ActivationKey` whose value is 1234, the script path is `D:\InstallApp.bat –key {ActivationKey}`. The guest agent runs the command `D:\InstallApp.bat –key 1234`. Your script file can then be programmed to accept and use this value. Insert {Owner} to pass the machine owner name to the script. You can also pass custom property values as parameters to the script by inserting {*YourCustomProperty*} in the path string. For example, entering the value **`\\vra-scripts.mycompany.com\scripts\changeIP.bat`** runs the `changeIP.bat` script from a shared location, but entering the value **`\\vra-scripts.mycompany.com\scripts\changeIP.bat {VirtualMachine.Network0.Address}`** runs the changeIP script but also passes the value of the `VirtualMachine.Network0.Address` property to the script as a parameter.

Table 1-24. Custom Properties for Customizing Provisioned Machines with a Guest Agent (continued)

Custom Property	Description
`VirtualMachine.ScriptPath.Decrypt`	Allows vRealize Automation to obtain an encrypted string that is passed as a properly formatted `VirtualMachine.SoftwareN.ScriptPath` custom property statement to the gugent command line. You can provide an encrypted string, such as your password, as a custom property in a command-line argument. This allows you to store encrypted information that the guest agent can decrypt and understand as a valid command-line argument. For example, the `VirtualMachine.Software0.ScriptPath = c:\dosomething.bat` *`password`* custom property string is not secure as it contains an actual password. To encrypt the password, you can create a vRealize Automation custom property, for example `MyPassword = password`, and enable encryption by selecting the available check box. The guest agent decrypts the **`[MyPassword]`** entry to the value in the custom property `MyPassword` and runs the script as `c:\dosomething.bat password`. - Create custom property **`MyPassword =`** ***`password`*** where *password* is the value of your actual password. Enable encryption by selecting the available check box. - Set custom property `VirtualMachine.ScriptPath.Decrypt` as **`VirtualMachine.ScriptPath.Decrypt = true`**. - Set custom property `VirtualMachine.Software0.ScriptPath` as **`VirtualMachine.Software0.ScriptPath = c:\dosomething.bat [MyPassword]`**. If you set `VirtualMachine.ScriptPath.Decrypt` to false, or do not create the `VirtualMachine.ScriptPath.Decrypt` custom property, then the string inside the square brackets ([and]) is not decrypted.
`VirtualMachine.SoftwareN.ISOName`	Specifies the path and filename of the ISO file relative to the datastore root. The format is *`/folder_name/subfolder_name/file_name`*`.iso`. If a value is not specified, the ISO is not mounted.
`VirtualMachine.SoftwareN.ISOLocation`	Specifies the storage path that contains the ISO image file to be used by the application or script. Format the path as it appears on the host reservation, for example `netapp-1:it_nfs_1`. If a value is not specified, the ISO is not mounted.

Custom Properties for BMC BladeLogic Configuration Manager Integration

vRealize Automation includes custom properties that you can use to provide additional controls for BMC BladeLogic Configuration Manager integration.

Table 1-25. Custom Properties Required for BMC BladeLogic Configuration Manager Integrations

Custom Property	Description
`VirtualMachine.EPI.Type`	Specifies the type of external provisioning infrastructure.
`VirtualMachine.Admin.Owner`	Specifies the user name of the machine owner.
`BMC.Software.Install`	Set to True to enable BMC BladeLogic Configuration Manager integration.
`EPI.Server.Name`	Specifies the name of the external provisioning infrastructure server, for example, the name of the server hosting BMC BladeLogic. If at least one general BMC EPI agent was installed without specifying a BMC BladeLogic Configuration Manager host, this value directs the request to the desired server. If only dedicated BMC EPI agents for specific BMC BladeLogic Configuration Manager hosts were installed, this value must exactly match the server name configured for one of these agents.
`BMC.Service.Profile`	Specifies the name of the default authentication profile on the BMC BladeLogic server.
`BMC.Software.BatchLocation`	Specifies the location in BMC BladeLogic configuration where software jobs are deployed. This value must match the appropriate value of `Vrm.Software.IdNNNN`. For example, a valid value could be `/Application Deployment`.
`VMware.VirtualCenter.OperatingSystem`	Specifies the vCenter Server guest operating system version (`VirtualMachineGuestOsIdentifier`) with which vCenter Server creates the machine. This operating system version must match the operating system version to be installed on the provisioned machine. Administrators can create property groups using one of several property sets, for example, `VMware[OS_Version]Properties`, that are predefined to include the correct `VMware.VirtualCenter.OperatingSystem` values. This property is for virtual provisioning. For related information, see the enumeration type `VirtualMachineGuestOsIdentifier` in vSphere API/SDK Documentation. For a list of currently accepted values, see the vCenter Server documentation.

Custom Properties To Make BMC BladeLogic Configuration Manager Software Jobs Available

Configure BMC BladeLogic Configuration Manager jobs for vRealize Automation integrations. Make all software jobs available to machine requesters to select from, or specify a software job to apply to all machines provisioned from the blueprint.

Table 1-26. Custom Properties to Make Software Jobs Available

Custom Property	Description
`LoadSoftware`	Set to True to enable software install options.

`Vrm.Software.Id`*`NNNN`*	Specifies a software job or policy to be applied to all machines provisioned from the blueprint. Set the value to `job_type=job_path`, where `job_type` is the numeral that represents the BMC BladeLogic job type and `job_path` is the location of the job in BMC BladeLogic, for example `4=/Utility/putty`. *NNNN* is a number from 1000 to 1999. The first property must start with 1000 and increment in numerical order for each additional property.
	`1 – AuditJob`
	`2 – BatchJob`
	`3 – ComplianceJob`
	`4 – DeployJob`
	`5 – FileDeployJob`
	`6 – NSHScriptJob`
	`7 – PatchAnalysisJob`
	`8 – SnapshotJob`

Optional Custom Properties for BMC BladeLogic Configuration Manager Integrations

You can also use optional custom properties that are commonly used with BMC BladeLogic Configuration Manager blueprints.

Table 1-27. Optional Custom Properties for BMC BladeLogic Configuration Manager Integrations

Property	Definition
`BMC.AddServer.Delay`	Specifies the number of seconds to wait before adding the machine to BMC BladeLogic Configuration Manager. The default is 30.
`BMC.AddServer.Retry`	Specifies the number of seconds to wait before retrying if the first attempt to add the machine to BMC BladeLogic Configuration Manager is unsuccessful. The default is 100.

Custom Properties for HP Server Automation Integration

vRealize Automation includes custom properties that you can use to provide additional controls for HP Server Automation integration. Some custom properties are required for HP Server Automation integration. Other custom properties are optional.

Required Custom Properties for HP Server Automation Integration

Certain custom properties are required for a blueprint to work with HP Server Automation.

Table 1-28. Required Custom Properties for HP Server Automation Integration

Property	Definition
`VMware.VirtualCenter.OperatingSystem`	Specifies the vCenter Server guest operating system version (`VirtualMachineGuestOsIdentifier`) with which vCenter Server creates the machine. This operating system version must match the operating system version to be installed on the provisioned machine. Administrators can create property groups using one of several property sets, for example, `VMware[OS_Version]Properties`, that are predefined to include the correct `VMware.VirtualCenter.OperatingSystem` values. This property is for virtual provisioning.
`VirtualMachine.EPI.Type`	Specifies the type of external provisioning infrastructure.
`EPI.Server.Name`	Specifies the name of the external provisioning infrastructure server, for example, the name of the server hosting BMC BladeLogic. If at least one general BMC EPI agent was installed without specifying a BMC BladeLogic Configuration Manager host, this value directs the request to the desired server.
`Opsware.Software.Install`	Set to True to allow HP Server Automation to install software.
`Opsware.Server.Name`	Specifies the fully qualified name of the HP Server Automation server.
`Opsware.Server.Username`	Specifies the user name provided when a password file in the agent directory was created, for example opswareadmin. This user name requires administrative access to the HP Server Automation instance.
`Opsware.BootImage.Name`	Specifies the boot image value as defined in HP Server Automation for the 32-bit WinPE image, for example winpe32. The property is not required when provisioning by cloning.
`Opsware.Customer.Name`	Specifies a customer name value as defined in HP Server Automation, for example MyCompanyName.
`Opsware.Facility.Name`	Specifies a facility name value as defined in HP Server Automation, for example Cambridge.
`Opsware.Machine.Password`	Specifies the default local administrator password for an operating system sequence WIM image such as `Opsware.OSSequence.Name` as defined in HP Server Automation, for example P@ssword1.
`Opsware.OSSequence.Name`	Specifies the operating system sequence name value as defined in HP Server Automation, for example Windows 2008 WIM.
`Opsware.Realm.Name`	Specifies the realm name value as defined in HP Server Automation, for example Production.
`Opsware.Register.Timeout`	Specifies the time, in seconds, to wait for creation of a provisioning job to complete.

Table 1-28. Required Custom Properties for HP Server Automation Integration (continued)

Property	Definition
`VirtualMachine.CDROM.Attach`	Set to False to provision the machine without a CD-ROM device. The default is True.
`Linux.ExternalScript.Name`	Specifies the name of an optional customization script, for example `config.sh`, that the Linux guest agent runs after the operating system is installed. This property is available for Linux machines cloned from templates on which the Linux agent is installed.
`Linux.ExternalScript.LocationType`	Specifies the location type of the customization script named in the `Linux.ExternalScript.Name` property. This can be either local or nfs.
`Linux.ExternalScript.Path`	Specifies the local path to the Linux customization script or the export path to the Linux customization on the NFS server. The value must begin with a forward slash and not include the file name, for example `/scripts/linux/config.sh`.

Optional Custom Properties for HP Server Automation Integration

Certain custom properties are optional for a blueprint to work with HP Server Automation.

Table 1-29. Optional Custom Properties for HP Server Automation Integration

Property	Definition
`Opsware.ProvFail.Notify`	(Optional) Specifies the notification email address for HP Server Automation to use in the event of provisioning failure, for example provisionfail@lab.local.
`Opsware.ProvFail.Notify`	(Optional) Specifies the HP Server Automation user to whom ownership is assigned if provisioning fails.
`Opsware.ProvSuccess.Notify`	(Optional) Specifies the notification email address for HP Server Automation to use if provisioning is successful.
`Opsware.ProvSuccess.Owner`	(Optional) Specifies the HP Server Automation user to whom ownership is assigned if provisioning is successful.

Custom Properties That Make HP Server Automation Software Jobs Available

Depending on how your fabric administrator configures HP Server Automation jobs for vRealize Automation integration, you might have a choice between making all software jobs available to machine requesters to select, or you can specify jobs to apply to all machines provisioned from your blueprint.

Table 1-30. Custom Properties to Make Software Jobs Available

Property	Definition
`LoadSoftware`	Set to True to enable software install options.
`Vrm.Software.Id`	(Optional) Specifies an HP Server Automation policy to be applied to all machines provisioned from the blueprint. *NNNN* is a number from 1000 to 1999. The first property must start with 1000 and increment in numerical order for each additional property.

Custom Properties Grouped by Name

You can use custom properties to provide additional vRealize Automation controls.

Custom properties have been grouped here by name. To explore custom properties grouped by function, see Custom Properties Grouped by Function.

Custom Properties Underscore (_)

A list of vRealize Automation custom properties that begin with an underscore (_).

Table 1-31. Custom Properties Underscore (_) Table

Property	Description
`_debug_deployment`	Except for scale operations which allow partially successful deployments, the default behavior is to destroy the entire deployment if any of the individual resources fail to provision. You can override the default behavior by setting the `_debug_deployment` custom property value to true. If provisioning fails, the debugging custom property stops the resources from being rolled back so you can identify which of the components failed to provision successfully. In other words, by setting `_debug_deployment` to true, you can more easily debug customization and first-boot (for example, agent) issues because the setting ensures that machines are not destroyed after a provisioning failure. Otherwise, the setting doesn't directly change anything about the provisioning process, or affect guest agent or customization (for example, settings our outcomes relative to a vCenter customization spec). Note: A failed catalog item is normally inaccessible because it is immediately rolled back on failure. But when `_debug_deployment` is set to true, vRealize Automation treats the otherwise failed deployment as partially successful, which enables its accessibility. To apply the custom property to a blueprint, add `_debug_deployment` to the **Blueprint Properties** page using the **Properties** tab when you create or edit a blueprint. The `_debug_deployment` property is consumed at the software provisioning level, not the guest agent or machine provisioning level. You can also configure vRealize Automation to not delete virtual machines after deployment failure by using settings in the `VRMAgent.exe.config` file.
`_deploymentName`	When added to a blueprint, this property allows you to specify a custom name for the deployment by setting the value of `_deploymentName` to your custom string. If more than one instance of this deployment is provisioned in a single request, your custom name becomes a prefix. If you want users to specify their own deployment names, set this custom property to allow override. The following two caveats are required for usage: ■ You must add this property at the blueprint level, not at the component level. For example, when creating or editing a blueprint, click the **Properties** tab and then select **Custom Properties > New** to add the `_deploymentName` property to the blueprint. Do not add the property to a machine or other component in the blueprint. ■ You must add this property as a separate property and not as a member of a property group.

Custom Properties A

A list of vRealize Automation custom properties that begin with the letter A.

Table 1-32. Custom Properties A Table

Property	**Description**
`AD.Lookup.Department`	Specifies the cost center value that is included in a notification email sent to approvers. This property value must be specified in the blueprint.
`agent.download.url`	When using port forwarding, specifies the private IP address of your Amazon AWS tunnel machine and port for your software agent file, for example **`https://`*`Private_IP:1443`*`/software-service/resources/nobel-agent.jar`**. Specifies the URL for the VPN agent on your deployment. The URL format is `https://` *`Private_IP`*`:1443/software-service//resources/noble-agent.jar` You can add this property, in combination with `software.agent.service.url` and `software.ebs.url`, to a reservation or the compute resource endpoint. You can also use this property to specify a private address and port when using PAT or NAT translation and port forwarding.
`amazon.AmazonEC2Config.ServiceURL`	Specifies the Amazon configuration service URL for Amazon GovCloud, for example `amazon.AmazonEC2Config.ServiceURL`=https://ec2.us-gov-west-1.amazonaws.com.
`amazon.ElasticLoadBalancingConfig.ServiceURL`	Specifies the Amazon load balancer configuration service URL for Amazon GovCloud, for example `amazon.ElasticLoadBalancingConfig.ServiceURL`=https://elasticloadbalancing.us-gov-west-1.amazonaws.com.
`Amazon.ElasticLoadBalancer.Names`	Assigns machines that are provisioned by a blueprint to the elastic load balancers that match the specified values. This property is valid for vSphere, Amazon, and Hyper-V configurations.
`Amazon.Extensions.UserData`	Specifies the name of an Amazon user data script to be run during the first boot cycle when an instance is launched. The property supports string substitution from other custom properties to allow for dynamic requests. You can add the property either to the overall vRealize Automation blueprint or to an AWS machine component in the blueprint. For information about Amazon user data scripts, see the Running Commands on Your Linux Instance at Launch topic in *Amazon Elastic Compute Cloud* product documentation. You can pass a series of custom properties to the `Amazon.Extensions.UserData` property by including them in a file whose name begins with `Amazon.CustomProperty.Shell`.

Table 1-32. Custom Properties A Table (continued)

Property	Description
`amazon.IAMInstanceProfile.ARN`	Specifies the AWS Identity and Access Management (IAM) instance profile Amazon Resource Names (ARNs) when requesting an AWS instance. When you add this property, for example `amazon.IAMInstanceProfile.ARN` = *IAM Instance Profile ARN(s) value*, to a blueprint and then request provisioning from the catalog, the provisioned Amazon virtual machine or instance contains the specified IAM role. The DEM reads and includes the property specification, for example amazon.IAMInstanceProfile.ARN = IAM Instance Profile ARN(s) value, in the Amazon RunInstanceRequest workflow.
`Amazon.Instance.Id`	Specifies the Amazon instance ID of a machine provisioned on an Amazon EC2 endpoint. This property is valid for vSphere and Amazon configurations.
`Amazon.Instance.GroupName`	Specifies the name of the existing AWS placement group for the associated Amazon endpoint. The placement group must exist in the target availability zone prior to vRealize Automation data collection of the endpoint. Add the `Amazon.Instance.GroupName` custom property to a blueprint to specify which AWS placement group is used during machine provisioning.
`Amazon.elasticIpAddress.ipAddress`	Specifies the Amazon IP address where *ipAddress* is the specific IP address to assign to the instance.
`Amazon.Placement.Tenancy`	Set to = dedicated to specify that the AWS connection be specific to a dedicated tenant. This property is valid for use with VPC subnets.
`Amazon.Storage.Encrypt`	If set to true, specifies whether the Amazon EBS storage disks attached to the EC2 machine should be encrypted or not encrypted. Default is false. The property only applies to new EBS volume encryptions. Volumes that are part of an Amazon Machine Image (AMI) definition maintain their AMI settings regardless of this property.
`Amazon.Storage.iops`	Specifies the input/output operations per second (IOPS) for the associated storage device. Currently, this property is only supported when the `Amazon.Storage.Type` property value is io1. For more information, see Amazon EBS volume types documentation. Add the `Amazon.Storage.iops` custom property to a blueprint to specify the IOPS. The io1 storage type is the only AWS storage type in which you can set IOPS.

Table 1-32. Custom Properties A Table (continued)

Property	Description
`Amazon.Storage.Type`	Specifies the Amazon EBS volume type to use for disk storage relative to the associated Amazon endpoint. All disks are provisioned with the specified type. You cannot specify a different volume type for each disk. Set the property value to one of the API Names values provided in Amazon EBS volume types documentation, for example io1 or gp2. Add the `Amazon.Storage.Type` custom property to a blueprint to specify the EBS volume type to use during machine provisioning.
`Azure.Windows.ScriptPath`	Specifies the path to the downloaded script that configures tunneling for Windows-based systems. Update the path as appropriate for your deployment.
`Azure.Linux.ScriptPath`	Specifies the path to the downloaded script that configures tunneling for Linux-based systems. Update the path as appropriate for your deployment.

Custom Properties B

A list of vRealize Automation custom properties that begin with the letter B.

Table 1-33. Custom Properties B Table

Property	Definition
`BMC.AddServer.Delay`	Specifies the number of seconds to wait before adding the machine to BMC BladeLogic Configuration Manager. The default is 30.
`BMC.AddServer.Retry`	Specifies the number of seconds to wait before retrying if the first attempt to add the machine to BMC BladeLogic Configuration Manager is unsuccessful. The default is 100.
`BMC.Service.Profile`	Specifies the name of the default authentication profile on the BMC BladeLogic server.
`BMC.Software.BatchLocation`	Specifies the location in BMC BladeLogic configuration where software jobs are deployed. This value must match the appropriate value of `Vrm.Software.IdNNNN`. For example, a valid value could be `/Application Deployment`.
`BMC.Software.Install`	Set to True to enable BMC BladeLogic Configuration Manager integration.

