VMware Integrated OpenStack Management API

Programming Guide
Version 7.0

TE CHNI CA L WHI TE P A P E R

J UNE 20 20

V E RS I ON 1. 0

Table of Contents

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Overview

VMware Integrated OpenStack includes a Kubernetes CRD (custom resource definition) based API that provides a way to control the VIO management cluster deployment. This document describes how to access the VIO LCM cluster and CRD resources.

Before you begin

- The VIO 7.0 LCM server must be installed and running.
- To access the VIO LCM server remotely, the Kubernetes command-line tool, kubectl, must be configured to communicate with your LCM server. See Kubernetes documentation: Install and Set Up kubectl.
- You must understand how to access the VIO LCM Kubernetes Cluster. See Kubernetes documentation: Access Clusters.
- You must have a basic understanding of custom resources. See Kubernetes documentation: Custom Resources.

CRD based APIs

VIO deployment creation supports the following CRDs.

CRD: viodeployments.vio-operator.vio.vmware.com

`URL: /apis/vio-operator.vio.vmware.com/v1alpha1/namespaces/default/viodeployments`

This API creates a VIO management cluster.

Prerequisite & Limitations:

- VIO LCM servers are running with no deployment instances created in the LCM.
- VIO supports a single deployment instance.
- Once the status enters the provisioning state, objects represented by the CR are being created and you should not modify the VIO Deployment CR. The CR only provisions for deployment. Any change to the CR made after deployment is ignored.

CRD: osdeployments.vio.vmware.com

`URL: /apis/vio.vmware.com/v1alpha1/osdeployments`

This API queries a deployed VIO management cluster status.

Authentication

Access to the VIO LCM server requires either basic or token authentication.

Basic Authentication

To access the VIO LCM cluster APIs using basic authentication, you must provide VIO admin credentials, and the credentials must be base64 encoded ID and password joined by a single colon. For example, if the user name is “admin” and password is “password”, use “`admin:password`” to generate the base64 encoded string. For more information on base64 encoding generation, see https://www.base64encode.org/.

Include the basic authentication token in the API call.

```
# curl --header "Authorization: Basic YWRtaW46Vk13YXJlMSE=" --insecure -X GET
$APISERVER/api

{
  "kind": "APIVersions",
```

```
  "versions": [
    "v1"
  ],
  "serverAddressByClientCIDRs": [
    {
      "clientCIDR": "0.0.0.0/0",
      "serverAddress": "192.168.111.161:6443"
    }
  ]
}
```

Token Authentication

You can also use token-based authentication to access VIO LCM cluster APIs.

For example:

```
# kubectl config view -o jsonpath='{"Cluster name\tServer\n"}{range
.clusters[*]}{.name}{"\t"}{.cluster.server}{"\n"}{end}'

Cluster          name          Server          kubernetes

https://192.168.111.161:6443

# export CLUSTER_NAME="kubernetes"

# APISERVER=$(kubectl config view -o
jsonpath="{.clusters[?(@.name==\"$CLUSTER_NAME\")].cluster.server}")

# TOKEN=$(kubectl get secrets -o
jsonpath="{.items[?(@.metadata.annotations['kubernetes\.io/serviceaccount\.name']=='default
')].data.token}"|base64 --decode)

# curl -X GET $APISERVER/api --header "Authorization: Bearer $TOKEN" –insecure
{
  "kind": "APIVersions",
  "versions": [
    "v1"
  ],
  "serverAddressByClientCIDRs": [
    {
      "clientCIDR": "0.0.0.0/0",
      "serverAddress": "192.168.111.161:6443"
    }
  ]
}
```

viodeployments.vio-operator.vio.vmware.com

To access the viodeployments.vio-operator.vio.vmware.com CRD, you can use kubectl or the Kubernetes API.

Access using kubectl

1. Get the CR definition and spec:

   ```
   kubectl describe crd viodeployments.vio-operator.vio.vmware.com
   ```

2. Edit the spec file based on your environment in json or yaml format.

3. Create the deployment: `kubectl create -f your-deployment-cr.yaml`

4. Modify the deployment: `kubectl edit viodeployment <name of the viodeployment>`

5. Check the deployment status: `kubectl get viodeployment <name of the CR> -o json`

   For example:

```
# kubectl get viodeployment
NAME      AGE apitest   62m

# kubectl get viodeployment apitest -o json
```

Review the status to check the spec validity and make modifications required to pass the validation.

