Provider APIs

DCS Provider APIs

DCSCluster [infrastructure.clus DCSIpHostnamePool [infrastru DCSMachin

DCSMachineTemplate [infrastructure.cluster.x-k8s.io/v1beta1]

HCS Provider APIs

HCSCluster [infrastructure.clus HCSMachine [infrastructure.clu HCSMachin

HCSMachineConfigPool [infrastructure.cluster.x-k8s.io/v1beta1]

vSphere Provider APIs

VSphereCluster [infrastructure.
VSphereClusterIdentity [infrast
VSphereMa

VSphereClusterTemplate
VSphereVM [infrastructure.clus
VSphereMa

VSphereMachineConfigPool [infrastructure.cluster.x-k8s.io/v1beta1]

VSphereFailureDomain [infrast
VSphereDe

Bare Metal Provider APIs

BaremetalCluster [infrastructur
BaremetalMachine [infrastructu
BaremetalM

MachineInventoryPool [infrastr
MachineRegistration [elementa
SeedImage

MachineInventory [elemental.cattle.io/v1beta1]

User-Provisioned Infrastructure Provider APIs

UserProvisionedCluster	UserProvisionedMachine	UserProvisi

DCS Provider APIs

DCSCluster [infrastructure.clus DCSIpHostnamePool [infrastru DCSMachin

DCSMachineTemplate [infrastructure.cluster.x-k8s.io/v1beta1]

DCSCluster [infrastructure.cluster.x-k8s.io/v1beta1]

Description

DCSCluster is the Schema for the dcsclusters API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DCSClusterSpec defines the desired state of DCSCluster.
`status`	`object`	DCSClusterStatus defines the observed state of DCSCluster.

.spec

Description

DCSClusterSpec defines the desired state of DCSCluster.

Type

`object`

Required

`controlPlaneLoadBalancer` `credentialSecretRef` `networkType`

Property	Type	Description
`controlPlaneEndpoint`	`object`	ControlPlaneEndpoint represents the endpoint used to communicate with the control plane.

Property	Type	Description
`controlPlaneHA`	`object`	ControlPlaneHA enables provider-managed DRS mutual-exclusion rules for control-plane VMs. The controller maintains the rule after VM URNs exist; DCS is responsible for executing DRS placement.
`controlPlaneLoadBalancer`	`object`	
`credentialSecretRef`	`object`	CredentialSecretRef
`encryptionProviderConfigRef`	`object`	EncryptionProviderConfigRef references a Secret containing a pre-existing encryption-provider.conf for the cluster. When set, the controller uses this config instead of generating a new random encryption key. Required for disaster recovery: the standby cluster must use the same encryption key as the primary cluster so that restored etcd data can be decrypted by the API server. The referenced Secret must contain a key "encryption-provider.conf".
`networkType`	`string`	
`site`	`string`	

.spec.controlPlaneEndpoint

Description

ControlPlaneEndpoint represents the endpoint used to communicate with the control plane.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	host is the hostname on which the API server is serving.
`port`	`integer`	port is the port on which the API server is serving.

.spec.controlPlaneHA

Description

ControlPlaneHA enables provider-managed DRS mutual-exclusion rules for control-plane VMs. The controller maintains the rule after VM URNs exist; DCS is responsible for executing DRS placement.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled turns on provider-managed DRS mutual exclusion for control-plane VMs.

.spec.controlPlaneLoadBalancer

Type

`object`

Required

`host` `port` `type`

Property	Type	Description
`host`	`string`	
`interface`	`string`	Interface is the NIC used by bootstrap VIP and alive. Empty means auto-detect by node IP.
`port`	`integer`	
`type`	`string`	
`vrid`	`integer`	

.spec.credentialSecretRef

Description

CredentialSecretRef

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.encryptionProviderConfigRef

Description

EncryptionProviderConfigRef references a Secret containing a pre-existing encryption-provider.conf for the cluster. When set, the controller uses this config instead of generating a new random encryption key. Required for disaster recovery: the standby cluster must use the same encryption key as the primary cluster so that restored etcd data can be decrypted by the API server. The referenced Secret must contain a key "encryption-provider.conf".

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.status

Description

DCSClusterStatus defines the observed state of DCSCluster.

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions provide observations of the operational state of a Cluster API resource.

Property	Type	Description
`controlPlaneHA`	`object`	ControlPlaneHA records the provider-managed DRS rule currently owned by this cluster. Members are a snapshot from the latest reconcile.
`failureDomains`	`object`	FailureDomains is a slice of FailureDomains.
`ready`	`boolean`	

.status.conditions

Description

Conditions provide observations of the operational state of a Cluster API resource.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This

Property	Type	Description
		should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see

Property	Type	Description
		.node.status.conditions), the ability to deconflict is important.

.status.controlPlaneHA

Description

ControlPlaneHA records the provider-managed DRS rule currently owned by this cluster. Members are a snapshot from the latest reconcile.

Type

`object`

Property	Type	Description
`clusterUrn`	`string`	ClusterUrn is the DCS compute cluster URN where the rule is maintained.
`members`	`array`	Members is the latest control-plane VM membership snapshot.
`ruleIndex`	`integer`	RuleIndex is the DCS rule index returned by cluster detail. It must be used together with RuleName validation.
`ruleName`	`string`	RuleName is the deterministic provider-generated DRS rule name.

.status.controlPlaneHA.members

Description

Members is the latest control-plane VM membership snapshot.

Type

`array`

.status.controlPlaneHA.members[]

Description

ControlPlaneHAMember records one control-plane VM included in the DRS rule.

Type

`object`

Property	Type	Description
`machineName`	`string`	MachineName is the owning CAPI Machine name when known.
`vmUrn`	`string`	VMUrn is the DCS VM URN used as the DRS rule member.

.status.failureDomains

Description

FailureDomains is a slice of FailureDomains.

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsclusters`
  - `DELETE`: delete collection of DCSCluster
  - `GET`: list objects of kind DCSCluster
  - `POST`: create a new DCSCluster
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsclusters/{name}`
  - `DELETE`: delete the specified DCSCluster
  - `GET`: read the specified DCSCluster
  - `PATCH`: partially update the specified DCSCluster
  - `PUT`: replace the specified DCSCluster
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsclusters/{name}/status`
  - `GET`: read status of the specified DCSCluster
  - `PATCH`: partially update status of the specified DCSCluster
  - `PUT`: replace status of the specified DCSCluster

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsclusters

HTTP method

`DELETE`

Description

delete collection of DCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind DCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`DCSClusterList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSCluster` schema
201 - Created	`DCSCluster` schema
202 - Accepted	`DCSCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsclusters/{name}

HTTP method

`DELETE`

Description

delete the specified DCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified DCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`DCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSCluster` schema
201 - Created	`DCSCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsclusters/{name}/status

HTTP method

`GET`

Description

read status of the specified DCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`DCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSCluster` schema

HTTP code	Response body
201 - Created	`DCSCluster` schema
401 - Unauthorized	Empty

DCSIpHostnamePool [infrastructure.cluster.x-k8s.io/v1beta1]

Description

DCSIpHostnamePool is the Schema for the dcsiphostnamepools API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DCSIpHostnamePoolSpec defines the desired state of DCSIpHostnamePool.
`status`	`object`	DCSIpHostnamePoolStatus defines the observed state of DCSIpHostnamePool.

.spec

Description

DCSIpHostnamePoolSpec defines the desired state of DCSIpHostnamePool.

Type

`object`

Required

`pool`

Property	Type	Description
`pool`	`array`	

.spec.pool

Type

`array`

.spec.pool[]

Type

`object`

Property	Type	Description
`additionNic`	`array`	
`dns`	`string`	
`dvSwitchName`	`string`	add for multi nic support if nic DVSwitchName or PortGroupName is nil, we will use vmtemplate DVSwitchName and PortGroupName
`gateway`	`string`	
`hostname`	`string`	
`ip`	`string`	
`ipv6`	`string`	
`machineName`	`string`	
`mask`	`string`	
`persistentDisk`	`array`	PersistentDisk defines disks that should survive VM upgrades (delete-recreate). Each entry is bound to (ip, slot); slot determines attach order and guest device name.
`portGroupName`	`string`	

Property	Type	Description
`portGroupUrn`	`string`	

.spec.pool[].additionNic

Type

`array`

.spec.pool[].additionNic[]

Type

`object`

Property	Type	Description
`dns`	`string`	
`dvSwitchName`	`string`	
`gateway`	`string`	
`ip`	`string`	
`ipv6`	`string`	
`mask`	`string`	
`portGroupName`	`string`	
`portGroupUrn`	`string`	

.spec.pool[].persistentDisk

Description

PersistentDisk defines disks that should survive VM upgrades (delete-recreate). Each entry is bound to (ip, slot); slot determines attach order and guest device name.

Type

`array`

.spec.pool[].persistentDisk[]

Description

PersistentDisk describes a disk that survives VM upgrades on an IP slot. Identity key is (ip, slot). Path/Format are guest-side concerns: when both are nil the disk runs in raw block mode — the controller still attaches it to the VM, but Ignition skips mkfs/mount, leaving the device for guest-side applications (e.g. Ceph OSD) to manage.

Type

`object`

Required

`quantityGB` `slot`

Property	Type	Description
`datastoreClusterName`	`string`	Provide exactly one of DatastoreClusterName or DatastoreName.
`datastoreName`	`string`	
`format`	`string`	Format is the filesystem (xfs/ext4/...). Optional: when nil and Path is also nil, the disk runs in raw block mode; when Path is set but Format is nil, the Ignition template falls back to ext4. Immutable after creation.

Property	Type	Description
`isThin`	`boolean`	IsThin is creation-only: it is sent only when creating new DCS persistent volumes, and is not validated or reconciled for existing volumes. Nil keeps the platform default.
`mountOptions`	`array`	
`options`	`array`	Options: mkfs flags. Immutable after creation.
`path`	`string`	Path is the guest mount point (e.g. /var/lib/etcd). Optional: when nil, Ignition does not render mkfs/mount for this disk (raw block mode).
`pciType`	`string`	PciType defaults to "VIRTIO". Immutable after creation.
`quantityGB`	`integer`	
`slot`	`integer`	Slot is the disk position (>= 0), unique per IP. Determines attach order, sequenceNum and guest device name. Immutable.

.spec.pool[].persistentDisk[].mountOptions

Type

`array`

.spec.pool[].persistentDisk[].mountOptions[]

Type

`string`

.spec.pool[].persistentDisk[].options

Description

Options: mkfs flags. Immutable after creation.

Type

`array`

.spec.pool[].persistentDisk[].options[]

Type

`string`

.status

Description

DCSIpHostnamePoolStatus defines the observed state of DCSIpHostnamePool.

Type

`object`

Property	Type	Description
`persistentDiskStatus`	`array`	PersistentDiskStatus survives DCSMachine delete-recreate; used for volume coordination and crash recovery.

.status.persistentDiskStatus

Description

PersistentDiskStatus survives DCSMachine delete-recreate; used for volume coordination and crash recovery.

Type

`array`

.status.persistentDiskStatus[]

Description

PersistentDiskStatus is the runtime state of a persistent volume at a (ip, slot).

Type

`object`

Required

`ip` `slot`

Property	Type	Description
`attachedVmUrn`	`string`	
`datastoreUrn`	`string`	
`ip`	`string`	
`lastError`	`string`	
`lastTransitionTime`	`string`	
`name`	`string`	Name is the DCS volume name; format: {poolName}-{sha256(ip/slot)[:8]}. Used for idempotent CreateVolume-by-name lookup.
`ownerMachineName`	`string`	

Property	Type	Description
`ownerMachineUID`	`string`	OwnerMachineUID is retained across Machine delete so rolling upgrade can hand the record to the next Machine rather than orphan it.
`path`	`string`	
`phase`	`string`	DiskPhase is the coordination phase of a persistent disk record.
`slot`	`integer`	
`volumeUrn`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsiphostnamepools`
  - `DELETE`: delete collection of DCSIpHostnamePool
  - `GET`: list objects of kind DCSIpHostnamePool
  - `POST`: create a new DCSIpHostnamePool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsiphostnamepools/{name}`
  - `DELETE`: delete the specified DCSIpHostnamePool
  - `GET`: read the specified DCSIpHostnamePool
  - `PATCH`: partially update the specified DCSIpHostnamePool
  - `PUT`: replace the specified DCSIpHostnamePool

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsiphostnamepools/{name}/status`
  - `GET`: read status of the specified DCSIpHostnamePool
  - `PATCH`: partially update status of the specified DCSIpHostnamePool
  - `PUT`: replace status of the specified DCSIpHostnamePool

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsiphostnamepools

HTTP method

`DELETE`

Description

delete collection of DCSIpHostnamePool

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind DCSIpHostnamePool

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePoolList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DCSIpHostnamePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSIpHostnamePool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePool` schema
201 - Created	`DCSIpHostnamePool` schema
202 - Accepted	`DCSIpHostnamePool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsiphostnamepools/{name}

HTTP method

`DELETE`

Description

delete the specified DCSIpHostnamePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema

HTTP code	Response body
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified DCSIpHostnamePool

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DCSIpHostnamePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DCSIpHostnamePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing

Parameter	Type	Description
		unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSIpHostnamePool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePool` schema
201 - Created	`DCSIpHostnamePool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsiphostnamepools/{name}/status

HTTP method

`GET`

Description

read status of the specified DCSIpHostnamePool

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DCSIpHostnamePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DCSIpHostnamePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSIpHostnamePool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSIpHostnamePool` schema
201 - Created	`DCSIpHostnamePool` schema
401 - Unauthorized	Empty

DCSMachine [infrastructure.cluster.x-k8s.io/v1beta1]

Description

DCSMachine is the Schema for the dcsmachines API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DCSMachineSpec defines the desired state of DCSMachine.
`status`	`object`	DCSMachineStatus defines the observed state of DCSMachine.

.spec

Description

DCSMachineSpec defines the desired state of DCSMachine.