Custom Properties C

A list of vRealize Automation custom properties that begin with the letter C.

Table 1-34. Custom Properties C Table

Property	Definition
`Cisco.Organization.Dn`	Specifies the distinguished name of the Cisco UCS Manager organization in which Cisco UCS machines provisioned by the business group are placed, for example org-root/org-Engineering. If the specified organization does not exist in the Cisco UCS Manager instance that is managing the machine, provisioning fails. This property is available for business groups only.
`CloneFrom`	Specifies the name of an existing machine or virtualization platform object to clone from, for example a template in vCenter Server such as Win2k8tmpl.
`CloneSpec`	Specifies the name of a customization specification on a cloned machine, for example a predefined `SysPrep` object in vCenter Server such as Win2k Customization Spec. The default value is specified on the blueprint.
`Command.DiskPart.Options`	When you use WIM-based virtual provisioning on ESX server hosts, set to Align=64 to use the recommended alignment parameters when you format and partition the machine's disk. This property is not available for physical provisioning.
`Command.FormatDisk.Options`	When you use WIM-based virtual provisioning on ESX server hosts, set to /A:32K to use the recommended alignment parameters when you format and partition the machine's disk. This property is not available for physical provisioning.
`containers.ipam.driver`	For use with containers only. Specifies the IPAM driver to be used when adding a Containers network component to a blueprint. The supported values depend on the drivers installed in the container host environment in which they are used. For example, a supported value might be `infoblox` or `calico` depending on the IPAM plug-ins that are installed on the container host. This property name and value are case-sensitive. The property value is not validated when you add it. If the specified driver does not exist on the container host at provisioning time, an error message is returned and provisioning fails.
`containers.network.driver`	For use with containers only. Specifies the network driver to be used when adding a Containers network component to a blueprint. The supported values depend on the drivers installed in the container host environment in which they are used. By default, Docker-supplied network drivers include bridge, overlay, and macvlan, while Virtual Container Host (VCH)-supplied network drivers include the bridge driver. Third-party network drivers such as `weave` and `calico` might also be available, depending on what network plug-ins are installed on the container host. This property name and value are case-sensitive. The property value is not validated when you add it. If the specified driver does not exist on the container host at provisioning time, an error message is returned and provisioning fails.
`Container`	For use with containers only. The default value is `App.Docker` and is required. Do not modify this property.

Table 1-34. Custom Properties C Table (continued)

Property	Definition
`Container.Auth.User`	For use with containers only. Specifies the user name for connecting to the Containers host.
`Container.Auth.Password`	For use with containers only. Specifies either the password for the user name or the public or private key password to be used. Encrypted property value is supported.
`Container.Auth.PublicKey`	For use with containers only. Specifies the public key for connecting to the Containers host.
`Container.Auth.PrivateKey`	For use with containers only. Specifies private key for connecting to the Containers host. Encrypted property value is supported.
`Container.Connection.Protocol`	For use with containers only. Specifies the communication protocol. The default value is `API` and is required. Do not modify this property.
`Container.Connection.Scheme`	For use with containers only. Specifies the communication scheme. The default is `https`.
`Container.Connection.Port`	For use with containers only. Specifies the Containers connection port. The default is 2376.
`Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.MachineActivated`	For use with containers only. Specifies the event broker property to expose all Containers properties and is used for registering a provisioned host. The default value is `Container*` and is required. Do not modify this property.
`Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.Disposing`	For use with containers only. Specifies the event broker property to expose all Containersproperties above and is used for unregistering a provisioned host. The default value is `Container*` and is required. Do not modify this property.

Custom Properties E

A list of vRealize Automation custom properties that begin with the letter E.

Table 1-35. Custom Properties E Table

Property	Definition
`EPI.Server.Collection`	Specifies the name of the Citrix provisioning collection to which the machine is to be registered.
`EPI.Server.Name`	Specifies the name of the external provisioning infrastructure server, for example, the name of the server hosting BMC BladeLogic. If at least one general BMC EPI agent was installed without specifying a BMC BladeLogic Configuration Manager host, this value directs the request to the desired server. If only dedicated BMC EPI agents for specific BMC BladeLogic Configuration Manager hosts were installed, this value must exactly match the server name configured for one of these agents. Specifies the name of the server hosting HP Server Automation. If at least one general Opsware EPI agent was installed without specifying a server automation server, this value directs the request to the desired server. If only dedicated EPI agents for specific HP server automation servers were installed, this value must exactly match the server name configured for one of these agents. If at least one general EPI agent of the appropriate type (`VirtualMachine.EPI.Type`) was installed without specifying a server, this value directs the request to the desired server. If only dedicated EPI agents for specific servers of the appropriate type were installed, this value must exactly match the server name configured for one of these agents.
`EPI.Server.Port`	Specifies the port on which to contact the provisioning server. If you are using a Citrix provisioning server, omit to specify the default port value of 54321.
`EPI.Server.Site`	Specifies the name of the Citrix provisioning site that contains the collection and store identified by the `EPI.Server.Collection` and `EPI.Server.Store` properties, for example site1.
`EPI.Server.Store`	Specifies the name of the Citrix provisioning store that contains the vDisk identified by the `EPI.Server.VDiskName` property, for example store1.
`EPI.Server.VDiskName`	Specifies the name of the Citrix provisioning vDisk from which to provision, for example disk1.
`ext.policy.activedirectory.customizationWorkflowTag`	The tag that you added to a custom vRealize Orchestrator workflow. The Active Directory policy searches for custom workflows with the specified tag and, if found, uses the tagged workflow when an Active Directory record is created.
`ext.policy.activedirectory.customizationDeleteWorkflowTag`	The tag that you added to a custom vRealize Orchestrator workflow. The Active Directory policy searches for custom workflows with the specified tag and, if found, uses the tagged workflow when an Active Directory record is deleted.
`ext.policy.activedirectory.domain`	The domain that you want to user rather than the domain in the current Active Directory policy. Overrides `ext.policy.activedirectory.system.domain` value that is specified in the Active Directory policy.

Table 1-35. Custom Properties E Table (continued)

Property	Definition
`ext.policy.activedirectory.endpoint.id`	The policy identifier to use to specify a policy or override policy. The ID that you provide must be for an existing Active Directory policy. Overrides `ext.policy.activedirectory.system.endpoint.id`value that is specified in the Active Directory policy.
`ext.policy.activedirectory.id`	The user-readable ID for the Active Directory policy. Overrides the `ext.policy.activedirectory.system.id` value that is specified in the Active Directory policy.
`ext.policy.activedirectory.ignore`	Indicates that the machine is not added to Active Directory by a policy. It skips the Active Directory policy.
`ext.policy.activedirectory.machineName`	The name of the machine in Active Directory that you want to use rather than the name in the current Active Directory policy. Overrides `ext.policy.activedirectory.system.machineName` value that is specified in the Active Directory policy.
`ext.policy.activedirectory.orgunit`	The organizational unit that you want to use rather than the organizational unit in the current Active Directory policy. Overrides the `ext.policy.activedirectory.system.orgunit` value that is specified in the Active Directory policy.
`ext.policy.activedirectory.system.domain`	System property for the domain of the machine in Active Directory. If you modify this property, which is used by the defined policies, you can deactivate the policy. Use `ext.policy.activedirectory.domain` to override the policy value.
`ext.policy.activedirectory.system.endpoint.id`	System property for the name of the Active Directory vRealize Orchestrator endpoint. If you modify this property, which is used by the defined policies, you can deactivate the policy. Use `ext.policy.activedirectory.endpoint.id` to override the policy value.
`ext.policy.activedirectory.system.id`	System property for the user-readable ID for the Active Directory policy. If you modify this property, which is used by the defined policies, you can deactivate the policy. Use `ext.policy.activedirectory.id` to override the policy value.
`ext.policy.activedirectory.system.machineName`	System property for the name of the machine in Active Directory. If you modify this property, which is used by the defined policies, you can deactivate the policy. Use `ext.policy.activedirectory.machineName` to override the policy value.
`ext.policy.activedirectory.system.orgunit`	System property for the distinguished name of the Active Directory organizational unit. If you modify this property, which is used by the defined policies, you can deactivate the policy. Use `ext.policy.activedirectory.orgunit` to override the policy value.

Custom Properties H

A list of vRealize Automation custom properties that begin with the letter H.

Table 1-36. Custom Properties H Table

Property	Definition
`Hostname`	Specifies the host machine name, overriding the generated machine name contained in the `VirtualMachine.Admin.Name` property. If `Hostname` is not used, the `VirtualMachine.Admin.Name` value is used as the machine name. The maximum number of allowed characters for the `Hostname` value is 15.
`Hyperv.Network.Type`	Specifies the network adapter type of the virtual machine. This property is valid for use with Hyper-V (SCVMM) only. When the value is set to synthetic, specifies that the blueprint be allowed to provision a Generation-2 machine on a Hyper-V (SCVMM) 2012 R2 resource. Generation-2 provisioning also requires that the blueprint includes the `Scvmm.Generation2 = true` property setting. The legacy value is not compatible with WinXP or Server 2003 x64 guest operating systems. The default value is synthetic.

Custom Properties I

A list of vRealize Automation custom properties that begin with the letter I.

Table 1-37. Custom Properties I Table

Property	Definition
`Image.ISO.Location`	Values for this property are case sensitive. Specifies the location of the ISO image from which to boot. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning. For virtual provisioning with vCenter Server, this specifies the name of a datastore in the instance that will be accessible to the provisioning compute resource. For virtual provisioning with XenServer, this specifies the name of a storage repository. For physical provisioning, this specifies the HTTP URL of the web-accessible location of the image.
`Image.ISO.Name`	Values for this property are case sensitive. Specifies the name of the ISO image from which to boot, for example /ISO/Microsoft/WinPE.iso. The format of this value depends on your platform. For details, see documentation provided for your platform. This property is required for WIM-based provisioning, Linux Kickstart and autoYaST provisioning, and SCCM-based provisioning. For virtual provisioning with vCenter Server, this svalue specifies the path to the image in the datastore specified by `Image.ISO.Location`. The value must use forward slashes and begin with a forward slash. For virtual provisioning with XenServer, this value specifies the name of the image in the storage repository specified by `Image.ISO.Location`. In virtual provisioning with Hyper-V, this value specifies the full local path to the image. For physical provisioning, this value specifies the file name of the image.
`Image.ISO.UserName`	Specifies the user name to access the CIFS share in the format *username@domain*. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.
`Image.ISO.Password`	Specifies the password associated with the `Image.ISO.UserName` property. For Dell iDRAC integrations where the image is located on a CIFS share that requires authentication to access.

Table 1-37. Custom Properties I Table (continued)

Property	Definition
`Image.WIM.Path`	Specifies the UNC path to the WIM file from which an image is extracted during WIM-based provisioning. The path format is \\server\share$ format, for example `\\lab-ad\dfs$`.
`Image.WIM.Name`	Specifies the name of the WIM file as located by the `Image.WIM.Path` property.
`Image.WIM.Index`	Specifies the index used to extract the correct image from the WIM file.
`Image.Network.User`	Specifies the user name with which to map the WIM image path (`Image.WIM.Path`) to a network drive on the provisioned machine. This is typically a domain account with access to the network share.
`Image.Network.Password`	Specifies the password associated with the `Image.Network.User` property.
`Image.Network.Letter`	Specifies the drive letter to which the WIM image path is mapped on the provisioned machine. The default value is K.
`Infrastructure.Admin.MachineObjectOU`	Specifies the organizational unit (OU) of the machine. When machines are placed in the required OU by the business group OU setting, this property is not required.
`Infrastructure.Admin.ADUser`	Specifies the domain administrator user ID. This identifier is used to query Active Directory users and groups when an anonymous bind cannot be used.
`Infrastructure.Admin.ADPassword`	Specifies the password associated with the `Infrastructure.Admin.ADUser` domain administrator user ID.
`Infrastructure.Admin.DefaultDomain`	Specifies the default domain on the machine.
`Infrastructure.ResourcePool.Name`	Specifies the resource pool to which the machine belongs, if any. The default is the value specified in the reservation from which the machine was provisioned.

Custom Properties L

A list of vRealize Automation custom properties that begin with the letter L.

Table 1-38. Custom Properties L Table

Property	Description
`Linux.ExternalScript.LocationType`	Specifies the location type of the customization script named in the `Linux.ExternalScript.Name` property. This can be either local or nfs. You must also specify the script location using the `Linux.ExternalScript.Path` property. If the location type is nfs, also use the `Linux.ExternalScript.Server` property.
`Linux.ExternalScript.Name`	Specifies the name of an optional customization script, for example `config.sh`, that the Linux guest agent runs after the operating system is installed. This property is available for Linux machines cloned from templates on which the Linux agent is installed. If you specify an external script, you must also define its location by using the `Linux.ExternalScript.LocationType` and `Linux.ExternalScript.Path` properties.

Table 1-38. Custom Properties L Table (continued)

Property	Description
`Linux.ExternalScript.Path`	Specifies the local path to the Linux customization script or the export path to the Linux customization on the NFS server. The value must begin with a forward slash and not include the file name, for example `/scripts/linux/config.sh`.
`Linux.ExternalScript.Server`	Specifies the name of the NFS server, for example lab-ad.lab.local, on which the Linux external customization script named in `Linux.ExternalScript.Name` is located.
`LoadSoftware`	Set to True to enable software install options.

Custom Properties M

A list of vRealize Automation custom properties that begin with the letter M.

Table 1-39. Custom Properties M Table

Property	Description
`MaximumProvisionedMachines`	Specifies the maximum number of linked clones for one machine snapshot. The default is unlimited.
`Machine.SSH`	Set to True to enable the **Connect Using SSH** option on the vRealize Automation Items page for Linux machines provisioned from this blueprint. If set to True and the **Connect using RDP or SSH** machine operation is enabled in the blueprint, all Linux machines that are provisioned from the blueprint display the **Connect Using SSH** option to entitled users. The **Connect Using SSH** option requires that your browser has a plug-in that supports SSH, for example the FireSSH SSH terminal client for Mozilla Firefox and Google Chrome. When the plug-in is present, selecting **Connect Using SSH** displays an SSH console and prompts for your administrator credentials.

Custom Properties N

A list of vRealize Automation custom properties that begin with the letter N.

Table 1-40. Custom Properties N Table

Property	Description
`NSX.Edge.ApplianceSize`	Specifies the allowed NSX edge appliance size types for the provisioned machine or deployment. The options are: ■ `compact` For small deployments, POCs, and single service use. ■ CPU = 1 ■ RAM = 512 MB ■ Disk = 512 MB ■ `large` For small to medium or multi-tenant deployments. ■ CPU = 2 ■ RAM = 1 GB ■ Disk = 512 MB ■ `quadlarge` For high throughput equal-cost multi-path routing (ECMP) or high performance firewall deployments. ■ CPU = 4 ■ RAM = 1 GB ■ Disk = 512 MB ■ `xlarge` For L7 load balancing and dedicated core deployments. ■ CPU = 6 ■ RAM = 8 GB ■ Disk = 4.5GB (4GB Swap) For related information, see System Requirements for NSX.
`NSX.Edge.HighAvailability`	When set to true (`NSX.Edge.HighAvailability`=true), enables high availability (HA) mode on the NSX edge machine that is deployed from the blueprint. When used with `NSX.Edge.HighAvailability.PortGroup`=*port_group_name*, this property allows you to configure an NSX edge during blueprint authoring. You can add this property to an NSX load balancer component in the vRealize Automation blueprint or to the vRealize Automation blueprint itself. Must be used in conjunction with `NSX.Edge.HighAvailability.PortGroup=` *port_group_name*.

Table 1-40. Custom Properties N Table (continued)

Property	Description
`NSX.Edge.HighAvailability.PortGroup`	Creates an internal interface or internal vNIC attached to the specified port group name, for example `NSX.Edge.HighAvailability.PortGroup=VM Network` where `VM Network` is an HA (high availability) distributed (vLAN-backed) or NSX logical switch port group. NSX HA mode requires at least one internal network interface, or vNIC. When used with `NSX.Edge.HighAvailability`=true, this property allows you to configure high availability (HA) an NSX edge during blueprint authoring. When using one arm load balancer with HA enabled, you must specify a separate port group for the HA. **Note** The specified port group network cannot be a member of the reservation pool, as the property's use of the port group conflicts with the normal deployment's use of the port group, resulting in the following error: `Portgroup must be unique within an Edge...` Must be used in conjunction with `NSX.Edge.HighAvailability`=true.

Table 1-40. Custom Properties N Table (continued)

Property	Description
`NSX.Validation.Disable.Single.Edge.Uplink`	When set to true, the NSX validation that checks for the following conditions is disabled: ■ All on-demand NAT networks on the blueprint source the same external network. ■ All on-demand routed networks on the blueprint that use the load balancer VIP source the same external network. ■ All on-demand load balancer components on the blueprint have VIPs on the same external network or on-demand networks backed by the same external network. Disabling this validation check can result in a deployment that succeeds but in which some network components might be inaccessible. If not present or if set to false, the validation check is enabled (default). A single NSX edge can only support one external network as its uplink network. Multiple IPs from the same external network are supported. While a blueprint can contain any number of external or on-demand network components, NSX only supports one external network as the uplink network. This property can only be specified at the blueprint level. It cannot be specified on a component in the blueprint canvas.
`NSX.Validation.Disable.Blueprint.NSXT`	When set to true, all NSX-T validation is disabled for the blueprint **Finish** action. If not present or if set to false, the NSX-T validation check is enabled (default). For example, if you have overlapping subnets in the blueprint, an error message appears when you click **Finish** in the blueprint and the overlap prevents you from finishing the blueprint, although you can save it. If you want to finish the blueprint, you can add `NSX.Validation.Disable.Blueprint.NSXT` by using the **Blueprint Properties** page and then finish the blueprint. The property only disables NSX-T validations for the blueprint **Finish** action.

Custom Properties O

A list of vRealize Automation custom properties that begin with the letter O.

Table 1-41. Custom Properties O Table

Property	Description
`Opsware.BootImage.Name`	Specifies the boot image value as defined in HP Server Automation for the 32-bit WinPE image, for example winpe32. The property is not required when provisioning by cloning.
`Opsware.Customer.Name`	Specifies a customer name value as defined in HP Server Automation, for example MyCompanyName.
`Opsware.Facility.Name`	Specifies a facility name value as defined in HP Server Automation, for example Cambridge.
`Opsware.Machine.Password`	Specifies the default local administrator password for an operating system sequence WIM image such as `Opsware.OSSequence.Name` as defined in HP Server Automation, for example P@ssword1.
`Opsware.OSSequence.Name`	Specifies the operating system sequence name value as defined in HP Server Automation, for example Windows 2008 WIM.
`Opsware.ProvFail.Notify`	(Optional) Specifies the notification email address for HP Server Automation to use in the event of provisioning failure, for example provisionfail@lab.local.
`Opsware.ProvFail.Owner`	(Optional) Specifies the HP Server Automation user to whom ownership is assigned if provisioning fails.
`Opsware.ProvSuccess.Notify`	(Optional) Specifies the notification email address for HP Server Automation to use if provisioning is successful.
`Opsware.ProvSuccess.Owner`	(Optional) Specifies the HP Server Automation user to whom ownership is assigned if provisioning is successful.
`Opsware.Realm.Name`	Specifies the realm name value as defined in HP Server Automation, for example Production.
`Opsware.Register.Timeout`	Specifies the time, in seconds, to wait for creation of a provisioning job to complete.
`Opsware.Server.Name`	Specifies the fully qualified name of the HP Server Automation server.
`Opsware.Server.Username`	Specifies the user name provided when a password file in the agent directory was created, for example opswareadmin. This user name requires administrative access to the HP Server Automation instance.
`Opsware.Software.Install`	Set to True to allow HP Server Automation to install software.

