vRealize Automation Extensibility Migration Guide

TECHNICAL WHITE PAPER

Table of Contents

Revision History

Date	Version	Description
January 2017	1.0	Initial version

Validated with product versions:

Product	Version
vRealize Automation	6.2.x, 7.1, 7.2
vRealize Orchestrator	6.0.x, 7.1, 7.2

Introduction

This document outlines the changes that were introduced in the extensibility feature between vRealize Automation 6.x and 7.x. It also describes the steps you must take to adapt the extensibility use cases and workflows, so that they continue working as expected after the upgrade to vRealize Automation 7.x.

In this document, you can find description of:

- How to update your existing extensibility implementation in order to work with vRealize Automation 7.x.
- How to migrate extensibility solution to use the new mechanism provided by vRealize Automation Event Broker Service.

This document complements the vRealize Automation installation and configuration documentation, available in the vRealize Automation Information Center.

Extensibility Concepts

vRealize Automation, previously known as vCloud Automation Center, is designed to be extensible, so that it can simplify and accelerate the integration with other infrastructure tools and processes. vRealize Automation provides several extensibility options that cover a variety of use cases.

One of the most common extensibility use cases is extending or customising the machine provisioning process by using vRealize Orchestrator. With the Orchestrator plug-ins, you can integrate vRealize Automation with third-party management systems, such as Active Directory (AD), IP address management (IPAM), Customer Relationship Management (CRM), or others.

Extensibility in vRealize Automation 6.x

vCloud Automation Center 6.x introduces the extensibility kit using the vRealize Orchestrator vCloud Automation Center plug-in. You can customize the life cycle of machine provisioning through vRealize Orchestrator.

For more information about installing and configuring the vCloud Automation Center extensibility kit, see Extensibility in vCloud Automation Center 6.x.

Extensibility in vRealize Automation 7.x

As a new feature in vRealize Automation 7.0 the Event Broker Service replaces the current extensibility mechanism. You can subscribe to different events in vRealize Automation, such as blueprint life cycle events, approval events, system log events, and others.

NOTE The vRealize Orchestrator vCloud Automation Center 6.x plug-in is deprecated in vRealize Automation 7.0. For this reason, it is best that you migrate all your extensibility use cases to the event broker solution.

Event Broker Details

To use the vRealize Automation extensibility with Event Broker, you subscribe to a topic and select one or more vRealize Orchestrator workflows that run based on specified criteria.

vRealize Automation 7.0 includes a predefined set of topics:

- Post-Approval
- Pre-Approval
- EventLog Default Event
- Blueprint configuration
- Resource Reclamation Completion Event
- Business configuration
- Orchestration server configuration (XaaS) - obsolete
- Machine lifecycle
- Machine provisioning

vRealize Automation 7.1 includes the topics of the previous version and the following additional topics:

- Container Operations
- IPAM IP allocation/deallocation completion
- Orchestration server configuration
- Provision Container

vRealize Automation 7.2 includes the topics of the previous version and the following additional topics:

- Blueprint component completed
- Blueprint component requested
- Catalog item request completed
- Catalog item requested
- Component action completed
- Component action requested
- Deployment action completed
- Deployment action requested
- IPAM IP lifecycle event completion

With the provided topics, you can implement a variety of new use cases that are impossible with the vRealize Orchestrator vCloud Automation Center 6.x plug-in.

Extend vRealize Automation Machine Provisioning by Using Event Broker

The following procedure provides an example of using Event Broker to extend the machine provisioning.

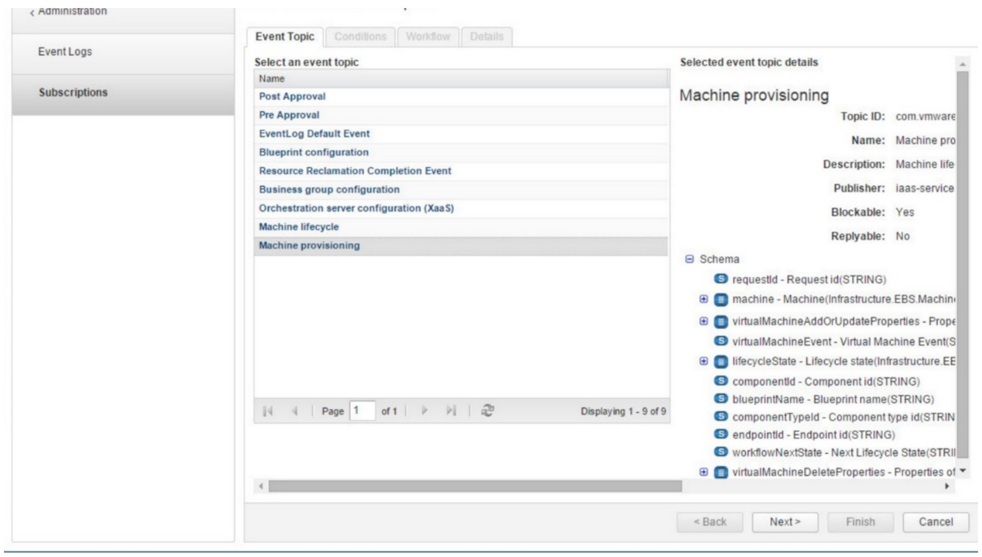
Prerequisites

- Log in to the vRealize Automation console as a tenant administrator.
- Select **Administration** > **Events** > **Subscriptions** > **New**.

Procedure

1. In the **Event Topic** tab under New Workflow Subscription, click **Machine provisioning**.

FIGURE 1 - THE EVENT TOPIC TAB IN THE VREALIZE AUTOMATION 7.X SUBSCRIPTION WIZARD



Select an event topic

Name
Post Approval
Pre Approval
EventLog Default Event
Blueprint configuration
Resource Reclamation Completion Event
Business group configuration
Orchestration server configuration (XaaS)
Machine lifecycle
Machine provisioning

Displaying 1 - 9 of 9

Selected event topic details

Machine provisioning

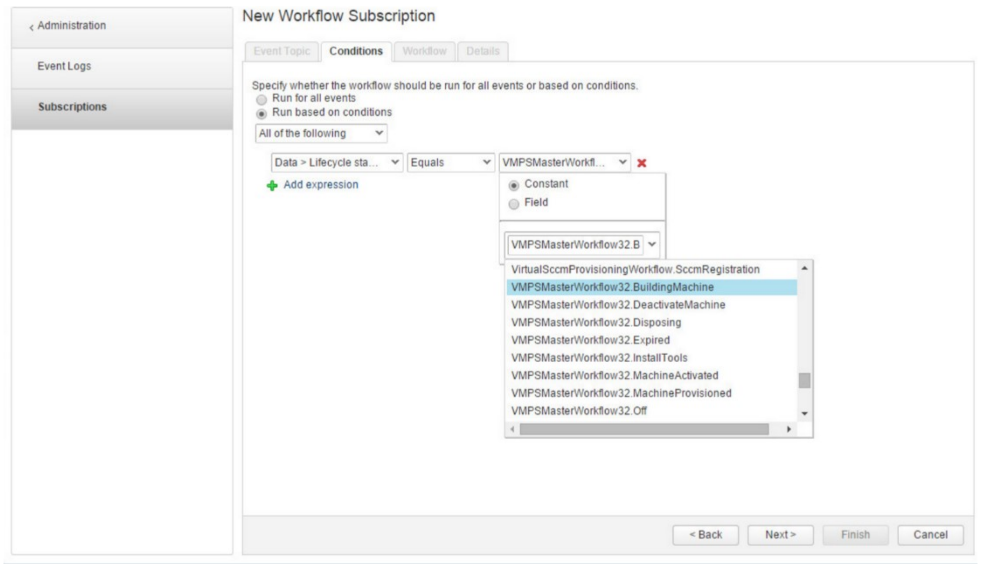
Topic ID:	com.vmware
Name:	Machine pro
Description:	Machine life
Publisher:	iaas-service
Blockable:	Yes
Replyable:	No

Schema
- requestId - Request id(STRING)
- machine - Machine(Infrastructure.EBS.Machin
- virtualMachineAddOrUpdateProperties - Prope
- virtualMachineEvent - Virtual Machine Event(S
- lifecycleState - Lifecycle state(Infrastructure.EE
- componentId - Component id(STRING)
- blueprintName - Blueprint name(STRING)
- componentTypeId - Component type id(STRIN
- endpointId - Endpoint id(STRING)
- workflowNextState - Next Lifecycle State(STRII
- virtualMachineDeleteProperties - Properties of

Every topic has a defined schema that provides the information available for a specific event. The **Machine provisioning** topic gives access to all the data related to the provisioning request, and to the requested machine.

2. In the **Conditions** tab, specify the criteria for triggering a vRealize Orchestrator workflow.

FIGURE 2 - THE CONDITIONS TAB IN VREALIZE AUTOMATION 7.X SUBSCRIPTION WIZARD



Specify whether the workflow should be run for all events or based on conditions.
○ Run for all events
● Run based on conditions
All of the following

Data > Lifecycle sta... | Equals | VMPSMasterWorkfl...
● Constant
○ Field
VMPSMasterWorkflow32.B

VirtualSccmProvisioningWorkflow.SccmRegistration
VMPSMasterWorkflow32.BuildingMachine
VMPSMasterWorkflow32.DeactivateMachine
VMPSMasterWorkflow32.Disposing
VMPSMasterWorkflow32.Expired
VMPSMasterWorkflow32.InstallTools
VMPSMasterWorkflow32.MachineActivated
VMPSMasterWorkflow32.MachineProvisioned
VMPSMasterWorkflow32.Off

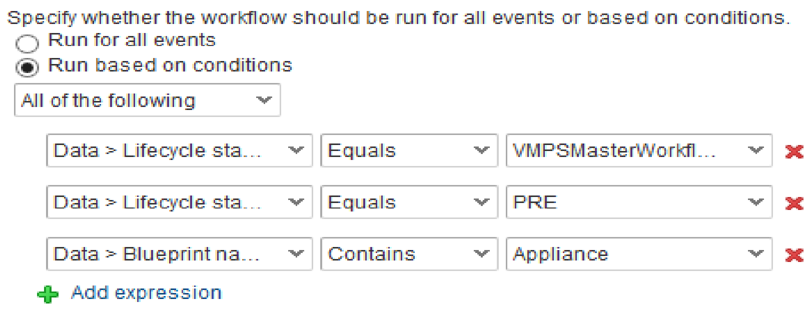
The conditions are based on the following data:

- Data
  - Blueprint name
  - Component ID
  - Component type ID
  - Endpoint ID
  - Lifecycle state
  - Machine
  - Next life cycle state (output)
  - Properties of the virtual machine to be added or updated (output)
  - Properties of the virtual machine to be deleted (output)
  - Request ID
  - Virtual machine event (output)
- Description
- Event topic ID
- Event type
- Event type name
- Source identity
- Source type

- Target ID
- Target type
- Tenant ID
- Time stamp
- User name

You create a subscription to an event by selecting a specific machine life cycle state, event, and phase. The Event Broker supports criteria definitions that are based on pre-events or post-events, so that you can construct more advanced conditions.