Type

`object`

Required

`ipHostPoolRef` `vmTemplateName`

Property	Type	Description
`ipHostPoolRef`	`object`	LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Property	Type	Description
`location`	`object`	
`providerID`	`string`	
`resource`	`object`	
`vmConfig`	`object`	
`vmTemplateName`	`string`	
`vmUrn`	`string`	

.spec.ipHostPoolRef

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.location

Type

`object`

Required

`type`

Property	Type	Description
`name`	`string`	
`type`	`string`	
`urn`	`string`	

.spec.resource

Type

`object`

Required

`type`

Property	Type	Description
`name`	`string`	
`type`	`string`	
`urn`	`string`	

.spec.vmConfig

Type

`object`

Required

`dcsMachineCpuSpec` `dcsMachineMemorySpec`

Property	Type	Description
`cdRomDatastoreClusterName`	`string`	
`cdRomDatastoreName`	`string`	
`dcsMachineCpuSpec`	`object`	
`dcsMachineDiskSpec`	`array`	
`dcsMachineMemorySpec`	`object`	
`dvSwitchName`	`string`	
`portGroupName`	`string`	
`portGroupUrn`	`string`	

.spec.vmConfig.dcsMachineCpuSpec

Type

`object`

Required

`quantity`

Property	Type	Description
`quantity`	`integer`	

.spec.vmConfig.dcsMachineDiskSpec

Type

`array`

.spec.vmConfig.dcsMachineDiskSpec[]

Type

`object`

Required

`quantity`

Property	Type	Description
`datastoreClusterName`	`string`	
`datastoreName`	`string`	
`datastoreUrn`	`string`	
`format`	`string`	
`mountOptions`	`array`	
`options`	`array`	
`path`	`string`	
`quantity`	`integer`	
`systemVolume`	`boolean`	
`wipeFilesystem`	`boolean`	

.spec.vmConfig.dcsMachineDiskSpec[].mountOptions

Type

`array`

.spec.vmConfig.dcsMachineDiskSpec[].mountOptions[]

Type

`string`

.spec.vmConfig.dcsMachineDiskSpec[].options

Type

`array`

.spec.vmConfig.dcsMachineDiskSpec[].options[]

Type

`string`

.spec.vmConfig.dcsMachineMemorySpec

Type

`object`

Required

`quantity`

Property	Type	Description
`quantity`	`integer`	

.status

Description

DCSMachineStatus defines the observed state of DCSMachine.

Type

`object`

Property	Type	Description
`additionalNic`	`array`	

Property	Type	Description
`addresses`	`array`	MachineAddresses is a slice of MachineAddress items to be used by infrastructure providers.
`cdRomDatastoreUrn`	`string`	
`cdRomDetached`	`boolean`	
`cdRomFile`	`string`	
`cdRomFileDeleted`	`boolean`	
`conditions`	`array`	Conditions provide observations of the operational state of a Cluster API resource.
`failureDomains`	`object`	FailureDomains is a slice of FailureDomains.
`networkConfig`	`object`	DCSMachineNetworkConfig is the Pool IP entry projected onto a DCSMachine. It mirrors DCSIpHostname but excludes pool-owned fields that have no meaning on a per-Machine record (notably PersistentDisk, which is coordinated by the Pool).
`ready`	`boolean`	

.status.additionalNic

Type

`array`

.status.additionalNic[]

Type

`object`

Property	Type	Description
`dns`	`string`	
`dvSwitchName`	`string`	
`gateway`	`string`	
`ip`	`string`	
`ipv6`	`string`	
`mask`	`string`	
`portGroupName`	`string`	
`portGroupUrn`	`string`	

.status.addresses

Description

MachineAddresses is a slice of MachineAddress items to be used by infrastructure providers.

Type

`array`

.status.addresses[]

Description

MachineAddress contains information for the node's address.

Type

`object`

Required

`address` `type`

Property	Type	Description
`address`	`string`	address is the machine address.
`type`	`string`	type is the machine address type, one of Hostname, ExternalIP, InternalIP, ExternalDNS or InternalDNS.

.status.conditions

Description

Conditions provide observations of the operational state of a Cluster API resource.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across

Property	Type	Description
		resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.failureDomains

Description

FailureDomains is a slice of FailureDomains.

Type

`object`

.status.networkConfig

Description

DCSMachineNetworkConfig is the Pool IP entry projected onto a DCSMachine. It mirrors DCSIpHostname but excludes pool-owned fields that have no meaning on a per-Machine record (notably PersistentDisk, which is coordinated by the Pool).

Type

`object`

Property	Type	Description
`additionNic`	`array`	
`dns`	`string`	
`dvSwitchName`	`string`	
`gateway`	`string`	
`hostname`	`string`	

Property	Type	Description
`ip`	`string`	
`ipv6`	`string`	
`machineName`	`string`	
`mask`	`string`	
`portGroupName`	`string`	
`portGroupUrn`	`string`	

.status.networkConfig.additionNic

Type

`array`

.status.networkConfig.additionNic[]

Type

`object`

Property	Type	Description
`dns`	`string`	
`dvSwitchName`	`string`	
`gateway`	`string`	
`ip`	`string`	
`ipv6`	`string`	
`mask`	`string`	

Property	Type	Description
`portGroupName`	`string`	
`portGroupUrn`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachines`
  - `DELETE`: delete collection of DCSMachine
  - `GET`: list objects of kind DCSMachine
  - `POST`: create a new DCSMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachines/{name}`
  - `DELETE`: delete the specified DCSMachine
  - `GET`: read the specified DCSMachine
  - `PATCH`: partially update the specified DCSMachine
  - `PUT`: replace the specified DCSMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachines/{name}/status`
  - `GET`: read status of the specified DCSMachine
  - `PATCH`: partially update status of the specified DCSMachine
  - `PUT`: replace status of the specified DCSMachine

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachines

HTTP method

`DELETE`

Description

delete collection of DCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind DCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachine` schema
201 - Created	`DCSMachine` schema
202 - Accepted	`DCSMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachines/{name}

HTTP method

`DELETE`

Description

delete the specified DCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified DCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachine` schema
201 - Created	`DCSMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachines/{n

ame}/status

HTTP method

`GET`

Description

read status of the specified DCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the

Parameter	Type	Description
		default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachine` schema
201 - Created	`DCSMachine` schema
401 - Unauthorized	Empty

DCSMachineTemplate [infrastructure.cluster.x-k8s.io/v1beta1]

Description

DCSMachineTemplate is the Schema for the dcsmachinetemplates API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DCSMachineTemplateSpec defines the desired state of DCSMachineTemplate.
`status`	`object`	DCSMachineTemplateStatus defines the observed state of DCSMachineTemplate.

.spec

Description

DCSMachineTemplateSpec defines the desired state of DCSMachineTemplate.

Type

`object`

Required

`template`

Property	Type	Description
`template`	`object`	

.spec.template

Type

`object`

Required

`spec`

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create. This is a copy of customizable fields from metav1.ObjectMeta. ObjectMeta is embedded in `Machine.Spec`, `MachineDeployment.Template` and `MachineSet.Template`, which are not top-level Kubernetes objects. Given that metav1.ObjectMeta has lots of special cases and read-only fields which end up in the generated CRD validation, having it as a subset simplifies the API and some issues that can impact user experience. During the upgrade to controller-tools@v2 ↗ for v1alpha2, we noticed a failure would occur running Cluster API test suite against the new CRDs, specifically `spec.metadata.creationTimestamp in body must be of type string: "null"`. The investigation showed that `controller-tools@v2` behaves differently than its previous version when handling types from metav1 package. In more details, we found that embedded (non-top level) types that embedded `metav1.ObjectMeta` had validation properties, including for `creationTimestamp` (metav1.Time). The `metav1.Time` type specifies a custom json marshaller that, when IsZero() is true, returns `null` which breaks validation because the field isn't marked as nullable.

Property	Type	Description
		In future versions, controller-tools@v2 might allow overriding the type and validation for embedded types. When that happens, this hack should be revisited.
`spec`	`object`	DCSMachineSpec defines the desired state of DCSMachine.

.spec.template.spec

Description

DCSMachineSpec defines the desired state of DCSMachine.

Type

`object`

Required

`ipHostPoolRef` `vmTemplateName`

Property	Type	Description
`ipHostPoolRef`	`object`	LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.
`location`	`object`	
`providerID`	`string`	
`resource`	`object`	
`vmConfig`	`object`	

Property	Type	Description
`vmTemplateName`	`string`	
`vmUrn`	`string`	

.spec.template.spec.ipHostPoolRef

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.template.spec.location

Type

`object`

Required

`type`

Property	Type	Description
`name`	`string`	
`type`	`string`	
`urn`	`string`	

.spec.template.spec.resource

Type

`object`

Required

`type`

Property	Type	Description
`name`	`string`	
`type`	`string`	
`urn`	`string`	

.spec.template.spec.vmConfig

Type

`object`

Required

`dcsMachineCpuSpec` `dcsMachineMemorySpec`

Property	Type	Description
`cdRomDatastoreClusterName`	`string`	
`cdRomDatastoreName`	`string`	

Property	Type	Description
`dcsMachineCpuSpec`	`object`	
`dcsMachineDiskSpec`	`array`	
`dcsMachineMemorySpec`	`object`	
`dvSwitchName`	`string`	
`portGroupName`	`string`	
`portGroupUrn`	`string`	

.spec.template.spec.vmConfig.dcsMachineCpuSpec

Type

`object`

Required

`quantity`

Property	Type	Description
`quantity`	`integer`	

.spec.template.spec.vmConfig.dcsMachineDiskSpec

Type

`array`

.spec.template.spec.vmConfig.dcsMachineDiskSpec[]

Type

`object`

Required

`quantity`

Property	Type	Description
`datastoreClusterName`	`string`	
`datastoreName`	`string`	
`datastoreUrn`	`string`	
`format`	`string`	
`mountOptions`	`array`	
`options`	`array`	
`path`	`string`	
`quantity`	`integer`	
`systemVolume`	`boolean`	
`wipeFilesystem`	`boolean`	

.spec.template.spec.vmConfig.dcsMachineDiskSpec[].mountOptions

Type

`array`

.spec.template.spec.vmConfig.dcsMachineDiskSpec[].mountOptions[]

Type

`string`

.spec.template.spec.vmConfig.dcsMachineDiskSpec[].options

Type

`array`

.spec.template.spec.vmConfig.dcsMachineDiskSpec[].options[]

Type

`string`

.spec.template.spec.vmConfig.dcsMachineMemorySpec

Type

`object`

Required

`quantity`

Property	Type	Description
`quantity`	`integer`	

.status

Description

DCSMachineTemplateStatus defines the observed state of DCSMachineTemplate.

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachinetemplates`
  - `DELETE`: delete collection of DCSMachineTemplate
  - `GET`: list objects of kind DCSMachineTemplate
  - `POST`: create a new DCSMachineTemplate
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachinetemplates/{name}`
  - `DELETE`: delete the specified DCSMachineTemplate
  - `GET`: read the specified DCSMachineTemplate
  - `PATCH`: partially update the specified DCSMachineTemplate
  - `PUT`: replace the specified DCSMachineTemplate
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachinetemplates/{name}/status`
  - `GET`: read status of the specified DCSMachineTemplate
  - `PATCH`: partially update status of the specified DCSMachineTemplate
  - `PUT`: replace status of the specified DCSMachineTemplate

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachinetemplates

HTTP method

`DELETE`

Description

delete collection of DCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind DCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplateList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplate` schema
201 - Created	`DCSMachineTemplate` schema
202 - Accepted	`DCSMachineTemplate` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachinetemplates/{name}

HTTP method

`DELETE`

Description

delete the specified DCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified DCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplate` schema
201 - Created	`DCSMachineTemplate` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/dcsmachinetemplates/{name}/status

HTTP method

`GET`

Description

read status of the specified DCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DCSMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DCSMachineTemplate` schema

HTTP code	Response body
201 - Created	`DCSMachineTemplate` schema
401 - Unauthorized	Empty

Huawei Cloud Stack Provider APIs

HCSCluster [infrastructure.clus HCSMachine [infrastructure.clu HCSMachin

HCSMachineConfigPool [infrastructure.cluster.x-k8s.io/v1beta1]

HCSCluster [infrastructure.cluster.x-k8s.io/v1beta1]

Description

HCSCluster is the Schema for the hcsclusters API

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	spec defines the desired state of HCSCluster
`status`	`object`	status defines the observed state of HCSCluster

.spec

Description

spec defines the desired state of HCSCluster

Type

`object`

Required

`identityRef` `network` `networkType`

Property	Type	Description
`controlPlaneEndpoint`	`object`	ControlPlaneEndpoint specifies the API endpoint where the control plane can be accessed.
`controlPlaneHA`	`object`	ControlPlaneHA configures the provider-managed control-plane server group.
`controlPlaneLoadBalancer`	`object`	
`encryptionProviderConfigRef`	`object`	EncryptionProviderConfigRef references a Secret containing a pre-existing encryption-provider.conf for the cluster. When set, the controller uses this config instead of generating a new random encryption key. Required for disaster recovery: the standby cluster must use the same encryption key as the primary cluster so that restored etcd data can be decrypted by the API server. The referenced Secret must contain a key "encryption-provider.conf".
`identityRef`	`object`	IdentityRef is a reference to the identity used to provision the cluster.
`network`	`object`	Network is the network configuration of the cluster.

Property	Type	Description
`networkType`	`string`	NetworkType is the network type of the cluster(kube-ovn only supported now).

.spec.controlPlaneEndpoint

Description

ControlPlaneEndpoint specifies the API endpoint where the control plane can be accessed.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	host is the hostname on which the API server is serving.
`port`	`integer`	port is the port on which the API server is serving.

.spec.controlPlaneHA

Description

ControlPlaneHA configures the provider-managed control-plane server group.

Type

`object`

Required

`enabled` `policy`

Property	Type	Description
`enabled`	`boolean`	Enabled controls whether provider creates and uses a control-plane server group.
`policy`	`string`	Policy is the HCS server group policy used for control-plane ECS placement.