Custom Properties P

A list of vRealize Automation custom properties that begin with the letter P.

Table 1-42. Custom Properties P Table

Property	Description
`Plugin.AdMachineCleanup.Delete`	Set to True to delete the accounts of destroyed machines, instead of disabling them.
`Plugin.AdMachineCleanup.Execute`	Set to True to enable the Active Directory cleanup plug-in. By default, each machine’s account is deactivated when it is destroyed.

Table 1-42. Custom Properties P Table (continued)

Property	Description
`Plugin.AdMachineCleanup.MoveToOu`	Moves the account of destroyed machines to a new Active Directory organizational unit. The value is the organization unit to which you are moving the account. This value must be in *ou=OU, dc=dc* format, for example ou=trash,cn=computers,dc=lab,dc=local.
`Plugin.AdMachineCleanup.UserName`	Specifies an Active Directory account user name with sufficient privileges to perform Active Directory actions such as delete, deactivate, rename, or move Active Directory accounts. The value must be in *domain\username* format, for example lab\administrator. This property is required if the vRealize Automation manager service does not have these rights in a domain, which can occur when you provision machines in more than one domain.
`Plugin.AdMachineCleanup.Password`	Specifies the password associated to the `Plugin.AdMachineCleanup.UserName` property.
`Plugin.AdMachineCleanup.Domain`	Specifies the Active Directory domain name that contains the machine account to be destroyed.
`Plugin.AdMachineCleanup.RenamePrefix`	Renames the accounts of destroyed machines by adding a prefix. The value is the prefix string to prepend, for example destroyed_.
`Pxe.Clean.ScriptName`	Specifies the name of an EPI PowerShell script installed in the vRealize Automation Model Manager, to run on the machine after it is provisioned. The value is the name assigned to the script when it is uploaded to the Model Manager, for example `clean.ps1`.
`Pxe.Setup.ScriptName`	Specifies a custom EPI PowerShell script to run on the machine before it is started by using the PXE network boot program. The value is the name assigned to the script when it is uploaded to the model manager, for example `setup.ps1`.

Custom Properties R

A list of vRealize Automation custom properties that begin with the letter R.

Table 1-43. Custom Properties R Table

Property	Description
`ReservationPolicyID`	Specifies the reservation policy ID, not the reservation policy name. For example, the name that is returned by the vRealize Orchestrator property `getApplicableReservationPolicies` is the reservation policy name, not the reservation policy ID.

Custom Properties S

A list of vRealize Automation custom properties that begin with the letter S.

Table 1-44. Custom Properties S Table

Property	**Description**
`SysPrep.`*Section.Key* ■ `SysPrep.GuiUnattended.AdminPassword` ■ `SysPrep.GuiUnattended.EncryptedAdminPassword` ■ `SysPrep.GuiUnattended.TimeZone`	Specifies information to be added to the SysPrep answer file on machines during the WinPE stage of provisioning. Information that already exists in the SysPrep answer file is overwritten by these custom properties. *Section* represents the name of the section of the SysPrep answer file, for example GuiUnattended or UserData. *Key* represents a key name in the section. For example, to set the time zone of a provisioned machine to West Pacific Standard Time, define the custom property `GuiUnattended.UserData.TimeZone` and set the value to 275. For a full list of sections, keys, and accepted values, see the System Preparation Utility for Windows documentation. The following *Section.Key* combinations can be specified for WIM-based provisioning: ■ GuiUnattended ■ AdminPassword ■ EncryptedAdminPassword ■ TimeZone ■ UserData ■ ProductKey ■ FullName ■ ComputerName ■ OrgName ■ Identification ■ DomainAdmin ■ DomainAdminPassword ■ JoinDomain ■ JoinWorkgroup
`Sysprep.Identification.DomainAdmin`	Specifies a user name with administrator-level access to the target domain in Active Directory. Do not include the user domain in the credentials that you send to vCloud Director or vCloud Air.
`Sysprep.Identification.DomainAdminPassword`	Specifies the password to associate with the `Sysprep.Identification.DomainAdmin` property.
`Sysprep.Identification.JoinDomain`	Specifies the name of the domain to join in Active Directory.
`Sysprep.Identification.JoinWorkgroup`	Specifies the name of the workgroup to join if not using a domain.
`SysPrep.UserData.ComputerName`	Specifies a machine name, for example lab-client005.
`SysPrep.UserData.FullName`	Specifies the full name of a user.
`SysPrep.UserData.OrgName`	Specifies the organization name of the user.

Table 1-44. Custom Properties S Table (continued)

Property	Description
`SysPrep.UserData.ProductKey`	Specifies the Windows product key.
`SCCM.Collection.Name`	Specifies the name of the SCCM collection that contains the operating system deployment task sequence.
`SCCM.CustomVariable.`*Name*	Specifies the value of a custom variable, where *Name* is the name of any custom variable to be made available to the SCCM task sequence after the provisioned machine is registered with the SCCM collection. The value is determined by your choice of custom variable. If your integration requires it, you can use `SCCM.RemoveCustomVariablePrefix` to remove the `SCCM.CustomVariable.` prefix from your custom variable.
`SCCM.Server.Name`	Specifies the fully qualified domain name of the SCCM server on which the collection resides, for example lab-sccm.lab.local.
`SCCM.Server.SiteCode`	Specifies the site code of the SCCM server.
`SCCM.Server.UserName`	Specifies a user name with administrator-level access to the SCCM server.
`SCCM.Server.Password`	Specifies the password associated with the `SCCM.Server.UserName` property.
`SCCM.RemoveCustomVariablePrefix`	Set to *true* to remove the prefix `SCCM.CustomVariable.` from SCCM custom variables you created by using the custom property `SCCM.CustomVariable.`*Name*.
`Scvmm.Generation2`	When set to true, specifies that the blueprint be allowed to provision a Generation-2 machine on a Hyper-V (SCVMM) 2012 R2 resource. Generation-2 provisioning also requires that the blueprint includes the `Hyperv.Network.Type = synthetic` property setting.
`Snapshot.Policy.AgeLimit`	Sets the age limit, in days, for snapshots that can be applied to machines. This property applies to vSphere provisioning. When a snapshot exceeds the age limit, the Apply option is no longer available. When the snapshot age limit is reached, the snapshot remains but you can no longer revert to it. You can delete the snapshot using the vSphere client.

Table 1-44. Custom Properties S Table (continued)

Property	Description
`Snapshot.Policy.Limit`	Sets the number of snapshots allowed per machine. The default setting is one snapshot per machine. This property applies to vSphere provisioning. When set to 0, the blueprint option to create a snapshot is hidden for all users except for support and manager roles. Snapshots are shown in a hierarchical structure. ■ Depth – Maximum is 31. ■ Width – There is no limit.
`software.agent.service.url`	When using port forwarding, specifies the private IP address of your Amazon AWS tunnel machine and port for the vRealize Automation software service API, for example **`https://`*`Private_IP:1443`*`/software-service/api`**. You can add this property, in combination with `software.ebs.url` and `agent.download.url`, to a reservation or the compute resource endpoint. You can also use this property to specify a private address and port when using PAT or NAT and port forwarding.
`software.agent.task.timeout.seconds`	Specifies the timeout period, in seconds, for software scripts that are executing on agents. By default, the timeout period for software scripts that are executing on agents is 6 hours.
`software.ebs.url`	When using port forwarding, specifies the private IP address of your Amazon AWS tunnel machine and port for the vRealize Automation event broker service, for example **`https://`*`Private_IP:1443`*`/event-broker-service/api`**. You can add this property, in combination with `software.agent.service.url` and `agent.download.url`, to a reservation or the compute resource endpoint. You can also use this property to specify a private address and port when using PAT or NAT and port forwarding.

Table 1-44. Custom Properties S Table (continued)

Property	Description
`software.http.proxyHost`	Specifies the host name, or address, of the proxy server. For software content properties to use the proxy server, you must use both `software.http.proxyHost` and `software.http.proxyPort`. **Note** You can use the software proxy settings to define a content property type value for a software component. Content properties are URLs that are downloaded by the agent. The agent uses the variable as a file path to the locally downloaded file. However, you can use the software proxy settings to download through the proxy host instead of from the URL.
`software.http.proxyPassword`	Specifies the password for the user name with which to authenticate to the proxy server. Use in combination with `software.http.proxyUser`. The `software.http.proxyPassword` setting is required if you use the `software.http.proxyUser` setting. **Note** You can use the software proxy settings to define a content property type value for a software component. Content properties are URLs that are downloaded by the agent. The agent uses the variable as a file path to the locally downloaded file. However, you can use the software proxy settings to download through the proxy host instead of from the URL.
`software.http.proxyPort`	Specifies the port number of the proxy server. For software content properties to use the proxy server, you must use both `software.http.proxyHost` and `software.http.proxyPort`. There is no default `software.http.proxyPort` value. **Note** You can use the software proxy settings to define a content property type value for a software component. Content properties are URLs that are downloaded by the agent. The agent uses the variable as a file path to the locally downloaded file. However, you can use the software proxy settings to download through the proxy host instead of from the URL.

Table 1-44. Custom Properties S Table (continued)

Property	Description
`software.http.proxyUser`	Specifies the user name with which to authenticate to the proxy server. Use in combination with `software.http.proxyPassword`. The `software.http.proxyUser` setting is optional. The `software.http.proxyPassword` setting is required if you use the `software.http.proxyUser` setting. **Note** You can use the software proxy settings to define a content property type value for a software component. Content properties are URLs that are downloaded by the agent. The agent uses the variable as a file path to the locally downloaded file. However, you can use the software proxy settings to download through the proxy host instead of from the URL.
`software.http.noProxyList`	Specifies a list of hosts, and optional ports, that cannot use the proxyHost. The original content property downloads directly from URLs that match the patterns in the list. The `software.http.noProxyList` setting is only applicable if the proxy server is configured. For example, for the following comma separated list: `"buildweb.eng.vmware.com,confluence.eng.vmware.com:443,*.eng.vmware.com:80"` The following statements apply: - Any URL whose HOST is "buildweb.eng.vmware.com" cannot use the proxy server. - Any URL whose HOST is "confluence.eng.vmware.com" and the whose PORT is 443 cannot use the proxy server. - Any URL whose HOST is anything under the "eng.vmware.com" namespace and whose PORT is 80 cannot use the proxy server. **Note** You can use the software proxy settings to define a content property type value for a software component. Content properties are URLs that are downloaded by the agent. The agent uses the variable as a file path to the locally downloaded file. However, you can use the software proxy settings to download through the proxy host instead of from the URL.

Custom Properties V

A list of vRealize Automation custom properties that begin with the letter V.

Although general support for vCloud Networking and Security has ended, the VCNS custom properties continue to be valid for NSX purposes. See the Knowledge Base article 2144733.

Table 1-45. Custom Properties V Table

Property	Description
`VbScript.PreProvisioning.Name`	Specifies the full path of a Visual Basic script to be run before a machine is provisioned. For example, `%System-Drive%\Program Files(x86)\VMware\vCAC Agents\EPI_Agent\Scripts\SendEmail.vbs`. The script file must reside on the system on which the Visual Basic script EPI agent is installed.
`VbScript.PostProvisioning.Name`	Specifies the full path of a Visual Basic script to be run after a machine is provisioned. For example, `%System-Drive%\Program Files(x86)\VMware\vCAC Agents\EPI_Agent\Scripts\SendEmail.vbs`. The script file must reside on the system on which the Visual Basic script EPI agent is installed.
`VbScript.UnProvisioning.Name`	Specifies the full path of a Visual Basic script to be run when a machine is destroyed. For example, `%System-Drive%\Program Files (x86)\VMware\vCAC Agents\EPI_Agent\Scripts\SendEmail.vb`. The script file must reside on the system on which the Visual Basic script EPI agent is installed.
`VCloud.Lease.Sync.TimeBufferMins`	Specifies a threshold integer value for a compute resource such that lease synchronization between vCloud Director and vRealize Automation only occur for vCloud Director or vCloud Air-provisioned machines that are set to expire in vCloud Director or vCloud Air in that time period. If a conflict is found, the lease value is synchronized to match the lease length defined in vRealize Automation. The default `VCloud.Lease.Sync.TimeBufferMins` value is 720 minutes, which is 12 hours. If `VCloud.Lease.Sync.TimeBufferMins` is not present, the default value is used. For example, if the default values are used, vRealize Automation runs the lease synchronization check workflow every 45 minutes, which is the workflow default, and only the leases of machines that are set to expire within 12 hours are changed to match the lease length defined in vRealize Automation.
`VCloud.Owner.UseEndpointAccount`	Set to true to assign the endpoint account as the vCloud Air or vCloud Director machine owner for provisioning and import operations. For change ownership operations, the owner is not changed on the endpoint. If not specified or set to false, the vRealize Automation owner is the machine owner.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VCloud.Template.MakeIdenticalCopy`	Set to true to clone an identical copy of the vCloud Air or vCloud Director template for machine provisioning. The machine is provisioned as an identical copy of the template. Settings specified in the template, including storage path, supersede settings specified in the blueprint. The only changes from the template are the names of the cloned machines, which are generated from the machine prefix specified in the blueprint. vCloud Air or vCloud Director machines that are provisioned as identical copies can use networks and storage profiles that are not available in the vRealize Automation reservation. To avoid having unaccounted reservation allocations, verify that the storage profile or network specified in the template is available in the reservation.
`VCNS.LoadBalancerEdgePool.Names.`*name*	Specifies the NSX load balancing pools to which the virtual machine is assigned during provisioning. The virtual machine is assigned to all service ports of all specified pools. The value is an *edge/pool* name or a list of *edge/pool* names separated by commas. Names are case-sensitive. **Note** You can add a machine IP address to an existing load balancer by using the `VCNS.LoadBalancerEdgePool.Names` custom property. vRealize Automation and NSX use the first member of the specified edge load balancer pool to determine the new member port and monitor port settings. However, NSX 6.2 does not require that the member port setting be specified. To avoid provisioning failure when using `VCNS.LoadBalancerEdgePool.Names` with NSX 6.2 to add a machine to an existing pool, you must specify a port value for the first member of the load balancer pool in NSX. Appending a name allows you to create multiple versions of a custom property. For example, the following properties might list load balancing pools set up for general use and machines with high, moderate, and low performance requirements: ■ `VCNS.LoadBalancerEdgePool.Names` ■ `VCNS.LoadBalancerEdgePool.Names.moderate` ■ `VCNS.LoadBalancerEdgePool.Names.high` ■ `VCNS.LoadBalancerEdgePool.Names.low`
`VCNS.SecurityGroup.Names.`*name*	Specifies the NSX security group or groups to which the virtual machine is assigned during provisioning. The value is a security group name or a list of names separated by commas. Names are case-sensitive. Appending a name allows you to create multiple versions of the property, which can be used separately or in combination. For example, the following properties can list security groups intended for general use, for the sales force, and for support: ■ `VCNS.SecurityGroup.Names` ■ `VCNS.SecurityGroup.Names.sales` ■ `VCNS.SecurityGroup.Names.support`

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VCNS.SecurityGroup.Names.`*blueprint_name*	When using NSX, specifies the Edge Pool with which to associate the blueprint.
`VCNS.SecurityTag.Names.`*name*	Specifies the NSX security tag or tags to which the virtual machine is associated during provisioning. The value is a security tag name or a list of names separated by commas. Names are case-sensitive. Appending a name allows you to create multiple versions of the property, which can be used separately or in combination. For example, the following properties can list security tags intended for general use, for the sales force, and for support: ■ `VCNS.SecurityTag.Names` ■ `VCNS.SecurityTag.Names.sales` ■ `VCNS.SecurityTag.Names.support`
`VirtualMachine.Admin.UseGuestAgent`	If the guest agent is installed as a service on a template for cloning, set to True on the machine blueprint to enable the guest agent service on machines cloned from that template. When the machine is started, the guest agent service is started. Set to False to deactivate the guest agent. If set to False, the enhanced clone workfow will not use the guest agent for guest operating system tasks, reducing its functionality to `VMwareCloneWorkflow`. If not specified or set to anything other than False, the enhanced clone workflow sends work items to the guest agent. This property does not apply to Amazon Web Services provisioning.
`VirtualMachine.Admin.NameCompletion`	Specifies the domain name to include in the fully qualified domain name of the machine that the RDP or SSH files generate for the user interface options **Connect Using RDP** or **Connect Using SSH** option. For example, set the value to myCompany.com to generate the fully qualified domain name *`my-machine-name`*`.myCompany.com` in the RDP or SSH file.
`VirtualMachine.Admin.ConnectAddress`	Specifies the RDP connection address of the machine to which an RDP file is downloaded when the user interface option **Connect Using RDP** is used or attached to automatic emails. Do not use in a blueprint or property group unless you require the user to be prompted and you have not supplied a default value.
`VirtualMachine.Admin.ConnectAddress.Regex`	Used by a vRealize Automation administrator to define a regular expression to match an IP address for terminal connections, such as an RDP connection. If matched, the IP address is saved under the `VirtualMachine.Admin.ConnectAddress` custom property. Otherwise, the first available IP address is designated. For example, setting the property value to `10.10.0.` allows selection of an IP address from a 10.10.0.* subnet that is assigned to the virtual machine. If the subnet has not been assigned, the property is ignored. This property is available for use with OpenStack.