STATE	DESCRIPTION
ERROR	Short message telling which configuration is wrong
CREATING	Saving the VIO Deployment spec to etcd
VALIDATING	Now LCM takes charge, doing validation again
WAITING FOR CONTROLLERS	Creating the kubernetes worker node
PROVISIONING	Provisioning services that form OpenStack control plane
PROVISIONING ERROR	Something wrong happened during provisioning
RECONFIGURING	LCM is reconfiguring services, edit sub CR might trigger this
RUNNING	Provisioning is successful; all services are up and running

If needed, check `vio-api-app-0` pod logs for more detailed information.

```
# kubectl logs vio-api-app-0
```

Or check the vio-operator for more debug log information.

```
# kubectl get pod | grep vio-operator vio-operator-568cbb7f5b-xqpzw
1/1     Running   0          21d
# kubectl logs vio-operator-568cbb7f5b-xqpzw
```

Access using the Kubernetes API

1. Get viodeployments list:

```
# curl --header "Authorization: Bearer $TOKEN" --insecure -X GET
$APISERVER/apis/vio-
operator.vio.vmware.com/v1alpha1/namespaces/default/viodeployments/
```

2. Create a viodeployment

   You could put your deployment json data into a file, for example `viodeploy.json`

```
# curl --header "Authorization: Bearer $TOKEN" --insecure --header 'Content-Type:
application/json' -X POST --data @viodeploy.json
$APISERVER/apis/viooperator.vio.vmware.com/v1alpha1/namespaces/default/viodeployment
s
```

3. Get single viodeployment

   In the following example, `apitest` is the instance name

```
# curl --header "Authorization: Bearer $TOKEN" --insecure -X GET
$APISERVER/apis/vio-
operator.vio.vmware.com/v1alpha1/namespaces/default/viodeployments/apitest
```

   This API also returns the deployment status. The detailed value could refer to the same command in kubectl CLI section.

   For example:

```
# curl --header "Authorization: Bearer $TOKEN" --insecure -X GET
$APISERVER/apis/viooperator.vio.vmware.com/v1alpha1/namespaces/default/viodeployment
s/apitest {
    "apiVersion": "vio-operator.vio.vmware.com/v1alpha1",
    "kind": "VIODeployment",
    "metadata": {
        "creationTimestamp": "2020-05-28T06:21:39Z",
        "generation": 1,
        "name": "apitest",
        "namespace": "default",
        "resourceVersion": "6789925",         "selfLink":
"/apis/viooperator.vio.vmware.com/v1alpha1/namespaces/default/viodeployments/
apitest",
        "uid": "baab8b2e-5a97-4ce3-af3e-bf77ba98a1d2"
    },
    "spec": {
        "endpoints": {
            "hostname": "string",
            "private_vip": "string",
            "public_vip": "string"
        },
        "ha_enabled": true,
        "ip_access_enabled": true,
        ……
        ……
        ……
    },
    "status": {
        "history": [
            {
                "state": "ERROR: vCenter password of string should be base64
encoded",
                "time": "2020-05-28T06:21:39.996535Z"
            }
        ],
        "state": "ERROR: vCenter password of string should be base64 encoded",
        "time": "2020-05-28T06:21:39.996535Z"
    }
}
```

osdeployments.vio.vmware.com

To access the osdeployments.vio.vmware.com CRD, you can use kubectl or the Kubernetes API.

Access using kubectl

1. Get the CRD definition and spec:

   ```
   kubectl get crd osdeployments.vio.vmware.com
   ```

2. Get the current deployment instance name, this CR is under the Openstack namespace.

   For example:

   ```
   # kubectl get osdeployments.vio.vmware.com -n openstack
   NAME            AGE osdeployment1   21d
   ```

3. Get the deployment instance details, for example:

   With this CR, you could query the VIO LCM deployment latest status, including the spec configurations, and the desired and observed number for specific services: Nova, Cinder, Glance, Keystone, etc.