.spec.controlPlaneLoadBalancer

Type

`object`

Property	Type	Description
`elbVirsubnetL4Ips`	`array`	
`elbVirsubnetL7Ips`	`array`	
`id`	`string`	
`liseners`	`array`	
`name`	`string`	Name represents the name of the load balancer as a pointer to a string, used for identification and optionally included in JSON output.
`pools`	`array`	

Property	Type	Description
`publicIpOption`	`object`	PublicIpOption specifies configuration options for the public IP of the control plane load balancer, optional in JSON output.
`selector`	`string`	Selector specifies a label query used to identify resources, represented as a pointer to a string, and is optional in JSON output.
`vipAddress`	`string`	
`vipDomainName`	`string`	
`vipSubnetId`	`string`	
`vipSubnetName`	`string`	
`vpcId`	`string`	VpcId is a pointer to a string that specifies the unique ID of the VPC, optionally included in JSON output.
`vpcName`	`string`	VpcName is a pointer to a string representing the name of the VPC, optionally included in JSON output.

.spec.controlPlaneLoadBalancer.elbVirsubnetL4Ips

Type

`array`

.spec.controlPlaneLoadBalancer.elbVirsubnetL4Ips[]

Type

`object`

Property	Type	Description
`ips`	`array`	IPs is a list of IP addresses associated with the virtual subnet, serialized as a JSON array of strings and optional.
`subnetName`	`string`	SubnetName is a pointer to a string that specifies the name of the subnet, optionally included in JSON output.

.spec.controlPlaneLoadBalancer.elbVirsubnetL4Ips[].ips

Description

IPs is a list of IP addresses associated with the virtual subnet, serialized as a JSON array of strings and optional.

Type

`array`

.spec.controlPlaneLoadBalancer.elbVirsubnetL4Ips[].ips[]

Type

`string`

.spec.controlPlaneLoadBalancer.elbVirsubnetL7Ips

Type

`array`

.spec.controlPlaneLoadBalancer.elbVirsubnetL7Ips[]

Type

`object`

Property	Type	Description
`ips`	`array`	IPs is a list of IP addresses associated with the virtual subnet, serialized as a JSON array of strings and optional.
`subnetName`	`string`	SubnetName is a pointer to a string that specifies the name of the subnet, optionally included in JSON output.

.spec.controlPlaneLoadBalancer.elbVirsubnetL7Ips[].ips

Description

IPs is a list of IP addresses associated with the virtual subnet, serialized as a JSON array of strings and optional.

Type

`array`

.spec.controlPlaneLoadBalancer.elbVirsubnetL7Ips[].ips[]

Type

`string`

.spec.controlPlaneLoadBalancer.liseners

Type

`array`

.spec.controlPlaneLoadBalancer.liseners[]

Type

`object`

Property	Type	Description
`id`	`string`	
`protocolPort`	`integer`	

.spec.controlPlaneLoadBalancer.pools

Type

`array`

.spec.controlPlaneLoadBalancer.pools[]

Type

`object`

Property	Type	Description
`healthMonitorID`	`string`	
`id`	`string`	
`protocolPort`	`integer`	

.spec.controlPlaneLoadBalancer.publicIpOption

Description

PublicIpOption specifies configuration options for the public IP of the control plane load balancer, optional in JSON output.

Type

`object`

Required

`ipVersion` `networkType`

Property	Type	Description
`ipVersion`	`string`	IpVersion represents the IP version of the public IP address as a pointer to a string, used for identification and optionally included in JSON output.
`networkType`	`string`	NetworkType specifies the type of network configuration for the control plane load balancer, serialized as a JSON string.

.spec.encryptionProviderConfigRef

Description

EncryptionProviderConfigRef references a Secret containing a pre-existing encryption-provider.conf for the cluster. When set, the controller uses this config instead of generating a new random encryption key. Required for disaster recovery: the standby cluster must use the same encryption key as the primary cluster so that restored etcd data can be decrypted by the API server. The referenced Secret must contain a key "encryption-provider.conf".

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.identityRef

Description

IdentityRef is a reference to the identity used to provision the cluster.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name is the name of the identity secret to use.

.spec.network

Description

Network is the network configuration of the cluster.

Type

`object`

Property	Type	Description
`securityGroup`	`object`	SecurityGroups represents the security group configuration of the cluster, optional in JSON output.

Property	Type	Description
`subnets`	`array`	Subnets represents the subnet configuration of the cluster, optional in JSON output.
`vpc`	`object`	Vpc represents the VPC configuration of the cluster, optional in JSON output.

.spec.network.securityGroup

Description

SecurityGroups represents the security group configuration of the cluster, optional in JSON output.

Type

`object`

Property	Type	Description
`id`	`string`	ID represents the unique ID of the security group as a pointer to a string, used for identification and optionally included in JSON output.
`name`	`string`	Name represents the name of the security group as a pointer to a string, used for identification and optionally included in JSON output.

.spec.network.subnets

Description

Subnets represents the subnet configuration of the cluster, optional in JSON output.

Type

`array`

.spec.network.subnets[]

Type

`object`

Property	Type	Description
`cidr`	`string`	Cidr represents the CIDR block of the subnet as a pointer to a string, used for identification and optionally included in JSON output.
`dhcp`	`boolean`	DHCP represents the DHCP flag of the subnet as a pointer to a bool, used for identification and optionally included in JSON output.
`gatewayIp`	`string`	
`id`	`string`	
`managed`	`boolean`	
`name`	`string`	Name represents the name of the subnet as a pointer to a string, used for identification and optionally included in JSON output.
`neutronSubnetId`	`string`	

Property	Type	Description
`primaryDNS`	`string`	PrimaryDNS represents the primary DNS of the subnet as a pointer to a string, used for identification and optionally included in JSON output.
`secondaryDNS`	`string`	SecondaryDNS represents the secondary DNS of the subnet as a pointer to a string, used for identification and optionally included in JSON output.

.spec.network.vpc

Description

Vpc represents the VPC configuration of the cluster, optional in JSON output.

Type

`object`

Property	Type	Description
`cidr`	`string`	Cidr specifies the CIDR block of the VPC as a pointer to a string, used for identification and optionally included in JSON output.
`externalGatewayInfo`	`object`	ExternalGatewayInfo specifies the external gateway information of the VPC as a pointer to a ExternalGatewayConfig struct, used for identification and optionally included in JSON output.

Property	Type	Description
`id`	`string`	ID specifies the unique ID of the VPC as a pointer to a string, used for identification and optionally included in JSON output.
`managed`	`boolean`	Managed represents the managed flag of the VPC as a pointer to a bool, used for identification and optionally included in JSON output.
`name`	`string`	Name specifies the name of the VPC as a pointer to a string, used for identification and optionally included in JSON output.

.spec.network.vpc.externalGatewayInfo

Description

ExternalGatewayInfo specifies the external gateway information of the VPC as a pointer to a ExternalGatewayConfig struct, used for identification and optionally included in JSON output.

Type

`object`

Property	Type	Description
`networkId`	`string`	NetworkId represents the id of the network as a pointer to a string, optionally included in JSON serialization.

Property	Type	Description
`networkName`	`string`	NetworkName represents the name of the network as a pointer to a string, optionally included in JSON serialization.

.status

Description

status defines the observed state of HCSCluster

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions defines current service state of the cluster.
`controlPlaneHA`	`object`	ControlPlaneHA records the provider-managed control-plane server group observed state.
`ready`	`boolean`	Ready indicates whether the cluster is in a ready state. It is an optional field that can be nil.

.status.conditions

Description

Conditions defines current service state of the cluster.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can

Property	Type	Description
		immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.controlPlaneHA

Description

ControlPlaneHA records the provider-managed control-plane server group observed state.

Type

`object`

Property	Type	Description
`id`	`string`	
`members`	`array`	
`name`	`string`	
`policy`	`string`	

.status.controlPlaneHA.members

Type

`array`

.status.controlPlaneHA.members[]

Type

`object`

Property	Type	Description
`machineName`	`string`	
`serverID`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsclusters`
  - `DELETE`: delete collection of HCSCluster
  - `GET`: list objects of kind HCSCluster
  - `POST`: create a new HCSCluster
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsclusters/{name}`
  - `DELETE`: delete the specified HCSCluster
  - `GET`: read the specified HCSCluster
  - `PATCH`: partially update the specified HCSCluster
  - `PUT`: replace the specified HCSCluster

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsclusters/{name}/status`
  - `GET`: read status of the specified HCSCluster
  - `PATCH`: partially update status of the specified HCSCluster
  - `PUT`: replace status of the specified HCSCluster

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsclusters

HTTP method

`DELETE`

Description

delete collection of HCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind HCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`HCSClusterList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new HCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSCluster` schema
201 - Created	`HCSCluster` schema
202 - Accepted	`HCSCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsclusters/{name}

HTTP method

`DELETE`

Description

delete the specified HCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified HCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`HCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified HCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified HCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSCluster` schema
201 - Created	`HCSCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsclusters/{name}/status

HTTP method

`GET`

Description

read status of the specified HCSCluster

HTTP responses

HTTP code	Response body
200 - OK	`HCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified HCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified HCSCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The

Parameter	Type	Description
		request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSCluster` schema
201 - Created	`HCSCluster` schema
401 - Unauthorized	Empty

HCSMachine [infrastructure.cluster.x-k8s.io/v1beta1]

Description

HCSMachine is the Schema for the hcsmachines API.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	spec defines the desired state of HCSMachine
`status`	`object`	status defines the observed state of HCSMachine

.spec

Description

spec defines the desired state of HCSMachine

Type

`object`

Property	Type	Description
`adminPassword`	`string`	AdminPassword is the admin password for the instance.

Property	Type	Description
`availabilityZone`	`string`	AvailabilityZone is the availability zone for the instance.
`configPoolRef`	`object`	ConfigPoolRef is a reference to the HCSMachineConfigPool resource that provides shared configuration for this machine.
`dataVolumes`	`array`	DataVolumes is a list of additional data volumes to attach to the instance.
`flavorName`	`string`	FlavorName is the name of the HCS instance flavor/type.
`imageName`	`string`	ImageName is the name of the image to use for the instance.
`jobId`	`string`	
`keyPairName`	`string`	KeyPairName is the name of the SSH key pair for instance access.
`providerID`	`string`	ProviderID is the unique identifier as specified by the cloud provider.

Property	Type	Description
`rootVolume`	`object`	RootVolume is the root disk volume configuration.
`serverId`	`string`	ServerId is the HCS ECS server instance ID.
`serverName`	`string`	

.spec.configPoolRef

Description

ConfigPoolRef is a reference to the HCSMachineConfigPool resource that provides shared configuration for this machine.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.

Property	Type	Description
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-

Property	Type	Description
		architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.dataVolumes

Description

DataVolumes is a list of additional data volumes to attach to the instance.

Type

`array`

.spec.dataVolumes[]

Description

Volume defines the disk volume configuration for HCS instances.

Type

`object`

Required

`size` `type`

Property	Type	Description
`format`	`string`	Format is the format of the volume.

Property	Type	Description
`mountPath`	`string`	Path is the path to the volume.
`size`	`integer`	Size is the volume size in GB.
`type`	`string`	Type is the volume type (e.g., SSD, SAS, SATA).

.spec.rootVolume

Description

RootVolume is the root disk volume configuration.

Type

`object`

Required

`size` `type`

Property	Type	Description
`format`	`string`	Format is the format of the volume.
`mountPath`	`string`	Path is the path to the volume.
`size`	`integer`	Size is the volume size in GB.

Property	Type	Description
`type`	`string`	Type is the volume type (e.g., SSD, SAS, SATA).

.status

Description

status defines the observed state of HCSMachine

Type

`object`

Property	Type	Description
`addresses`	`array`	Addresses is the list of addresses assigned to the machine.
`conditions`	`array`	Conditions defines current service state of the machine.
`failureMessage`	`string`	FailureMessage will be set in case of a terminal error.
`failureReason`	`string`	FailureReason will be set in case of a terminal error.
`instanceState`	`string`	InstanceState is the current state of the HCS instance (e.g., ACTIVE, BUILD, ERROR).

Property	Type	Description
`ready`	`boolean`	Ready indicates whether the machine infrastructure is ready.

.status.addresses

Description

Addresses is the list of addresses assigned to the machine.

Type

`array`

.status.addresses[]

Description

NodeAddress contains information for the node's address.

Type

`object`

Required

`address` `type`

Property	Type	Description
`address`	`string`	The node address.
`type`	`string`	Node address type, one of Hostname, ExternalIP or InternalIP.

.status.conditions

Description

Conditions defines current service state of the machine.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may

Property	Type	Description
		choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachines`
  - `DELETE`: delete collection of HCSMachine
  - `GET`: list objects of kind HCSMachine

  - `POST`: create a new HCSMachine

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachines/{name}`
  - `DELETE`: delete the specified HCSMachine
  - `GET`: read the specified HCSMachine
  - `PATCH`: partially update the specified HCSMachine
  - `PUT`: replace the specified HCSMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachines/{name}/status`
  - `GET`: read status of the specified HCSMachine
  - `PATCH`: partially update status of the specified HCSMachine
  - `PUT`: replace status of the specified HCSMachine

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachines

HTTP method

`DELETE`

Description

delete collection of HCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind HCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new HCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachine` schema
201 - Created	`HCSMachine` schema
202 - Accepted	`HCSMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachines/{name}

HTTP method

`DELETE`

Description

delete the specified HCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified HCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified HCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified HCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachine` schema

HTTP code	Response body
201 - Created	`HCSMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachines/{name}/status

HTTP method

`GET`

Description

read status of the specified HCSMachine

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified HCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified HCSMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachine` schema
201 - Created	`HCSMachine` schema

HTTP code	Response body
401 - Unauthorized	Empty

HCSMachineTemplate [infrastructure.cluster.x-k8s.io/v1beta1]

Description

HCSMachineTemplate is the Schema for the hcsmachinetemplates API. It is used by MachineDeployment and MachineSet to create HCSMachine resources.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	spec defines the desired state of HCSMachineTemplate
`status`	`object`	status defines the observed state of HCSMachineTemplate

.spec

Description

spec defines the desired state of HCSMachineTemplate

Type

`object`

Required

`template`

Property	Type	Description
`template`	`object`	Template is the HCSMachine resource template that will be used for creating machines.

.spec.template

Description

Template is the HCSMachine resource template that will be used for creating machines.

Type

`object`

Required

`spec`

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is the standard object metadata.
`spec`	`object`	Spec is the specification of the desired behavior of the machine.

.spec.template.spec

Description

Spec is the specification of the desired behavior of the machine.