Table 1-45. Custom Properties V Table (continued)

Property	**Description**
`VirtualMachine.Admin.ThinProvision`	Determines whether thin provisioning is used on ESX compute resources. Disk provisioning is abstracted from the underlying storage. Set to True to use thin provisioning. Set to False to use standard provisioning. This property is for virtual provisioning.
`VirtualMachine.Admin.CustomizeGuestOSDelay`	Specifies the time to wait after customization is complete and before starting the guest operating system customization. The value must be in HH:MM:SS format. If the value is not set, the default value is one minute (00:01:00). If you choose not to include this custom property, provisioning can fail if the virtual machine reboots before guest agent work items are completed, causing provisioning to fail. This property does not apply to Amazon Web Services provisioning.
`VirtualMachine.Admin.Datastore.Cluster.ResourceLeaseDurationSec`	When provisioning to multiple VMs and using SDRS, specifies a value in seconds, in the range of 30 to 3600, for reserving storage resources during the `RecommendDataStore` API call. You can add this property to a business group or blueprint or when you request provisioning. The lease lock is only applied to the datastore that is used by the deployment, not all datastores in the storage cluster. The lease lock is released when provisioning either completes or fails. If not specified, no lock is applied to the storage resources at provisioning time. Because of memory size considerations, requesting more than 10 VMs simultaneously can cause provisioning failures.
`VirtualMachine.Admin.NetworkInterfaceType`	Indicates the network adapter type that is supported and emulated by the guest operating system. Use to create a new virtual machine and assign a specific adapter type for a template cloning operation. Use to modify the network settings of a newly provisioned virtual machine. The following options are available: ■ E1000 (default) ■ VirtIO ■ RTL8139 ■ RTL8139 VirtIO
`VirtualMachine.Admin.Name`	Specifies the generated machine name for vSphere, for example CodyVM01. When creating custom workflows or plug-ins for customizing a virtual machine name, set this property to match the name of the virtual machine. This is an internal input property for the agent to name the virtual machine. **Note** This property is for vSphere only. The value specified in the blueprint has no effect on this property. This property is not intended to be used to prompt the user. Use the `HostName` property to prompt the user. If the property is set at runtime, the container name that is created in the hypervisor might not match the item record name.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.Admin.UUID`	Specifies the UUID of the machine. The guest agent records the value when the machine is created. The value becomes read-only. The value in the blueprint or property group has no effect on this property.
`VirtualMachine.Admin.AgentID`	Specifies the UUID of the guest agent. The guest agent recorsd the value when the machine is created. The value becomes read-only. The value in the blueprint or property group has no effect on this property.
`VirtualMachine.Admin.Owner`	Specifies the user name of the machine owner.
`VirtualMachine.Admin.Approver`	Specifies the user name of the group manager who approved the machine request.
`VirtualMachine.Admin.Description`	Specifies the description of the machine as entered or modified by its owner or an administrator.
`VirtualMachine.Admin.EncryptPasswords`	If set to True, specifies that the administrator passwords are encrypted.
`VirtualMachine.Admin.AdministratorEmail`	Specifies the manager email addresses or Active Directory accounts for the business group of the provisioning blueprint. Multiple email addresses are separated by a comma, for example `AlbertAdmin@VMware.com,WeiLeeMgr@VMware.com`.
`VirtualMachine.Admin.TotalDiskUsage`	Specifies the total disk space in GB that the machine uses, including all disks as specified by the `VirtualMachine.DiskN.Size` properties and the swap file as specified by the `VMware.Memory. Reservation` property. You specify the value in GB, but the disk space is stored by vRealize Automation in MB.
`VirtualMachine.Admin.Hostname`	Informs the administrator which host is used for provisioning the machine on the endpoint. The specified value is implemented on the machine and is populated during data collection. For example, if the compute resource of a machine is changed, a proxy agent updates the value of the machine’s `VirtualMachine.Admin.Hostname` property. **Note** This is an internal output property from the agent that is populated during the data collection process and identifies the host on which a machine resides.
`VirtualMachine.Admin.ClusterName`	Informs the administrator which cluster contains the compute resource for the machine to use. **Note** This is an internal output property from the agent that is populated during the data collection process and identifies the cluster in which a machine resides.
`VirtualMachine.Admin.ApplicationID`	List the application IDs that can be assigned to a machine.
`VirtualMachine.Admin.AddOwnerToAdmins`	Set to True (default) to add the machine’s owner, as specified by the `VirtualMachine.Admin.Owner` property, to the local administrators group on the machine. This property is not available for provisioning by cloning.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.Admin.AllowLogin`	Set to True (default) to add the machine owner to the local remote desktop users group, as specified by the `VirtualMachine.Admin.Owner` property.
`VirtualMachine.Admin.DiskInterfaceType`	Indicates the type of disk drivers. The following disk drivers are supported: ■ IDE (default) ■ VirtIO This property is for virtual provisioning.
`VirtualMachine.Admin.EagerZero`	When set to true, specifies that the machine disks are provisioned using the VMware provisioning format of eager zero. Thick provision eager zero is a type of thick virtual disk that supports clustering features such as fault tolerance. Space required for the virtual disk is allocated at creation time. In contrast to the flat format, the data remaining on the physical device is zeroed out when the virtual disk is created. It might take much longer to create disks in this format than to create other types of disks.
`VirtualMachine.Admin.ForceHost`	Specifies the name of the ESX host. The property is only honored if `VirtualMachine.Admin.HostSelectionPolicy` is set to EXACT_MATCH. **Note** This property is for vSphere only. When provisioning against a vSphere cluster, you can use the `VirtualMachine.Admin.ForceHost` property to specify the host on which a machine is to be provisioned. This property is used only if DRS is not set to automatic for the cluster. If the cluster has DRS enabled and is set to Automatic, vSphere relocates the provisioned machine when the machine is restarted.
`VirtualMachine.Admin.HostSelectionPolicy`	Optionally set to EXACT_MATCH to require the machine to be placed on the host specified by the `VirtualMachine.Admin.ForceHost` property. If the host is unavailable, the request results in a failure. If a host is not specified, the next best available host is selected. If set to EXACT_MATCH, an error occurs if the specified host does not have enough memory or is in maintenance mode. **Note** This property applies to vSphere only.
`VirtualMachine.Agent.CopyToDisk`	Set to True (default) to copy the guest agent executable file to `%System-Drive%\VRM\Build\Bin` on the machine's disk.
`VirtualMachine.Agent.GuiRunOnce`	Set to True to include guest agent execution in the `SysPrep.inf` run once section. Set to False for the Linux agent to stop the provisioning workflow.
`VirtualMachine.Agent.Reboot`	Set to True (default) to specify that the guest agent restarts the machine following installation of the guest operating system.
`VirtualMachine.CDROM.Attach`	Set to False to provision the machine without a CD-ROM device. The default is True.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.CPU.Count`	Specifies the number of CPUs, for example 2, allocated to a machine. The default is the value specified by the CPU setting on the blueprint. **Note** This custom property value is overridden by the CPU value on the blueprint when the machine is first provisioned.
`VirtualMachine.Customize.WaitComplete`	Set to True to prevent the provisioning workflow from sending work items to the guest agent until all customizations arecomplete. Set to False to allow work items to be created before customization is complete. This property does not apply to Amazon Web Services provisioning.
`VirtualMachine.Core.Count`	If set to a value greater than zero, specifies the number of cores per socket when provisioning the virtual machine. You can use this property on a blueprint to specify cores per virtual socket or total number of sockets. For example, your licensing terms might restrict software that is licensed per socket or available operating systems only recognize so many sockets and additional CPUs must be provisioned as additional cores.
`VirtualMachine.DiskN.Letter`	Specifies the drive letter or mount point of a machine's disk *N*. The default is C. For example, to specify the letter D for Disk 1, define the custom property as `VirtualMachine.Disk1.Letter` and enter the value D. Disk numbering must be sequential. When used in conjunction with a guest agent, this value specifies the drive letter or mount point under which an additional disk *N* is mounted by the guest agent in the guest operating system.
`VirtualMachine.DiskN.IsFixed`	Deactivates the editing of a specific disk when reconfiguring a machine. Set to True to deactivate display of the edit capacity option for a specific volume. The True value is case-sensitive. The *N* value is the 0-based index of the disk. Alternatively, you can set the `VirtualMachine.Disk`*N*`.IsFixed` custom property to True in the VirtualMachineProperties table in the database or use the Repository API to specify a URI value such as `.../Repository/Data/ManagementModelEntities.svc/VirtualMachines(guid'60D93A8A-F541-4CE0-A6C6-78973AC0F1D2')/VirtualMachineProperties`.
`VirtualMachine.DiskN.Label`	Specifies the label for a machine's disk *N*. The disk label maximum is 32 characters. Disk numbering must be sequential. When used with a guest agent, specifies the label of a machine's disk *N* inside the guest operating system.
`VirtualMachine.DiskN.Active`	Set to True (default) to specify that the machine's disk *N* is active. Set to False to specify that the machine's disk *N* is not active.
`VirtualMachine.DiskN.FS`	For use with Windows guest agent (gugent). Specifies the file system of the machine's disk *N*. The options are NTFS (default), FAT and FAT32. For example usage, see the `10_setupdisks.bat` Windows agent script.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.DiskN.FileSystem`	For use with Linux guest agent (gugent). Specifies the file system of the machine's disk *N*. The options are ext3, ext4, and XFS. For example usage, see the `30_DiskSetup.sh` Linux agent script.
`VirtualMachine.DiskN.Percent`	Specifies the percentage of the disk *N* to be formatted by a guest agent for the machine's use. That machine cannot use the remaining portion of the disk.
`VirtualMachine.DiskN.StorageReservationPolicy`	Specifies the storage reservation policy to use to find storage for disk *N*. Also assigns the named storage reservation policy to a volume. To use this property, substitute the volume number for *N* in the property name and specify a storage reservation policy name as the value. This property is equivalent to the storage reservation policy name specified on the blueprint. Disk numbering must be sequential. This property is valid for all Virtual and vCloud reservations. This property is not valid for Physical, Amazon, or OpenStack reservations. You can use `VirtualMachine.DiskN.StorageReservationPolicyMode` to prevent provisioning from failing if there is insufficient space on the datastores in a storage reservation policy. Use this custom property to allow vRealize Automation to select a datastore outside the specified storage reservation policy in cases where there is not sufficient space remaining on the datastores in the policy.
`VirtualMachine.DiskN.StorageReservationPolicyMode`	Allocates disk *N* to the best available storage reservation policy.
`VirtualMachine.DiskN.Storage`	Specifies the datastore on which to place the machine disk *N*, for example DATASTORE01. This property is also used to add a single datastore to a linked clone blueprint. *N* is the index (starting at 0) of the volume to assign. Enter the name of the datastore to assign to the volume. This is the datastore name as it appears in the Storage Path on the Edit Compute Resource page. Disk numbering must be sequential.
`VirtualMachine.EPI.Type`	Specifies the type of external provisioning infrastructure. Set to BMC for BMC BladeLogic integration. Set to CitrixProvisioning for Citrix provisioning server integration.
`VirtualMachine.EULA.AcceptAll`	Set to true to specify that all the EULAs for the VM templates of the vCloud Air or vCloud Director endpoints are accepted during provisioning.
`VirtualMachine.Host.TpmEnabled`	Limits virtual machine placement to hosts that have a Trust Protection Module (TPM) device installed and recognized by ESX and vSphere. The default value is False. All hosts in a cluster must have a Trust Protection Module device installed. If no acceptable hosts or clusters are found, the machine cannot be provisioned until this property is removed.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.Memory.Size`	Specifies the size of the machine's memory in MB, such as 1024. The default is the value specified by the memory setting on the blueprint. **Note** This custom property setting is overridden by the memory setting on the blueprint when the machine is first provisioned.
`VirtualMachine.NetworkN.Address`	Specifies the IP address of network device *N* in a machine provisioned with a static IP address. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks.
`VirtualMachine.NetworkN.AdditionAddress`*M*	Defines additional *M* IP address allocated for an OpenStack instance for network *N*, excluding the IP address set specified by the `VirtualMachine.Network`*N*`.Address`. property. More addresses are displayed on the Network tab in the Additional Addresses column. This property is used by OpenStack machine state data collection. While this property is only data-collected by the OpenStack endpoint, it is not specific to OpenStack and can be used for lifecycle extensibility by other endpoint types. This property is not supported for on-demand NAT or on-demand routed networks.
`VirtualMachine.NetworkN.AddressType`	Specifies how IP address allocation is supplied to the network provider, where Network*N* is the network number, starting with 0. The following values are available: - DHCP - Static - MANUAL (available for vCloud Air and vCloud Director only) The MANUAL value also requires that you specify an IP address. This property is available for configuring vCloud Air, vCloud Director, and vSphere machine components in the blueprint. Also see `VirtualMachine.NetworkN.Name`. This property is not supported for on-demand NAT or on-demand routed networks.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.NetworkN.MacAddressType`	Indicates whether the MAC address of network device *N* is generated or user-defined (static). This property is available for cloning. The default value is generated. If the value is static, you must also use `VirtualMachine.NetworkN.MacAddress` to specify the MAC address. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks.
`VirtualMachine.NetworkN.MacAddress`	Specifies the MAC address of a network device *N*. This property is available for cloning. If the value of `VirtualMachine.NetworkN.MacAddressType` is generated, this property contains the generated address. If the value of `VirtualMachine.NetworkN.MacAddressType` is static, this property specifies the MAC address. For virtual machines provisioned on ESX server hosts, the address must be in the range specified by VMware. For details, see vSphere documentation. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.NetworkN.Name`	Specifies the name of the network to connect to, for example the network device *N* to which a machine is attached. This is equivalent to a network interface card (NIC). By default, a network is assigned from the network paths available on the reservation on which the machine is provisioned. Also see `VirtualMachine.NetworkN.AddressType`. You can ensure that a network device is connected to a specific network by setting the value of this property to the name of a network on an available reservation. For example, if you give properties for N= 0 and 1, you get 2 NICs and their assigned value, provided the network is selected in the associated reservation. `VirtualMachine.Network`*N* custom properties are specific to blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks. For an example of how to use this custom property to dynamically set `VirtualMachine.Network0.Name` based on a consumer's selection from a list of predefined available networks, see the Adding a Network Selection Drop-Down in vRA 7 blog post.
`VirtualMachine.NetworkN.PortID`	Specifies the port ID to use for network device *N* when using a dvPort group with a vSphere distributed switch. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. This property is not supported for on-demand NAT or on-demand routed networks.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.NetworkN.NetworkProfileName`	Specifies the name of a network profile from which to assign a static IP address to network device *N* or from which to obtain the range of static IP addresses that can be assigned to network device *N* of a cloned machine, where *N*=0 for the first device, 1 for the second, and so on. The network profile that the property points to is used to allocate an IP address. The property determines the network that the machine attaches to, based on the reservation. Changing this property value after the network is assigned has no effect on the expected IP address values for the designated machines. With WIM-based provisioning for virtual machines, you can use this property to specify a network profile and network interface or you can use the Network section of the Virtual Reservation page. The following attributes of the network profile are available to enable static IP assignment in a cloning blueprint: ■ `VirtualMachine.NetworkN.SubnetMask` ■ `VirtualMachine.NetworkN.Gateway` ■ `VirtualMachine.NetworkN.PrimaryDns` ■ `VirtualMachine.NetworkN.SecondaryDns` ■ `VirtualMachine.NetworkN.PrimaryWins` ■ `VirtualMachine.NetworkN.SecondaryWins` ■ `VirtualMachine.NetworkN.DnsSuffix` ■ `VirtualMachine.NetworkN.DnsSearchSuffixes` `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. You cannot use this custom property to define an on-demand NAT or on-demand routed network profile name. Because on-demand network profile names are generated at allocation time (during provisioning), their names are unknown when creating or editing the blueprint. To specify NSX on-demand network information, use the applicable network component in the blueprint design canvas for your vSphere machine components.

Table 1-45. Custom Properties V Table (continued)