FIGURE 3 - A SET OF CONDITIONS TO BUILD CRITERIA UNDER THE CONDITIONS TAB IN VREALIZE AUTOMATION 7.X SUBSCRIPTION WIZARD



Specify whether the workflow should be run for all events or based on conditions.

( ) Run for all events
[✓ Run based on conditions]

All of the following

Data > Lifecycle sta...	Equals	VMPSMasterWorkfl...
Data > Lifecycle sta...	Equals	PRE
Data > Blueprint na...	Contains	Appliance

Master Workflow Life Cycle States have three phases: **PRE**, **POST**, and **EVENT**.

TABLE 1 - ALL STATES FOR MACHINE PROVISIONING TOPIC IN VREALIZE AUTOMATION 7.X

VMPS MasterWorkflow32 State	Pre Action	Post Action
Requested	supported	supported
WaitingToBuild	supported	supported
BuildingMachine	supported	supported
RegisterMachine	supported	supported
MachineProvisioned	supported	supported
MachineActivated	supported	supported

DeactivateMachine	supported	supported
UnprovisionMachine	supported	supported
Disposing	supported	supported

Based on a pre-event or a post-event, you can implement three basic scenarios.

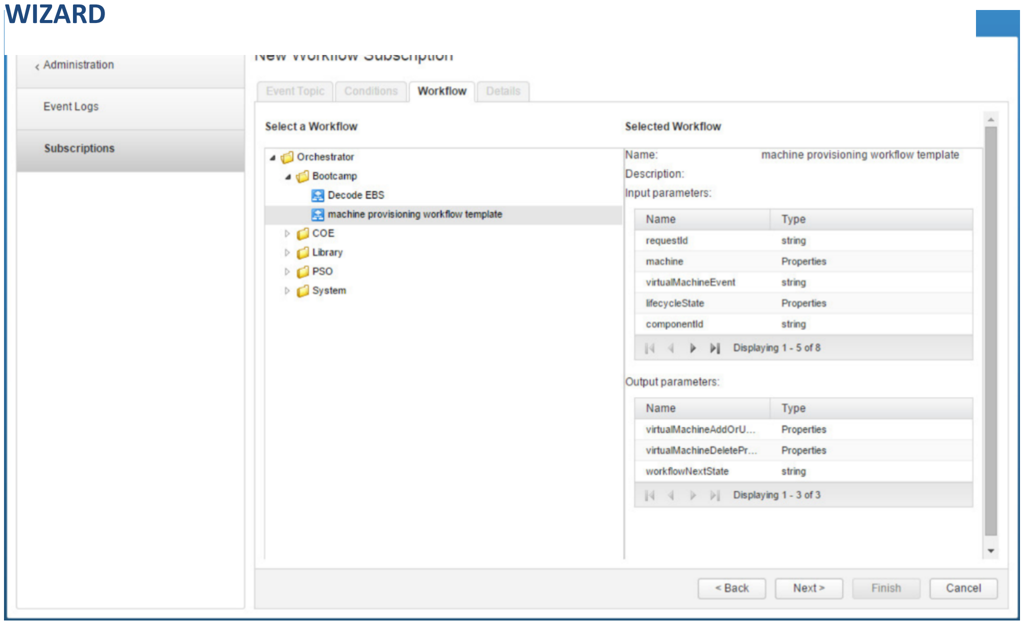
TABLE 2 - BASIC SCENARIOS FOR USING PRE-EVENTS AND POST-EVENTS IN VREALIZE AUTOMATION 7.2

Scenario	Description	Phase	Use Case
Add, update or delete machine properties	Modifies the machine data during the machine provisioning process. Requires subscription to be set to **Blocking**.	PRE-event and POST-event phases	
Set a `workflowNextState` output	Defines the next allowed state of the provisioning process based on an event, for example, a failed operation in a workflow.	PRE-event phase only	You configure an IPAM server to provide a static IP address to a virtual machine. If the request fails, the virtual machine takes a dynamic IP address.
Set `virtualMachineEvent`	Triggers a global IaaS event to manipulate the provisioning master workflow.	PRE-event phase only	You deploy a virtual machine, and you do not want the machine to start when it is deployed.

For more information about events in vRealize Automation, see *VMPS Master Workflow Life Cycle States* in the vRealize Automation 7.2 documentation.

3. Specify the vRealize Orchestrator workflow to be triggered.

FIGURE 4 - THE WORKFLOW TAB IN VREALIZE AUTOMATION 7.X SUBSCRIPTION WIZARD



Name: machine provisioning workflow template

Description:

Input parameters:

Name	Type
requestId	string
machine	Properties
virtualMachineEvent	string
lifecycleState	Properties
componentId	string

Displaying 1 - 5 of 8

Output parameters:

Name	Type
virtualMachineAddOrU...	Properties
virtualMachineDeletePr...	Properties
workflowNextState	string

Displaying 1 - 3 of 3

4. Bind event data to workflow parameters.

The event exposes data, or payload, in the form of a `LiteralMap` object, which contains objects of type `Literal`. The associated vRealize Orchestrator workflow uses input parameters to process this data.

To provide the payload of the event to the workflow, you must map the `Literal` objects to the workflow input parameters by using their names.

For example, if your workflow needs a parameter of type `String`, the event that triggers that workflow must contain a `Literal` object with a key that matches the name of the input parameter.

TABLE 3 - THE MOST TYPICAL MAPPINGS BETWEEN LITERAL OBJECTS AND INPUT PARAMETER TYPES IN VREALIZE AUTOMATION 7.X

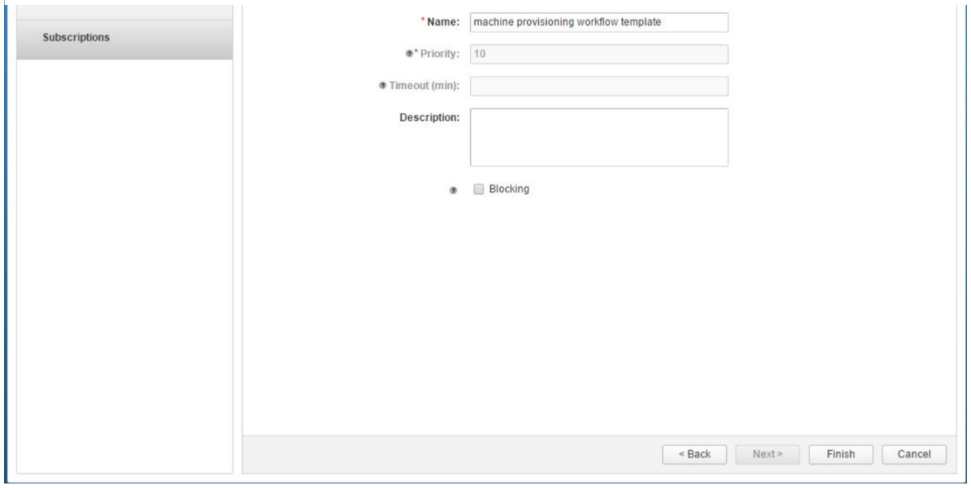
Literal Object	Workflow Parameter Type
StringLiteral	String
SecureStringLiteral	SecureString

MultipleLiteral containing Simple types, for example String	Array/<the simple type, for example String>
MultipleLiteral containing Complex type	Array/Properties
IntegerLiteral	Number
EntityReference	The `vcoType` of the object, for example `VC:VirtualMachine`
DateTimeLiteral	Date
ComplexLiteral	Properties
BooleanLiteral	Boolean

- If the workflow that you use has a single input parameter of type `Properties`, the entire event payload is bound to that parameter. It does not matter if the payload contains a `Literal` object with the same name.
- A `Literal` object that does not match any of the input parameters is added to the workflow run as a global parameter. This means that the input parameters are converted to parameters of type `String`. The name of the string matches the key of the `Literal` object and the value of the string matches the `value.toString()` method of the `Literal` object.
- Input parameters that do not match any incoming `Literal` object, are removed from the workflow run context.
- If a `Literal` object matches an input parameter but is not configured with the correct type, the preparation for the workflow run fails with a `ClassCastException` exception.