Type

`object`

Property	Type	Description
`adminPassword`	`string`	AdminPassword is the admin password for the instance.
`availabilityZone`	`string`	AvailabilityZone is the availability zone for the instance.
`configPoolRef`	`object`	ConfigPoolRef is a reference to the HCSMachineConfigPool resource that provides shared configuration for this machine.
`dataVolumes`	`array`	DataVolumes is a list of additional data volumes to attach to the instance.
`flavorName`	`string`	FlavorName is the name of the HCS instance flavor/type.
`imageName`	`string`	ImageName is the name of the image to use for the instance.
`jobId`	`string`	
`keyPairName`	`string`	KeyPairName is the name of the SSH key pair for instance access.

Property	Type	Description
`providerID`	`string`	ProviderID is the unique identifier as specified by the cloud provider.
`rootVolume`	`object`	RootVolume is the root disk volume configuration.
`serverId`	`string`	ServerId is the HCS ECS server instance ID.
`serverName`	`string`	

.spec.template.spec.configPoolRef

Description

ConfigPoolRef is a reference to the HCSMachineConfigPool resource that provides shared configuration for this machine.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}"

Property	Type	Description
		(where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.template.spec.dataVolumes

Description

DataVolumes is a list of additional data volumes to attach to the instance.

Type

`array`

.spec.template.spec.dataVolumes[]

Description

Volume defines the disk volume configuration for HCS instances.

Type

`object`

Required

`size` `type`

Property	Type	Description
`format`	`string`	Format is the format of the volume.
`mountPath`	`string`	Path is the path to the volume.
`size`	`integer`	Size is the volume size in GB.
`type`	`string`	Type is the volume type (e.g., SSD, SAS, SATA).

.spec.template.spec.rootVolume

Description

RootVolume is the root disk volume configuration.

Type

`object`

Required

`size` `type`

Property	Type	Description
`format`	`string`	Format is the format of the volume.
`mountPath`	`string`	Path is the path to the volume.
`size`	`integer`	Size is the volume size in GB.
`type`	`string`	Type is the volume type (e.g., SSD, SAS, SATA).

.status

Description

status defines the observed state of HCSMachineTemplate

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachinetemplates`
  - `DELETE`: delete collection of HCSMachineTemplate
  - `GET`: list objects of kind HCSMachineTemplate
  - `POST`: create a new HCSMachineTemplate
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachinetemplates/{name}`
  - `DELETE`: delete the specified HCSMachineTemplate
  - `GET`: read the specified HCSMachineTemplate
  - `PATCH`: partially update the specified HCSMachineTemplate
  - `PUT`: replace the specified HCSMachineTemplate
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachinetemplates/{name}/status`
  - `GET`: read status of the specified HCSMachineTemplate
  - `PATCH`: partially update status of the specified HCSMachineTemplate
  - `PUT`: replace status of the specified HCSMachineTemplate

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachinetemplates

HTTP method

`DELETE`

Description

delete collection of HCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind HCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplateList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new HCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplate` schema
201 - Created	`HCSMachineTemplate` schema
202 - Accepted	`HCSMachineTemplate` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachinetemplates/{name}

HTTP method

`DELETE`

Description

delete the specified HCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified HCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified HCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified HCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplate` schema
201 - Created	`HCSMachineTemplate` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachinetemplates/{name}/status

HTTP method

`GET`

Description

read status of the specified HCSMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified HCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified HCSMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineTemplate` schema

HTTP code	Response body
201 - Created	`HCSMachineTemplate` schema
401 - Unauthorized	Empty

HCSMachineConfigPool [infrastructure.cluster.x-k8s.io/v1beta1]

Description

HCSMachineConfigPool is the Schema for the hcsmachineconfigpools API. It provides a pool of pre-defined machine configurations (hostname, network settings) that can be claimed by HCSMachine resources.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	spec defines the desired state of HCSMachineConfigPool
`status`	`object`	status defines the observed state of HCSMachineConfigPool

.spec

Description

spec defines the desired state of HCSMachineConfigPool

Type

`object`

Property	Type	Description
`configs`	`array`	Configs lists machine configurations claimable by HCSMachines.

.spec.configs

Description

Configs lists machine configurations claimable by HCSMachines.

Type

`array`

.spec.configs[]

Description

MachineConfig defines the configuration for a single machine in the pool.

Type

`object`

Required

`hostname` `networks`

Property	Type	Description
`hostname`	`string`	Hostname is the hostname to assign to the machine.
`networks`	`array`	Networks is the list of network configurations for the machine.
`persistentDisks`	`array`	PersistentDisks declares EVS disks that survive HCSMachine delete-recreate. Each disk is bound to the (Hostname, Slot) identity and re-attached to the new ECS on rolling upgrade. Slots must be unique and contiguous from 0.

.spec.configs[].networks

Description

Networks is the list of network configurations for the machine.

Type

`array`

.spec.configs[].networks[]

Description

MachineConfigNetworkConfig defines the network configuration for a machine.

Type

`object`

Required

`ipAddress`

Property	Type	Description
`ipAddress`	`string`	IpAddress is the static IP address to assign to the machine.
`subenetName`	`string`	SubenetName is the deprecated legacy key kept for backward compatibility. Use subnetName instead.
`subnetId`	`string`	SubnetId is the ID of the subnet to use.
`subnetName`	`string`	SubnetName is the name of the subnet to use.

.spec.configs[].persistentDisks

Description

PersistentDisks declares EVS disks that survive HCSMachine delete-recreate. Each disk is bound to the (Hostname, Slot) identity and re-attached to the new ECS on rolling upgrade. Slots must be unique and contiguous from 0.

Type

`array`

.spec.configs[].persistentDisks[]

Description

PersistentDisk defines a persistent EVS disk attached to a fixed machine slot in the pool. Persistent disks survive HCSMachine delete-recreate cycles (e.g. rolling upgrades), while temporary disks declared in HCSMachineTemplate.spec.dataVolumes[] are removed together with their ECS.

Type

`object`

Required

`size` `slot` `type`

Property	Type	Description
`format`	`string`	Format is the filesystem format; empty normalises to xfs. Already-formatted disks are not re-formatted. Immutable once accepted (validating webhook).
`mountOptions`	`array`	MountOptions overrides the default ("defaults,noatime"); takes effect on next rebuild.

Property	Type	Description
`mountPath`	`string`	MountPath is the guest filesystem mount path; empty means block device only (no fs_setup/mount). Immutable once accepted (validating webhook).
`size`	`integer`	Size is the EVS volume size in GB. CRD lower bound is 1 to accept legacy claimed volumes; the controller enforces the EVS [10, 32768] range only for new volumes. Immutable once accepted (validating webhook).
`slot`	`integer`	Slot is the disk position within the MachineConfig. Unique and contiguous from 0; combined with hostname it keys the underlying EVS volume.
`type`	`string`	Type is the EVS volume type name (e.g. SSD, SATA, hcs_global_ssd). Availability is checked at reconcile via CinderListVolumeTypes; no static whitelist. Immutable once accepted (validating webhook).

.spec.configs[].persistentDisks[].mountOptions

Description

MountOptions overrides the default ("defaults,noatime"); takes effect on next rebuild.

Type

`array`

.spec.configs[].persistentDisks[].mountOptions[]

Type

`string`

.status

Description

status defines the observed state of HCSMachineConfigPool

Type

`object`

Property	Type	Description
`availableCount`	`integer`	AvailableCount is the number of available configurations in the pool.
`persistentDiskStatus`	`array`	PersistentDiskStatus records the runtime state of every persistent disk owned by this pool, keyed by (Hostname, Slot).
`totalCount`	`integer`	TotalCount is the total number of configurations in the pool.
`usedConfigs`	`array`	UsedConfigs is the list of configurations that have been claimed by HCSMachines.

.status.persistentDiskStatus

Description

PersistentDiskStatus records the runtime state of every persistent disk owned by this pool, keyed by (Hostname, Slot).

Type

`array`

.status.persistentDiskStatus[]

Description

PersistentDiskStatus records the runtime state of a single persistent disk, keyed by (Hostname, Slot). Spec fields (size/type/format/mountPath) are not duplicated here.

Type

`object`

Required

`hostname` `slot`

Property	Type	Description
`attachedServerID`	`string`	AttachedServerID is the ECS server id where the disk is currently attached. Cleared during reconcileDelete after the EVS has returned to available.
`availabilityZone`	`string`	AvailabilityZone of the EVS volume; the controller locks the ECS AZ to this value.
`device`	`string`	Device is the guest-visible device path (e.g. /dev/vdc) computed from the current (len(spec.dataVolumes), slot). Cleared during reconcileDelete.

Property	Type	Description
`hostname`	`string`	Hostname identifies which MachineConfig the disk belongs to.
`lastError`	`string`	LastError records the last error message, set when Phase=Error.
`lastTransitionTime`	`string`	LastTransitionTime marks the most recent phase change.
`name`	`string`	Name is the stable EVS volume name produced by PersistentVolumeName().
`ownerMachineName`	`string`	OwnerMachineName is the name of the HCSMachine currently bound to this disk.
`ownerMachineUID`	`string`	OwnerMachineUID is the UID of the HCSMachine currently bound to this disk.
`phase`	`string`	Phase is the current lifecycle phase.
`slot`	`integer`	Slot is the disk slot within the MachineConfig (matches spec).

Property	Type	Description
`volumeID`	`string`	VolumeID is the HCS EVS volume id.

.status.usedConfigs

Description

UsedConfigs is the list of configurations that have been claimed by HCSMachines.

Type

`array`

.status.usedConfigs[]

Description

UsedConfig represents a machine configuration that has been claimed by a HCSMachine.

Type

`object`

Required

`hostname` `machineRef`

Property	Type	Description
`hostname`	`string`	HostName is the hostname of the claimed configuration.
`machineRef`	`string`	MachineRef is the name of the HCSMachine that claimed this configuration.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachineconfigpools`
  - `DELETE`: delete collection of HCSMachineConfigPool
  - `GET`: list objects of kind HCSMachineConfigPool
  - `POST`: create a new HCSMachineConfigPool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachineconfigpools/{name}`
  - `DELETE`: delete the specified HCSMachineConfigPool
  - `GET`: read the specified HCSMachineConfigPool
  - `PATCH`: partially update the specified HCSMachineConfigPool
  - `PUT`: replace the specified HCSMachineConfigPool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachineconfigpools/{name}/status`
  - `GET`: read status of the specified HCSMachineConfigPool
  - `PATCH`: partially update status of the specified HCSMachineConfigPool
  - `PUT`: replace status of the specified HCSMachineConfigPool

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachineconfigpools

HTTP method

`DELETE`

Description

delete collection of HCSMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind HCSMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPoolList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new HCSMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachineConfigPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPool` schema
201 - Created	`HCSMachineConfigPool` schema
202 - Accepted	`HCSMachineConfigPool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachineconfigpools/{name}

HTTP method

`DELETE`

Description

delete the specified HCSMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified HCSMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified HCSMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified HCSMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachineConfigPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPool` schema
201 - Created	`HCSMachineConfigPool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/hcsmachineconfigpools/{name}/status

HTTP method

`GET`

Description

read status of the specified HCSMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified HCSMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified HCSMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HCSMachineConfigPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HCSMachineConfigPool` schema

HTTP code	Response body
201 - Created	`HCSMachineConfigPool` schema
401 - Unauthorized	Empty

VMware vSphere Provider APIs

VSphereCluster [infrastructure. VSphereClusterIdentity [infrast VSphereMa

VSphereClusterTemplate VSphereVM [infrastructure.clus VSphereMa

VSphereMachineConfigPool [infrastructure.cluster.x-k8s.io/v1beta1]

VSphereFailureDomain [infrast VSphereDe

VSphereCluster [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereCluster is the Schema for the vsphereclusters API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereClusterSpec defines the desired state of VSphereCluster.
`status`	`object`	VSphereClusterStatus defines the observed state of VSphereClusterSpec.

.spec

Description

VSphereClusterSpec defines the desired state of VSphereCluster.

Type

`object`

Property	Type	Description
`clusterModules`	`array`	ClusterModules hosts information regarding the anti-affinity vSphere constructs for each of the objects responsible for creation of VM objects belonging to the cluster.

Property	Type	Description
`controlPlaneEndpoint`	`object`	ControlPlaneEndpoint represents the endpoint used to communicate with the control plane.
`controlPlaneLoadBalancer`	`object`	ControlPlaneLoadBalancer describes the control plane endpoint's load balancer. Leaving it nil is allowed and means the same as type=external: the endpoint is provided by the user and the provider manages no VIP. The field is set when the cluster is created or never: it cannot be added to an existing cluster, and once set every field is immutable and the field cannot be cleared. The endpoint, the apiserver serving certificate and the guest runtime are all derived from it and cannot be changed in place.
`disableClusterModule`	`boolean`	DisableClusterModule is used to explicitly turn off the ClusterModule feature. This should work along side NodeAntiAffinity feature flag. If the NodeAntiAffinity feature flag is turned off, this will be disregarded.
`failureDomainSelector`	`object`	FailureDomainSelector is the label selector to use for failure domain selection for the control plane nodes of the cluster. If not set (`nil`), selecting failure domains will be

Property	Type	Description
		disabled. An empty value (`{}`) selects all existing failure domains. A valid selector will select all failure domains which match the selector.
`identityRef`	`object`	IdentityRef is a reference to either a Secret or VSphereClusterIdentity that contains the identity to use when reconciling the cluster.
`server`	`string`	Server is the address of the vSphere endpoint.
`thumbprint`	`string`	Thumbprint is the colon-separated SHA-1 checksum of the given vCenter server's host certificate

.spec.clusterModules

Description

ClusterModules hosts information regarding the anti-affinity vSphere constructs for each of the objects responsible for creation of VM objects belonging to the cluster.

Type

`array`

.spec.clusterModules[]

Description

ClusterModule holds the anti affinity construct `ClusterModule` identifier in use by the VMs owned by the object referred by the TargetObjectName field.

Type

`object`

Required

`controlPlane` `moduleUUID` `targetObjectName`

Property	Type	Description
`controlPlane`	`boolean`	ControlPlane indicates whether the referred object is responsible for control plane nodes. Currently, only the KubeadmControlPlane objects have this flag set to true. Only a single object in the slice can have this value set to true.
`moduleUUID`	`string`	ModuleUUID is the unique identifier of the `ClusterModule` used by the object.
`targetObjectName`	`string`	TargetObjectName points to the object that uses the Cluster Module information to enforce anti-affinity amongst its descendant VM objects.