Property	Description
■ `VirtualMachine.NetworkN.SubnetMask` ■ `VirtualMachine.NetworkN.Gateway` ■ `VirtualMachine.NetworkN.PrimaryDns` ■ `VirtualMachine.NetworkN.SecondaryDns` ■ `VirtualMachine.NetworkN.PrimaryWins` ■ `VirtualMachine.NetworkN.SecondaryWins` ■ `VirtualMachine.NetworkN.DnsSuffix` ■ `VirtualMachine.NetworkN.DnsSearchSuffixes`	Configures attributes of the network profile specified in `VirtualMachine.NetworkN.NetworkProfileName`. `VirtualMachine.Network`*N* custom properties are specific to individual blueprints and machines. When a machine is requested, network and IP address allocation is performed before the machine is assigned to a reservation. Because blueprints are not guaranteed to be allocated to a specific reservation, do not use this property on a reservation. When specifying values for multiple DNS search suffixes using `VirtualMachine.NetworkN.DnsSearchSuffixes`, you can use commas to separate values for a Windows deployment. These properties are not supported for on-demand NAT or on-demand routed networks.
`VirtualMachine.Rdp.File`	Specifies the file that contains RDP settings to be used when opening an RDP link to the machine. Can be used together with, or as an alternative to, `VirtualMachine.Rdp.Setting`*N*. The file must be located in the *vRA_installation_dir*\Server\Website\Rdp folder. You must create the `Rdp` directory. For related information, see `VirtualMachine.Rdp.SettingN`.
`VirtualMachine.Rdp.SettingN`	Specifies the RDP settings to be used when opening an RDP link to the machine. *N* is a unique number used to distinguish one RDP setting from another. For example, to specify the RDP authentication level so that no authentication requirement is specified, define the custom property `VirtualMachine.Rdp.Setting1` and set the value to authentication level:i:3. For information about available RDP settings, and their correct syntax, see Microsoft Windows RDP documentation such as RDP Settings for Remote Desktop Services in Windows Server. For related information, see `VirtualMachine.Rdp.File`.
`VirtualMachine.Reconfigure.DisableHotCpu`	Set to true to specify that the reconfigure machine action restarts the specified machine. By default, the reconfigure machine action does not restart the machine. Performing a hot add of CPU, memory, or storage causes the reconfigure machine action to fail and not to restart the machine unless the `Hot Add` setting is enabled in vSphere for the machine or template. You can add `VirtualMachine.Reconfigure.DisableHotCpu=true` to a machine component in a vRealize Automation blueprint to deactivate the `Hot Add` setting and force the machine to restart regardless of the vSphere `Hot Add` setting. The custom property is only available for machine types that support hardware reconfiguration, which are vSphere, vCloud Air, and vCloud Director.
`VirtualMachine.Request.Layout`	Specifies the property layout to be used in the virtual machine request page. The value must match the name of the layout to be used.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.Software.Execute`	When set to True, supports guest agents for Amazon Web Services provisioning. Use this property with the `VirtualMachine.SoftwareN.Name` and `VirtualMachine.SoftwareN.ScriptPath` custom properties to configure and use guest agents in Amazon Web Services provisioning.
`VirtualMachine.SoftwareN.Name`	Specifies the descriptive name of a software application *N* or script to install or run during provisioning. This is an optional and information-only property. It serves no real function for the enhanced clone workflow or the guest agent but it is useful for a custom software selection in a user interface or for software use reporting.
`VirtualMachine.SoftwareN.ScriptPath`	Specifies the full path to an application's install script. The path must be a valid absolute path as seen by the guest operating system and must include the name of the script filename. You can pass custom property values as parameters to the script by inserting {*CustomPropertyName*} in the path string. For example, if you have a custom property named `ActivationKey` whose value is 1234, the script path is `D:\InstallApp.bat –key {ActivationKey}`. The guest agent runs the command `D:\InstallApp.bat –key 1234`. Your script file can then be programmed to accept and use this value. You can also pass custom property values as parameters to the script by inserting {*YourCustomProperty*} in the path string. For example, entering the value **`\\vra-scripts.mycompany.com\scripts\changeIP.bat`** runs the `changeIP.bat` script from a shared location, but entering the value **`\\vra-scripts.mycompany.com\scripts\changeIP.bat {VirtualMachine.Network0.Address}`** runs the changeIP script but also passes the value of the `VirtualMachine.Network0.Address` property to the script as a parameter. Insert {Owner} to pass the machine owner name to the script.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.ScriptPath.Decrypt`	Allows vRealize Automation to obtain an encrypted string that is passed as a properly formatted `VirtualMachine.SoftwareN.ScriptPath` custom property statement to the gugent command line. You can provide an encrypted string, such as your password, as a custom property in a command-line argument. This allows you to store encrypted information that the guest agent can decrypt and understand as a valid command-line argument. For example, the `VirtualMachine.Software0.ScriptPath = c:\dosomething.bat` *`password`* custom property string is not secure as it contains an actual password. To encrypt the password, you can create a vRealize Automation custom property, for example `MyPassword = password`, and enable encryption by selecting the available check box. The guest agent decrypts the **`[MyPassword]`** entry to the value in the custom property `MyPassword` and runs the script as `c:\dosomething.bat password`. ■ Create custom property **`MyPassword =`** ***`password`*** where *password* is the value of your actual password. Enable encryption by selecting the available check box. ■ Set custom property `VirtualMachine.ScriptPath.Decrypt` as **`VirtualMachine.ScriptPath.Decrypt = true`**. ■ Set custom property `VirtualMachine.Software0.ScriptPath` as **`VirtualMachine.Software0.ScriptPath = c:\dosomething.bat [MyPassword]`**. If you set `VirtualMachine.ScriptPath.Decrypt` to false, or do not create the `VirtualMachine.ScriptPath.Decrypt` custom property, then the string inside the square brackets ([and]) is not decrypted.
`VirtualMachine.SoftwareN.ISOName`	Specifies the path and filename of the ISO file relative to the datastore root. The format is *`/folder_name/subfolder_name/file_name`*`.iso`. If a value is not specified, the ISO is not mounted.
`VirtualMachine.SoftwareN.ISOLocation`	Specifies the storage path that contains the ISO image file to be used by the application or script. Format the path as it appears on the host reservation, for example `netapp-1:it_nfs_1`. If a value is not specified, the ISO is not mounted.
`VirtualMachine.Storage.Name`	Identifies the storage path on which the machine resides. The default is the value specified in the reservation that was used to provision the machine.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VirtualMachine.Storage.AllocationType`	Stores collected groups to a single datastore. A distributed environment stores disks round-robin style. Specify one of the following values: - Collected Keep all disks together. - Distributed Allow disks to be placed on any datastore or datastore cluster that is available in the reservation. For an example of how to use the `VirtualMachine.Storage.AllocationType` property to create datastore clusters, see the Keeping Multiple Disks Together blog post.
`VirtualMachine.Storage.Cluster.Automation.Enabled`	If set to True, the storage cluster automation on the machine is enabled. If set to False, then storage cluster automation is deactivated on the machine. The storage cluster automation type is determined by the `VirtualMachine.Storage.Cluster.Automation.Behavior` custom property.
`VirtualMachine.Storage.Cluster.Automation.Behavior`	Specifies an SDRS behavior type when `VirtualMachine.Storage.Cluster.Automation.Enabled` is set to True. The available behavior type values are automated or manual. The `VirtualMachine.Storage.Cluster.Automation.Enabled` and `VirtualMachine.Storage.Cluster.Automation.Behavior` properties are set after the machine is provisioned and after inventory data collection is finished. If automation is deactivated, `VirtualMachine.Storage.Cluster.Automation.Behavior` is not present on the machine.
`VirtualMachine.Storage.ReserveMemory`	Set to True to manage vSwap storage allocation to ensure availability and set allocation in the reservation. vSwap allocation is considered when you create or reconfigure a virtual machine. vSwap allocation checking is only available for vSphere endpoints. **Note** If you do not specify the `VirtualMachine.Storage.ReserveMemory` custom property when you create or provision the machine from vRealize Automation, swap space availability is not ensured. If you add the property for an already provisioned machine, and the allocated reservation is full, the storage allocated in the reservation might exceed the actual allocated storage.
`VirtualMachine.VDI.Type`	Specifies the type of virtual desktop infrastructure. For XenDesktop provisioning, set to XenDesktop.
`VMware.AttributeN.Name`	Specifies the name of an attribute in vRealize Orchestrator. For example, it specifies the value of the attribute used in the `VMware.AttributeN.Name` property. Replace the letter *N* with a number, starting with 0 and increasing for each attribute to set.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VMware.AttributeN.Value`	Specifies the value of the attribute used in the `VMware.AttributeN.Name` property. Replace the letter *N* with a number, starting with 0 and increasing for each attribute to set.
`VMware.Endpoint.Openstack.IdentityProvider.Domain.Name`	Allows vRealize Automation to support required Keystone V3 domain-name authentication. If Keystone V3 is in effect, you can use the property to designate a specific domain for the OpenStack endpoint to authenticate with a Keystone V3 OpenStack identity provider. ■ For new endpoints, add the custom property to designate a specific domain. ■ For upgraded or migrated endpoints, add the custom property only if data collection fails after upgrade or migration.
`VMware.Endpoint.Openstack.IdentityProvider.Version`	Specifies the version of OpenStack identity provider (Keystone) to use when authenticating an OpenStack endpoint. Configure a value of `3` to authenticate with Keystone V3 OpenStack identity provider. If you use any other value, or do not use this custom property, authentication defaults to Keystone V2.
`VMware.Endpoint.Openstack.Release`	Deprecated. Specifies the OpenStack release, for example Havana or Icehouse, when creating an OpenStack endpoint. Required for 6.2.1, 6.2.2, and 6.2.3 OpenStack provisioning.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VMware.Endpoint.NSX.HideDiscoveredSecurityObjects`	Set to true to hide newly discovered security objects in the active tenant for the NSX endpoints to which the security objects are associated. Otherwise, all new security objects are available to all tenants after data collection, provided that the security object is for an endpoint in which you have a reservation. This option allows you to prevent users from accessing security objects when you want to assign those objects to a single tenant or to mask from all tenants. Set to false to toggle back to global, which enables all new security objects to be available to all tenants after data collection, provided that the security object is for an endpoint in which you have a reservation. To take effect, the fabric administrator adds the `VMware.Endpoint.NSX.HideDiscoveredSecurityObjects` custom property to the associated NSX endpoint that is associated to a vSphere endpoint. The setting applies to the next inventory data collection. Existing security objects remain unchanged. To change the tenancy setting of a security object that has already been data-collected, such as existing security objects after upgrading to the current vRealize Automation release, you can edit the security object's Tenant ID setting programmatically by using the vRealize Automation REST API or vRealize CloudClient. The available Tenant ID settings for the NSX endpoint are as follows: ■ "`<global>`" - the security object is available to all tenants. This is the default setting for existing security objects after upgrade to this release and for all new security objects that you create. ■ "`<unscoped>`" - the security object is not available to any tenants. Only the system administrator can access the security object. This is an ideal setting when defining security objects that are to eventually be assigned to a specific tenant. ■ "*`tenant_id_name`*" - the security object is only available to a single, named tenant.
`VMware.Hardware.Version`	Specifies the VM hardware version to be used for vSphere settings. Supported values are currently vmx-04, vmx-07, vmx-08, vmx-09 and vmx-10. This property is applicable for VM Create and VM Update workflows and is available only for basic workflow blueprints.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VMware.VirtualCenter.OperatingSystem`	Specifies the vCenter Server guest operating system version (`VirtualMachineGuestOsIdentifier`) with which vCenter Server creates the machine. This operating system version must match the operating system version to be installed on the provisioned machine. Administrators can create property groups using one of several property sets, for example, `VMware[OS_Version]Properties`, that are predefined to include the correct `VMware.VirtualCenter.OperatingSystem` values. This property is for virtual provisioning. When this property has a non-Windows value, the **Connect Using RDP** user interface option is deactivated. The property can be used in a virtual, cloud or physical blueprint. For related information, see the enumeration type `VirtualMachineGuestOsIdentifier` in vSphere API/SDK Documentation. For a list of currently accepted values, see the vCenter Server documentation.
`VMware.SCSI.Type`	For vCloud Air, vCloud Director, or vSphere machine components in blueprints, specifies the SCSI machine type using one of the following case-sensitive values: ■ buslogic Use BusLogic emulation for the virtual disk. ■ lsilogic Use LSILogic emulation for the virtual disk (default). ■ lsilogicsas Use LSILogic SAS 1068 emulation for the virtual disk. ■ pvscsi Use para-virtualization emulation for the virtual disk. ■ none Use if a SCSI controller does not exist for this machine. The `VMware.SCSI.Type` property is not available for use with the CloneWorkflow provisioning workflow. If you specify the CloneWorkflow provisioning workflow when configuring your machine component in the blueprint design canvas, you cannot use the `VMware.SCSI.Type` property.
`VMware.SCSI.Sharing`	Specifies the sharing mode of the machine's VMware SCSI bus. Possible values are based on the `VirtualSCSISharing` ENUM value and include noSharing, physicalSharing, and virtualSharing. If you specify the CloneWorkflow provisioning workflow when configuring your machine component in the blueprint design canvas, the `VMware.SCSI.Sharing` property is not available. The `VMware.SCSI.Sharing` property is not available for use with the CloneWorkflow provisioning workflow. If you specify the CloneWorkflow provisioning workflow when configuring your machine component in the blueprint design canvas, you cannot use the `VMware.SCSI.Sharing` property.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VMware.Memory.Reservation`	Defines the amount of reserved memory for the virtual machine in MB, for example 1024. Setting this value also reduces the size of the virtual machine swap file on disk by the amount specified.
`VMware.Network.Type`	Specifies the network to connect the VM as specified in the reservation. The network adapter on the machine must be connected to a unique network. The following adapter type values are available: - Flexible (default) - VirtualPCNet32 (not compatible with vSphere). - E1000 or VirtualE1000 - VMXNET or VirtualVMXNET - VMXNET2 - VMXNET3 Set to E1000 when provisioning Windows 32-bit virtual machines on ESX server hosts to ensure that machines are created with the correct network adapter. This property is not used for physical provisioning.
`VMware.Ovf.Thumbprint`	If the OVF resides on an HTTPS server that has a certificate, this property stores the value of that certificate's thumbprint and is used to validate that certificate. It has no relevance when the OVF is hosted on an HTTP server. The property is automatically created when you import an OVF by using the `ImportOvfWorkflow` provisioning workflow in the blueprint component's user interface. If you create the blueprint programmatically with vRealize Automation REST APIs or vRealize CloudClient, you must manually create the property. **Note** The thumbprint can be stored in a comma-separated format to support a certificate chain. When the `VMware.Ovf.TrustAllCertificates` is present and set to true, the `VMware.Ovf.Thumbprint` property is ignored.
`VMware.Ovf.TrustAllCertificates`	When this property is present and set to true, the `Vmware.Ovf.Thumbprint` property is ignored and no certificate validation is performed when you import an OVF by using the `ImportOvfWorkflow` provisioning workflow.
`VMware.Ovf.Configuration.`*X*	An OVF can contain user-configurable properties, for example a property that sets the root password of a VM provisioned from the OVF. When you import an OVF into a blueprint, the user-configurable properties that are defined in the OVF are parsed and converted into custom properties of the form `Vmware.Ovf.Configuration.`*X*, where *X* is the name of the user-configurable property from the OVF.
`VMware.VCenterOrchestrator.EndpointName`	Overrides a specified endpoint setting or specifies that a particular endpoint be used during the vRealize Automation IaaS provisioning process. The value of this property can be set to an applicable vRealize Orchestrator endpoint, such as external VRO, available in the environment.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VMware.VirtualCenter.Folder`	Specifies the name of the inventory folder in the data center in which to put the virtual machine. The default is VRM, which is also the vSphere folder in which vRealize Automation places provisioned machines if the property is not used. This value can be a path with multiple folders, for example `production\email servers`. A proxy agent creates the specified folder in vSphere if the folder does not exist. Folder names are case-sensitive. This property is available for virtual provisioning.
`VDI.Server.Website`	Specifies the server name of the Citrix Web interface site to use in connecting to the machine. If the value of `VDI.Server.Name` is a XenDesktop farm, this property must have an appropriate value or the machine owner cannot connect to the machine using XenDesktop. If this property is not specified, the `VDI.Server.Name` property determines the desktop delivery controller to connect to, which must be the name of a server that hosts a desktop delivery controller. **Note** If the Citrix Web Interface (WI) has been replaced with StoreFront (SF), you can use this property instead of `VDI.Server.Name` to connect to the XenDesktop server. An example value is `VDI.Server.Website=sqa-xddc-7.sqa.local/Citrix/StoreWeb`. See `VDI.Server.Name` for more information.
`VDI.Server.Name`	Specifies the server name, which hosts the desktop delivery controller, to register with, or the name of a XenDesktop farm that contains desktop delivery controllers with which to register. If the value is a XenDesktop farm name, the `VDI.Server.Website` property value must be the URL of an appropriate Citrix web interface site to use in connecting to the machine. If the value is a server name, and at least one general XenDesktop VDI agent was installed without specifying a desktop delivery controller server, this value directs the request to the desired server. If the value is a server name, and only dedicated XenDesktop VDI agents for specific DDC servers were installed, this value must exactly match the server name configured for a dedicated agent. **Note** For more information about how to make StoreFront the default page in IIS, see Citrix documentation. See also `VDI.Server.Website`. **Note** Changes in the Citrix web interface protocol have impacted how the `VDI.Server.Name` default value is recognized. The value of the `VDI.Server.Name` property is used as the default connection string to open the Citrix web interface when users connect to a virtual desktop. It is always the DNS/IP of the XD server. If that value does not connect to the Citrix interface, you are unable to access your VMs. However, you can use the `VDI.Server.Website` custom property when the Citrix web interface is hosted on a server other than the XenDesktop server. When this property is present on the VM, it is used instead of `VDI.Server.Name`.

Table 1-45. Custom Properties V Table (continued)

Property	Description
`VDI.Server.Group`	For XenDesktop 5, specifies the name of the XenDesktop group to add machines to and the name of the catalog to which the group belongs, in the *group_name;catalog_name* format. For XenDesktop 4, specifies the name of the XenDesktop group to which machines are to be added. XenDesktop 4 preassigned groups are supported.
`VDI.ActiveDirectory.Interval`	Specifies an optional interval value in time span format for virtual desktop infrastructure machine Active Directory registration check. The default value is 00:00:15 (15 seconds).
`VDI.ActiveDirectory.Timeout`	Specifies an optional timeout value to wait before retrying Active Directory registration. The default value is 00:00:15 (30 minutes).
`VDI.ActiveDirectory.Delay`	Specifies an optional delay time value in time span format between successfully adding a machine to Active Directory and initiation of XenDesktop registration. The default value is 00:00:05 (5 seconds).
`Vrm.DataCenter.Location`	Allows you to use a blueprint to provision machines on more than one compute resource. You can add the `Vrm.DataCenter.Location` property to a blueprint, or enable the **Display Location on Request** option in the blueprint, to require that the user supply a data center location when they request machine provisioning. **Note** If you enable the **Display Location on Request** option on the blueprint, you do not need to also add the custom property. Data center locations are configured in a `DataCenterLocations.xml` file, which provides the location values that are applied to compute resources. For related information about adding data center locations, see *Configuring vRealize Automation*. Because the `Vrm.DataCenter.Location` property cannot access the contents of the `DatacenterLocations.xml` file, you must rely on users to provide property values that match the locations provided in the `DataCenterLocations.xml` file. Use this property if you want to use the data center location value as input to an external action for another custom property.

Table 1-45. Custom Properties V Table (continued)

Property	**Description**
`Vrm.DataCenter.Policy`	Specifies whether provisioning must use a compute resource associated with a particular location, or if any location is suitable. To enable this feature, you must add data center to a location file. Associate each compute resource with a location. Set to Exact (default) to provision a requested machine on a compute resource associated with the location specified on the blueprint. The request fails if no reservations match the requested location. If the property is not present, the Exact default is used. Set to NonExact to provision a requested machine on a compute resource with sufficient capacity and associated with the location specified on the blueprint. If that compute resource is not available, then use the next available compute resource with sufficient capacity without regard to location.
`Vrm.ProxyAgent.Uri`	Allows you to override the default `Vrm.ProxyAgent.Uri` value that is derived from the VMPS endpoint address in the vRealize Automation Manager Service configuration file. The configuration setting is often set to the local machine but you might want to set it to the virtual IP (VIP) address. You can specify the `Vrm.ProxyAgent.Uri` custom property on a blueprint. Sample syntax is as follows: `Vrm.ProxyAgent.Uri`=https://*loadbalancer-vip*/VMPS2Proxy
`Vrm.Software.Id`*NNNN* This row is specific to BMC BladeLogic.	Specifies a software job or policy to be applied to all machines provisioned from the blueprint. Set the value to `job_type=job_path`, where `job_type` is the numeral that represents the BMC BladeLogic job type and `job_path` is the location of the job in BMC BladeLogic, for example `4=/Utility/putty`. *NNNN* is a number from 1000 to 1999. The first property must start with 1000 and increment in numerical order for each additional property.

```
1 – AuditJob
2 – BatchJob
3 – ComplianceJob
4 – DeployJob
5 – FileDeployJob
6 – NSHScriptJob
7 – PatchAnalysisJob
8 – SnapshotJob
```

Property	**Description**
`Vrm.Software.Id`*NNNN* This row is specific to HP Server Automation.	(Optional) Specifies an HP Server Automation policy to be applied to all machines provisioned from the blueprint. *NNNN* is a number from 1000 to 1999. The first property must start with 1000 and increment in numerical order for each additional property.

Custom Properties X

A list of vRealize Automation custom properties that begin with the letter X.

Table 1-46. Custom Properties X Table

Property	Description
`Xen.Platform.Viridian`	For virtual provisioning, set to False when you provision Windows virtual machines on a XenServer host or pool. The default is True. This property is not used in physical provisioning.

Using the Property Dictionary

You can use the property dictionary to define new custom property definitions and property groups.

You define a property to support a specific data type and a display control style within that data type. You can also create reusable property groups to simplify adding multiple properties.

Using Property Definitions

Many custom properties are supplied with vRealize Automation. You can also define new properties to create unique custom properties and provide greater control for provisioning machines.

When you add a property to a blueprint or reservation, you can determine if a user must be prompted for a property value and if the property value must be encrypted.

You can specify how a property is rendered, for example if should display as a checkbox or as a drop-down menu with values obtained from a custom vRealize Orchestrator workflow.

You can also use properties to control how your custom workflows function. For information about using vRealize Automation Designer to define and work with custom workflows, see *Life Cycle Extensibility*.