5. Specify the subscription details.

FIGURE 5 - THE SUBSCRIPTION DETAILS SCREEN IN VREALIZE AUTOMATION 7.X SUBSCRIPTION WIZARD



* Name: machine provisioning workflow template
* Priority: 10
Timeout (min):
Description:
☐ Blocking

Blocking and Non-Blocking Subscriptions

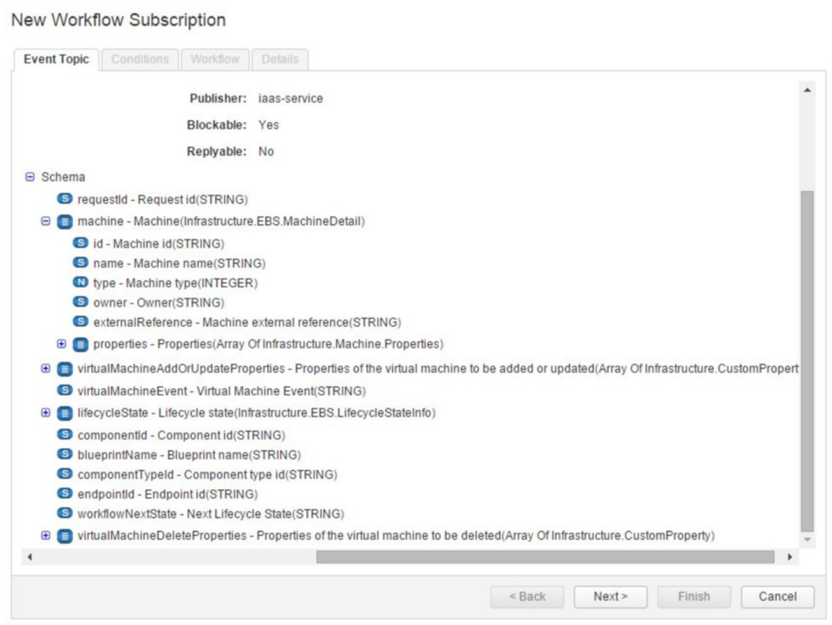
With Event Broker, you can specify whether an event subscription is blocking or non-blocking. A blocking subscription waits for the extensibility workflow to complete.

TABLE 4 - THE DIFFERENCES BETWEEN BLOCKING AND NON-BLOCKING EVENTS IN VREALIZE AUTOMATION 7.X

	Blocking Subscription	Non-Blocking Subscription
Workflow fails	The run fails only if it is expected to fail at that state.	The request succeeds.
Workflow return values	Returns are processed.	Returns are discarded.
Parallelism	Workflows run in a sequence per single request. Workflows run in parallel for multiple requests.	Workflows run in parallel.
Use	Manipulate and edit data during the events.	Runs a task without further intervention.

After you create an event subscription, you customize the payload data for the vRealize Orchestrator workflows. The event topic schema describes the data that is sent to the vRealize Orchestrator workflow.

FIGURE 6 - THE EVENT TOPIC SCHEMA IN VREALIZE AUTOMATION 7.X SUBSCRIPTION WIZARD


New Workflow Subscription

Event Topic | Conditions | Workflow | Details

Publisher: iaas-service
Blockable: Yes
Replyable: No

- Schema
  - requestId - Request id(STRING)
  - machine - Machine(Infrastructure.EBS.MachineDetail)
    - id - Machine id(STRING)
    - name - Machine name(STRING)
    - type - Machine type(INTEGER)
    - owner - Owner(STRING)
    - externalReference - Machine external reference(STRING)
    - properties - Properties(Array Of Infrastructure.Machine.Properties)
  - virtualMachineAddOrUpdateProperties - Properties of the virtual machine to be added or updated(Array Of Infrastructure.CustomPropert
  - virtualMachineEvent - Virtual Machine Event(STRING)
  - lifecycleState - Lifecycle state(Infrastructure.EBS.LifecycleStateInfo)
  - componentId - Component id(STRING)
  - blueprintName - Blueprint name(STRING)
  - componentTypeId - Component type id(STRING)
  - endpointId - Endpoint id(STRING)
  - workflowNextState - Next Lifecycle State(STRING)
  - virtualMachineDeleteProperties - Properties of the virtual machine to be deleted(Array Of Infrastructure.CustomProperty)

The event topic schema includes the machine properties. For performance reasons, by default, no machine properties are sent from state to state or based on a triggered event. You can include machine properties in the event payload. To achieve this, you set a custom property on endpoint, reservation, blueprint, request, or others.

You define the custom property name in the `Extensibility.Lifecycle.Properties.{workflowName}.{stateName}` format. The value is a comma-separated list of properties that can also include wildcard entries. Events can be defined on a global level, in this case `{stateName}` is the name of the workflow.

For example, to send all hidden properties and all properties that start with `Virtual` to the `BuildingMachine` state, set a custom property with the `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.BuildingMachine` name and `__*,Virtual*` value. This example shows the two patterns for hidden and virtual properties, separated by a comma.

NOTE Hidden properties start with a double underscore (__) character in their name.

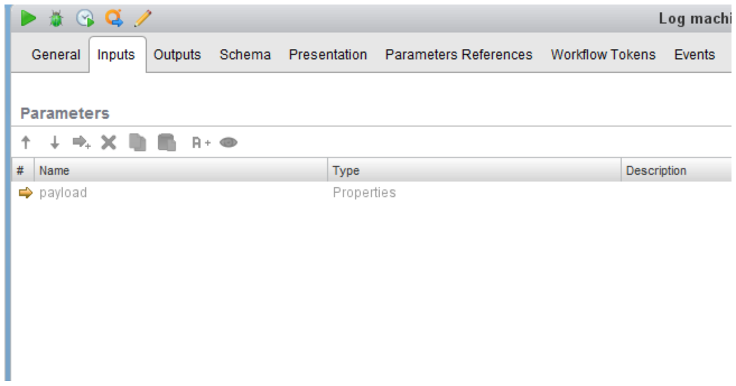
Passing Input Parameters to the vRealize Orchestrator Workflow

You can choose among several methods to pass information to a workflow.

TABLE 5 - VALID OPTIONS FOR DEFINING WORKFLOW INPUT PARAMETERS

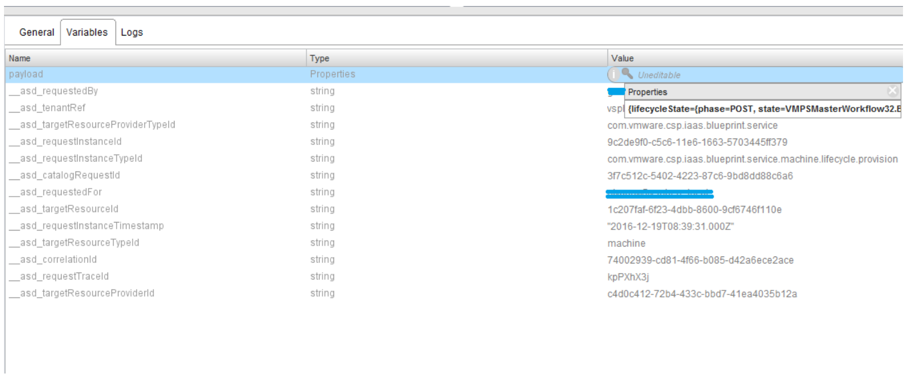
Method	Description
String representation of JSON data	The target workflow has a single input parameter of type `String`. **NOTE** This method requires an extra effort to parse the event payload.
Object of type `Properties`	The object contains the entire subscription information. The target workflow has a single input parameter of type `Properties`.
Schema-provided attributes as workflow input parameters	The target workflow has several input parameters that match the event topic schema parameters.

FIGURE 7 - SINGLE INPUT PARAMETER OF TYPE PROPERTIES OF A VREALIZE ORCHESTRATOR WORKFLOW



#	Name	Type	Description
	payload	Properties	

FIGURE 8 - A SUBSCRIPTION PAYLOAD AND HIDDEN INPUT PARAMETERS AS INPUT PARAMETERS FOR A VREALIZE ORCHESTRATOR WORKFLOW

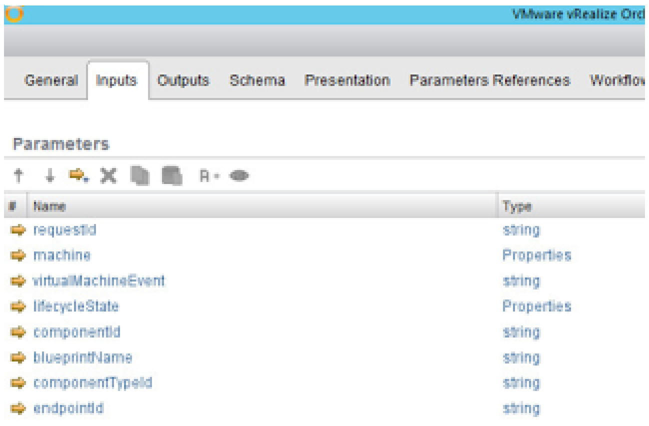


Name	Type	Value
payload	Properties	Uneditable
__asd_requestedBy	string	[redacted]
__asd_tenantRef	string	vspl
__asd_targetResourceProviderTypeId	string	com.vmware.csp.iaas.blueprint.service
__asd_requestInstanceId	string	9c2de9f0-c5c6-11e6-1663-5703445ff379
__asd_requestInstanceTypeId	string	com.vmware.csp.iaas.blueprint.service.machine.lifecycle.provision
__asd_catalogRequestId	string	3f7c512c-5402-4223-87c6-9bd8dd88c6a6
__asd_requestedFor	string	[redacted]
__asd_targetResourceId	string	1c207faf-6f23-4dbb-8600-9cf6746f110e
__asd_requestInstanceTimestamp	string	"2016-12-19T08:39:31.000Z"
__asd_targetResourceTypeId	string	machine
__asd_correlationId	string	74002939-cd81-4f66-b085-d42a6ece2ace
__asd_requestTraceId	string	kpPXhX3j
__asd_targetResourceProviderId	string	c4d0c412-72b4-433c-bbd7-41ea4035b12a

Properties
{lifecycleState={phase=POST, state=VMPSMasterWorkflow32.E

For more information about samples, see Samples.

FIGURE 9 - INPUT PARAMETERS OF A VREALIZE ORCHESTRATOR WORKFLOW THAT MATCH THE INPUT PARAMETERS OF A MACHINE PROVISIONING TOPIC SCHEMA



#	Name	Type
	requestId	string
	machine	Properties
	virtualMachineEvent	string
	lifecycleState	Properties
	componentId	string
	blueprintName	string
	componentTypeId	string
	endpointId	string

When an event triggers a workflow run, some of the event headers or event class fields become part of the workflow run as global input parameters.