.spec.controlPlaneEndpoint

Description

ControlPlaneEndpoint represents the endpoint used to communicate with the control plane.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	The hostname on which the API server is serving.
`port`	`integer`	The port on which the API server is serving.

.spec.controlPlaneLoadBalancer

Description

ControlPlaneLoadBalancer describes the control plane endpoint's load balancer. Leaving it nil is allowed and means the same as type=external: the endpoint is provided by the user and the provider manages no VIP. The field is set when the cluster is created or never: it cannot be added to an existing cluster, and once set every field is immutable and the field cannot be cleared. The endpoint, the apiserver serving certificate and the guest runtime are all derived from it and cannot be changed in place.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	Host is the control plane endpoint address. For type=internal it is the IPv4 VIP.
`interface`	`string`	Interface pins the guest NIC that holds the VIP. Empty means auto-detect by matching the node's primary IP.

Property	Type	Description
`port`	`integer`	Port is the control plane endpoint port.
`type`	`string`	Type selects who owns the control plane VIP.
`vrid`	`integer`	VRID is the keepalived VRRP router ID. Only used when type=internal. The 1-255 range is validated by the webhook, not by the CRD schema, so that type=external objects are not rejected for leaving it at zero.

.spec.failureDomainSelector

Description

FailureDomainSelector is the label selector to use for failure domain selection for the control plane nodes of the cluster. If not set (`nil`), selecting failure domains will be disabled. An empty value (`{}`) selects all existing failure domains. A valid selector will select all failure domains which match the selector.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is

Property	Type	Description
		"key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.failureDomainSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.failureDomainSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.failureDomainSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.failureDomainSelector.matchExpressions[].values[]

Type

`string`

.spec.failureDomainSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.identityRef

Description

IdentityRef is a reference to either a Secret or VSphereClusterIdentity that contains the identity to use when reconciling the cluster.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	Kind of the identity. Can either be VSphereClusterIdentity or Secret
`name`	`string`	Name of the identity.

.status

Description

VSphereClusterStatus defines the observed state of VSphereClusterSpec.

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions defines current service state of the VSphereCluster.

Property	Type	Description
`failureDomains`	`object`	FailureDomains is a list of failure domain objects synced from the infrastructure provider.
`ready`	`boolean`	
`v1beta2`	`object`	v1beta2 groups all the fields that will be added or modified in VSphereCluster's status with the V1Beta2 version.
`vCenterVersion`	`string`	VCenterVersion defines the version of the vCenter server defined in the spec.

.status.conditions

Description

Conditions defines current service state of the VSphereCluster.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across

Property	Type	Description
		resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.failureDomains

Description

FailureDomains is a list of failure domain objects synced from the infrastructure provider.

Type

`object`

.status.v1beta2

Description

v1beta2 groups all the fields that will be added or modified in VSphereCluster's status with the V1Beta2 version.

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions represents the observations of a VSphereCluster's current state. Known condition types are Ready, FailureDomainsReady, VCenterAvailable, ClusterModulesReady, KubeOvnAppReleaseReady, SelfBuiltLoadBalancerReady and Paused.

.status.v1beta2.conditions

Description

conditions represents the observations of a VSphereCluster's current state. Known condition types are Ready, FailureDomainsReady, VCenterAvailable, ClusterModulesReady, KubeOvnAppReleaseReady, SelfBuiltLoadBalancerReady and Paused.

Type

`array`

.status.v1beta2.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set

Property	Type	Description
		based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclusters`
  - `DELETE`: delete collection of VSphereCluster
  - `GET`: list objects of kind VSphereCluster

  - `POST`: create a new VSphereCluster

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclusters/{name}`
  - `DELETE`: delete the specified VSphereCluster
  - `GET`: read the specified VSphereCluster
  - `PATCH`: partially update the specified VSphereCluster
  - `PUT`: replace the specified VSphereCluster
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclusters/{name}/status`
  - `GET`: read status of the specified VSphereCluster
  - `PATCH`: partially update status of the specified VSphereCluster
  - `PUT`: replace status of the specified VSphereCluster

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclusters

HTTP method

`DELETE`

Description

delete collection of VSphereCluster

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereCluster

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereCluster` schema
201 - Created	`VSphereCluster` schema
202 - Accepted	`VSphereCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclusters/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereCluster

HTTP responses

HTTP code	Response body
200 - OK	`VSphereCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereCluster` schema

HTTP code	Response body
201 - Created	`VSphereCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclusters/{name}/status

HTTP method

`GET`

Description

read status of the specified VSphereCluster

HTTP responses

HTTP code	Response body
200 - OK	`VSphereCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VSphereCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VSphereCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereCluster` schema
201 - Created	`VSphereCluster` schema

HTTP code	Response body
401 - Unauthorized	Empty

VSphereClusterIdentity [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereClusterIdentity defines the account to be used for reconciling clusters.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereClusterIdentitySpec contains a secret reference and a group of allowed namespaces.
`status`	`object`	VSphereClusterIdentityStatus contains the status of the VSphereClusterIdentity.

.spec

Description

VSphereClusterIdentitySpec contains a secret reference and a group of allowed namespaces.

Type

`object`

Property	Type	Description
`allowedNamespaces`	`object`	AllowedNamespaces is used to identify which namespaces are allowed to use this account. Namespaces can be selected with a label selector. If this object is nil, no namespaces will be allowed

Property	Type	Description
`secretName`	`string`	SecretName references a Secret inside the controller namespace with the credentials to use

.spec.allowedNamespaces

Description

AllowedNamespaces is used to identify which namespaces are allowed to use this account. Namespaces can be selected with a label selector. If this object is nil, no namespaces will be allowed

Type

`object`

Property	Type	Description
`selector`	`object`	Selector is a standard Kubernetes LabelSelector. A label query over a set of resources.

.spec.allowedNamespaces.selector

Description

Selector is a standard Kubernetes LabelSelector. A label query over a set of resources.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.allowedNamespaces.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.allowedNamespaces.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.allowedNamespaces.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.allowedNamespaces.selector.matchExpressions[].values[]

Type

`string`

.spec.allowedNamespaces.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status

Description

VSphereClusterIdentityStatus contains the status of the VSphereClusterIdentity.

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions defines current service state of the VSphereClusterIdentity.
`ready`	`boolean`	
`v1beta2`	`object`	v1beta2 groups all the fields that will be added or modified in VSphereClusterIdentity's status with the V1Beta2 version.

.status.conditions

Description

Conditions defines current service state of the VSphereClusterIdentity.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.

Property	Type	Description
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.v1beta2

Description

v1beta2 groups all the fields that will be added or modified in VSphereClusterIdentity's status with the V1Beta2 version.

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions represents the observations of a VSphereClusterIdentity's current state. Known condition types are Available and Paused.

.status.v1beta2.conditions

Description

conditions represents the observations of a VSphereClusterIdentity's current state. Known condition types are Available and Paused.

Type

`array`

.status.v1beta2.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.

Property	Type	Description
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vphereclusteridentities`
  - `DELETE`: delete collection of VSphereClusterIdentity
  - `GET`: list objects of kind VSphereClusterIdentity
  - `POST`: create a new VSphereClusterIdentity
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vphereclusteridentities/{name}`
  - `DELETE`: delete the specified VSphereClusterIdentity
  - `GET`: read the specified VSphereClusterIdentity
  - `PATCH`: partially update the specified VSphereClusterIdentity
  - `PUT`: replace the specified VSphereClusterIdentity
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vphereclusteridentities/{name}/status`
  - `GET`: read status of the specified VSphereClusterIdentity
  - `PATCH`: partially update status of the specified VSphereClusterIdentity
  - `PUT`: replace status of the specified VSphereClusterIdentity

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vphereclusteridentities

HTTP method

`DELETE`

Description

delete collection of VSphereClusterIdentity

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereClusterIdentity

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentityList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereClusterIdentity

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereClusterIdentity` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentity` schema
201 - Created	`VSphereClusterIdentity` schema
202 - Accepted	`VSphereClusterIdentity` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vsphereclusteridentities/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereClusterIdentity

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereClusterIdentity

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentity` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereClusterIdentity

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentity` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereClusterIdentity

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereClusterIdentity` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentity` schema
201 - Created	`VSphereClusterIdentity` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vsphereclusteridentities/{name}/status

HTTP method

`GET`

Description

read status of the specified VSphereClusterIdentity

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentity` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VSphereClusterIdentity

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentity` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VSphereClusterIdentity

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereClusterIdentity` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterIdentity` schema
201 - Created	`VSphereClusterIdentity` schema

HTTP code	Response body
401 - Unauthorized	Empty

VSphereMachine [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereMachine is the Schema for the vspheremachines API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereMachineSpec defines the desired state of VSphereMachine.
`status`	`object`	VSphereMachineStatus defines the observed state of VSphereMachine.

.spec

Description

VSphereMachineSpec defines the desired state of VSphereMachine.

Type

`object`

Required

`network` `template`

Property	Type	Description
`additionalDisksGiB`	`array`	AdditionalDisksGiB holds the sizes of additional disks of the virtual machine, in GiB Defaults to the eponymous property

Property	Type	Description
		value in the template from which the virtual machine is cloned.
`cloneMode`	`string`	CloneMode specifies the type of clone operation. The LinkedClone mode is only supported for templates that have at least one snapshot. If the template has no snapshots, then CloneMode defaults to FullClone. When LinkedClone mode is enabled the DiskGiB field is ignored as it is not possible to expand disks of linked clones. Defaults to LinkedClone, but fails gracefully to FullClone if the source of the clone operation has no snapshots.
`customVMXKeys`	`object`	CustomVMXKeys is a dictionary of advanced VMX options that can be set on VM Defaults to empty map
`dataDisks`	`array`	DataDisks are additional disks to add to the VM that are not part of the VM's OVA template.
`datacenter`	`string`	Datacenter is the name, inventory path, managed object reference or the managed object ID of the datacenter in which the virtual machine is created/located. Defaults to * which selects the default datacenter.

Property	Type	Description
`datastore`	`string`	Datastore is the name, inventory path, managed object reference or the managed object ID of the datastore in which the virtual machine is created/located.
`diskGiB`	`integer`	DiskGiB is the size of a virtual machine's disk, in GiB. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`failureDomain`	`string`	FailureDomain is the failure domain unique identifier this Machine should be attached to, as defined in Cluster API. For this infrastructure provider, the name is equivalent to the name of the VSphereDeploymentZone.
`folder`	`string`	Folder is the name, inventory path, managed object reference or the managed object ID of the folder in which the virtual machine is created/located.
`guestSoftPowerOffTimeout`	`string`	GuestSoftPowerOffTimeout sets the wait timeout for shutdown in the VM guest. The VM will be powered off forcibly after the timeout if the VM is still up and running when the PowerOffMode is set to trySoft.

Property	Type	Description
		This parameter only applies when the PowerOffMode is set to trySoft. If omitted, the timeout defaults to 5 minutes.
`hardwareVersion`	`string`	HardwareVersion is the hardware version of the virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Check the compatibility with the ESXi version before setting the value.
`machineConfigPoolRef`	`object`	MachineConfigPoolRef is a reference to the VSphereMachineConfigPool to use for this machine.
`memoryMiB`	`integer`	MemoryMiB is the size of a virtual machine's memory, in MiB. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`namingStrategy`	`object`	NamingStrategy allows configuring the naming strategy used when calculating the name of the VSphereVM.
`network`	`object`	Network is the network configuration for this machine's VM.

Property	Type	Description
`numCPUs`	`integer`	NumCPUs is the number of virtual processors in a virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`numCoresPerSocket`	`integer`	NumCoresPerSocket is the number of cores per socket in a virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`os`	`string`	OS is the Operating System of the virtual machine Defaults to Linux
`pciDevices`	`array`	PciDevices is the list of pci devices used by the virtual machine.
`powerOffMode`	`string`	PowerOffMode describes the desired behavior when powering off a VM. There are three supported power off modes: hard, soft, and trySoft. The first mode, hard, is the equivalent of a physical system's power cord being ripped from the wall. The soft mode requires the VM's guest to have VM Tools installed and attempts to gracefully shut down the VM. Its variant, trySoft, first attempts a graceful shutdown,

Property	Type	Description
		and if that fails or the VM is not in a powered off state after reaching the GuestSoftPowerOffTimeout, the VM is halted. If omitted, the mode defaults to hard.
`providerID`	`string`	ProviderID is the virtual machine's BIOS UUID formatted as vsphere://12345678-1234-1234-1234-123456789abc
`resourcePool`	`string`	ResourcePool is the name, inventory path, managed object reference or the managed object ID in which the virtual machine is created/located.
`server`	`string`	Server is the IP address or FQDN of the vSphere server on which the virtual machine is created/located.
`snapshot`	`string`	Snapshot is the name of the snapshot from which to create a linked clone. This field is ignored if LinkedClone is not enabled. Defaults to the source's current snapshot.
`storagePolicyName`	`string`	StoragePolicyName of the storage policy to use with this Virtual Machine

Property	Type	Description
`tagIDs`	`array`	TagIDs is an optional set of tags to add to an instance. Specified tagIDs must use URN-notation instead of display names.
`template`	`string`	Template is the name, inventory path, managed object reference or the managed object ID of the template used to clone the virtual machine.
`thumbprint`	`string`	Thumbprint is the colon-separated SHA-1 checksum of the given vCenter server's host certificate When this is set to empty, this VirtualMachine would be created without TLS certificate validation of the communication between Cluster API Provider vSphere and the VMware vCenter server.

.spec.additionalDisksGiB

Description

AdditionalDisksGiB holds the sizes of additional disks of the virtual machine, in GiB Defaults to the eponymous property value in the template from which the virtual machine is cloned.

Type

`array`

.spec.additionalDisksGiB[]

Type

`integer`

.spec.customVMXKeys

Description

CustomVMXKeys is a dictionary of advanced VMX options that can be set on VM Defaults to empty map

Type

`object`

.spec.dataDisks

Description

DataDisks are additional disks to add to the VM that are not part of the VM's OVA template.