Best Practices for Naming Property Definitions

To avoid naming conflicts with supplied vRealize Automation custom properties, use a standard and meaningful prefix for all property names that you create. Use a prefix such as a company or feature name followed by a dot for all new property names. VMware reserves all property names that do not contain a dot (.). Property names that do not follow this recommendation may conflict with vRealize Automation custom properties. In that event, the vRealize Automation custom property takes precedence over property definitions that you create.

General Procedures

The following steps describe the general procedure for creating and using new property definitions:

1 Create a new property definition and associate it with a data type that allows for a specific type of content, such as boolean or integer content. Use a standard naming convention for the new property name such as *my_grouping_prefix.my_property_name*.

2. Associate a property definition with a display type, such as a check box or drop-down menu. Available display types are derived from the selected data type.
3. Add the property to a blueprint either individually or as part of a property group.

   Add the property to a blueprint and specify if the property value must be encrypted.

   Add the property to a blueprint and specify if the user should be prompted to specify a property value.
4. As a machine requestor, specify required values as prompted.

Using vRealize Orchestrator Script Actions

You can populate the property value in a drop-down menu by using vRealize Orchestrator script actions. Using vRealize Orchestrator script actions also enables you to populate a drop-down menu value based on the values specified for another property.

You can use the `vra content list --type property-definition` vRealize CloudClient command to list all property definitions in the current vRealize Automation tenant. You can also use the `vra content list --type property-group` vRealize CloudClient command to list all property groups.

For a tutorial on creating dynamic property definitions to filter the options that are available to users, see the How to use dynamic property definitions blog post.

Limitations

If you create a property definition where `Data type` equals `String`, `Display as` equals `Dropdown`, and you use a vRealize Orchestrator action that returns properties that populate the drop-down list, the list is in random order. You cannot specify the order.

Create and Test Custom Property Definitions

You create a custom properties definition that determines how the custom property appears in vRealize Automation. You can add the custom property to a blueprint so that you can verify that the property displays the check box, drop-down menu, or other control type as expected.

To create and test the custom property definitions, you need a blueprint that is already entitled to you or to a test user account to which you have access. This test blueprint allows you to create the custom property, add it to a blueprint, and then verify that the custom property has the expected appearance. After you validate the custom property, you can add it to your production blueprints as needed.

Prerequisites

- Verify that you have a blueprint to which you are adding the action. See *Configuring vRealize Automation*.
- Verity that the blueprint is entitled to you so that you can test the custom properties in the blueprint. See *Configuring vRealize Automation*.

- Log in to vRealize Automation as a **tenant administrator** or **fabric administrator**.

Procedure

1 Create Custom Property Definitions

  You create custom property definitions that determine how the custom property appears in vRealize Automation. You can validate the custom property in a test blueprint before adding it to your production blueprints.

2 Add a Custom Property to a Blueprint

  You can add custom properties to many parts of vRealize Automation, including approval policies, business groups, endpoints, and reservation policies. However, only the machine blueprints support the display options that you configure as property definitions. Adding a custom property to a blueprint as a simple way to verify that the custom property appears in the user interface as you designed it in the property definition.

3 Verify the Custom Property in the Catalog Request Form

  As creator of the custom property definitions that run vRealize Orchestrator actions, you should test your custom properties to ensure that the correct values appear in the request form.

Create Custom Property Definitions

You create custom property definitions that determine how the custom property appears in vRealize Automation. You can validate the custom property in a test blueprint before adding it to your production blueprints.

- Create a Property Definition

  You can create property definitions to allow for additional levels of vRealize Automation customization. When you create a property definition, you specify a data type for the property, for example a string or a boolean type.

- Create a Custom Property That Validates Against a Regular Expression

  You create custom property definition that evaluates a regular expression when you want service catalog users to provide validated data on the catalog request form.

- Create a vRealize Orchestrator Action Custom Property Definition

  You create a custom property definition that includes a vRealize Orchestrator action so that you can add the custom property to a blueprint. The action runs when the service catalog user is configuring the custom property in the request form. The action retrieves the data that is displayed in the form.

- Bind Custom Properties to Create a Parent-Child Relationship

  To create a parent-child relationship between custom properties, you bind the parent to the child. When you add the parent and child custom properties to a blueprint, the requesting user selects a value for the parent property. The selected parent value determines the possible values for the child property.

Create a Property Definition

You can create property definitions to allow for additional levels of vRealize Automation customization. When you create a property definition, you specify a data type for the property, for example a string or a boolean type.

To avoid conflict with supplied vRealize Automation custom properties, use a naming format of *my_prefix.my_property_name1*. For example, use a standard prefix such as company name, followed by a dot (.), followed by a descriptive name.

Properties that you create that do not follow this recommendation may conflict with vRealize Automation-supplied custom properties. In that event, the vRealize Automation custom properties take precedence over properties that you create.

Prerequisites

Log in to vRealize Automation as a **tenant administrator** or **fabric administrator**.

Procedure

1. Select **Administration > Property Dictionary > Property Definitions**.
2. Click **New** (✚).
3. Enter the new property definition name in the **Name** text box.

   Use a standard naming convention for the new property name such as *my_grouping_prefix.my_property_name*.

   The **Name** value is stored internally as the property identifier (ID).
4. Accept the generated value in the **Label** text box.

   The **Label** value is automatically populated with the value that you entered in the **Name** text box. If you enter a **Label** value first, the **Name** text box is populated with the same value.

   The **Label** value is displayed in the user interface when requesting properties, for example when adding a property to a blueprint, as the property name.
   The **Label** value can contain a wider range of characters than the **Name** value.
5. In the **Visibility** section, select **All tenants** or **This tenant** to determine where the property is to be available.

   If you are logged in with only tenant administrator privileges, then only **This tenant** is available. If you are logged in with only fabric administrator privileges, then only **All tenants** is available.

   You cannot change the **All tenants** or **This tenant** setting after you create the item.
6. (Optional) Enter a property description in the **Description** text box.

   Describe the intent of the property definition and any other helpful information about the property.

7 (Optional) Enter a value in the **Display order** text box.

The number that you enter controls how the property name appears on the request form. The following ordering rules apply:

- The display order applies only to properties that are configured with **Prompt User** or **Show in Request Form** settings.
- All properties with a display order appear before properties with no display order.
- Properties with a display order are sorted from lowest to highest value. Negative numbers are allowed.
- All properties are ordered alphabetically, with all display order properties appearing before non-display order properties.
- If two properties have the same display order value, they are sorted alphabetically.

8 Select a property definition data type from the **Data type** drop-down menu.

Table 1-47. Data type

Data type	Display as
Boolean	Allows for a boolean value. The Display as options are **Checkbox** and **Yes/No**.
Datetime	Allows for a value entered in a date and time format. The Display as option is **Date Time Picker**.
Decimal	Allows for an integer or decimal value. The Display as options are **Dropdown**, **Slider**, and **Textbox**.
Integer	Allows for an integer value. The Display as options are **Dropdown**, **Slider**, and **Textbox**.
Secure String	Allows for secure or encrypted content such as a password. The Display as option is either a password that requires confirmation or **Textbox**.
String	Allows for a string value. The Display as options are **Dropdown**, **Email**, **Hyperlink**, **Textarea**, and **Textbox**.

9 If the **Required** option is available, select **Yes** or **No** from the drop-down menu to specify if a value must be supplied for this property.

10 Select a display control type for this property in the **Display as** drop-down menu. Available options are derived from your **Data type** selection as shown in the above table.

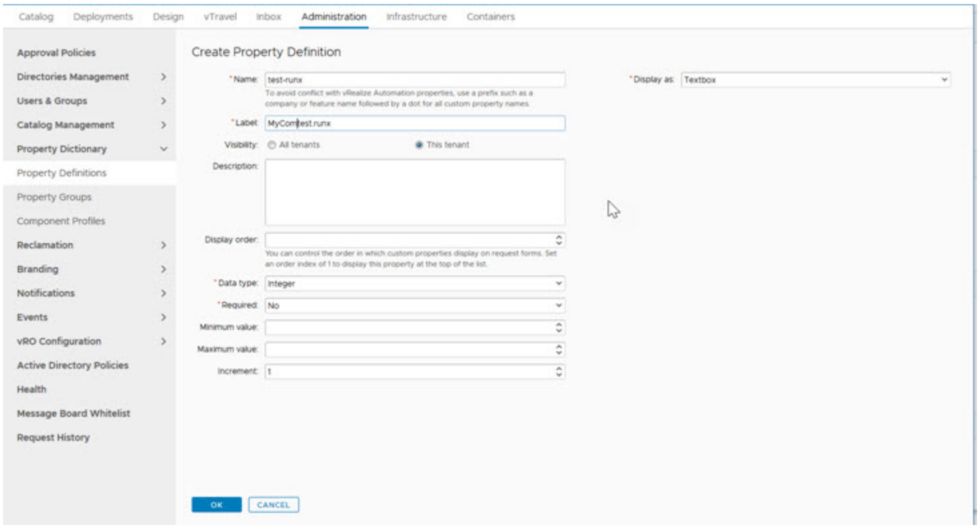
Table 1-48. Display as

Display as	Description
CheckBox	Provides a single check box control.
Date Time Picker	Provides a date and time control that adheres to a *YYYY-MM-DD* or *MM/DD/YYYY* format and a time in *HH:MM* format, 24-hour clock or followed by AM or PM.

Table 1-48. Display as (continued)

Display as	Description
Dropdown	Provides a drop-down menu control.
Email	Provides an email control.
Hyperlink	Displays a link with the property display name as the link text and the property value as the URL.
Slider	Provides a slider control for a range of values.
Testarea	Provides a text area in which to enter or display information.
Textbox	Provides a text box in which to enter a value.
Yes/No	Specifies a Yes or No value.

11 Based on your data type, make any additional selections as presented in the user interface. A sample page is shown below:



Create Property Definition

*Name: test-runx

To avoid conflict with vRealize Automation properties, use a prefix such as a company or feature name followed by a dot for all custom property names.

*Label: MyComtest.runx

Visibility: ○ All tenants ● This tenant

Description:

Display order:

You can control the order in which custom properties display on request forms. Set an order index of 1 to display this property at the top of the list.

*Data type: Integer

*Required: No

Minimum value:

Maximum value:

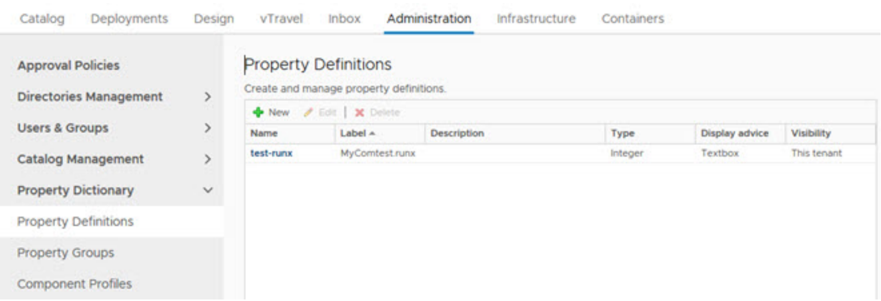
Increment: 1

*Display as: Textbox

12 Click **OK**.

Results

The property is created and available on the Property Definitions page.



Property Definitions

Create and manage property definitions.

Name	Label	Description	Type	Display advice	Visibility
test-runx	MyComtest.runx		Integer	Textbox	This tenant

Create a Custom Property That Validates Against a Regular Expression

You create custom property definition that evaluates a regular expression when you want service catalog users to provide validated data on the catalog request form.

For example, to add an alphanumeric text box where the requesting user provides an application or function name that is limited to five to ten characters with no special characters. For this scenario, you use a regular expression custom property configured to something similar to `^[a-zA-Z0-9]{5,10}$`.

Prerequisites

- Ensure that you have a regular expression that validates the provided values as expected.
- Log in to vRealize Automation as a **tenant administrator** or **fabric administrator**.

Procedure

1. Select **Administration > Property Dictionary > Property Definitions**.
2. Click the **New** icon (✚).
3. Enter the options.

Option	Description
Name	Enter a value using a standard naming convention for the new property name such as `my_grouping_prefix.my_property_name`.
Label	The label is populated based on the name. You can change the label to provide a more readable name.
Visibility	The action custom properties are only available in the current tenant. To make them available in another tenant, you must configure them when you are logged in to that tenant.
Description	Describe the intent of the property definition and any other helpful information about the property.
Display order	The number that you enter controls how the property name appears on the request form. The following ordering rules apply: ■ The display order applies only to properties that are configured with **Prompt User** or **Show in Request Form** settings. ■ All properties with a display order appear before properties with no order index. ■ Properties with a display order are sorted from lowest to highest order index value. You can use negative numbers. ■ All properties are ordered alphabetically, with all display ordered properties appearing before non-display ordered properties. ■ If two properties have the same display order value, then they are sorted alphabetically.
Data type	Select **String** in the drop-down menu.
Required	Select **No** in the drop-down menu.

Option	Description
Display as	Select **Textbox** in the drop-down menu.
Valid user input	Enter the regular expression.

4. Enter a value in the testing text box to verify that the expression works.
5. Click **OK**.

Results

The custom property definition is added to the list and it is available to add to a blueprint.

What to do next

Add the custom property to a machine blueprint. See Add a Custom Property or Property Group Using the Properties Tab on a Blueprint's Machine Component.

Create a vRealize Orchestrator Action Custom Property Definition

You create a custom property definition that includes a vRealize Orchestrator action so that you can add the custom property to a blueprint. The action runs when the service catalog user is configuring the custom property in the request form. The action retrieves the data that is displayed in the form.

Prerequisites

- Review the configuration details for the custom property you are creating. See Configuration Details for the vRealize Orchestrator Action Custom Property Definitions.
- Log in to vRealize Automation as a **tenant administrator** or **fabric administrator**.

Procedure

1. Select **Administration > Property Dictionary > Property Definitions**.
2. Click the **New** icon (✚).
3. Enter the options.

Option	Description
Name	Consult the configuration details. Some of the custom properties require specific names or formats. Where you can, use a standard naming convention for the new property name such as `my_grouping_prefix.my_property_name`.
Label	The label is populated based on the name. You can change the label to provide a more readable name.
Visibility	The action custom properties are only available in the current tenant. To make them available in another tenant, you must configure them when you are logged in to that tenant.

Option	Description
Description	Describe the intent of the property definition and any other helpful information about the property.
Display order	The number that you enter controls where the property name appears on the request form. The following ordering rules apply: ■ The display order applies only to properties that are configured with **Prompt User** or **Show in Request Form** settings. ■ All properties with a display order appear before properties with no order index. ■ Properties with a display order are sorted from lowest to highest value. You can use negative numbers. ■ All properties are ordered alphabetically, with all display order properties appearing before non-display order properties. ■ If two properties have the same display order value, they are sorted alphabetically.

4. Consult the configuration details to determine what you must provide for the values.

   The following values are provided in the configuration details:

   - Data type
   - Display as
   - Values
   - Action folder
   - Script action
   - Input parameters

5. Click **OK**.

Results

The custom property definition is added to the list and it is available to add to a blueprint.

What to do next

Add the custom property to a blueprint. Whether you add it as a machine or as a network property depends on the property. See Add a Custom Property to a Blueprint.

Bind Custom Properties to Create a Parent-Child Relationship

To create a parent-child relationship between custom properties, you bind the parent to the child. When you add the parent and child custom properties to a blueprint, the requesting user selects a value for the parent property. The selected parent value determines the possible values for the child property.

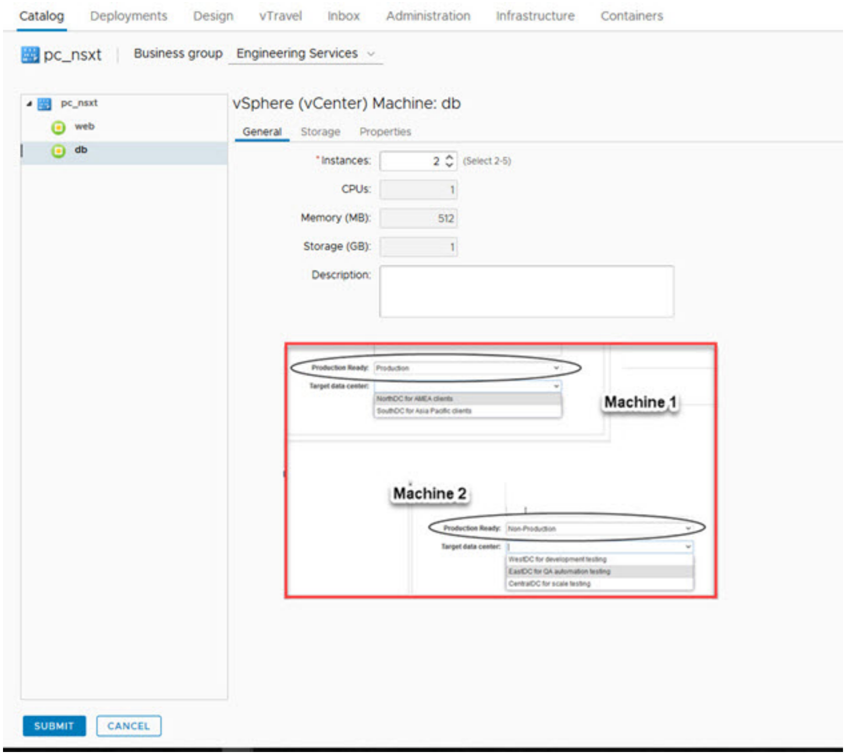
- The parent custom property definition can be a static list or an external value that is determined by an vRealize Orchestrator action. It provides possible input parameters to a child property definition.

- The child custom property definition must call a vRealize Orchestrator action. In the child custom property, you bind the parent custom property so that it provides an input parameter value.

For example, your development team works on production and non-production systems. You also have five data centers. Three of the data centers are your development testing data centers and the other two are where you provide services to your internal clients. To ensure that developers can deploy the same blueprint to either environment, the testing or the internal clients data centers, you create and bind two custom property definition. Using the first custom property, the requesting user can select either the production or non-production environment. Based the environment that the user selects in the request form, the second custom property displays one the following values:

- The list of three testing data centers for the non-production environments.
- The two internal clients data centers as production environments.

The following screen illustrates the catalog request page for the Machine 1 (db), with a snippet section illustrating the property to bind from the Machine 1 (db) to the property in the Machine 2 (web).



pc_nsxt | Business group: Engineering Services

vSphere (vCenter) Machine: db

*Instances: 2 (Select 2-5)
CPUs: 1
Memory (MB): 512
Storage (GB): 1
Description:

Production Ready: Production
Target data center:
NorthDC for AMEA clients
SouthDC for Asia Pacific clients

Production Ready: Non-Production
Target data center:
WestDC for development testing
EastDC for QA automation testing
CentralDC for scale testing

The goal of this procedure is to create two custom properties that you bind in parent-child relationship. With the binding, you can select the appropriate location based on the selected production state.