TABLE 6 - THE BINDING BETWEEN EVENT HEADERS AND THE NAMES OF THE GLOBAL PARAMETERS THAT ARE SENT TO A WORKFLOW

Global Parameter Name of a Workflow	Event Header	Sample Data	Details
__asd_catalogRequestId	Id	559de3b8-1a16-4b2e-a47e-5990f7456c19	ID of the event.
__asd_requestedBy	userName	demouser@test.com	
__asd_requestedFor	userName	demouser@test.com	
__asd_requestInstanceId	id	559de3b8-1a16-4b2e-a47e-5990f7456c19	
__asd_requestInstanceTimestamp	timeStamp	2015-04-02T14:11:21.000+03:00	
__asd_requestInstanceTypeId	eventTopicId	com.vmware.csp.iaas.blueprint.service.machine.lifecycle.provision	
__asd_targetResourceId	targetId	a6bdf2dd-539c-43ff-acd2-c843938ab30f	
__asd_targetResourceProviderId	sourceIdentity	799957e6-7557-4cde-a7fe-87e90095f574	ID of the object, for which the event is published. For example, the ID of the newly provisioned virtual machine.
__asd_targetResourceProviderTypeId	sourceType	com.vmware.csp.iaas.blueprint.service	ID of the service that published the event, for example the IaaS service ID.
__asd_targetResourceTypeId	targetType	machine	
__asd_tenantRef	tenantId	Dev	
__asd_traceId	trace-id	Eadeu9pe	Request ID of the unique distributed trace.

NOTE The **Sample Data** column contains real data from an event that IaaS triggers during a machine provisioning. The workflow receives this data and runs according to the subscription.

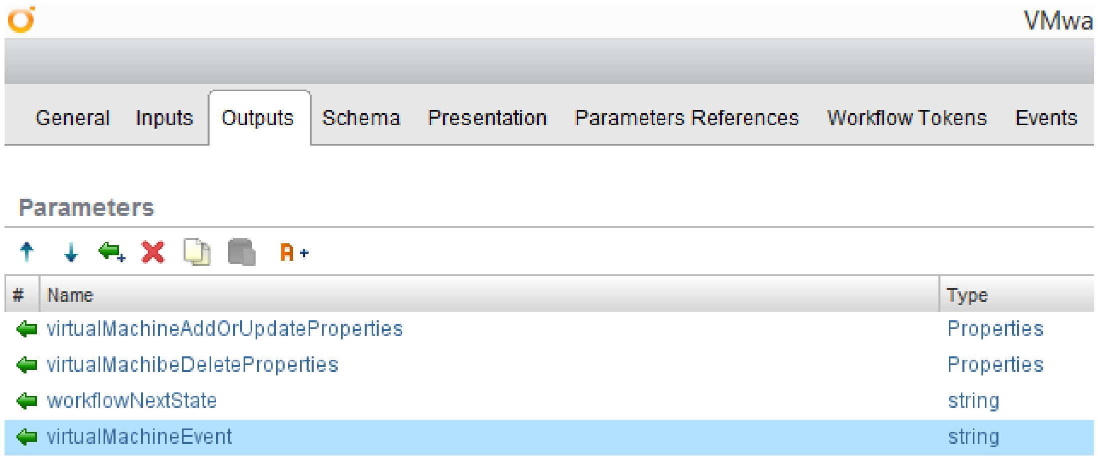
FIGURE 10 - EXAMPLE DATA THAT CONTAINS HIDDEN PROPERTIES

__asd_requestedBy	string	givanov@vsphere.local
__asd_tenantRef	string	vsphere.local
__asd_targetResourceProviderTypeId	string	com.vmware.csp.iaas.blueprint.service
__asd_requestInstanceId	string	0d669650-527f-11e6-dc5a-b4168ba88b5e
__asd_requestInstanceTypeId	string	com.vmware.csp.iaas.blueprint.service.machine.lifecycle.active
__asd_catalogRequestId	string	4e78064d-da8c-475d-8d76-6488bd165a09
__asd_requestedFor	string	givanov@vsphere.local
__asd_targetResourceId	string	7d01ee0d-a396-49ef-9490-0d3bb4f06d57
__asd_requestInstanceTimestamp	string	"2016-07-25T15:47:11.000Z"
__asd_targetResourceTypeId	string	machine
__asd_correlationId	string	88bf628f-2035-4e67-81e3-58bf7fee0378
__asd_requestTraceId	string	ykGIJygs
__asd_targetResourceProviderId	string	c4d0c412-72b4-433c-bbd7-41ea4035b12a

Passing Workflow Output Parameters to vRealize Automation

The values of the output parameters modify the machine provisioning properties, define the next life cycle state, and trigger a machine event.

FIGURE 11 - EXAMPLES OF OUTPUT PARAMETERS OF A WORKFLOW



#	Name	Type
	virtualMachineAddOrUpdateProperties	Properties
	virtualMachibeDeleteProperties	Properties
	workflowNextState	string
	virtualMachineEvent	string

Why Migrate Extensibility Use Cases to Event Broker?

vRealize Automation, previously known as vCloud Automation Center, began as a two-component product that used Java and .Net stack. Each vRealize Automation 7.x release moves the product components further from .Net to Java. As a result, the changes in vRealize Automation affect the functionalities of the vRealize Orchestrator plug-ins that manage vRealize Automation entities. For this reason, VMware recommends that you reimplement the extensibility solutions based on vCloud Automation Center 6.x, so that they work with the Event Broker Service and the vRealize Automation 7.x plug-in.

The long-term goal is to use the extensibility feature only with the event broker solution.

As a short-term solution, you can adapt your existing extensibility workflows, without migrating them to Event Broker.

Adapting Existing Extensibility Workflows

After upgrading vRealize Automation from 6.x to 7.x, you must first adapt the extensibility workflows based on the changes in the vRealize Orchestrator vCloud Automation Center 7.x plug-in.

What Was Changed Between the Versions of the vRealize Orchestrator Plug-In?

From the perspective of the vRealize Orchestrator plug-ins, it is the vRealize Automation plug-in that now manages the migrated infrastructure entities, also known as Model Manager Entities, that the vCloud Automation Center plug-in handled previously.

TABLE 7 - INFRASTRUCTURE ENTITIES THAT WERE TRANSFORMED FROM .NET TO JAVA

vCloud Automation Center 6.x Entity Type	vRealize Automation 7.x Entity Type
Infrastructure Machine Blueprint	Composite Blueprint
Infrastructure Property Layout	Property Groups
Infrastructure Approval Policy	Approval Policy
Infrastructure Property Definition	Property Definition

Some entities still exist in both vCloud Automation Center and vRealize Automation plug-ins, but it is best to manage these entities from the vRealize Automation plug-in.

TABLE 8 - THE INFRASTRUCTURE ENTITIES THAT WERE MIGRATED FROM THE VCLOUD AUTOMATION CENTER PLUG-IN TO THE VREALIZE AUTOMATION PLUG-IN

vCloud Automation Center Plug-In Entity Type	vRealize Automation Plug-In Entity Type
`VCAC:Blueprint` (removed as a type in vRealize Automation 7.x)	`VCACCAFE:CompositeBlueprint`
`VCAC:Reservation` (still exists as a type in vRealize Automation 7.x)	`VCACCAFE:Reservation`
Reservation Policy (not available as a plug-in type)	`VCACCAFE:ReservationPolicy`
Reservation Policy Type (not exposed as a plug-in type)	`VCACCAFE:ReservationPolicyType`

Reservation Alert (not exposed as a plug-in type)	`VCACCAFE:ReservationAlert`
Reservation Alert Policy (not exposed as a plug-in type)	`VCACCAFE: ReservationAlertPolicy`
Reservation Alert Type (not exposed as a plugin type)	`VCACCAFE: ReservationAlertType`

TABLE 9 - CHANGES IN THE VCLOUD AUTOMATION CENTER PLUG-IN AND IN THE VREALIZE AUTOMATION PLUG-IN.

Plug-In Name	Changes
vCloud Automation Center plug-in	• The plug-in is not backward compatible and works only with vRealize Automation 7.x. • Added CAFÉ authentication support. • Added HTTP connection caching. • The vRealize Automation plug-in and the IaaS plug-in are included in a single package. • Completely removed two inventory objects: o `VCAC:Blueprint` o `VCAC:ApprovalPolicy` • Deprecated the following inventory objects: ○ Provisioning groups ○ Reservations ○ Machine Prefixes • Because of the composite blueprint, removed all Discovery workflows: ○ Import a vCenter Virtual Machine ○ Import an IaaS Virtual Machine • Updated the **Assign a state change** workflow to a blueprint. The virtual machines now use the composite blueprint. • Restricted CRUD operations on certain model entities.
vRealize Automation plug-in	• Introduced management of entities from type Business Group. **NOTE** Use the `VCACCAFE:Subtenant` type. The `VCACCAFE:BusinessGroup` type is deprecated. • Introduced management of entities from type Reservation. • Introduced management of entities from type

	Reservation Policy. • Introduced management of entities from type Property Definition. • Introduced management of entities from type Composite Blueprint. • Introduced a vRealize Orchestrator workflow for customizing IaaS catalog item request.

Prerequisites

After you upgrade to vRealize Automation 7.x, you must confirm that the following requirements are fulfilled.

- All system components are running.
- The vRealize Automation services show as REGISTERED in the component registry.
- All vRealize Orchestrator plug-ins are compatible with the configured external systems and configured to work with external systems.
- All external systems, that are configured to work with vRealize Automation and vRealize Orchestrator, are accessible.

Migrating Extensibility to Event Broker

There is no automated way of migrating extensibility from vCloud Automation Center plug-in to Event Broker. This process is manual and requires some modifications in both vRealize Automation and vRealize Orchestrator.

In vRealize Orchestrator, you must modify all extensibility-related workflows. You must replace all input parameters of all custom extensibility workflows with the types provided by the Event Broker event topic.

In vRealize Automation, you must remove all unused custom properties that are related to the vCloud Automation Center 6.x extensibility.

Migrating extensibility to use the Event Broker Service involves several stages:

Uninstall the vCloud Automation Center Plug-In Extensibility

You can uninstall the vCloud Automation Center plug-in extensibility by running a vCloud Automation Center plug-in workflow.