Type

`array`

.spec.dataDisks[]

Description

VSphereDisk is an additional disk to add to the VM that is not part of the VM OVA template.

Type

`object`

Required

`name` `sizeGiB`

Property	Type	Description
`name`	`string`	Name is used to identify the disk definition. Name is required and needs to be unique so that it can be used to clearly identify purpose of the disk.

Property	Type	Description
`provisioningMode`	`string`	ProvisioningMode specifies the provisioning type to be used by this vSphere data disk. If not set, the setting will be provided by the default storage policy.
`sizeGiB`	`integer`	SizeGiB is the size of the disk in GiB.

.spec.machineConfigPoolRef

Description

MachineConfigPoolRef is a reference to the VSphereMachineConfigPool to use for this machine.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2

Property	Type	Description
		in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.namingStrategy

Description

NamingStrategy allows configuring the naming strategy used when calculating the name of the VSphereVM.

Type

`object`

Property	Type	Description
`template`	`string`	Template defines the template to use for generating the name of the VSphereVM object. If not defined, it will fall back to `{{ .machine.name }}`. The templating has the following data available: • `.machine.name`: The name of the Machine object. The templating also has the following funcs available: • `trimSuffix`: same as strings.TrimSuffix • `trunc`: truncates a string, e.g. `trunc 2 "hello"` or `trunc -2 "hello"` Notes: • While the template offers some flexibility, we would like the name to link to the Machine name to ensure better user experience when troubleshooting • Generated names must be valid Kubernetes names as they are used to create a VSphereVM object and usually also as the name of the Node object. • Names are automatically truncated at 63 characters. Please note that this can lead to name conflicts, so we highly recommend to use a template which leads to a name shorter than 63 characters.

.spec.network

Description

Network is the network configuration for this machine's VM.

Type

`object`

Required

`devices`

Property	Type	Description
`devices`	`array`	Devices is the list of network devices used by the virtual machine.
`preferredAPIServerCidr`	`string`	PreferredAPIServerCIDR is the preferred CIDR for the Kubernetes API server endpoint on this machine Deprecated: This field is going to be removed in a future release.
`routes`	`array`	Routes is a list of optional, static routes applied to the virtual machine.

.spec.network.devices

Description

Devices is the list of network devices used by the virtual machine.

Type

`array`

.spec.network.devices[]

Description

NetworkDeviceSpec defines the network configuration for a virtual machine's network device.

Type

`object`

Required

`networkName`

Property	Type	Description
`addressesFromPools`	`array`	AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.
`deviceName`	`string`	DeviceName may be used to explicitly assign a name to the network device as it exists in the guest operating system.
`dhcp4`	`boolean`	DHCP4 is a flag that indicates whether or not to use DHCP for IPv4 on this device. If true then IPAddrs should not contain any IPv4 addresses.

Property	Type	Description
`dhcp4Overrides`	`object`	DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides ↗)
`dhcp6`	`boolean`	DHCP6 is a flag that indicates whether or not to use DHCP for IPv6 on this device. If true then IPAddrs should not contain any IPv6 addresses.
`dhcp6Overrides`	`object`	DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides ↗)
`gateway4`	`string`	Gateway4 is the IPv4 gateway used by this device. Required when DHCP4 is false.
`gateway6`	`string`	Gateway6 is the IPv6 gateway used by this device. Required when DHCP6 is false.

Property	Type	Description
`ipAddrs`	`array`	IPAddrs is a list of one or more IPv4 and/or IPv6 addresses to assign to this device. IP addresses must also specify the segment length in CIDR notation. Required when DHCP4, DHCP6 and SkipIPAllocation are false.
`macAddr`	`string`	MACAddr is the MAC address used by this device. It is generally a good idea to omit this field and allow a MAC address to be generated. Please note that this value must use the VMware OUI to work with the in-tree vSphere cloud provider.
`mtu`	`integer`	MTU is the device's Maximum Transmission Unit size in bytes.
`nameservers`	`array`	Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf↗).
`networkName`	`string`	NetworkName is the name, managed object reference or the managed object ID of the vSphere network to which the device will be connected.
`routes`	`array`	Routes is a list of optional, static routes applied to the device.

Property	Type	Description
`searchDomains`	`array`	SearchDomains is a list of search domains used when resolving IP addresses with DNS.
`skipIPAllocation`	`boolean`	SkipIPAllocation allows the device to not have IP address or DHCP configured. This is suitable for devices for which IP allocation is handled externally, eg. using Multus CNI. If true, CAPV will not verify IP address allocation.

.spec.network.devices[].addressesFromPools

Description

AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.

Type

`array`

.spec.network.devices[].addressesFromPools[]

Description

TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.network.devices[].dhcp4Overrides

Description

DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.

Property	Type	Description
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take

Property	Type	Description
		precedence.
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.network.devices[].dhcp6Overrides

Description

DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.

Property	Type	Description
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take precedence.
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.network.devices[].ipAddrs

Description

IPAddrs is a list of one or more IPv4 and/or IPv6 addresses to assign to this device. IP addresses must also specify the segment length in CIDR notation. Required when DHCP4, DHCP6 and SkipIPAllocation are false.

Type

`array`

.spec.network.devices[].ipAddrs[]

Type

`string`

.spec.network.devices[].nameservers

Description

Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf).

Type

`array`

.spec.network.devices[].nameservers[]

Type

`string`

.spec.network.devices[].routes

Description

Routes is a list of optional, static routes applied to the device.

Type

`array`

.spec.network.devices[].routes[]

Description

NetworkRouteSpec defines a static network route.

Type

`object`

Required

`metric` `to` `via`

Property	Type	Description
`metric`	`integer`	Metric is the weight/priority of the route.
`to`	`string`	To is an IPv4 or IPv6 address.
`via`	`string`	Via is an IPv4 or IPv6 address.

.spec.network.devices[].searchDomains

Description

SearchDomains is a list of search domains used when resolving IP addresses with DNS.

Type

`array`

.spec.network.devices[].searchDomains[]

Type

`string`

.spec.network.routes

Description

Routes is a list of optional, static routes applied to the virtual machine.

Type

`array`

.spec.network.routes[]

Description

NetworkRouteSpec defines a static network route.

Type

`object`

Required

`metric` `to` `via`

Property	Type	Description
`metric`	`integer`	Metric is the weight/priority of the route.
`to`	`string`	To is an IPv4 or IPv6 address.
`via`	`string`	Via is an IPv4 or IPv6 address.

.spec.pciDevices

Description

PciDevices is the list of pci devices used by the virtual machine.

Type

`array`

.spec.pciDevices[]

Description

PCIDeviceSpec defines virtual machine's PCI configuration.

Type

`object`

Property	Type	Description
`customLabel`	`string`	CustomLabel is the hardware label of a virtual machine's PCI device. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`deviceId`	`integer`	DeviceID is the device ID of a virtual machine's PCI, in integer. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with VGPUProfile as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.
`vGPUProfile`	`string`	VGPUProfile is the profile name of a virtual machine's vGPU, in string. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with DeviceID and VendorID as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.

Property	Type	Description
`vendorId`	`integer`	VendorId is the vendor ID of a virtual machine's PCI, in integer. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with VGPUProfile as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.

.spec.tagIDs

Description

TagIDs is an optional set of tags to add to an instance. Specified tagIDs must use URN-notation instead of display names.

Type

`array`

.spec.tagIDs[]

Type

`string`

.status

Description

VSphereMachineStatus defines the observed state of VSphereMachine.

Type

`object`

Property	Type	Description
`addresses`	`array`	Addresses contains the VSphere instance associated addresses.
`conditions`	`array`	Conditions defines current service state of the VSphereMachine.
`failureMessage`	`string`	FailureMessage will be set in the event that there is a terminal problem reconciling the Machine and will contain a more verbose string suitable for logging and human consumption. This field should not be set for transitive errors that a controller faces that are expected to be fixed automatically over time (like service outages), but instead indicate that something is fundamentally wrong with the Machine's spec or the configuration of the controller, and that manual intervention is required. Examples of terminal errors would be invalid combinations of settings in the spec, values that are unsupported by the controller, or the responsible controller itself being critically misconfigured. Any transient errors that occur during the reconciliation of Machines can be added as events to the Machine object and/or logged in the controller's output.
`failureReason`	`string`	FailureReason will be set in the event that there is a terminal problem reconciling the Machine and will contain a succinct value suitable for machine interpretation.

Property	Type	Description
		This field should not be set for transitive errors that a controller faces that are expected to be fixed automatically over time (like service outages), but instead indicate that something is fundamentally wrong with the Machine's spec or the configuration of the controller, and that manual intervention is required. Examples of terminal errors would be invalid combinations of settings in the spec, values that are unsupported by the controller, or the responsible controller itself being critically misconfigured. Any transient errors that occur during the reconciliation of Machines can be added as events to the Machine object and/or logged in the controller's output.
`network`	`array`	Network returns the network status for each of the machine's configured network interfaces.
`ready`	`boolean`	Ready is true when the provider resource is ready.
`v1beta2`	`object`	v1beta2 groups all the fields that will be added or modified in VSphereMachine's status with the V1Beta2 version.

.status.addresses

Description

Addresses contains the VSphere instance associated addresses.

Type

`array`

.status.addresses[]

Description

MachineAddress contains information for the node's address.

Type

`object`

Required

`address` `type`

Property	Type	Description
`address`	`string`	address is the machine address.
`type`	`string`	type is the machine address type, one of Hostname, ExternalIP, InternalIP, ExternalDNS or InternalDNS.

.status.conditions

Description

Conditions defines current service state of the VSphereMachine.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.

Property	Type	Description
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.network

Description

Network returns the network status for each of the machine's configured network interfaces.

Type

`array`

.status.network[]

Description

NetworkStatus provides information about one of a VM's networks.

Type

`object`

Required

`macAddr`

Property	Type	Description
`connected`	`boolean`	Connected is a flag that indicates whether this network is currently connected to the VM.

Property	Type	Description
`ipAddrs`	`array`	IPAddrs is one or more IP addresses reported by vm-tools.
`macAddr`	`string`	MACAddr is the MAC address of the network device.
`networkName`	`string`	NetworkName is the name of the network.

.status.network[].ipAddrs

Description

IPAddrs is one or more IP addresses reported by vm-tools.

Type

`array`

.status.network[].ipAddrs[]

Type

`string`

.status.v1beta2

Description

v1beta2 groups all the fields that will be added or modified in VSphereMachine's status with the V1Beta2 version.

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions represents the observations of a VSphereMachine's current state. Known condition types are Ready, VirtualMachineProvisioned and Paused.

.status.v1beta2.conditions

Description

conditions represents the observations of a VSphereMachine's current state. Known condition types are Ready, VirtualMachineProvisioned and Paused.

Type

`array`

.status.v1beta2.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.

Property	Type	Description
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachines`
  - `DELETE` : delete collection of VSphereMachine
  - `GET` : list objects of kind VSphereMachine
  - `POST` : create a new VSphereMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachines/{name}`
  - `DELETE` : delete the specified VSphereMachine
  - `GET` : read the specified VSphereMachine
  - `PATCH` : partially update the specified VSphereMachine
  - `PUT` : replace the specified VSphereMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachines/{name}/status`
  - `GET` : read status of the specified VSphereMachine
  - `PATCH` : partially update status of the specified VSphereMachine
  - `PUT` : replace status of the specified VSphereMachine

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachines

HTTP method

`DELETE`

Description

delete collection of VSphereMachine

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereMachine

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachine` schema
201 - Created	`VSphereMachine` schema
202 - Accepted	`VSphereMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachines/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereMachine

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachine` schema
201 - Created	`VSphereMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachines/{name}/status

HTTP method

`GET`

Description

read status of the specified VSphereMachine

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VSphereMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VSphereMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachine` schema

HTTP code	Response body
201 - Created	`VSphereMachine` schema
401 - Unauthorized	Empty

VSphereClusterTemplate [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereClusterTemplate is the Schema for the vsphereclustertemplates API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereClusterTemplateSpec defines the desired state of VSphereClusterTemplate.

.spec

Description

VSphereClusterTemplateSpec defines the desired state of VSphereClusterTemplate.

Type

`object`

Required

`template`

Property	Type	Description
`template`	`object`	VSphereClusterTemplateResource describes the data for creating a VSphereCluster from a template.

.spec.template

Description

VSphereClusterTemplateResource describes the data for creating a VSphereCluster from a template.

Type

`object`

Required

`spec`

Property	Type	Description
`spec`	`object`	VSphereClusterSpec defines the desired state of VSphereCluster.

.spec.template.spec

Description

VSphereClusterSpec defines the desired state of VSphereCluster.

Type

`object`

Property	Type	Description
`clusterModules`	`array`	ClusterModules hosts information regarding the anti-affinity vSphere constructs for each of the objects responsible for creation of VM objects belonging to the cluster.
`controlPlaneEndpoint`	`object`	ControlPlaneEndpoint represents the endpoint used to communicate with the control plane.

Property	Type	Description
`controlPlaneLoadBalancer`	`object`	ControlPlaneLoadBalancer describes the control plane endpoint's load balancer. Leaving it nil is allowed and means the same as type=external: the endpoint is provided by the user and the provider manages no VIP. The field is set when the cluster is created or never: it cannot be added to an existing cluster, and once set every field is immutable and the field cannot be cleared. The endpoint, the apiserver serving certificate and the guest runtime are all derived from it and cannot be changed in place.
`disableClusterModule`	`boolean`	DisableClusterModule is used to explicitly turn off the ClusterModule feature. This should work along side NodeAntiAffinity feature flag. If the NodeAntiAffinity feature flag is turned off, this will be disregarded.
`failureDomainSelector`	`object`	FailureDomainSelector is the label selector to use for failure domain selection for the control plane nodes of the cluster. If not set (`nil`), selecting failure domains will be disabled. An empty value (`{}`) selects all existing failure domains. A valid selector will select all failure domains which match the selector.

Property	Type	Description
`identityRef`	`object`	IdentityRef is a reference to either a Secret or VSphereClusterIdentity that contains the identity to use when reconciling the cluster.
`server`	`string`	Server is the address of the vSphere endpoint.
`thumbprint`	`string`	Thumbprint is the colon-separated SHA-1 checksum of the given vCenter server's host certificate

.spec.template.spec.clusterModules

Description

ClusterModules hosts information regarding the anti-affinity vSphere constructs for each of the objects responsible for creation of VM objects belonging to the cluster.