   ```
   # kubectl get osdeployments.vio.vmware.com -n openstack osdeployment1 -o json
   {
       "apiVersion": "vio.vmware.com/v1alpha1",
       "kind": "OSDeployment",
       "metadata": {
           "name": "osdeployment1",
           "namespace": "openstack",
   "selfLink":
   "/apis/vio.vmware.com/v1alpha1/namespaces/openstack/osdeployments/osdeployment1",
       },
       "spec": {
           "admin_domain_name": "default",
           "datastore": "vdnetSharedStorage",
           "ha-enabled": true,
           "ip_access_enabled": true,
           "openstack_endpoints": {
               "private_vip": "192.168.111.160",
               "public_vip": "192.168.112.200"
           },
           "region_name": "RegionOne",
           "services": [
               {
                   "conf": "Keystone:keystone1",
                   "name": "keystone1",
                   "service": "keystone"
               },
           ]
       },
       "status": {
           "deploymentProgress": 100,
           "nodes": {
               "desired": 3,
               "observed": 3
   },          "processedSpec":
   {
               "admin_domain_name": "default",
               "datastore": "vdnetSharedStorage",
               "ha-enabled": true,
   ```

```
                "ip_access_enabled": true,
                "openstack_endpoints": {
                    "private_vip": "192.168.111.160",
                    "public_vip": "192.168.112.200"
                },
                "region_name": "RegionOne",
                "services": [
                    {
                        "conf": "Keystone:keystone1",
                        "name": "keystone1",
                        "service": "keystone"
                    },
                    {
                        "conf": "NovaCompute:compute-b8b6aa6c-c12",
                        "name": "compute-b8b6aa6c-c12",
                        "service": "nova-compute"
                    }
                ],
            },
            "services": {
                "keystone": {
                    "keystone1": {
                        "controllers": [
                            {
                                "desired": 2,
                                "exclude-start-stop": false,
                                "kind": "deployment",
                                "name": "keystone-api",
                                "observed": 2
                            }
                        ],
                        "failedJobs": [],
                        "isReady": true,
                        "validation": {
                            "numberError": 0
                        }
                    }
                },
            "state": "RUNNING"
        }
    }
```


Access using the Kubernetes API

1. Get the current deployment.

   ```
   curl --header "Authorization: Bearer $TOKEN" --insecure -X GET
   $APISERVER/apis/vio.vmware.com/v1alpha1/namespaces/openstack/osdeployments
   ```

2. Get the deployment instance details.

   For example, `osdeployment1` is the deployment name:

   ```
   curl --header "Authorization: Bearer $TOKEN" --insecure -X GET
   $APISERVER/apis/vio.vmware.com/v1alpha1/namespaces/openstack/osdeployments/osdepl
   oyment1
   ```

Schema Definition

viodeployments.vio-operator.vio.vmware.com schema

To get the schema, query the CRD definition of VIO Deployment.

```
# kubectl get viodeployments apitest -o json
{
    "apiVersion": "vio-operator.vio.vmware.com/v1alpha1",
    "kind": "VIODeployment",
    "metadata": {
        "creationTimestamp": "2020-05-28T06:21:39Z",
        "generation": 1,
        "name": "apitest",
        "namespace": "default",
        "resourceVersion": "6789925",         "selfLink":
"/apis/viooperator.vio.vmware.com/v1alpha1/namespaces/default/viodeployments/
apitest",
        "uid": "baab8b2e-5a97-4ce3-af3e-bf77ba98a1d2"
    },
    "spec": {
        "endpoints": {
            "hostname": "string",
            "private_vip": "string",
            "public_vip": "string"
        },
        "ha_enabled": true,
        "ip_access_enabled": true,
...
        ...
    },
    "status": {
        "history": [
        ],
        "state": "",
        "time": "2020-05-28T06:21:39.996535Z"
    }
}
```

Sub field explanations

Name

The name of the deployment is used as prefix for the sub CRs. For example, if the name is “viodemo”, the vCenter CR is viodemo-vcenter01, viodemo-vcenter02 and so forth. Exception: The novacompute CRs does not follow this convention.

Version

The VIO API is currently version 2.0.

ha_enabled

The flag for HA/non-HA deployment can be true|false:

- If set to true, multiple services are deployed.
- If set to false, only one service is deployed.

ip_access_enabled

The flag for how to access the OpenStack deployment. Recommended value: true.

Log_insight

The log insight server IP/Port for gathering log.

Sample:

```
    "log_insight": {
      "ip": "10.0.0.100",
      "port": 9000
    }
```

Topology

The master/worker number and flavor.