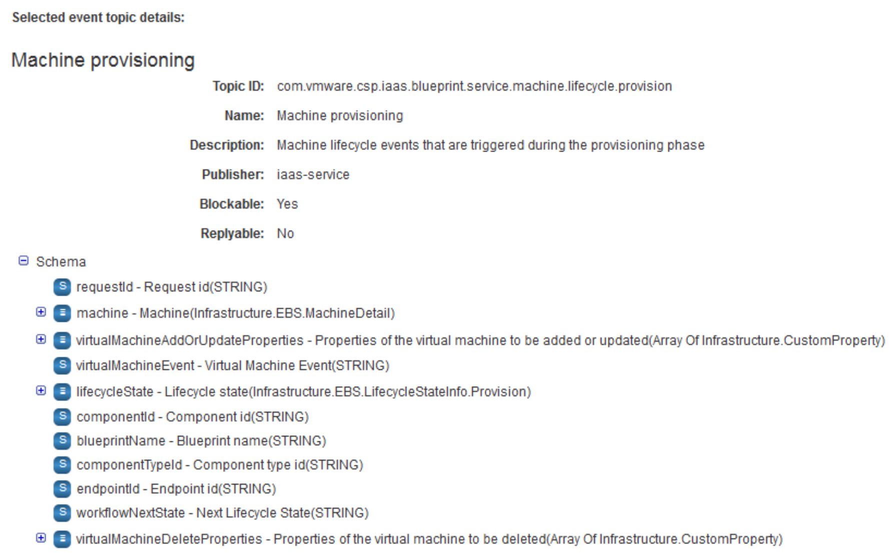
1. Log in to the Orchestrator client as an administrator.
2. Click the **Workflows** view.
3. In the hierarchical list of workflows, expand **Library** > **vRealize Automation** > **Infrastructure Administration** > **Extensibility > Installation**.

4. Run the **Uninstall vCO customization** workflow.

Rework Extensibility Workflows to Use Event Broker

1. After you remove the extensibility implementation provided by the vCloud Automation Center plug-in, you make a copy of the existing extensibility workflows and rework the duplicate workflows to become compatible with Event Broker.
2. Check the schema of the Machine provisioning event topic to retrieve information about the data payload and the type of the data elements.

FIGURE 12 - THE SCHEMA OF THE MACHINE PROVISIONING EVENT TOPIC



Selected event topic details:

Machine provisioning

Topic ID: com.vmware.csp.iaas.blueprint.service.machine.lifecycle.provision
Name: Machine provisioning
Description: Machine lifecycle events that are triggered during the provisioning phase
Publisher: iaas-service
Blockable: Yes
Replyable: No

Schema
- requestId - Request id(STRING)
- machine - Machine(Infrastructure.EBS.MachineDetail)
- virtualMachineAddOrUpdateProperties - Properties of the virtual machine to be added or updated(Array Of Infrastructure.CustomProperty)
- virtualMachineEvent - Virtual Machine Event(STRING)
- lifecycleState - Lifecycle state(Infrastructure.EBS.LifecycleStateInfo.Provision)
- componentId - Component id(STRING)
- blueprintName - Blueprint name(STRING)
- componentTypeId - Component type id(STRING)
- endpointId - Endpoint id(STRING)
- workflowNextState - Next Lifecycle State(STRING)
- virtualMachineDeleteProperties - Properties of the virtual machine to be deleted(Array Of Infrastructure.CustomProperty)

3. Based on the machine provisioning topic schema, update the input and output parameters of the workflows.

Workflow input parameters

The Event Broker payload can bind to a single workflow input parameter by using either a `Properties` type, or several input parameters that are based on the event topic schema.

TABLE 10 - THE EXTENSIBILITY DATA PAYLOAD ITEMS OF THE VCLOUD AUTOMATION CENTER AND THE VREALIZE AUTOMATION

vCloud Automation Center 6.x	vRealize Automation 7.x
virtualMachineEntity / Type: `vCAC:Entity`	requestId / Type: `String`
vCACVm / Type: `vCAC:VirtualMachine`	machine / Type: `Properties`
vCACVmProperties / Type: `Properties`	endpointId / Type: `String`
vcacHost / Type: `vCAC:Host`	componentTypeId / Type: `String`

externalWFStub / Type: `String`	lifecycleState / Type: `Properties`
	componentId / Type: `String`
	blueprintName / Type: `String`

NOTE Payload data is the information that is configured in a blueprint, or becomes active and available during the provisioning process.

Workflow output parameters

Most of the extensibility use cases require retrieving or configuring virtual machine data.

a. Configure the machine data use cases.
   - Configure a custom machine name based on some logic.
   - Configure the name of the inventory folder in the data center, in which you put the virtual machine.
   - Configure the machine network parameters based on the external system data.

For all of the above use cases, you must update the output parameters of the vRealize Orchestrator workflows that are used for machine customization in the extensibility use cases.

TABLE 11 - THE OUTPUT PARAMETERS OF THE VREALIZE ORCHESTRATOR WORKFLOWS THAT ARE USED IN MACHINE CUSTOMIZATION EXTENSIBILITY USE CASES

Parameter	Description
`virtualMachineAddOrUpdateProperties`	An object with type `Properties` that collects all machine properties that are used for machine customization.
`virtualMachineDeleteProperties`	An object with type `Properties` that collects all machine properties that must be removed.
`workflowNextState`	Defines the next state of the infrastructure state machine workflow.
`virtualMachineEvent`	Triggers an event on the target machine.

b. Retrieve the machine data use cases.

Retrieve the virtual machine parameters and send the data to an external system, for example, a configuration management database, such as Service Now or a Microsoft Active Directory server.

You can retrieve the machine data by using the event broker payload and then pass this data as a vRealize Orchestrator `Properties` object. Additional data is also available in the internal workflow input parameters, which are the parameters that start with a double underscore (__) character.

You can retrieve the value of such an input parameter with a script, for example:

```
var value = System.getContext().getParameter(<paramName>)
```

Based on the workflow logic and retrieved data, you configure the workflow output parameters.

- `virtualMachineAddOrUpdateProperties`
- `virtualMachineDeleteProperties`
- `workflowNextState`
- `virtualMachineEvent`

4. Retrieve a list of all machine parameters and machine custom properties that are used in workflow scripting.

The following list shows sample data that is based on the machine provisioning topic schema.

```
        Machine properties sample data for clone from template single component CI


    {
        owner=fritz@xxxx,
        name=VC1-VM18,
        id=efd9f11f-b5ba-4230-8709-7339ad63ffeb,
        type=0.0,
        properties={
                VirtualMachine.Storage.Cluster.Name=StorageCluster,
                VirtualMachine.Cafe.Blueprint.Id=CloneVM_Test,
                Vrm.DataCenter.Location=Boston,
                __InterfaceType=vSphere,
                VirtualMachine.Disk1.Storage.Cluster.Name=StorageCluster,
                __VirtualMachine.Allocation.InitialMachineState=SubmittingRequest,
                VirtualMachine.Cafe.Blueprint.Component.TypeId=Infrastructure.CatalogItem.Machine.Virtual.vSphere,
                __api.request.id=ae42002d-010f-412c-ae56-50325cfc6faf,
                VirtualMachine.Disk0.Size=5,
                VirtualMachine.Disk0.StorageReservationPolicy=Storage1,
                VirtualMachine.Disk1.StorageReservationPolicy=Storage1,
                __api.request.callback.service.id=c77cf2fa-bc87-4172-8a5a-9cc3ebef2d15,
                VirtualMachine.Cafe.Blueprint.Name=CloneVM_Test,
                Cafe.Shim.VirtualMachine.TotalStorageSize=6,
                VirtualMachine.Disk1.Size=1,
                __Legacy.Workflow.ImpersonatingUser=,
                __Legacy.Workflow.User=fritz@coke.sqa-horizon.local,
                __VirtualMachine.ProvisioningWorkflowName=CloneWorkflow,
                VirtualMachine.Disk1.Storage=StorageCluster,
                Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.BuildingMachine=*,
                VirtualMachine.Disk1.Label=Disk2,
                VirtualMachine.Disk0.Storage.Cluster.ExternalReferenceId=group-p1555,
                VirtualMachine.Network0.AddressType=Static,
                __Cafe.Request.VM.LeaseDays=1,
                __request_reason=,
                __Cafe.Root.Request.Id=10230d73-b226-41b8-b272-8902c4ecb361,
                _number_of_instances=1,
                IAAS_6X_UUID=e4d67d2d-d051-4b3c-b9d1-20cd81c0104e,
                VirtualMachine.Disk0.IsClone=true,
                        __Cafe.Request.VM.ArchiveDays=1,
                        __clonefromid=61ed2987-cbd6-4fed-a31f-34bf19c14f2c,
                        __Cafe.Request.BlueprintType=0,
                        __Notes=vm desc,
                        VirtualMachine.Memory.Size=1024,
                        VirtualMachine.Cafe.Blueprint.Component.Cluster.Index=0,
                        VirtualMachine.CPU.Count=1,
                        VirtualMachine.Disk0.user_created=true,
                        __Clone_Type=CloneWorkflow,
                        callExternalWorkflowActivityPostWaitingToBuild=False,
                        VirtualMachine.Admin.TotalDiskUsage=6144,
                        VirtualMachine.Network0.ExternalAddress=,
                        __trace_id=0rCbogwD,
                        __iaas_request_binding_id=8860aab6-8123-4ad1-a888-a59ef7b98a7d,
                        VirtualMachine.Cafe.Blueprint.Component.Id=Machine,
                        VirtualMachine.Disk1.Storage.Cluster.ExternalReferenceId=group-p1555,
                        VirtualMachine.Network0.Address=192.168.1.160,
                        VirtualMachine.Disk0.Storage.Cluster.Name=StorageCluster,
                        VirtualMachine.Storage.Cluster.ExternalReferenceId=group-p1555,
                        VirtualMachine.Disk0.Storage=StorageCluster,
                        Vrm.ProxyAgent.Uri=https: //sof-vcac-vm336.sof-mbu.eng.vmware.com/VMPS2Proxy,
                        __reservationPolicyID=c1c503a5-072c-416c-aecc-9f9c8843264d,
                        VirtualMachine.Disk1.IsClone=false,
                        VirtualMachine.Disk1.Letter=D: ,
                        VirtualMachine.Storage.Name=StorageCluster,
                        __clonefrom=xxxxx,
                        VirtualMachine.Network0.Name=VMNetwork
            }
}
```

5. From the list in Step 4, find the new custom property name based on its value.

For more information, see the Custom Properties Reference for vRealize Automation 6.2.x and the Custom Properties Reference for vRealize Automation 7.x documents.