Type

`array`

.spec.template.spec.clusterModules[]

Description

ClusterModule holds the anti affinity construct `ClusterModule` identifier in use by the VMs owned by the object referred by the TargetObjectName field.

Type

`object`

Required

`controlPlane` `moduleUUID` `targetObjectName`

Property	Type	Description
`controlPlane`	`boolean`	ControlPlane indicates whether the referred object is responsible for control plane nodes. Currently, only the KubeadmControlPlane objects have this flag set to true. Only a single object in the slice can have this value set to true.
`moduleUUID`	`string`	ModuleUUID is the unique identifier of the `ClusterModule` used by the object.
`targetObjectName`	`string`	TargetObjectName points to the object that uses the Cluster Module information to enforce anti-affinity amongst its descendant VM objects.

.spec.template.spec.controlPlaneEndpoint

Description

ControlPlaneEndpoint represents the endpoint used to communicate with the control plane.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	The hostname on which the API server is serving.

Property	Type	Description
`port`	`integer`	The port on which the API server is serving.

.spec.template.spec.controlPlaneLoadBalancer

Description

ControlPlaneLoadBalancer describes the control plane endpoint's load balancer. Leaving it nil is allowed and means the same as type=external: the endpoint is provided by the user and the provider manages no VIP. The field is set when the cluster is created or never: it cannot be added to an existing cluster, and once set every field is immutable and the field cannot be cleared. The endpoint, the apiserver serving certificate and the guest runtime are all derived from it and cannot be changed in place.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	Host is the control plane endpoint address. For type=internal it is the IPv4 VIP.
`interface`	`string`	Interface pins the guest NIC that holds the VIP. Empty means auto-detect by matching the node's primary IP.
`port`	`integer`	Port is the control plane endpoint port.

Property	Type	Description
`type`	`string`	Type selects who owns the control plane VIP.
`vrid`	`integer`	VRID is the keepalived VRRP router ID. Only used when type=internal. The 1-255 range is validated by the webhook, not by the CRD schema, so that type=external objects are not rejected for leaving it at zero.

.spec.template.spec.failureDomainSelector

Description

FailureDomainSelector is the label selector to use for failure domain selection for the control plane nodes of the cluster. If not set (`nil`), selecting failure domains will be disabled. An empty value (`{}`) selects all existing failure domains. A valid selector will select all failure domains which match the selector.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.failureDomainSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.failureDomainSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.failureDomainSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.failureDomainSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.failureDomainSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.identityRef

Description

IdentityRef is a reference to either a Secret or VSphereClusterIdentity that contains the identity to use when reconciling the cluster.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	Kind of the identity. Can either be VSphereClusterIdentity or Secret
`name`	`string`	Name of the identity.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclustertemplates`
  - `DELETE`: delete collection of VSphereClusterTemplate
  - `GET`: list objects of kind VSphereClusterTemplate
  - `POST`: create a new VSphereClusterTemplate
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclustertemplates/{name}`
  - `DELETE`: delete the specified VSphereClusterTemplate

- `GET`: read the specified VSphereClusterTemplate
- `PATCH`: partially update the specified VSphereClusterTemplate
- `PUT`: replace the specified VSphereClusterTemplate

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclustertemplates

HTTP method

`DELETE`

Description

delete collection of VSphereClusterTemplate

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereClusterTemplate

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterTemplateList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereClusterTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereClusterTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterTemplate` schema
201 - Created	`VSphereClusterTemplate` schema
202 - Accepted	`VSphereClusterTemplate` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vsphereclustertemplates/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereClusterTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereClusterTemplate

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereClusterTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereClusterTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereClusterTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereClusterTemplate` schema
201 - Created	`VSphereClusterTemplate` schema
401 - Unauthorized	Empty

VSphereVM [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereVM is the Schema for the vspherevms API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereVMSpec defines the desired state of VSphereVM.
`status`	`object`	VSphereVMStatus defines the observed state of VSphereVM.

.spec

Description

VSphereVMSpec defines the desired state of VSphereVM.

Type

`object`

Required

`network` `template`

Property	Type	Description
`additionalDisksGiB`	`array`	AdditionalDisksGiB holds the sizes of additional disks of the virtual machine, in GiB Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`biosUUID`	`string`	BiosUUID is the VM's BIOS UUID that is assigned at runtime after the VM has been created. This field is required at runtime for other controllers that read this CRD as unstructured data.
`bootstrapRef`	`object`	BootstrapRef is a reference to a bootstrap provider-specific resource that holds configuration details. This field is optional in case no bootstrap data is required to create a VM.
`cloneMode`	`string`	CloneMode specifies the type of clone operation. The LinkedClone mode is only supported for templates that have at least one snapshot. If the template has no snapshots, then CloneMode defaults to FullClone. When LinkedClone mode is enabled the DiskGiB field is ignored as it is not possible to expand disks of linked clones. Defaults to LinkedClone, but fails gracefully to FullClone if the source of the clone operation has no snapshots.

Property	Type	Description
`customVMXKeys`	`object`	CustomVMXKeys is a dictionary of advanced VMX options that can be set on VM Defaults to empty map
`dataDisks`	`array`	DataDisks are additional disks to add to the VM that are not part of the VM's OVA template.
`datacenter`	`string`	Datacenter is the name, inventory path, managed object reference or the managed object ID of the datacenter in which the virtual machine is created/located. Defaults to * which selects the default datacenter.
`datastore`	`string`	Datastore is the name, inventory path, managed object reference or the managed object ID of the datastore in which the virtual machine is created/located.
`diskGiB`	`integer`	DiskGiB is the size of a virtual machine's disk, in GiB. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`folder`	`string`	Folder is the name, inventory path, managed object reference or the managed

Property	Type	Description
		object ID of the folder in which the virtual machine is created/located.
`guestSoftPowerOffTimeout`	`string`	GuestSoftPowerOffTimeout sets the wait timeout for shutdown in the VM guest. The VM will be powered off forcibly after the timeout if the VM is still up and running when the PowerOffMode is set to trySoft. This parameter only applies when the PowerOffMode is set to trySoft. If omitted, the timeout defaults to 5 minutes.
`hardwareVersion`	`string`	HardwareVersion is the hardware version of the virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Check the compatibility with the ESXi version before setting the value.
`memoryMiB`	`integer`	MemoryMiB is the size of a virtual machine's memory, in MiB. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`network`	`object`	Network is the network configuration for this machine's VM.

Property	Type	Description
`numCPUs`	`integer`	NumCPUs is the number of virtual processors in a virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`numCoresPerSocket`	`integer`	NumCoresPerSocket is the number of cores per socket in a virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`os`	`string`	OS is the Operating System of the virtual machine Defaults to Linux
`pciDevices`	`array`	PciDevices is the list of pci devices used by the virtual machine.
`powerOffMode`	`string`	PowerOffMode describes the desired behavior when powering off a VM. There are three supported power off modes: hard, soft, and trySoft. The first mode, hard, is the equivalent of a physical system's power cord being ripped from the wall. The soft mode requires the VM's guest to have VM Tools installed and attempts to gracefully shut down the VM. Its variant, trySoft, first attempts a graceful shutdown,

Property	Type	Description
		and if that fails or the VM is not in a powered off state after reaching the GuestSoftPowerOffTimeout, the VM is halted. If omitted, the mode defaults to hard.
`resourcePool`	`string`	ResourcePool is the name, inventory path, managed object reference or the managed object ID in which the virtual machine is created/located.
`server`	`string`	Server is the IP address or FQDN of the vSphere server on which the virtual machine is created/located.
`snapshot`	`string`	Snapshot is the name of the snapshot from which to create a linked clone. This field is ignored if LinkedClone is not enabled. Defaults to the source's current snapshot.
`storagePolicyName`	`string`	StoragePolicyName of the storage policy to use with this Virtual Machine
`tagIDs`	`array`	TagIDs is an optional set of tags to add to an instance. Specified tagIDs must use URN-notation instead of display names.

Property	Type	Description
`template`	`string`	Template is the name, inventory path, managed object reference or the managed object ID of the template used to clone the virtual machine.
`thumbprint`	`string`	Thumbprint is the colon-separated SHA-1 checksum of the given vCenter server's host certificate When this is set to empty, this VirtualMachine would be created without TLS certificate validation of the communication between Cluster API Provider vSphere and the VMware vCenter server.

.spec.additionalDisksGiB

Description

AdditionalDisksGiB holds the sizes of additional disks of the virtual machine, in GiB Defaults to the eponymous property value in the template from which the virtual machine is cloned.

Type

`array`

.spec.additionalDisksGiB[]

Type

`integer`

.spec.bootstrapRef

Description

BootstrapRef is a reference to a bootstrap provider-specific resource that holds configuration details. This field is optional in case no bootstrap data is required to create a VM.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.customVMXKeys

Description

CustomVMXKeys is a dictionary of advanced VMX options that can be set on VM Defaults to empty map

Type

`object`

.spec.dataDisks

Description

DataDisks are additional disks to add to the VM that are not part of the VM's OVA template.

Type

`array`

.spec.dataDisks[]

Description

VSphereDisk is an additional disk to add to the VM that is not part of the VM OVA template.

Type

`object`

Required

`name` `sizeGiB`

Property	Type	Description
`name`	`string`	Name is used to identify the disk definition. Name is required and needs to be unique so that it can be used to clearly identify purpose of the disk.
`provisioningMode`	`string`	ProvisioningMode specifies the provisioning type to be used by this vSphere data disk. If not set, the setting will be provided by the default storage policy.
`sizeGiB`	`integer`	SizeGiB is the size of the disk in GiB.

.spec.network

Description

Network is the network configuration for this machine's VM.

Type

`object`

Required

`devices`

Property	Type	Description
`devices`	`array`	Devices is the list of network devices used by the virtual machine.
`preferredAPIServerCidr`	`string`	PreferredAPIServerCIDR is the preferred CIDR for the Kubernetes API server endpoint on this machine Deprecated: This field is going to be removed in a future release.
`routes`	`array`	Routes is a list of optional, static routes applied to the virtual machine.

.spec.network.devices

Description

Devices is the list of network devices used by the virtual machine.

Type

`array`

.spec.network.devices[]

Description

NetworkDeviceSpec defines the network configuration for a virtual machine's network device.

Type

`object`

Required

`networkName`

Property	Type	Description
`addressesFromPools`	`array`	AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.
`deviceName`	`string`	DeviceName may be used to explicitly assign a name to the network device as it exists in the guest operating system.
`dhcp4`	`boolean`	DHCP4 is a flag that indicates whether or not to use DHCP for IPv4 on this device. If true then IPAddrs should not contain any IPv4 addresses.
`dhcp4Overrides`	`object`	DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides↗)

Property	Type	Description
`dhcp6`	`boolean`	DHCP6 is a flag that indicates whether or not to use DHCP for IPv6 on this device. If true then IPAddrs should not contain any IPv6 addresses.
`dhcp6Overrides`	`object`	DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides↗)
`gateway4`	`string`	Gateway4 is the IPv4 gateway used by this device. Required when DHCP4 is false.
`gateway6`	`string`	Gateway6 is the IPv6 gateway used by this device. Required when DHCP6 is false.
`ipAddrs`	`array`	IPAddrs is a list of one or more IPv4 and/or IPv6 addresses to assign to this device. IP addresses must also specify the segment length in CIDR notation. Required when DHCP4, DHCP6 and SkipIPAllocation are false.
`macAddr`	`string`	MACAddr is the MAC address used by this device. It is generally a good idea to omit this field and

Property	Type	Description
		allow a MAC address to be generated. Please note that this value must use the VMware OUI to work with the in-tree vSphere cloud provider.
`mtu`	`integer`	MTU is the device's Maximum Transmission Unit size in bytes.
`nameservers`	`array`	Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf↗).
`networkName`	`string`	NetworkName is the name, managed object reference or the managed object ID of the vSphere network to which the device will be connected.
`routes`	`array`	Routes is a list of optional, static routes applied to the device.
`searchDomains`	`array`	SearchDomains is a list of search domains used when resolving IP addresses with DNS.

Property	Type	Description
`skipIPAllocation`	`boolean`	SkipIPAllocation allows the device to not have IP address or DHCP configured. This is suitable for devices for which IP allocation is handled externally, eg. using Multus CNI. If true, CAPV will not verify IP address allocation.

.spec.network.devices[].addressesFromPools

Description

AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.

Type

`array`

.spec.network.devices[].addressesFromPools[]

Description

TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in

Property	Type	Description
		the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.network.devices[].dhcp4Overrides

Description

DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.

Property	Type	Description
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take precedence.

Property	Type	Description
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.network.devices[].dhcp6Overrides

Description

DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.

Property	Type	Description
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take precedence.
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.network.devices[].ipAddrs

Description

IPAddrs is a list of one or more IPv4 and/or IPv6 addresses to assign to this device. IP addresses must also specify the segment length in CIDR notation. Required when DHCP4, DHCP6 and SkipIPAllocation are false.

Type

`array`

.spec.network.devices[].ipAddrs[]

Type

`string`

.spec.network.devices[].nameservers

Description

Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf).

Type

`array`

.spec.network.devices[].nameservers[]

Type

`string`

.spec.network.devices[].routes

Description

Routes is a list of optional, static routes applied to the device.

Type

`array`

.spec.network.devices[].routes[]

Description

NetworkRouteSpec defines a static network route.

Type

`object`

Required

`metric` `to` `via`

Property	Type	Description
`metric`	`integer`	Metric is the weight/priority of the route.
`to`	`string`	To is an IPv4 or IPv6 address.
`via`	`string`	Via is an IPv4 or IPv6 address.

.spec.network.devices[].searchDomains

Description

SearchDomains is a list of search domains used when resolving IP addresses with DNS.

Type

`array`

.spec.network.devices[].searchDomains[]

Type

`string`

.spec.network.routes

Description

Routes is a list of optional, static routes applied to the virtual machine.

Type

`array`

.spec.network.routes[]

Description

NetworkRouteSpec defines a static network route.