Prerequisites

- For this example, create a vRealize Orchestrator action that provides data center names as location information. Name the action datacenters_prod, add an input parameter named prod as a string type, and use this sample script for the action script.

  ```
  if(prod == null) {
      return ['Empty1', 'Empty2'];
  } else if (prod.equals('nonprod')) {
      return ['WestDC for development testing', 'EastDC for QA automation testing', 'CentralDC for
  scale testing'];
  } else {
      return ['NorthDC for AMEA clients', 'SouthDC for Asia Pacific clients'];
  }
  ```

  For information about developing workflows, and about creating and using vRealize Orchestrator script actions, see *Developing with VMware vRealize Orchestrator* in vRealize Orchestrator product documentation.

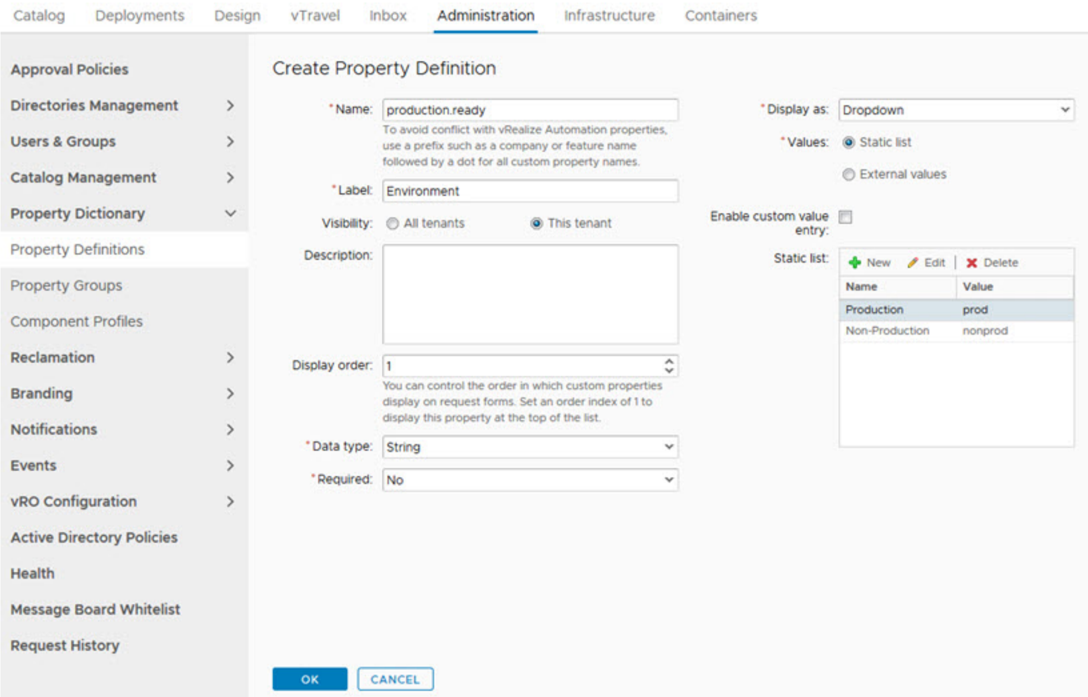
- Log in to vRealize Automation as a **tenant administrator** or **fabric administrator**.

Procedure

1. Create a custom property definition so that users can select production or non-production environment.

   a. Select **Administration > Property Dictionary > Property Definitions**.

   b. Configure the production.ready custom property.



Create Property Definition

*Name: production.ready

To avoid conflict with vRealize Automation properties, use a prefix such as a company or feature name followed by a dot for all custom property names.

*Label: Environment

Visibility: ○ All tenants ● This tenant

Description:

Display order: 1

You can control the order in which custom properties display on request forms. Set an order index of 1 to display this property at the top of the list.

*Data type: String

*Required: No

*Display as: Dropdown

*Values: ● Static list ○ External values

Enable custom value entry: ☐

Static list:

Name	Value
Production	prod
Non-Production	nonprod

Option	Example Values
Name	`production.ready`
Label	`Environment`
Description	`Select the production or non-production environment.`
Display order	`1` You select 1 to ensure that this custom property appears first in the blueprint.
Data type	String
Display as	Dropdown
Values	Static list
Static list values	Add the following key-pair pairs. ■ `Production` and `prod` ■ `Non-Production` and `nonprod`

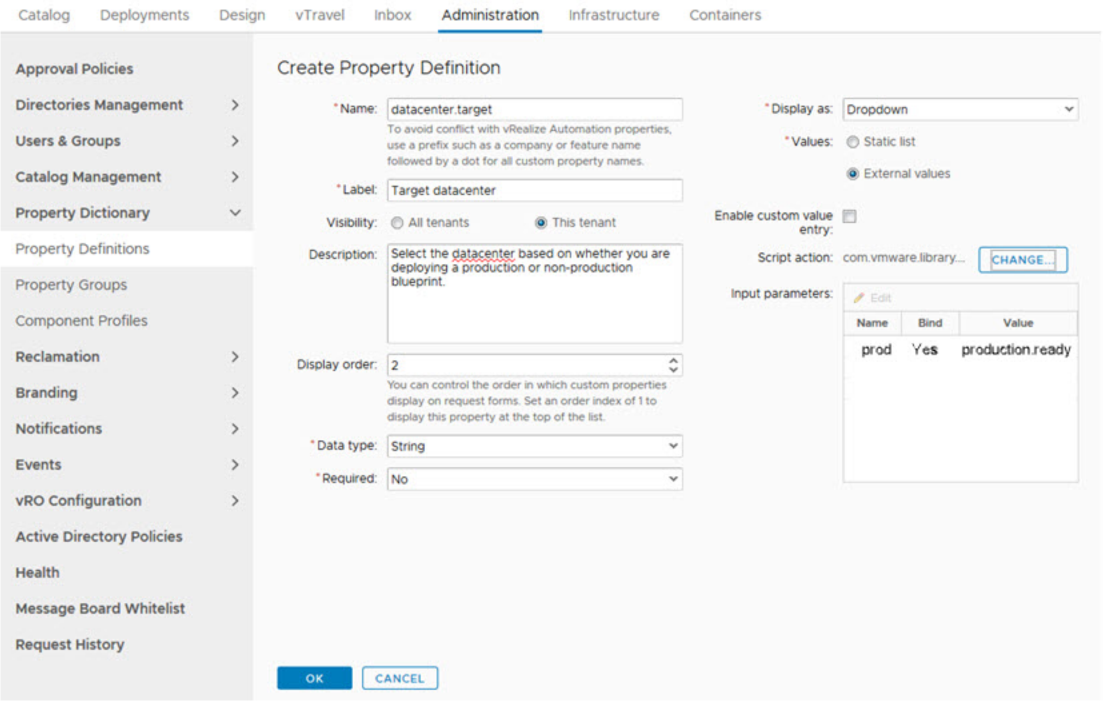
   c. Click **OK**.

The `production.ready` custom property is configured and ready to use.

2 Create a vRealize Orchestrator action custom property definition that runs your custom location action.

   a Select **Administration > Property Dictionary > Property Definitions**.

   b Configure the datacenter.target custom property.



Create Property Definition

*Name: datacenter.target
To avoid conflict with vRealize Automation properties, use a prefix such as a company or feature name followed by a dot for all custom property names.
*Label: Target datacenter
Visibility: ○ All tenants ● This tenant
Description: Select the datacenter based on whether you are deploying a production or non-production blueprint.
Display order: 2
You can control the order in which custom properties display on request forms. Set an order index of 1 to display this property at the top of the list.
*Data type: String
*Required: No
*Display as: Dropdown
*Values: ○ Static list ● External values
Enable custom value entry: ☐
Script action: com.vmware.library...
Input parameters:

Name	Bind	Value
prod	Yes	production.ready

Option	Example Values
Name	`datacenter.target`
Label	`Target data center`
Description	`Select the datacenter based on whether you are deploying a production or non-production blueprint.`
Display order	2 You select 2 to ensure that this custom property is listed after the `production.ready` custom property in the blueprint.
Data type	String
Display as	Dropdown
Values	External values
Script action	Click **Select** and locate your datacenters_prod action.

   The input parameters table includes a prod parameter.

   c In the Input parameters table, select the prod row and click **Edit**.

   d Select the **Bind** check box.

   e Select **production.ready** in the drop-down menu.

f Click **OK**.

g Click **OK**.

The `datacenter.target` custom property is configured and ready to use.

What to do next

- Because of the relationship between the two property definitions, add the two property definitions to a property group. See Create a Property Group.
- Add your production-datacenter property group to a blueprint. See Add a Custom Property or Property Group Using the Properties Tab on a Blueprint's Machine Component.

Add a Custom Property to a Blueprint

You can add custom properties to many parts of vRealize Automation, including approval policies, business groups, endpoints, and reservation policies. However, only the machine blueprints support the display options that you configure as property definitions. Adding a custom property to a blueprint as a simple way to verify that the custom property appears in the user interface as you designed it in the property definition.

Some custom properties are associated with the virtual machine blueprint on the **Properties** tab and some are on the **Network** tab.

- Add a Custom Property or Property Group Using the Properties Tab on a Blueprint's Machine Component

  You add a custom property as a machine custom property so that service catalog users can select of configure the values when they request the item. You can add individual properties or property groups.

- Add a Custom Property Using the Network Tab on a Blueprint's Machine Component

  Add a custom property as a network custom property so that service catalog users can select the necessary network profile value when they deploy the blueprint.

Add a Custom Property or Property Group Using the Properties Tab on a Blueprint's Machine Component

You add a custom property as a machine custom property so that service catalog users can select of configure the values when they request the item. You can add individual properties or property groups.

In this workflow, you add the custom properties to validate that they are working as expected in blueprints. You can also add custom properties to business groups, approval policies, and other components.

Prerequisites

- Verify that you created the required property definition. See Create Custom Property Definitions.

- If you are adding a property group, verify that you added the relevant property definitions to a property group. See Create a Property Group. To test the visual functions of the property definitions, you must select **Show in request** when you add the property to the group.
- If you are adding a vRealize Orchestrator action as a custom property, review the configuration details to ensure that you addrf the custom property in the correct location. See Configuration Details for the vRealize Orchestrator Action Custom Property Definitions.
- Verify that you created the blueprint to which you are adding the custom property. See *Configuring vRealize Automation*.
- Log in to vRealize Automation as an **infrastructure architect**.

Procedure

1. Select **Design > Blueprints**.
2. Select the blueprint to which you are adding the custom property and click **Edit**.
3. Click the target machine component.

   The configuration options for the virtual machine appear on the canvas.
4. Click the **Properties** tab, and then click the **Custom Properties** tab or the **Property Groups** tab.
   - To add a custom property, click **New** and select the property definition in the drop-down menu.

Option	Description
Name	Name of the selected custom property definition.
Value	(Optional) Enter a default value.
Encrypted	When adding custom properties that run vRealize Orchestrator actions, do not encrypt the value.
Overridable	Select this option to ensure that the requesting user can select a value on the request form.
Show in request	Select this option to ensure that the requesting user can see the property and select a value on the request form.

   - To add a property group, click **Add** and select the group.
5. Click **OK**.

   The custom property is added to the blueprint.
6. Click **Finish**.
7. Publish the finished blueprint.

Results

The blueprint includes the custom property.

What to do next

Test the custom property in the request form. See Verify the Custom Property in the Catalog Request Form.

Add a Custom Property Using the Network Tab on a Blueprint's Machine Component

Add a custom property as a network custom property so that service catalog users can select the necessary network profile value when they deploy the blueprint.

Prerequisites

- Verify that you have the required custom property definition. See Create a vRealize Orchestrator Action Custom Property Definition.
- If you are adding a vRealize Orchestrator action as a custom property, review the configuration details to ensure that you added the custom property in the correct location. See Configuration Details for the vRealize Orchestrator Action Custom Property Definitions.
- Verify that you created the blueprint to which you are adding the custom property. See *Configuring vRealize Automation*.
- Log in to vRealize Automation as an **infrastructure architect**.

Procedure

1 Select **Design > Blueprints**.

2 Select the blueprint that you want to edit.

The blueprint opens in the design canvas.

3 In the design canvas, click the virtual machine component that you want to edit.

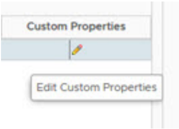
The configuration options for the virtual machine appear on the canvas.

4 Click the machine component's **Network** tab.

5 Click **New** to add a new network row.

6 In the new row, select a network and an assignment type (static IP or DHCP), specify an address if using a static IP, and click **OK**.

7 In the new row, click the Edit icon in the Custom Properties column to assign a custom property.



8 Click **New**, select the custom property, configure the options described in the following table, and click **OK**.

Option	Description
Name	Select an existing custom property name from the drop-down menu.
Value	(Optional) Enter a default value.
Encrypted	When adding custom properties that run vRealize Orchestrator actions, do not encrypt the value.
Overridable	Select this option to ensure that the requesting user can select a value on the request form.
Show in request	Select this option to ensure that the requesting user can see the property and select a value on the request form.

The network, with its configured custom property, is added to the blueprint.

9 Click **Finish**.

10 Publish the finished blueprint.

Results

The blueprint includes the custom property.

What to do next

Test the custom property in the request form. See Verify the Custom Property in the Catalog Request Form

Verify the Custom Property in the Catalog Request Form

As creator of the custom property definitions that run vRealize Orchestrator actions, you should test your custom properties to ensure that the correct values appear in the request form.

Prerequisites

- Add the custom property to the appropriate location in the blueprint. See Add a Custom Property to a Blueprint.
- Verity that the blueprint is entitled to you so that you can test the custom properties in the blueprint. See *Configuring vRealize Automation*.
- Log in to the vRealize Automation as a user with access to the test blueprint.

Procedure

1 Click **Catalog** to display the catalog items that you are entitled to use.

Published blueprints appear on the Catalog page as catalog items.

2 Click **Request** on the catalog item.

3 On the request form, click the machine to which you added the custom property.

4. On the machine's **Properties** tab, select the custom property and click the drop-down arrow.

   The vRealize Orchestrator action runs and retrieves the values it is configured to display. Verify that the expected values appear.

What to do next

Add the custom property to your production blueprints where needed.

Configuration Details for the vRealize Orchestrator Action Custom Property Definitions

You create custom property definitions that run vRealize Orchestrator actions to retrieve key value pairs from external files or from vRealize Automation configuration information. You add the custom properties to blueprints so that they appear in the catalog request forms.

The service catalog user requesting the item can select a value to include in the deployment. When the user clicks the drop-down menu to select a value, the vRealize Orchestrator action runs, retrieving the data that is displayed in the menu for the user to select.

The configuration workflows for each vRealize Orchestrator actions property definition are similar, but some of the details vary. For example, there are differences in prerequisites and limitations, and where you apply the custom property in the blueprint might vary.

- Network Custom Property Definition

  You add a custom property to retrieve network names from the vRealize Automation database when you want users to select the network in the request form. The network selector custom property uses a vRealize Orchestrator action to retrieve the values.

- Reservation Policy Custom Property Definition

  You add a custom property definition to retrieve reservation policy names that are applicable to the requesting users when they select the policy in the request form. The reservation policy selector custom property definition uses a vRealize Orchestrator action to retrieve the values.

- PowerShell Script Custom Property Definition

  You add a custom property to run a PowerShell script when you want to use a script to retrieve data to populate the custom property in the request form. The PowerShell script custom property uses a vRealize Orchestrator action to run the script and retrieve the values.

- Database Query Custom Property Definition

  You add a custom property to query a database when you want to retrieve values from that database to populate the custom property on the request form. The database custom property uses a vRealize Orchestrator action to run the query and retrieve the values.

- Custom Action Custom Property Definition

  You add a custom property to retrieve data from a source using a custom vRealize Orchestrator action when you want users to select the retrieved values in the request form.

Network Custom Property Definition

You add a custom property to retrieve network names from the vRealize Automation database when you want users to select the network in the request form. The network selector custom property uses a vRealize Orchestrator action to retrieve the values.

Limitations

Plan for the following limitations when you use the network selector custom property.

- The name of the custom property must be `VirtualMachine.Network0.Name`. This name is required. You cannot create multiple property definitions for the network selector.
- The action retrieves all the network names for the requesting user without validating that it applies to the target vCenter Server instance. A service catalog user might select a network that is not applicable to the selected target. If the wrong network is selected, the catalog request fails.
- The action retrieves network names for the requesting user only. If you submit a request on behalf of other users, the networks are for you. For example, Network A and Network C are associated with Business Group 1, so the BG 1 users see only Network A and C, not B.

Prerequisites

If you use an external vRealize Orchestrator server, verity that it is set up correctly. See *Configuring vRealize Automation*.

Custom Property Configuration Values

You use these options to create the custom property.

Table 1-49. Network Custom Property Configuration Values

Option	Value
Name	You must use `VirtualMachine.Network0.Name`. See Custom Properties V.
Data type	String
Display as	Dropdown
Values	External
Action folder	com.vmware.vra.networks
Script action	getApplicableNetworks This script action is an example script. You can create specific actions for your environment.
Input parameters	No required parameters.

Blueprint Configuration

Add the custom property on the blueprint **Network** tab. See Add a Custom Property Using the Network Tab on a Blueprint's Machine Component.

Reservation Policy Custom Property Definition

You add a custom property definition to retrieve reservation policy names that are applicable to the requesting users when they select the policy in the request form. The reservation policy selector custom property definition uses a vRealize Orchestrator action to retrieve the values.

Limitations

Plan for the following limitations when you use the reservation policy selector custom property.

- The name of the custom property must be ReservationPolicyID. This name is required. You cannot create multiple property definitions for the reservation policy selector.
- The action retrieves all reservation policies applicable to the requesting user without validating that it applies to the target endpoint, for example a vCenter Server instance or some other platform. A service catalog user might select a reservation that is not applicable to the selected blueprint target system. If the user selects the wrong reservation, the catalog request fails.
- The action retrieves reservation policies for the requesting user only. If you submit a request on behalf of another user, the reservation policies are for you. For example, Reservation 1 and Reservation 3 are associated with Business Group 1, so the BG 1 users see only Reservations 1 and 3, not 2.

Prerequisites

If you use an external vRealize Orchestrator server, verity that it is set up correctly. See *Configuring vRealize Automation*.

Custom Property Configuration Values

You use these options to create the custom property.

Table 1-50. Reservation Policy Custom Property Configuration Values

Option	Value
Name	You must use ReservationPolicyID.
Data type	String
Display as	Dropdown
Values	External
Action folder	com.vmware.vra.reservations
Script action	getApplicableReservationPolicies This script action is an example script. You can create specific actions for your environment.
Input parameters	No required parameters.

Blueprint Configuration

You can add a custom property to the blueprint **Properties** tab to associate the property with the overall blueprint.

PowerShell Script Custom Property Definition

You add a custom property to run a PowerShell script when you want to use a script to retrieve data to populate the custom property in the request form. The PowerShell script custom property uses a vRealize Orchestrator action to run the script and retrieve the values.