6. Adapt your workflows and rework all scripting elements, so that they use the data payload provided by the Event Broker topic schema.
7. Create the required subscriptions by selecting the event topic, defining the criteria and conditions, and the workflow to be triggered.
8. Set the Event Broker custom properties that define the payload on applicable entities, such as endpoint, reservation, blueprint, request, and others.
   a. First try to use all the data that the Event Broker provides, by filling in the property value with a wildcard (*) and all values that can be overridden.
   b. When you know what input data is required, set a regular expression to limit the data traffic. This minimizes the network traffic and load on the system in general.

Example for custom property state names:

- `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.BuildingMachine`
- `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.Disposing`

- `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.Expired`
- `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.MachineProvisioned`
- `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.Requested`
- `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.UnprovisionMachine`

9. Remove the old extensibility properties from the blueprint.
   a. In the Orchestrator client, expand **Library** > **vRealize Automation** > **Infrastructure Administration** > **Extensibility.**
   b. Run the **Remove a state change workflow from a blueprint and its virtual machines** for all composite blueprints.

User Scenarios

Customizing Machine Provisioning Through Advanced Service Designer

Description

You have an instance of vCloud Automation Center 6.x, where an Advanced Service Designer catalog item is exposed for provisioning a customizable virtual machine in the vRealize Automation infrastructure. This Advanced Service Designer catalog item uses a modified version of the **Request a catalog Item** workflow in vRealize Orchestrator. This workflow adds request data properties to make the request of infrastructure virtual machine customizable.

Problem

In the described setup, the custom solutions for requesting a virtual machine through the Advanced Service Designer rely on triggering the `com.vmware.library.vcac.requestCatalogItem(...)` action in vRealize Orchestrator.

After the upgrade to vRealize Automation 7.x, the customized request through the Advanced Service Designer catalog item might not work correctly, because some of the custom properties are no longer valid, or their names changed. In addition to that, some of the custom properties that are valid in vRealize Automation 7.0 and 7.0.1 can no longer be used in vRealize Automation 7.1 and later.

Resolution

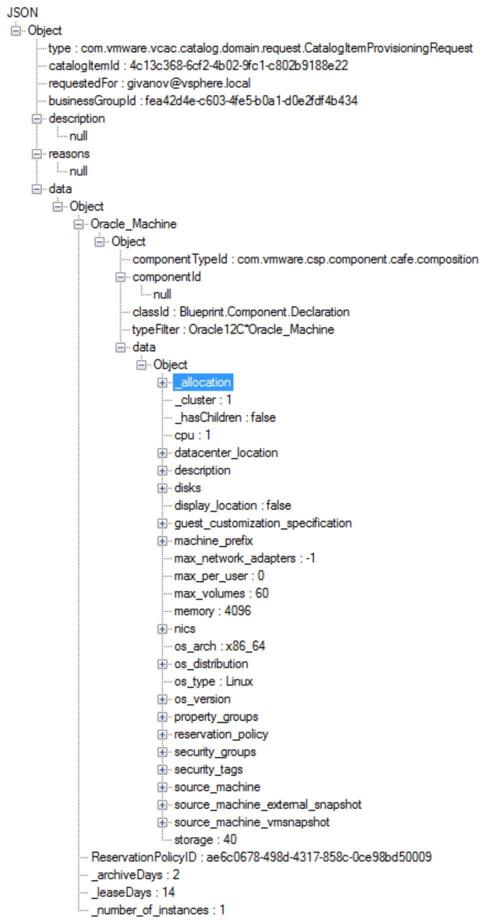
To make the customized request work with vRealize Automation 7.x, you must modify the custom solutions to use the following action:
`System.getModule("com.vmware.library.vcaccafe.request").requestCatalogItemWithProvisionin`

`gRequest(catalogItem, provisioningRequest)`. This action uses request data in JSON format. In this format, the request data is easy to modify.

Migration procedure

1. Retrieve the JSON representation of the request data.
   a. Log in to the Orchestrator client with an administrator account.
   b. Expand **Library** > **vRealize Automation** > **Samples**.
   c. Run the **Print catalog item provisioning request as JSON** workflow.
   d. Select a catalog item based on a composite blueprint. Retrieve the JSON output representation of a request data, printed in the workflow run console.

FIGURE 13 - THE JSON REPRESENTATION OF THE REQUESTED DATA



```
JSON
  Object
    type : com.vmware.vcac.catalog.domain.request.CatalogItemProvisioningRequest
    catalogItemId : 4c13c368-6cf2-4b02-9fc1-c802b9188e22
    requestedFor : givanov@vsphere.local
    businessGroupId : fea42d4e-c603-4fe5-b0a1-d0e2fdf4b434
    description
      null
    reasons
      null
    data
      Object
        Oracle_Machine
          Object
            componentTypeId : com.vmware.csp.component.cafe.composition
            componentId
              null
            classId : Blueprint.Component.Declaration
            typeFilter : Oracle12C*Oracle_Machine
            data
              Object
                _allocation
                _cluster : 1
                _hasChildren : false
                cpu : 1
                datacenter_location
                description
                disks
                display_location : false
                guest_customization_specification
                machine_prefix
                max_network_adapters : -1
                max_per_user : 0
                max_volumes : 60
                memory : 4096
                nics
                os_arch : x86_64
                os_distribution
                os_type : Linux
                os_version
                property_groups
                reservation_policy
                security_groups
                security_tags
                source_machine
                source_machine_external_snapshot
                source_machine_vmsnapshot
                storage : 40
        ReservationPolicyID : ae6c0678-498d-4317-858c-0ce98bd50009
        _archiveDays : 2
        _leaseDays : 14
        _number_of_instances : 1
```

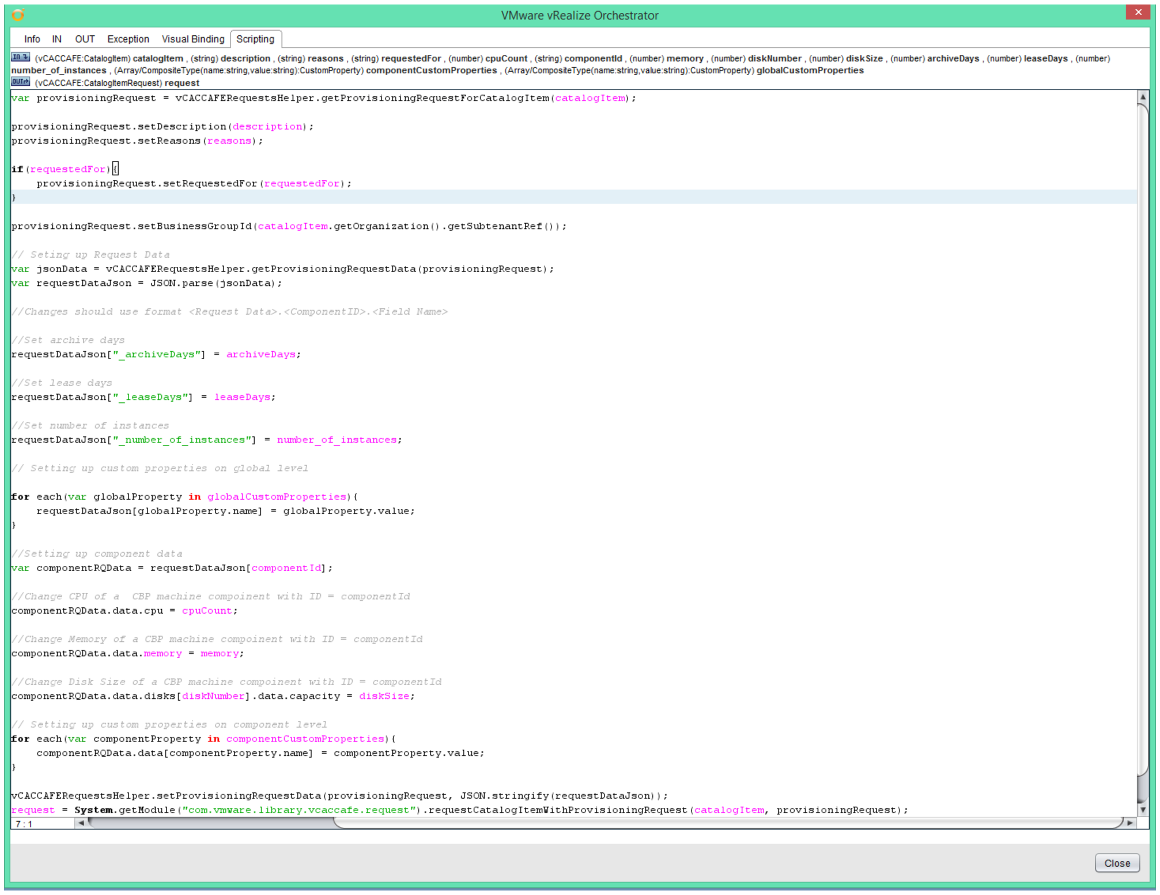
The request must use a specific format.

```
{Request Data}.{ComponentID}.{Field Name}
```

2. Adapt the scripting logic of the vRealize Orchestrator workflow in the existing customized request scenario.
    a. In the Orchestrator client, go to **Library** > **vRealize Automation** > **Requests**.
    b. Copy the workflow **Request a catalog item with provisioning request** workflow.
    c. Modify the scripting of the copied workflow.