- Schema supports a single master
- Flavors are small, medium, or large

The master setting cannot be changed because the master VM is presented when the VIO vAPP is deployed.

Sample:

```
    "topology": {
      "master": {
        "count": 1,
        "flavor": "small"
      },
      "worker": {
        "count": 3,
        "flavor": "medium"
      }
}
```

vCenters

The vCenter information for management vCenter and compute vCenter. You must provide one management vCenter that can also serve as a compute vCenter. You can add more compute vCenters as needed. The password for the vCenter should be base64 encoded.

Sample:

```
    "vcenters": [
      {
        "hostname": "192.168.111.4",
        "username": "administrator@vsphere.local",
        "password": "QWRtaW4hMjM=",
        "insecure": true,
        "is_management": true
      },
      {
        "hostname": "192.168.111.135",
        "username": "administrator@vsphere.local",
        "password": "QWRtaW4hMjM=",
        "insecure": true,
        "is_management": false
      }
    ]
```

Management_cluster

The locations to place worker or controller nodes: datacenter, datastore, and resource pool.

Sample:

```
"management_cluster": {
  "datacenter": "os-test-dc",
  "datastore": "vdnetSharedStorage",
  "resourcepool": "rp-vio"
}
```

Networks

The network information for management, api, and dvs_trunk_network. VIO supports both static IP and DHCP, but for production static IP is preferred. Currently, the cluster API only supports a single ip_block.

NSX Policy Sample:

```
"networks": [
  {
    "name": "VM Network",
    "dns": [
      "192.168.111.1"
    ],
    "gateway": "192.168.111.1",
    "netmask": "255.255.255.0",
    "type": "management",
    "ip_ranges": [
      {
        "begin_ip": "192.168.111.183",
        "end_ip": "192.168.111.185"
      }
    ]
  },
  {
    "name": "vio-dvpg",
    "dns": [
      "192.168.112.1"
    ],
    "gateway": "192.168.112.1",
    "type": "api"
  }
]
```

DVS sample

DVS Trunk Network is required for DVS deployment.

```
"networks": [
  {
    "name": "VM Network",
    "dns": [
      "192.168.111.1"
    ],
    "gateway": "192.168.111.1",
    "netmask": "255.255.255.0",
    "type": "management",
    "ip_ranges": [
      {
```

```
            "begin_ip": "192.168.111.183",
            "end_ip": "192.168.111.185"
          }
        ]
      },
{
"name":
"vio-
dvpg",
        "dns": [
          "192.168.112.1"
        ],
        "gateway": "192.168.112.1",
        "type": "api"
      },
      {
        "name": "vdnet-trunk",
        "type": "dvs_trunk_network",
        "ip_ranges": [
          {
            "begin_ip": "169.254.0.1",
            "end_ip": "169.254.0.254"
          }
        ]
      }
    ]
```

Region_name

The region name of OpenStack deployment.

Sample:

```
"region_name": "RegionOne"
```

Endpoints

The endpoint information for OpenStack deployment.

Sample:

```
   "endpoints": {
      "hostname": "demo.vio.vmware.com",
      "private_vip": "192.168.111.160",
      "public_vip": "192.168.112.200"
    }
```

Attributes

The additional attributes of this deployment. You can provide additional information about the deployment here.

NOTE: If a large environment includes many objects, discovery might require more time. To customize the duration for validation interval, set validation_wait_timeout.

Sample:

```
"attributes": {
      "validation_wait_timeout": 30
}
```

Openstack_info

The OpenStack core services information includes Keystone, Glance, Cinder, Nova, Neutron.

Keystone

For the Keystone related information, you can specify the domain, username, password for Keystone domains, and you can also specify LDAP information for Keystone. Sensitive information such as admin_password, ldap_password should be base64 encoded.