For example, as the cloud administrator you have a PowerShell script that retrieves user IDs from the Active Directory that is registered with vRealize Automation. The intent of the script is to retrieve and display John Smith when the actual value in Active Directory is JSmith01.

An advantage to using the PowerShell script action includes a central location for the script. You can either store the script on a central server and then run it on target virtual machines, or you can store it in vRealize Orchestrator and then run it on the target machines. A central location decreases maintenance time. Storing the scripts in vRealize Orchestrator when you have backup and restore configured ensures that you can restore the scripts if a system failure occurs.

Prerequisites

Verify that you have a working PowerShell script that returns key pair values. The script must be available on an accessible server or that the script is uploaded into vRealize Orchestrator.

Custom Property Configuration Values

You use these options to create the custom property.

Table 1-51. PowerShell Script Custom Property Configuration Values

Option	Value
Name	You can use any string.
Data type	String
Display as	Dropdown
Values	External
Action folder	com.vmware.vra.powershell

Table 1-51. PowerShell Script Custom Property Configuration Values (continued)

Option	Value
Script action	Select an action based on where the PowerShell script is located. ■ If the PowerShell script is on a central server, use executeExternalPowerShellScriptOnHostByName. ■ If the PowerShell script is uploaded into vRealize Orchestrator, use executePowershellScriptFromResourceOnHostByName. These script actions are example scripts. You can create specific actions for your environment. The `Resources/Sample/vRA/PowerShell/countries.ps1` sample PowerShell script is provided in the vRealize Orchestrator client as reference for use with the executePowershellScriptFromResourceOnHostByName action.
Input parameters	Configure the input parameters based on the selected action. Define parameters ■ If you use executeExternalPowerShellScriptOnHostByName: ■ **hostName**. Name of the central server where the script is located. ■ **externalPowershellScript**. Path to the PowerShell file on the host. ■ **Arguments**. Parameters to pass to the script. You separate the arguments with commas. For example, Argument1,Arguement2. ■ If you use executePowershellScriptFromResourceOnHostByName: ■ vRealize Orchestrator. Name of the vRealize Orchestrator instance you are using as the host. ■ **scriptResourcePath**. Path to the PowerShell file on the host. ■ **scriptResourceName**. Path to the PowerShell file as an uploaded resource in vRealize Orchestrator.

Blueprint Configuration

You can add a custom property to the blueprint **Properties** tab to associate the property with the overall blueprint.

Database Query Custom Property Definition

You add a custom property to query a database when you want to retrieve values from that database to populate the custom property on the request form. The database custom property uses a vRealize Orchestrator action to run the query and retrieve the values.

The action is supported for the following databases:

- Microsoft SQL Server
- MySQL
- Oracle
- PostgreSQL

Limitations

All retrieved values are converted to strings.

Prerequisites

Verify that the vRealize Orchestrator SQL Plug-In is installed and configured to connect to the target database.

Custom Property Configuration Values

You use these options to create the custom property.

Table 1-52. Database Query Custom Property Configuration Values

Option	Value
Name	You can use any string.
Data type	String
Display as	Dropdown
Values	External
Action folder	com.vmware.vra.sql
Script action	executeSQLSelectOnDatabase This script action is an example script. You can create specific actions for your environment.
Input parameters	■ **databaseName**. Name of the database to which vRealize Orchestrator is connected. ■ **sqlSelectQuery**. The SQL select query that you are running on the database to retrieve the values. For example, select * <table name>. ■ **keyColumnName**. Name of the database column that is the key for the key pair value. ■ **valueColumnName**. Name of the database column from which you are retrieving values.

Blueprint Configuration

You can add a custom property to the blueprint **Properties** tab to associate the property with the overall blueprint.

Custom Action Custom Property Definition

You add a custom property to retrieve data from a source using a custom vRealize Orchestrator action when you want users to select the retrieved values in the request form.

Limitations

The supported scripted actions include:

- Any and Array/Any
- Array/String and Array/Properties if you select the String data type in the definition form
- Array/Number if you select the Integer or Decimal data type in the definition form

Prerequisites

Verify that you have a working vRealize Orchestrator action. For information about developing workflows and creating and using vRealize Orchestrator script actions, see *Developing with VMware vCenter Orchestrator* .

The action script must accept the input parameter values. You can configure the values as key value pairs. You can present user-readable names for less friendly identifiers using key value pairs.

Custom Property Configuration Values

You use these options to create the custom property.

Table 1-53. Custom Action Custom Property Configuration Values

Option	Value
Name	You can use any string.
Data type	Decimal, Integer, or String
Display as	Dropdown
Values	External
Action folder	Location of your custom action.
Script action	Name of your custom action.
Input parameters	Depends on your custom action.

Blueprint Configuration

Usually you add the custom property on the blueprint Properties tab. Whether you add it to the Properties tab depends on your action. See Add a Custom Property to a Blueprint.

Using Property Groups

You can create property groups to collect properties into a single unit.

Property groups are logical and reusable groups of properties, that can include property definitions that you create or custom properties that are supplied. Property groups are designed to simplify the process of adding properties to blueprints or other vRealize Automation elements for which they are available. They provide a means by which logical groupings of properties can be added more efficiently than by adding the properties individually.

A property group typically contains properties that are commonly used together. For example, you can create a property group named WimImagingProperties that contains properties commonly used for WIM-based provisioning:

- `Image.ISO.Location`
- `Image.ISO.Name`
- `Image.Network.Password`
- `Image.Network.User`

- `Image.WIM.Index`
- `Image.WIM.Name`
- `Image.WIM.Path`

You can also create a property group for vCloud Air or vCloud Director machine provisioning that contains the following properties:

- `VirtualMachine.Network0.Name`
- `VCloud.Template.MakeIdenticalCopy`
- `VMware.SCSI.Type`
- `Sysprep.Identification.DomainAdmin`
- `Sysprep.Identification.DomainAdminPassword`
- `Sysprep.Identification.JoinDomain`

You can use the `vra content list --type property-definition` vRealize CloudClient command to list all property definitions in the current vRealize Automation tenant. You can also use the `vra content list --type property-group` vRealize CloudClient command to list all property groups.

Create a Property Group

You can organize specific custom properties into property groups to more easily add multiple custom properties to blueprints.

Prerequisites

Log in to vRealize Automation as a **tenant administrator** or **fabric administrator**.

Procedure

1 Select **Administration > Property Dictionary > Property Groups**.

2 Click **New** (✚).

3 Enter the new property group name and ID.

   If you enter the **Name** value first, the **ID** text box is populated with the same value.

4 In the **Visibility** section, select **All tenants** or **This tenant** to determine where the property is to be available.

   If you are logged in with only tenant administrator privileges, then only **This tenant** is available. If you are logged in with only fabric administrator privileges, then only **All tenants** is available.

   You cannot change the **All tenants** or **This tenant** setting after you create the item.

5 (Optional) Enter a description of the property group, for example `My Cloning Properties vSphere`.

6 Click **New** and add a property to the group.

Option	Description
Name	Add a new property or select an existing property from the drop-down menu. For example, enter `VirtualMachine.Storage.ReserveMemory`.
Value	(Optional) Enter a default property value. For example, enter `True`.
Encrypted	Select this option to specify that the property value be encrypted. For example, if the value is to be a password or other secure entry, using the encrypted option hides the value characters. When adding custom properties that run vRealize Orchestrator actions, do not encrypt the value.
Show in request	Select this option to specify that the requesting user can see the property and select a value on the request form when requesting machine provisioning.

7 Click **OK** to add the property to the group.

8 Add additional properties to the group.

9 Click **OK**.

Defining Component Profile Settings

You can use component profiles to configure advanced property management capabilities in vRealize Automation blueprints. Deployers can then use the `Size` and `Image` component profiles on a blueprint to select pre-defined value sets.

You can use the `Size` and `Image` component profiles, and their specified value sets, to map to a logical grouping such as Small, Medium, and Large or Dev, Test, and Production. By using these settings, you can reduce the number of blueprints that you need to maintain.

A component profile defines settings for a vSphere machine component in a blueprint. For example, you might define a component profile for a small size virtual machine deployment. You might define another component profile for a large size machine deployment. You can use vRealize Automation to define the following component profile types:

- Size

  See Configure Component Profile Size Settings for Catalog Deployments.

- Image

  See Configure Component Profile Image Settings for Catalog Deployments.

You can define multiple named value sets within the `Size` and `Image` component profile types and add one or more of the value sets to machine components in a blueprint. Each value set that you define for the component profile type contains the following configurable settings:

- Name that requesters see when they provision a machine
- Unique identifier for tenant

- Description
- Set of value choices for each option in the value set

You cannot define other component profile types.

When you request provisioning, you can select from available `Size` and `Image` options. When you choose one of the value sets, the corresponding property values are bound to the request.

Configure Component Profile Image Settings for Catalog Deployments

You can configure the component profile `Image` setting to control build information for vSphere machine components in the blueprint.

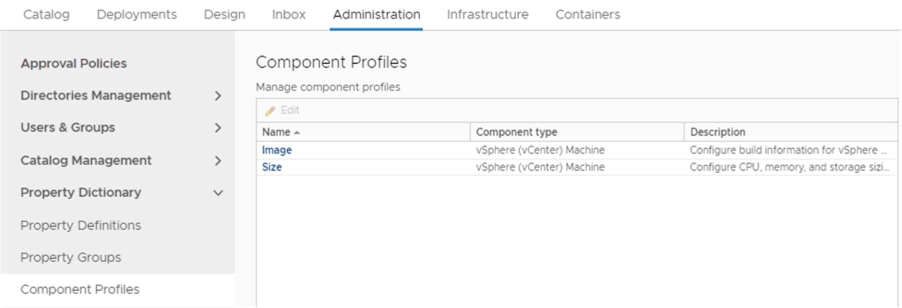
After you define value sets for the `Image` component profile, you can add one or more value sets to the component profile for a vSphere machine component in a blueprint. Users can then select an `Image` value set when they request a catalog item.

Prerequisites

Log in to vRealize Automation as an administrator with **tenant administrator** and **IaaS administrator** access rights.

Procedure

1 Select **Administration > Property Dictionary > Component Profiles**.



Component Profiles

Manage component profiles

Name ▲	Component type	Description
Image	vSphere (vCenter) Machine	Configure build information for vSphere ...
Size	vSphere (vCenter) Machine	Configure CPU, memory, and storage sizi...

2 Click **Image** in the Name column.

Information about the supplied image component property is displayed.

3 Click the **Value Sets** tab.

4 To define a new value set click **New** and configure the `Image` settings.

   a Enter a value in the **Display name** field to append to the ValueSet delimeter, for example `CloneA`.

   b Accept the default value shown in the **Name** text box, for example **ValueSet.CloneA**, or enter a custom name.

   c Enter a description such as `Build settings for cloning scenario A` in the **Description** text box.

d Select **Active** or **Inactive** in the **Status** drop down menu.

Select **Active** to allow the value set to be visible in the catalog provisioning request form.

e Select **Server** or **Desktop** as the blueprint type.

f Select the build action to use for this value set, for example select **Clone**.

Other action include:

- **Create**
- **Clone**
- **Linked Clone**
- **NetApp FlexClone**

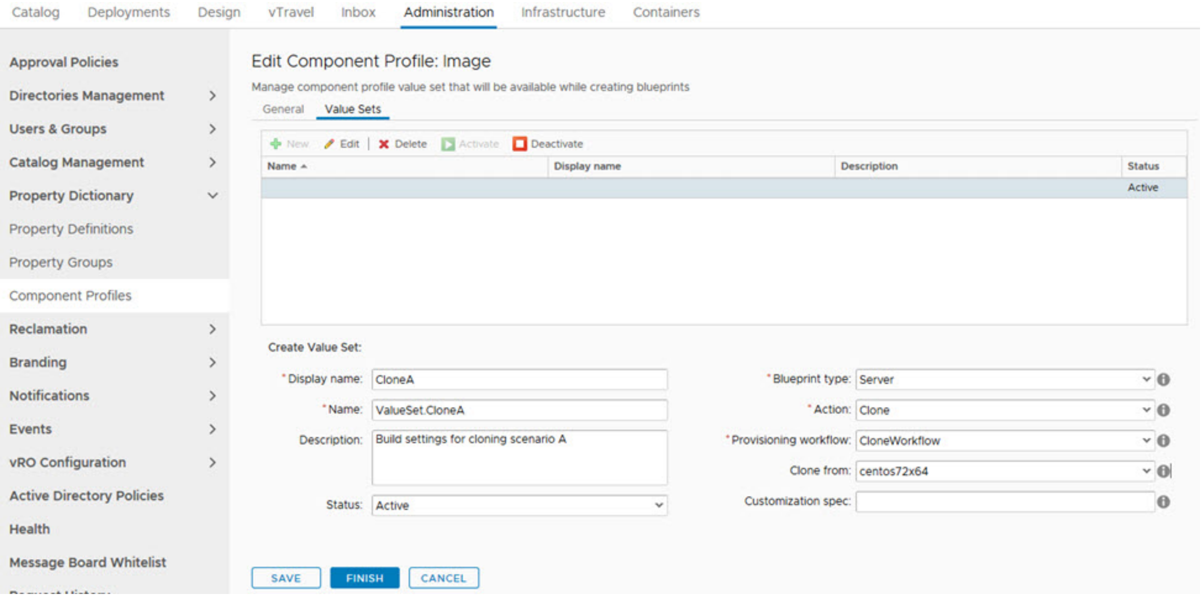
g Select the CloneWorkflow provisioning workflow.

h (Optional) Select a source machine to clone from, for example **centos7264**.

i (Optional) Enter the path to a vSphere customization specification.

5 Click **Save**.

6 When you are satisfied with your settings, click **Finish**.



Edit Component Profile: Image

Manage component profile value set that will be available while creating blueprints

Name	Display name	Description	Status
			Active

Create Value Set:

*Display name: CloneA

*Name: ValueSet.CloneA

Description: Build settings for cloning scenario A

Status: Active

*Blueprint type: Server

*Action: Clone

*Provisioning workflow: CloneWorkflow

Clone from: centos72x64

Customization spec:

What to do next

Add one or more value sets to the `Image` component profile by using the **Profiles** tab on a vSphere machine component. See *Configuring vRealize Automation*.

Configure Component Profile Size Settings for Catalog Deployments

You can configure the component profile `Size` setting to specify CPU, memory, and storage sizing for vSphere machine components in the blueprint.

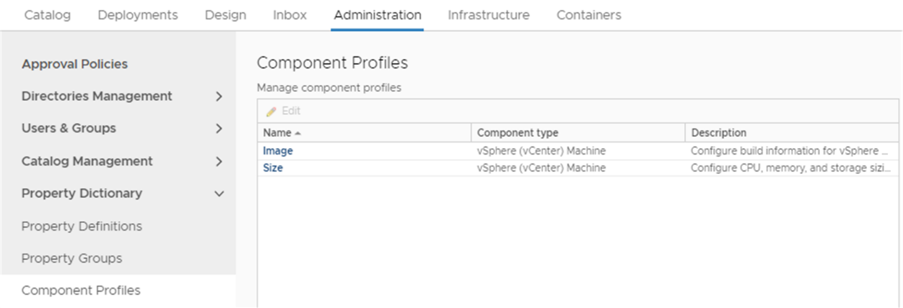
After you define value sets for the `Size` component profile, you can add one or more value sets to a component profile for a vSphere machine component in a blueprint. Users can then select a `Size` value set when they request a catalog item.

Prerequisites

Log in to vRealize Automation as an administrator with **tenant administrator** and **IaaS administrator** access rights.

Procedure

1 Select **Administration > Property Dictionary > Component Profiles**.



Name	Component type	Description
Image	vSphere (vCenter) Machine	Configure build information for vSphere ...
Size	vSphere (vCenter) Machine	Configure CPU, memory, and storage sizi...

2 Click **Size** in the Name column.

Information about the supplied `Size` component profile is displayed on the **General** tab.

3 Click the **Value Sets** tab.

4 To define a new value set, for example, for a large size deployment, click **New** and configure the `Size` settings.

   a Enter a value in the **Display name** field to append to the ValueSet delimeter, for example **`small_1`**.

   b Accept the default value shown in the **Name** text box as **ValueSet.small_1** or enter a custom name.

   c Enter a description such as **`small deployment`** in the **Description** text box.

   d Select **Active** or **Inactive** in the Status drop down menu.

   Select **Active** to allow the value set to be visible in the catalog provisioning request form.

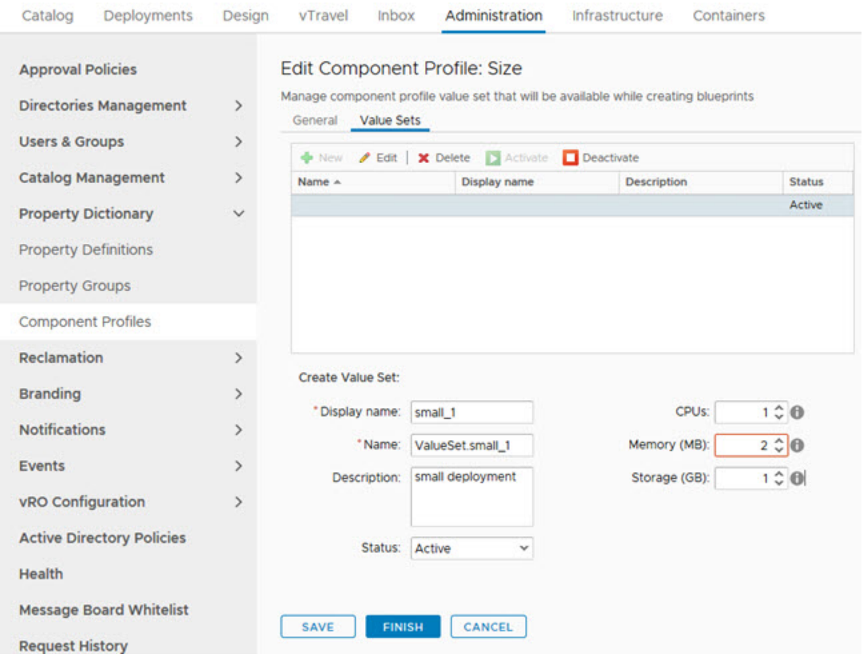
   e Enter the number of virtual CPUs on which the deployment can be run, for example 1.

   f Enter the amount of RAM to be used by virtual machines in the deployment, for example 2 MB.

   g Enter the amount of storage to be used by virtual machines in the deployment, for example 1 GB.

5 Click **Save**.

6 When you are satisfied with your settings , click **Finish**.



Edit Component Profile: Size

Manage component profile value set that will be available while creating blueprints

Create Value Set:

*Display name: small_1

*Name: ValueSet.small_1

Description: small deployment

Status: Active

CPUs: 1

Memory (MB): 2

Storage (GB): 1

What to do next

Add one or more value sets to the `Size` component profile by using the **Profiles** tab on a vSphere machine component. See *Configuring vRealize Automation*.