Examples

FIGURE 14 - A SAMPLE CONFIGURATION OF COMPONENT DATA AND CUSTOM PROPERTIES IN A REQUEST TO A SINGLE MACHINE CATALOG ITEM



IN (vCACCAFE:CatalogItem) **catalogItem** , (string) **description** , (string) **reasons** , (string) **requestedFor** , (number) **cpuCount** , (string) **componentId** , (number) **memory** , (number) **diskNumber** , (number) **diskSize** , (number) **archiveDays** , (number) **leaseDays** , (number) **number_of_instances** , (Array/CompositeType(name:string,value:string):CustomProperty) **componentCustomProperties** , (Array/CompositeType(name:string,value:string):CustomProperty) **globalCustomProperties**
OUT (vCACCAFE:CatalogItemRequest) **request**

```
var provisioningRequest = vCACCAFERequestsHelper.getProvisioningRequestForCatalogItem(catalogItem);

provisioningRequest.setDescription(description);
provisioningRequest.setReasons(reasons);

if(requestedFor){
    provisioningRequest.setRequestedFor(requestedFor);
}

provisioningRequest.setBusinessGroupId(catalogItem.getOrganization().getSubtenantRef());

// Seting up Request Data
var jsonData = vCACCAFERequestsHelper.getProvisioningRequestData(provisioningRequest);
var requestDataJson = JSON.parse(jsonData);

//Changes should use format <Request Data>.<ComponentID>.<Field Name>

//Set archive days
requestDataJson["_archiveDays"] = archiveDays;

//Set lease days
requestDataJson["_leaseDays"] = leaseDays;

//Set number of instances
requestDataJson["_number_of_instances"] = number_of_instances;

// Setting up custom properties on global level

for each(var globalProperty in globalCustomProperties){
    requestDataJson[globalProperty.name] = globalProperty.value;
}

//Setting up component data
var componentRQData = requestDataJson[componentId];

//Change CPU of a  CBP machine compoinent with ID = componentId
componentRQData.data.cpu = cpuCount;

//Change Memory of a CBP machine compoinent with ID = componentId
componentRQData.data.memory = memory;

//Change Disk Size of a CBP machine compoinent with ID = componentId
componentRQData.data.disks[diskNumber].data.capacity = diskSize;

// Setting up custom properties on component level
for each(var componentProperty in componentCustomProperties){
    componentRQData.data[componentProperty.name] = componentProperty.value;
}

vCACCAFERequestsHelper.setProvisioningRequestData(provisioningRequest, JSON.stringify(requestDataJson));
request = System.getModule("com.vmware.library.vcaccafe.request").requestCatalogItemWithProvisioningRequest(catalogItem, provisioningRequest);
```

7 : 1

For more information and sample workflows, see Samples.

Managing Blueprint Entities through the vRealize Orchestrator vCloud Automation Center Plug-In

Description

You have an instance of vCloud Automation Center 6.x, where vRealize Orchestrator extensibility workflows are configured to customize the provisioning of infrastructure virtual machines.

Problems

After the upgrade to vRealize Automation 7.x, several problems might occur.

The property `VirtualMachineTemplateID` of `VCAC:VirtualMachine` object now points to a system machine blueprint. This system machine blueprint is a hidden blueprint that is used for provisioning vSphere virtual machines as part of composite blueprints.

Some types are completely removed from the vCloud Automation Center plug-in. These types are replaced by similar ones, that are available in the vRealize Automation plug-in. For this reason, you must modify the custom solutions that are based on the vCloud Automation Center plug-in to use the types provided by the vRealize Automation plug-in.

For more information about entity mappings, see What was changed between the versions of the vRealize Orchestrator plug-in.

The management of some of the infrastructure components, such as Reservations, Business groups, Reservation Policies, Storage Policies, Property groups, Custom Properties, was transformed in the Java side of the product.

Resolution

If the property `VirtualMachineTemplateID` is used in scripting solutions to retrieve the composite blueprint objects, you have several options.

- You can use properties like `BlueprintName`, which is the name of the composite blueprint or `ComponentName`, which is the name of the component inside the composite blueprint.
- You can create a query for the composite blueprint representation which can be used to retrieve additional information.

You can also use the samples for entity management that are introduced in the vRealize Automation plug-in. These samples are available in the **Library** > **vRealize Automation** > **Samples** folder in the **Workflows** view of the Orchestrator client.

Using the vRealize Automation Plug-In and the vCloud Automation Center Plug-In

Description

You have an instance of vCloud Automation Center 6.x, where the vRealize Automation plug-in and the vCloud Automation Center plug-in manage the vCloud Automation Center and the infrastructure entities such as Business Groups, Reservations, and Storage Policies.

Problem

After an upgrade to vRealize Automation 7.1 or 7.2, the vCloud Automation Center plug-in no longer manages the infrastructure entities correctly.

NOTE Versions 7.0, 7.0.1 of vRealize Automation are not affected.

Resolution

Modify the custom scripts in vRealize Orchestrator to use the vRealize Automation plug-in for managing the infrastructure entities.

For more information and sample entities, see Samples.

Customizing Infrastructure Change Owner Operation Through Advanced Service Designer

Description

You have an instance of vCloud Automation Center 6.x, where a XaaS catalog item wraps the **Change owner** post-provision operation.

Problem

After upgrading to vRealize Automation 7.x, the **Change owner** post-provision operation does not work, even though the request is successful. The root cause of the problem is that, after the upgrade, this operation is not available on a virtual machine level and is moved to a deployment level.

For this reason, you must adapt your scripting solution to retrieve the deployment and run the **Change owner** post-provision operation on this deployment. The owner of all virtual machines under the deployment will be changed accordingly.

Resolution

For custom solutions that wrap the **Change owner** infrastructure operation, you can use the **Request a resource action with a request template** workflow. With a small modification to the workflow scripting, you can retrieve the request template in JSON format. After that, you modify the workflow again, based on the retrieved data, so that you can successfully run the **Change owner** operation from the workflow. For more information, see Samples.

vRealize Automation 7.1 introduces a built-in resource mapping for deployment catalog resources.

For versions of vRealize Automation 7.0 and 7.0.1, you must import the deployment resource mapping to vRealize Orchestrator, which is available in the Samples section.

Reconfiguring an Infrastructure Machine Through an Advanced Service Designer

Description

You have an instance of vCloud Automation Center 6.x, where a XaaS catalog item wraps the **Reconfigure virtual machine** post-provision operation.

Problem

After upgrading to vRealize Automation 7.x, the **Reconfigure virtual machine** post-provision operation might not work as expected, even though the request is successful. The root cause of the problem is that, after the upgrade, some of the custom properties used for **Reconfigure virtual machine** are no longer available, for example:

```
provider-Cafe.Shim.VirtualMachine.Reconfigure.CpuCount

provider-Cafe.Shim.VirtualMachine.Reconfigure.MemorySize.
```

Resolution

For custom solutions that wrap the **Reconfigure virtual machine** infrastructure operation, you can use the **Request a resource action with a request template** workflow. With a small modification to the workflow scripting, you can retrieve the request template in JSON format. After that, you modify the workflow again, based on the retrieved data, so that you can successfully run the **Reconfigure virtual machine** operation from the workflow. For more information, see Samples.

Samples

To use the listed examples, copy and paste them locally on your file system, so that you can import them to Orchestrator.

Catalog Item Samples

Request a catalog item based on composite blueprint.

Request a catalog item with provisioning request.workflow

Retrieve a component ID of a catalog item by type.

This sample is a vRealize Orchestrator action, which returns all component IDs filtered by type. For example, a vSphere machine type is `Infrastructure.CatalogItem.Machine.Virtual.vSphere`. If you have a multi-machine composite blueprint, this action returns a list of component IDs.

getCatalogItemComponentIds.action

Composite Blueprint Samples

Create a composite blueprint that uses a vSphere template.

CreateCloneVsphereVmCompositeBlueprint.workflow

Retrieve a composite blueprint data.

GetCompositeBlueprintData.workflow

Add global-level custom properties to a composite blueprint.

AddCompositeBlueprintGlobalCustomProperties.workflow

Add component custom properties to a composite blueprint.

AddCompositeBlueprintComponentCustomProperties.workflow

Property Groups and Property Definitions

The vRealize Automation plug-in 7.2 includes the workflows for managing property groups and property definitions. For versions of vRealize Automation 7.0, 7.0.1 and 7.1, you can use the following samples.

Retrieve property definitions.

getPropertyDefinition.workflow

Create a property definition.

createPropertyDefinition_7.1.workflow

Retrieve property groups.

getPropertyGroups.workflow

Create a property group.

createPropertyGroup_7.1.workflow

Compute and Storage Policy Samples

Create a reservation policy: Compute resource or Storage policy.

CreateReservationPolicy.workflow

vSphere Reservation Samples

Create a vSphere reservation.

CreateReservation.workflow

Update a vSphere reservation.

UpdateReservation.workflow

Na Open Data Protocol (OData) Query sample.

getAllItemsInBusinessGroup.workflow

Print an infrastructure resource action in JSON format.

Print a resource action data.workflow

Sample of the Reconfigure Virtual Machine workflow that wraps the original infrastructure resource action.

Reconfigure Infrastructure Vm.workflow

Change the owner of a deployment

Change Deployment Owner.workflow

A deployment catalog resource mapping action.

mapToCatalogResource.action

Event Broker Samples

Print payload for a machine provisioning subscription.

Log machine
provisioning payloa

Handle a failing request and send an email.

Handle Failing
Request.workflow

You can find additional examples for managing CAFÉ entities and infrastructure entities in the initial vRealize Automation content package. The content package is available on the file system of the upgraded vRealize Automation Appliance in the `/usr/lib/vcac/tools/initial-config` directory. The vRealize Orchestrator package name is `vra-initial-config-bundle-workflow.package`. In order to use the content package, you must import it to vRealize Orchestrator.

vRealize Automation Plug-In Enhancement
Together with the new functionalities, the vRealize Automation plug-in includes additional helper methods for retrieving the new type of entities.