Sample:

```
      "identity": {
        "admin_domain_name": "default",
        "admin_user": "admin",
        "admin_password": "cGFzc3dvcmQ=",
        "token_expiration_time": 7200
      }
            Another sample with LDAP information:
      "identity": {
        "admin_domain_name": "default",
        "admin_user": "admin",
        "admin_password": "dm13YXJl",
        "token_expiration_time": 7200,
        "ldap_backends": [
          {
            "ad_domain_names": "vio.com",
            "admin_user": "testuser@vio.com",
            "chase_referrals": false,
            "group_desc_attribute": "description",
            "group_filter": "(CN=VMware*)",
            "group_id_attribute": "cn",
            "group_member_attribute": "member",
            "group_members_are_ids": false,
            "group_name_attribute": "sAMAccountName",
            "group_objectclass": "group",
            "group_tree_dn": "OU=Distribution Groups,OU=Groups,OU=Corp,DC=vio,DC=com",
            "ldap_loadbalancer": false,
            "name": "domain1",
            "page_size": 100,
            "password": "cGFzc3dvcmQ=",
            "query_scope": "sub",
            "url": "ldap://server1.vio.com:389",
            "use_tls": false,
            "user": "vio-autouser@vio.com",
            "user_enabled_attribute": "userAccountControl",
"user_enabled_mask": 2,
            "user_filter": "(|(memberof=CN=VIO-RD,OU=Distribution
Groups,OU=Groups,OU=Corp,DC=vio,DC=com)(sAAMAccountName=vio-autouser))",
            "user_id_attribute": "cn",
            "user_mail_attribute": "mail",
            "user_name_attribute": "userPrincipalName",
            "user_objectclass": "organizationalPerson",
            "user_pass_attribute": "userPassword",
            "user_tree_dn": "cn=Users,dc=vio,dc=com"
          }
        ]
```

```
}
```

Glance

The Glance information for the OpenStack deployment. Each compute vCenter should have a backend section.

Sample:

```
"image": {
  "backends": [
    {
      "vcenter_name": "192.168.111.4",
      "datastores": [
        "vdnetSharedStorage"
      ]
    },
    {
      "vcenter_name": "192.168.111.135",
      "datastores": [
        "vdnetSharedStorage"
      ]
    }
  ]
}
```

Cinder

The Cinder information for the OpenStack deployment. Each compute cluster should have a Cinder backend section. The default driver is vmdk.

Sample:

```
"volume": {
  "backends": [
    {
      "availability_zone_name": "zone1",
      "clusters": [
        "compute_cluster"
],
       "vcenter_name": "192.168.111.4",
      "driver": "vmdk"
    },
    {
      "availability_zone_name": "zone2",
      "clusters": [
        "compute_cluster"
      ],
      "vcenter_name": "192.168.111.135"
    }
  ],
  "default_availability_zone_name": "zone1"
}
```

Nova

The compute clusters for OpenStack.

Sample:

```
"compute": {
  "compute_clusters": [
```

```
        {
          "vcenter_name": "192.168.111.4",
          "cluster_name": "compute_cluster",
          "datastore_regex": "vdnetSharedStorage",
          "availability_zone_name": "zone1"
        },
        {
          "vcenter_name": "192.168.111.135",
          "cluster_name": "compute_cluster",
          "datastore_regex": "vdnetSharedStorage",
          "availability_zone_name": "zone2"
        }
      ],
      "default_availability_zone_name": "zone1",
      "passthrough": true,
      "tenant_vdc": true
    }
  }
```

Neutron

The networks for OpenStack deployment. VIO supports dvs, nsxv, nsxt, or nsxp plugins.

For DVS, configure dvs_trunk_network in the network section and use the dvs backend.

Sample:

```
  "network": {
    "neutron_backend": "dvs",
    "dvs_name": "vio-dvs"
  }
}
```

For NSXT or NSXP, specify the following attributes:

- nsx_api_password and metadata_proxy_shared_secret should be base64 encoded. The field accepts both UUID and Name as input.
- NSXT is for NSX-T MP plug-in.
- NSXP is for NSX-T Policy plug-in.

Sample:

```
  "network": {
    "neutron_backend": "nsxt",
    "nsx": {
      "default_overlay_tz": "vio-overlay-tz",
      "default_tier0_router": "PLR-1 LogicalRouterTier0",
      "default_vlan_tz": "transportzone2",
      "dhcp_profile": "vio-dhcp-profile",
      "metadata_proxy": "vio-md-proxy",
      "metadata_proxy_shared_secret": "cGFzc3dvcmQ=",
      "nsx_api_managers": "192.168.111.146",
      "nsx_api_user": "admin",
      "nsx_api_password": "QWRtaW4hMjNBZG1pbg==",
      "insecure": true,
      "ens_support": true
    }
  }
```

For each service listed, you can fine tune the service by providing additional attributes such as the number of services for a particular service. In the following example, the spec limits the glance-api service to 1 instance.

