vRO - Cluster Node Replacement

Description:

The purpose of this workflow, is to replace a failed node in an Avi Controller Cluster. It will determine the failing node from the provided Avi Cluster, and retrieve the configuration of the mode.

This workflow allows for both OVFTool and Content Library Template deployment for the Node re deployment.

To complete the node removal, the Avi Cluster is put into single node Cluster configuration utilizing the current Leader Node. ONce the failed Node is re deployed, the cluster is built again using the original Leader and second Follower Controllers.

General:
- Avi Controller: <Drop Down list of vro Clients>
- Deployment Type: <Drop Down list of Deployment Types - ovfTool || Content Library>
- vCenter FQDN or IP Address: <Text Field for vCenter FQDN or IP Address>
- vCenter Connection: <Drop Down list of vRO vCenter Endpoints>
- vCenter Username: <Text Field for vCetner Username>
- vCenter Password: <Secure String for vCenter User Password>
- If set to true, the certificate is accepted silently and the certificate is added to the trusted store: <Boolean>
- OVA Path on VRO Appliance: <Text Field for OVA Template Path>
- Content Library Template : <Text Field for Content Library Template name>

vroClient:
Take controller selection and output Avi VRO Client Authenticated Session Object.

Action:
Execute vroClient callAction API GET request against "cluster/runtime" to retrieve Cluster Node State information.

Action:
Retrieve vCenter API Connection based on Input "vCenter Connection".

Action:
Retrieve Avi Configuration and VM Settings for failed Node VM.

Action:
Get VM power state, and power off VM if power state is ON. Also rename failed Node VM and append "-Old" to the end.

Action:
Export Cluster configuration as Array of Strings.

Export VM Configuration variables as set of variables.

Action:
Take the Input Cluster Configuration Array from the last Action.

Action:
Take the Cluster VIP IP from the Cluster Configuration Array and add elements to JSON Object.

Action:
Take the Leader Node information from the Cluster Configuration Array and add elements to JSON Object.

Action:
Take newly formed JSON Object, and execute vroClient callAction API PUT request against "cluster" to begin cluster creation.

Action:
Define the REST operation parameters (host, method and URL). the host parameter is the Input "Controller VIP IP Address" .

Action:
Continuously execute the REST operation for VIP UNTIL the response code is 200.

Decision:
Get Input "VM Deployment Type". Make Decision based on value.

Scriptable Task:
Define the OFV Command to deploy Controller 1 VM. OVF options defined by Input in Input Sections "VM Information" and "Cluster Information"

Scriptable Task:
Define the OFV Command to deploy Controller 2 VM. OVF options defined by Input in Input Sections "VM Information" and "Cluster Information"

Scriptable Task:
Define the OFV Command to deploy Controller 3 VM. OVF options defined by Input in Input Sections "VM Information" and "Cluster Information"

Scriptable Task:
Set the 3 OVF Commands as independant VRO Output Variables.

Nested Workflow Elements:
Run 3 instances of VRO default "Run SSH Command" workflows to execute the 3 OVF Commands. The SSH session are run locally against the vRO Appliance. SSH parameters (username, password, port,path and encoding are defined as hardcoded Workflow Variables. These 3 Workflow instances are run simaltaniously allowing us to deploy the VMs at the same time.

Scriptable Task:
Validate if Input "Controller 1 IP Address" begins with "https://". If it does not, then define new variable with "https://".

Define new variable for the Initial-Data API call for Controller 1.

Scriptable Task:
Validate if Input "Controller 2 IP Address" begins with "https://". If it does not, then define new variable with "https://".

Define new variable for the Initial-Data API call for Controller 2.

Scriptable Task:
Validate if Input "Controller 3 IP Address" begins with "https://". If it does not, then define new variable with "https://".

Define new variable for the Initial-Data API call for Controller 3.

Scriptable Task:
Set the new variables as independant VRO Output Variables.

Scriptable Task:
Sleep the script for 5 Minutes, to give the appliances time to come online.

Throw Exception:
By Default the VRO "Import a certificate from URL" workflow comes with a predefined Throw Exception action to handle any errors.

Workflow Elements:
Run an instance of VRO default "Import a certificate from URL" workflows to add the re deployed Controller certificate to the VRO Certificate Store. If this is not done, then any connections to the new Controller will fail.

Scriptable Task:
Utilized method outlined in https://gist.github.com/trisharia/7b62fcdf12600511b3d7e9b635981b2c.

Define the REST operation parameters (host, method and URL). the host parameter is the Input "Controller 1 IP Address" .

Scriptable Task:
Continuously execute the REST operation for Controller 1 UNTIL the response code is 200.

Action:
Take the Input "Cluster VIP IP Address" string and add elements to JSON Object.

Action:
Take the Input "Controller 1 IP Address", "Controller 2 IP Address" and "Controller 3 IP Address strings and add elements to JSON Object.

Action:
Take newly formed JSON Object, and execute vroClient callAction API PUT request against "cluster" to begin cluster creation.

Action:
Define the REST operation parameters (host, method and URL). the host parameter is the Input "Controller VIP IP Address" .

Action:
Continuously execute the REST operation for VIP UNTIL the response code is 200.

Action:
Retrieve vAPI Connection based on Input "vCenter Connection".

Action:
Retrieve vCenter API Connection based on Input "vCenter Connection".

Action:
Validate VMs dont already exist based on Inputs "Controller 1 VM Name", "Controller 2 VM Name" and "Controller 3 VM Name"

Action:
Retrieve vCenter Objects based on Inputs "Datacenter", "Cluster", "Datastore", "Folder", "Resource Pool" and "VM Network".

Action:
Deploy Node 1 using the vRO vAPI com_vmware_vcenter_ovf_library__item method. Once VM is deployed, Update vCPU, Memory and Disk configuration based on Inputs "VM Flavor" and "VM DIsk Size"

Action:
Deploy Node 2 using the vRO vAPI com_vmware_vcenter_ovf_library__item method. Once VM is deployed, Update vCPU, Memory and Disk configuration based on Inputs "VM Flavor" and "VM DIsk Size"

Action:
Deploy Node 3 using the vRO vAPI com_vmware_vcenter_ovf_library__item method. Once VM is deployed, Update vCPU, Memory and Disk configuration based on Inputs "VM Flavor" and "VM DIsk Size"