The following list shows the new helper methods that are available in the vRealize Automation plug-in.

```
vCACCafeEntitiesFinder class

findCatalogResourceTypes(vCACCAFEHost host, String query)        vCACCAFEResourceType[]
findCompositeBlueprints(vCACCAFEHost host, String query)  vCACCAFECompositeBlueprint[]
findCustomResourcesAndMappings(vCACCAFEHost host, String query)   vCACCAFECsResourceType[]
findEventTopics(vCACCAFEHost host, String query)       vCACCAFEEventTopic[]
findResourceMappings(vCACCAFEHost host, String query)      vCACCAFECsResourceType[]
findSubtenants(vCACCAFEHost host, String query)        vCACCAFESubtenant[]
findSystemWorkflowSubscriptions(vCACCAFEHost host, String query)    vCACCAFEWorkflowSubscription[]
findTenantWorkflowSubscriptions(vCACCAFEHost host, String query)    vCACCAFEWorkflowSubscription[]
getCatalogResourceTypes(vCACCAFEHost host)    vCACCAFEResourceType[]
getCompositeBlueprint(vCACCAFEHost host, String blueprintId)     vCACCAFECompositeBlueprint
getCompositeBlueprints(vCACCAFEHost host)      vCACCAFECompositeBlueprint[]
getEventTopic(vCACCAFEHost host, String eventTopicId)       vCACCAFEEventTopic
getEventTopics(vCACCAFEHost host)    vCACCAFEEventTopic[]
getResourceMapping(vCACCAFEHost host, String resourceMappingId)    vCACCAFECsResourceType
getResourceMappings(vCACCAFEHost host) java.util.List
getSubtenant(vCACCAFEHost host, String subtenantId)  vCACCAFESubtenant
getSubtenants(vCACCAFEHost host)     vCACCAFESubtenant[]
getSystemWorkflowSubscription(vCACCAFEHost host, String subscriptionId)   vCACCAFEWorkflowSubscription
getSystemWorkflowSubscriptions(vCACCAFEHost host)        vCACCAFEWorkflowSubscription[]
getTenantWorkflowSubscription(vCACCAFEHost host, String subscriptionId)   vCACCAFEWorkflowSubscription
getTenantWorkflowSubscriptions(vCACCAFEHost host) vCACCAFEWorkflowSubscription[]
```

Data Mapping in vRealize Automation and vCloud Automation Center

Extensibility in vCloud Automation Center has a different data model compared to the solution provided by vRealize Automation Event Broker Service. For the purposes of the migration and to preserve compatibility, the data payload entities and properties used in vCloud Automation Center are aligned to certain properties that are available in the event broker schema.

TABLE 12 - DATA PAYLOAD ENTITIES OF THE VCLOUD AUTOMATION CENTER MAPPED TO EVENT BROKER SCHEMA PROPERTIES

vCloud Automation Center 6.x Extensibility payload Object name: `virtualMachineEntity` Type: `VCACVirtualMachine`	vRealize Automation 7.x Event Broker machine provisioning schema payload Object name: payload Type: Properties
virtualMachineID	payload.machine.id
virtualMachineName	payload.machine.name
expires	n/a
initiatorType	n/a
notes	payload.machine.properties.get("__Notes")

guestOS	??
vmUniqueID	??
platformDetails	n/a
vmCreationDate	n/a
vmDeleteDate	n/a
lastLoggedDate	n/a
lastLoggedUser	n/a
lastPowerOffDate	n/a
lastPowerOnDate	n/a
ownerExists	n/a
usageIndex	n/a
usageIndexIgnoreBy	n/a
isDeleted	n/a
isMissing	n/a
isRogue	n/a
isRunning	n/a
recCreationTime	n/a
recDeleteTime	n/a
recUpdateTime	n/a
flags	n/a
text1	n/a
text2	n/a
virtualMachineState	n/a
vmCPUs	payload.machine.properties .get("VirtualMachine.CPU.Count")
VMTotalMemoryMB	payload.machine.properties .get("VirtualMachine.Memory.Size")
vmTotalStorageGB	payload.machine.properties.get("Cafe.Shim.VirtualMachine.TotalStorageSize")
guestOSFamily	n/a
currentTask	n/a
isTemplate	n/a
vmDNSName	n/a
vmUsedStorageGB	n/a
fileLevelCloneImageName	payload.machine.properties.get("__clonefrom")
vmInitialUsedSpace	n/a
vmEstimatedUsedSpace	n/a

storagePath	payload.machine.properties.get("VirtualMachine.Storage.Name")
connectToVdi	n/a
blueprintType	payload.machne.type
machineType	n/a
isManaged	n/a
isComponent	n/a
ParentMachineID	n/a
externalReferenceId	payload.machine.externalReference
VirtualMachineTemplateID	n/a
HostReservationID	n/a
HostID	n/a
BlueprintName	machine.properties.get("VirtualMachine.Cafe.Blueprint.Name") or payload.blueprintName
ComponentName	n/a
expireDays	n/a
NeedCatalogUpdate	n/a

TABLE 13 - DATA PAYLOAD PROPERTIES OF THE VCLOUD AUTOMATION CENTER MAPPED TO EVENT BROKER SCHEMA PROPERTIES

Description	vCloud Automation Center 6.x Extensibility payload Object name: `vCACVirtualMachineProperties` Type: Properties	vRealize Automation 7.x Event Broker machine provisioning schema payload Object name: payload Type: Properties
Request Layout	VirtualMachine.Request.Layout	n/a
Lease days	VirtualMachine.LeaseDays	Payload.machine.properties.get ("__Cafe.Request.VM.LeaseDays")
	__api.request.id	Payload. requestId
	__api.request.callback.service.id	Payload.machine.properties.get("__api.request.callback.service.id")
	__Clone_DeleteSnapshotWithBlueprint	
	__Clone_SnapshotId	
	__Clone_SnapshotText	

	__Clone_Type	Payload.machine.properties.get("__Virtual Machine.ProvisioningWorkflowName")
	__clonefrom	Payload.machine.properties.get("__Virtual Machine.ProvisioningWorkflowName")
	__clonefromid	Payload.machine.properties.get("__clonefro mid")
	__displayLocationToUser	
	__Legacy.Workflow.ImpersonatingUser	Payload.machine.properties.get("__Legacy. Workflow.ImpersonatingUser")
	__Legacy.Workflow.User	Payload.machine.properties.get("__Legacy. Workflow.User")
	__Notes	Payload.machine.properties.get("__Notes")
	__request_reason	Payload.machine.properties.get("__request _reason")
	__menusecurity_expire	n/a
	__menusecurity_connectVdi	n/a
	__menusecurity_connect	n/a
	__menusecurity_turnoff	n/a
	__menusecurity_turnon	n/a
	__menusecurity_destroy	n/a
	__menusecurity_reprovision	n/a
	__menusecurity_snapshotmanagement	n/a
	__menusecurity_connectVmrc	n/a
Blueprint ID	blueprintId	Payload.machine.properties.get("__Virtual Machine.Cafe.Blueprint.Id")
	Cafe.Shim.VirtualMachine.MaxCost	

	Cafe.Shim.VirtualMachine.MinCost	
	Custom.Common.AppService.HostnameString	
	Custom.Common.AppService.NoIndexOnFirst	
	Custom.Common.ComponentMachine.HostnameString	
	Custom.Common.ComponentMachine.NoIndexOnFirst	
	Custom.Common.Hostname.OwnerShortNameIdentifier	
	Custom.Common.SetCustomHostname.Execute	
Business group ID	provisioningGroupId	n/a
	VirtualMachine.Admin.AgentID	VirtualMachine.Admin.AgentID
	VirtualMachine.Admin.HostIdentity	
	VirtualMachine.Admin.Hostname	
VM total disk usage	VirtualMachine.Admin.TotalDiskUsage	VirtualMachine.Admin.TotalDiskUsage
	VirtualMachine.Admin.UseGuestAgent	VirtualMachine.Admin.UseGuestAgent
	VirtualMachine.Admin.UUID	

Online Resources

Title	URL
vRA 7.0 and Service Now Integration	http://fetacloud.com/vra-7-0-and-service-now-integration/
vRealize Automation 7 – Part 1, What's New – Spotlight Features	http://www.virtualjad.com/2015/10/vrealize-automation-7-part-1-whats-new-spotlight-features.html

Exploring the vRealize Automation 7.0 Event Broker – Part 1	https://virtualviking.net/2016/01/07/exploring-the-vrealize-automate-7-0-event-broker-service/
Exploring the vRealize Automation 7.0 Event Broker – Part 2	https://virtualviking.net/2016/01/20/exploring-the-vrealize-automation-7-0-event-broker-part-2/
Exploring the vRealize Automation 7.0 Event Broker – Part 3	http://virtualviking.net/2016/03/24/exploring-the-vrealize-automation-7-0-event-broker-part-3/
vRealize Automation 7 – Enabling Event Broker	http://extendingclouds.com/enabling-the-event-broker/
Enabling Event Broker – Dynamic install of the app authoring agent	https://extendingclouds.com/dynamic-install-of-the-app-authoring-agent/
UPDATED FOR VRA 7: How bout we let users set their default admin or root password?	https://extendingclouds.com/updated-for-vra-7-how-bout-we-let-users-set-their-default-admin-or-root-password/
vRealize Automation 7 – Custom Email Notifications with the Event Broker	http://extendingclouds.com/vrealize-automation-7-custom-email-notifications-with-the-event-broker/
vRealize Automation 7 – Event Broker subscription installation workflows	http://extendingclouds.com/vrealize-automation-7-event-subscription-installation-workflows/
vRealize Automation 7 extensibility though vRealize Orchestrator using Event Broker Service (EBS)	http://www.sddcmaster.com/2016/02/vrealie-automation-7-extensibility.html
Retooling the Infoblox vRA Plugin to Support Event Broker	http://www.storagegumbo.com/2016/03/retooling-infoblox-vra-plugin-to.html
vRealize Automation 7.0 – New Event Broker Enhances Lifecycle Extensibility	http://blogs.vmware.com/management/2015/11/vrealize-automation-7-0-new-event-broker-enhances-lifecycle-extensibility.html
vRealize Automation 7 Custom Hostname with Event Broker (EB) Subscription	http://www.definit.co.uk/2016/03/vrealize-automation-7-custom-hostname-with-event-broker-eb-subscription/
VMware vRealize Automation – vRA7 – The dawn of the vRealize Automation 7 Event Broker	http://dailyhypervisor.com/vmware-vrealize-automation-vra7-the-dawn-of-the-vrealize-automation-7-event-broker/
vRA Live! – Extensibility Videos Published	http://www.virtualjad.com/2015/05/vra-live-extensibility-videos-published.html
vRO Code – Finding All Items in a Business Group	http://www.automate-it.today/vro-code-finding-all-items-in-a-business-group/