Sample:

```
      "image": {
        "backends": [
          {
            "vcenter_name": "192.168.111.4",
            "datastores": [
              "vdnetSharedStorage"
            ]
          },
          {
            "vcenter_name": "192.168.111.135",
            "datastores": [
              "vdnetSharedStorage" ]
          }
        ],
        "attributes": {
          "pod": {
            "replicas": {
              "api": 1
            }
          }
        }
      }
```

CR Spec Examples in JSON format

To deploy a VIO management cluster with an NSX-T Policy or DVS network backend, you can use the examples provided and modify kubectl or API calls as needed. Or use the raw template to create a new CR spec.

Deployment with NSX-T Policy Backend Example

```
{
  "apiVersion": "vio-operator.vio.vmware.com/v1alpha1",
  "kind": "VIODeployment",
  "metadata": {
    "name": "site1"
  },
  "spec": {
    "attributes": {
      "validation_wait_timeout": 30
    },
    "name": "VIODemo",
    "version": "2.0",
    "vcenters": [
      {
        "hostname": "192.168.111.4",
        "username": "administrator@vsphere.local",
        "password": "QWRtaW4hMjM=",
        "insecure": true,
        "is_management": true
      },
      {
        "hostname": "192.168.111.135",
        "username": "administrator@vsphere.local",
```

```
        "password": "QWRtaW4hMjM=",
        "insecure": true,
        "is_management": false
      }
    ],
    "management_cluster": {
      "datacenter": "os-test-dc",
      "datastore": "vdnetSharedStorage",
      "resourcepool": "rp-xstack"
    },
    "networks": [
      {
        "name": "VM Network",
        "dns": [
          "192.168.111.1"
        ],
        "gateway": "192.168.111.1",
        "netmask": "255.255.255.0",
        "type": "management",
        "ip_ranges": [
          {
            "begin_ip": "192.168.111.183",
            "end_ip": "192.168.111.185"
          }
        ]
      },
      {
        "name": "vio-dvpg",
        "dns": [
          "192.168.112.1"
        ],
        "gateway": "192.168.112.1",
        "type": "api"
      }
    ],
    "endpoints": {
      "hostname": "demo.vio.vmware.com",
      "private_vip": "192.168.111.181",
      "public_vip": "192.168.112.201"
    },
    "openstack_info": {
      "identity": {
        "admin_domain_name": "default",
        "admin_user": "admin",
        "admin_password": "dm13YXJl",
"token_expiration_time": 7200
      },
      "image": {
        "backends": [
          {
            "vcenter_name": "192.168.111.4",
            "datastores": [
              "vdnetSharedStorage" ]
          },
          {
            "vcenter_name": "192.168.111.135",
```

```
            "datastores": [
              "vdnetSharedStorage"
            ]
          }
        ]
      },
      "network": {
        "neutron_backend": "nsxp",
        "nsx": {
          "default_overlay_tz": "vio-overlay-tz",
          "default_tier0_router": "PLR-1 LogicalRouterTier0",
          "default_vlan_tz": "transportzone2",
          "dhcp_profile": "vio-dhcp-profile",
          "metadata_proxy": "vio-md-proxy",
          "metadata_proxy_shared_secret": "cGFzc3dvcmQ=",
          "nsx_api_managers": "192.168.111.146",
          "nsx_api_user": "admin",
          "nsx_api_password": "QWRtaW4hMjNBZG1pbg==",
          "insecure": true,
          "ens_support": true
        }
      },
      "volume": {
        "backends": [
          {
            "availability_zone_name": "zone1",
            "clusters": [
              "compute_cluster"
            ],
            "vcenter_name": "192.168.111.4"
          },
          {
            "availability_zone_name": "zone2",
            "clusters": [
              "compute_cluster"
            ],
            "vcenter_name": "192.168.111.135"
          }
        ],
        "default_availability_zone_name": "zone1"
      },
      "compute": {
        "compute_clusters": [
{
            "vcenter_name": "192.168.111.4",
            "cluster_name": "compute_cluster",
            "datastore_regex": "vdnetSharedStorage",
            "availability_zone_name": "zone1"
          },
          {
            "vcenter_name": "192.168.111.135",
            "cluster_name": "compute_cluster",
            "datastore_regex": "vdnetSharedStorage",
            "availability_zone_name": "zone2"
          }
        ],
```

```
        "default_availability_zone_name": "zone1",
        "passthrough": true,
        "tenant_vdc": true
      }
    },
    "region_name": "RegionOne",
    "topology": {
      "master": {
        "count": 1,
        "flavor": "small"
      },
      "worker": {
        "count": 3,
        "flavor": "medium"
      }
    },
    "log_insight": {
      "ip": "192.168.111.50",
      "port": 9000
    },
    "ip_access_enabled": true,
    "ha_enabled": true
  }
}
```

Deployment with DVS Network Backend Example

```
{
  "apiVersion": "vio-operator.vio.vmware.com/v1alpha1",
  "kind": "VIODeployment",
  "metadata": {
    "name": "apitest"
  },
  "spec": {
    "attributes": {
      "validation_wait_timeout": 30
    },
    "name": "cidvs",
    "version": "2.0",
    "vcenters": [
      {
        "hostname": "192.168.111.21",
        "username": "administrator@vsphere.local",
        "password": "QWRtaW4hMjM=",
        "insecure": true,
        "is_management": true
      }
    ],
    "management_cluster": {
      "datacenter": "vio-datacenter",
      "datastore": "vdnetSharedStorage",
      "resourcepool": "rp_k8s"
    },
    "networks": [
      {
```

```
          "name": "VM Network",
          "dns": [
            "192.168.111.1"
          ],
          "gateway": "192.168.111.1",
          "netmask": "255.255.255.0",
          "type": "management",
          "ip_ranges": [
            {
              "begin_ip": "192.168.111.183",
              "end_ip": "192.168.111.185"
            }
          ]
        },
        {
          "name": "vio-dvpg",
          "dns": [
            "192.168.112.1"
          ],
          "gateway": "192.168.112.1",
          "type": "api"
        },
        {
          "name": "vdnet-trunk",
          "type": "dvs_trunk_network",
         "ip_ranges": [
            {
              "begin_ip": "169.254.0.1",
"end_ip": "169.254.0.254"
            }
          ]
        }
    ],
    "endpoints": {
      "hostname": "demo.vio.vmware.com",
      "private_vip": "192.168.111.160",
      "public_vip": "192.168.112.200"
    },
    "openstack_info": {
      "identity": {
        "admin_domain_name": "default",
        "admin_user": "admin",
        "admin_password": "cGFzc3dvcmQ=",
        "token_expiration_time": 7200
      },
      "image": {
        "backends": [
          {
            "vcenter_name": "192.168.111.21",
            "datastores": [
              "vdnetSharedStorage"
```

```
          ]
        }
      ],
      "attributes": {
        "pod": {
          "replicas": {
            "api": 1
          }
        }
      }
    },
    "network": {
      "neutron_backend": "dvs",
      "dvs_name": "vio-dvs"
    },
    "volume": {
      "backends": [
        {
         "availability_zone_name": "zone1",
          "clusters": [
            "compute_cluster"
          ],
          "vcenter_name": "192.168.111.21"
        }
      ],
      "default_availability_zone_name": "zone1"
    },
    "compute": {
      "compute_clusters": [
        {
          "vcenter_name": "192.168.111.21",
          "cluster_name": "compute_cluster",
          "datastore_regex": "vdnetSharedStorage",
          "availability_zone_name": "zone1"
        }
      ],
      "default_availability_zone_name": "zone1",
      "passthrough": true,
      "tenant_vdc": true
    }
  },
  "region_name": "RegionOne",
  "topology": {
    "master": {
      "count": 1,
      "flavor": "small"
    },
    "worker": {
      "count": 1,
      "flavor": "medium"
    }
```

```
    },
    "log_insight": {
      "ip": "192.168.111.50",
      "port": 9000
    },
    "ip_access_enabled": true,
    "ha_enabled": false
  }
}
```

Raw Template

```
{
  "apiVersion": "vio-operator.vio.vmware.com/v1alpha1",
  "kind": "VIODeployment",
  "metadata": {
    "name": "apitest"
  },
  "spec": {
  "attributes": {
    "property1": {},
    "property2": {}
  },
  "endpoints":  {
    "hostname": "string",
    "private_vip": "string",
    "public_vip": "string"
  },
  "ha_enabled": true,
  "ip_access_enabled": true,
  "log_insight": {
    "ip": "string",
    "port": 0
  },
  "management_cluster": {
    "datacenter": "string",
    "datastore": "string",
    "resourcepool": "string"
  },
  "name": "string",
  "networks": [
    {
      "dns": [
        "string"
      ],
      "gateway": "string",
      "ip_ranges": [
        {
          "begin_ip": "string",
          "end_ip": "string"
        }
      ],
      "name": "string",
      "netmask": "string",
      "type": "string"
```

```
    }
  ],
  "openstack_info": {
    "compute": {
      "attributes": {
        "property1": {},
        "property2": {}
      },
      "compute_clusters": [
        {
          "availability_zone_name": "string",
          "cluster_name": "string",
          "datastore_regex": "string",
          "dvs_moid": "string",
          "vcenter_name": "string"
        }
      ],
      "default_availability_zone_name": "string",
      "passthrough": true,
      "tenant_vdc": true
    },
    "identity": {
      "admin_domain_name": "string",
      "admin_password": "string",
      "admin_user": "string",
      "attributes": {
        "property1": {},
        "property2": {}
      },
      "ldap_backends": [
        {
          "ad_domain_controllers": "string",
          "ad_domain_names": "string",
          "ad_site": "string",
          "admin_user": "string",
          "chase_referrals": true,
          "group_desc_attribute": "string",
          "group_filter": "string",
          "group_id_attribute": "string",
          "group_member_attribute": "string",
          "group_members_are_ids": true,
          "group_name_attribute": "string",
          "group_objectclass": "string",
          "group_tree_dn": "string",
          "ldap_loadbalancer": true,
          "name": "string",
          "page_size": 0,
          "password": "string",
          "query_scope": "string",
"url": "string",           "use_tls":
true,
          "user": "string",
          "user_enabled_attribute": "string",
          "user_enabled_mask": 0,
          "user_filter": "string",
          "user_id_attribute": "string",
```

```
            "user_mail_attribute": "string",
            "user_name_attribute": "string",
            "user_objectclass": "string",
            "user_pass_attribute": "string",
            "user_tree_dn": "string"
          }
        ],
        "token_expiration_time": 0
      },
      "image": {
        "attributes": {
          "property1": {},
          "property2": {}
        },
        "backends": [
          {
            "datastores": [
              "string"
            ],
            "vcenter_name": "string"
          }
        ]
      },
      "network": {
        "attributes": {
          "property1": {},
          "property2": {}
        },
        "dns_designate_enabled": true,
        "dvs_name": "string",
        "neutron_backend": "string",
        "nsx": {
          "default_overlay_tz": "string",
          "default_tier0_router": "string",
          "default_vlan_tz": "string",
          "dhcp_profile": "string",
          "ens_support": true,
          "insecure": true,
          "metadata_proxy": "string",
          "metadata_proxy_shared_secret": "string",
"native_dhcp_metadata": true,          "nsx_api_managers": "string",
          "nsx_api_password": "string",
          "nsx_api_user": "string"
        },
        "nsxv": {
          "cluster_moid": "string",
          "datacenter_moid": "string",
          "datastore_id": "string",
          "dvs_id": "string",
          "external_network": "string",
"insecure": true,
          "nsx_api_managers": "string",
          "nsx_api_password": "string",
"nsx_api_user": "string",
          "resource_pool_id": "string",
          "vdn_scope_id": "string"
```

```
      }
    },
    "volume": {
      "attributes": {
        "property1": {},
        "property2": {}
      },
      "backends": [
        {
          "availability_zone_name": "string",
          "clusters": [
            "string"
          ],
          "driver": "string",
          "vcenter_name": "string"
        }
      ],
      "default_availability_zone_name": "string"
    }
  },
  "region_name": "string",
  "topology": {
    "master": {
      "count": 0,
      "flavor": "string"
    },
    "worker": {
      "count": 0,
      "flavor": "string"
    }
},
"vcenter
s": [
    {
      "hostname": "string",
      "insecure": true,
      "is_management": true,
      "name": "string",
      "password": "string",
      "username": "string"
    }
  ],
  "version": "string"
}
}
```