Type

`object`

Required

`metric` `to` `via`

Property	Type	Description
`metric`	`integer`	Metric is the weight/priority of the route.
`to`	`string`	To is an IPv4 or IPv6 address.
`via`	`string`	Via is an IPv4 or IPv6 address.

.spec.pciDevices

Description

PciDevices is the list of pci devices used by the virtual machine.

Type

`array`

.spec.pciDevices[]

Description

PCIDeviceSpec defines virtual machine's PCI configuration.

Type

`object`

Property	Type	Description
`customLabel`	`string`	CustomLabel is the hardware label of a virtual machine's PCI device. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`deviceId`	`integer`	DeviceID is the device ID of a virtual machine's PCI, in integer. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with VGPUProfile as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.
`vGPUProfile`	`string`	VGPUProfile is the profile name of a virtual machine's vGPU, in string. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with DeviceID and VendorID as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.

Property	Type	Description
`vendorId`	`integer`	VendorId is the vendor ID of a virtual machine's PCI, in integer. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with VGPUProfile as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.

.spec.tagIDs

Description

TagIDs is an optional set of tags to add to an instance. Specified tagIDs must use URN-notation instead of display names.

Type

`array`

.spec.tagIDs[]

Type

`string`

.status

Description

VSphereVMStatus defines the observed state of VSphereVM.

Type

`object`

Property	Type	Description
`addresses`	`array`	Addresses is a list of the VM's IP addresses. This field is required at runtime for other controllers that read this CRD as unstructured data.
`cloneMode`	`string`	CloneMode is the type of clone operation used to clone this VM. Since LinkedClone is the default but fails gracefully if the source of the clone has no snapshots, this field may be used to determine the actual type of clone operation used to create this VM.
`conditions`	`array`	Conditions defines current service state of the VSphereVM.
`failureMessage`	`string`	FailureMessage will be set in the event that there is a terminal problem reconciling the vspherevm and will contain a more verbose string suitable for logging and human consumption. This field should not be set for transitive errors that a controller faces that are expected to be fixed automatically over time (like service outages), but instead indicate that something is fundamentally wrong with the vm. Any transient errors that occur during the reconciliation of vspherevms can be added as events to the vspherevm object and/or logged in the controller's output.

Property	Type	Description
`failureReason`	`string`	FailureReason will be set in the event that there is a terminal problem reconciling the vspherevm and will contain a succinct value suitable for vm interpretation. This field should not be set for transitive errors that a controller faces that are expected to be fixed automatically over time (like service outages), but instead indicate that something is fundamentally wrong with the vm. Any transient errors that occur during the reconciliation of vspherevms can be added as events to the vspherevm object and/or logged in the controller's output.
`host`	`string`	Host describes the hostname or IP address of the infrastructure host that the VSphereVM is residing on.
`moduleUUID`	`string`	ModuleUUID is the unique identifier for the vCenter cluster module construct which is used to configure anti-affinity. Objects with the same ModuleUUID will be anti-affined, meaning that the vCenter DRS will best effort schedule the VMs on separate hosts.
`network`	`array`	Network returns the network status for each of the machine's configured network interfaces.
`ready`	`boolean`	Ready is true when the provider resource is ready. This field is required at runtime for other controllers that read

Property	Type	Description
		this CRD as unstructured data.
`retryAfter`	`string`	RetryAfter tracks the time we can retry queueing a task
`snapshot`	`string`	Snapshot is the name of the snapshot from which the VM was cloned if LinkedClone is enabled.
`taskRef`	`string`	TaskRef is a managed object reference to a Task related to the machine. This value is set automatically at runtime and should not be set or modified by users.
`v1beta2`	`object`	v1beta2 groups all the fields that will be added or modified in VSphereVM's status with the V1Beta2 version.
`vmRef`	`string`	VMRef is the VM's Managed Object Reference on vSphere. It can be used by consumers to programatically get this VM representation on vSphere in case of the need to retrieve informations. This field is set once the machine is created and should not be changed

.status.addresses

Description

Addresses is a list of the VM's IP addresses. This field is required at runtime for other controllers that read this CRD as unstructured data.

Type

`array`

.status.addresses[]

Type

`string`

.status.conditions

Description

Conditions defines current service state of the VSphereVM.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.

Property	Type	Description
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.network

Description

Network returns the network status for each of the machine's configured network interfaces.

Type

`array`

.status.network[]

Description

NetworkStatus provides information about one of a VM's networks.

Type

`object`

Required

`macAddr`

Property	Type	Description
`connected`	`boolean`	Connected is a flag that indicates whether this network is currently connected to the VM.
`ipAddrs`	`array`	IPAddrs is one or more IP addresses reported by vm-tools.
`macAddr`	`string`	MACAddr is the MAC address of the network device.
`networkName`	`string`	NetworkName is the name of the network.

.status.network[].ipAddrs

Description

IPAddrs is one or more IP addresses reported by vm-tools.

Type

`array`

.status.network[].ipAddrs[]

Type

`string`

.status.v1beta2

Description

v1beta2 groups all the fields that will be added or modified in VSphereVM's status with the V1Beta2 version.

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions represents the observations of a VSphereVM's current state. Known condition types are Ready, VirtualMachineProvisioned, VCenterAvailable and IPAddressClaimsFulfilled, GuestSoftPowerOffSucceeded, PCIDevicesDetached and Paused.

.status.v1beta2.conditions

Description

conditions represents the observations of a VSphereVM's current state. Known condition types are Ready, VirtualMachineProvisioned, VCenterAvailable and IPAddressClaimsFulfilled, GuestSoftPowerOffSucceeded, PCIDevicesDetached and Paused.

Type

`array`

.status.v1beta2.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.

Property	Type	Description
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspherevms`

- `DELETE`: delete collection of VSphereVM
- `GET`: list objects of kind VSphereVM
- `POST`: create a new VSphereVM

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspherevms/{name}`
  - `DELETE`: delete the specified VSphereVM
  - `GET`: read the specified VSphereVM
  - `PATCH`: partially update the specified VSphereVM
  - `PUT`: replace the specified VSphereVM
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspherevms/{name}/status`
  - `GET`: read status of the specified VSphereVM
  - `PATCH`: partially update status of the specified VSphereVM
  - `PUT`: replace status of the specified VSphereVM

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspherevms

HTTP method

`DELETE`

Description

delete collection of VSphereVM

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereVM

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVMList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereVM

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereVM` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVM` schema
201 - Created	`VSphereVM` schema
202 - Accepted	`VSphereVM` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspherevms/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereVM

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereVM

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVM` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereVM

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVM` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereVM

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereVM` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVM` schema

HTTP code	Response body
201 - Created	`VSphereVM` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspherevms/{name}/status

HTTP method

`GET`

Description

read status of the specified VSphereVM

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVM` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VSphereVM

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVM` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VSphereVM

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereVM` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereVM` schema
201 - Created	`VSphereVM` schema

HTTP code	Response body
401 - Unauthorized	Empty

VSphereMachineTemplate [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereMachineTemplate is the Schema for the vspheremachinetemplates API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereMachineTemplateSpec defines the desired state of VSphereMachineTemplate.

.spec

Description

VSphereMachineTemplateSpec defines the desired state of VSphereMachineTemplate.

Type

`object`

Required

`template`

Property	Type	Description
`template`	`object`	VSphereMachineTemplateResource describes the data needed to create a VSphereMachine from a template.

.spec.template

Description

VSphereMachineTemplateResource describes the data needed to create a VSphereMachine from a template.

Type

`object`

Required

`spec`

Property	Type	Description
`metadata`	`ObjectMeta`	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata ↗
`spec`	`object`	Spec is the specification of the desired behavior of the machine.

.spec.template.spec

Description

Spec is the specification of the desired behavior of the machine.

Type

`object`

Required

`network` `template`

Property	Type	Description
`additionalDisksGiB`	`array`	AdditionalDisksGiB holds the sizes of additional disks of the virtual machine, in GiB Defaults to the eponymous property

Property	Type	Description
		value in the template from which the virtual machine is cloned.
`cloneMode`	`string`	CloneMode specifies the type of clone operation. The LinkedClone mode is only supported for templates that have at least one snapshot. If the template has no snapshots, then CloneMode defaults to FullClone. When LinkedClone mode is enabled the DiskGiB field is ignored as it is not possible to expand disks of linked clones. Defaults to LinkedClone, but fails gracefully to FullClone if the source of the clone operation has no snapshots.
`customVMXKeys`	`object`	CustomVMXKeys is a dictionary of advanced VMX options that can be set on VM Defaults to empty map
`dataDisks`	`array`	DataDisks are additional disks to add to the VM that are not part of the VM's OVA template.
`datacenter`	`string`	Datacenter is the name, inventory path, managed object reference or the managed object ID of the datacenter in which the virtual machine is created/located. Defaults to * which selects the default datacenter.

Property	Type	Description
`datastore`	`string`	Datastore is the name, inventory path, managed object reference or the managed object ID of the datastore in which the virtual machine is created/located.
`diskGiB`	`integer`	DiskGiB is the size of a virtual machine's disk, in GiB. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`failureDomain`	`string`	FailureDomain is the failure domain unique identifier this Machine should be attached to, as defined in Cluster API. For this infrastructure provider, the name is equivalent to the name of the VSphereDeploymentZone.
`folder`	`string`	Folder is the name, inventory path, managed object reference or the managed object ID of the folder in which the virtual machine is created/located.
`guestSoftPowerOffTimeout`	`string`	GuestSoftPowerOffTimeout sets the wait timeout for shutdown in the VM guest. The VM will be powered off forcibly after the timeout if the VM is still up and running when the PowerOffMode is set to trySoft.

Property	Type	Description
		This parameter only applies when the PowerOffMode is set to trySoft. If omitted, the timeout defaults to 5 minutes.
`hardwareVersion`	`string`	HardwareVersion is the hardware version of the virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Check the compatibility with the ESXi version before setting the value.
`machineConfigPoolRef`	`object`	MachineConfigPoolRef is a reference to the VSphereMachineConfigPool to use for this machine.
`memoryMiB`	`integer`	MemoryMiB is the size of a virtual machine's memory, in MiB. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`namingStrategy`	`object`	NamingStrategy allows configuring the naming strategy used when calculating the name of the VSphereVM.
`network`	`object`	Network is the network configuration for this machine's VM.

Property	Type	Description
`numCPUs`	`integer`	NumCPUs is the number of virtual processors in a virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`numCoresPerSocket`	`integer`	NumCoresPerSocket is the number of cores per socket in a virtual machine. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`os`	`string`	OS is the Operating System of the virtual machine Defaults to Linux
`pciDevices`	`array`	PciDevices is the list of pci devices used by the virtual machine.
`powerOffMode`	`string`	PowerOffMode describes the desired behavior when powering off a VM. There are three supported power off modes: hard, soft, and trySoft. The first mode, hard, is the equivalent of a physical system's power cord being ripped from the wall. The soft mode requires the VM's guest to have VM Tools installed and attempts to gracefully shut down the VM. Its variant, trySoft, first attempts a graceful shutdown,

Property	Type	Description
		and if that fails or the VM is not in a powered off state after reaching the GuestSoftPowerOffTimeout, the VM is halted. If omitted, the mode defaults to hard.
`providerID`	`string`	ProviderID is the virtual machine's BIOS UUID formatted as vsphere://12345678-1234-1234-1234-123456789abc
`resourcePool`	`string`	ResourcePool is the name, inventory path, managed object reference or the managed object ID in which the virtual machine is created/located.
`server`	`string`	Server is the IP address or FQDN of the vSphere server on which the virtual machine is created/located.
`snapshot`	`string`	Snapshot is the name of the snapshot from which to create a linked clone. This field is ignored if LinkedClone is not enabled. Defaults to the source's current snapshot.
`storagePolicyName`	`string`	StoragePolicyName of the storage policy to use with this Virtual Machine

Property	Type	Description
`tagIDs`	`array`	TagIDs is an optional set of tags to add to an instance. Specified tagIDs must use URN-notation instead of display names.
`template`	`string`	Template is the name, inventory path, managed object reference or the managed object ID of the template used to clone the virtual machine.
`thumbprint`	`string`	Thumbprint is the colon-separated SHA-1 checksum of the given vCenter server's host certificate When this is set to empty, this VirtualMachine would be created without TLS certificate validation of the communication between Cluster API Provider vSphere and the VMware vCenter server.

.spec.template.spec.additionalDisksGiB

Description

AdditionalDisksGiB holds the sizes of additional disks of the virtual machine, in GiB Defaults to the eponymous property value in the template from which the virtual machine is cloned.

Type

`array`

.spec.template.spec.additionalDisksGiB[]

Type

`integer`

.spec.template.spec.customVMXKeys

Description

CustomVMXKeys is a dictionary of advanced VMX options that can be set on VM Defaults to empty map

Type

`object`

.spec.template.spec.dataDisks

Description

DataDisks are additional disks to add to the VM that are not part of the VM's OVA template.

Type

`array`

.spec.template.spec.dataDisks[]

Description

VSphereDisk is an additional disk to add to the VM that is not part of the VM OVA template.

Type

`object`

Required

`name` `sizeGiB`

Property	Type	Description
`name`	`string`	Name is used to identify the disk definition. Name is required and needs to be unique so that it can be used to clearly identify purpose of the disk.

Property	Type	Description
`provisioningMode`	`string`	ProvisioningMode specifies the provisioning type to be used by this vSphere data disk. If not set, the setting will be provided by the default storage policy.
`sizeGiB`	`integer`	SizeGiB is the size of the disk in GiB.

.spec.template.spec.machineConfigPoolRef

Description

MachineConfigPoolRef is a reference to the VSphereMachineConfigPool to use for this machine.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2

Property	Type	Description
		in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.template.spec.namingStrategy

Description

NamingStrategy allows configuring the naming strategy used when calculating the name of the VSphereVM.

Type

`object`

Property	Type	Description
`template`	`string`	Template defines the template to use for generating the name of the VSphereVM object. If not defined, it will fall back to `{{ .machine.name }}`. The templating has the following data available: • `.machine.name`: The name of the Machine object. The templating also has the following funcs available: • `trimSuffix`: same as strings.TrimSuffix • `trunc`: truncates a string, e.g. `trunc 2 "hello"` or `trunc -2 "hello"` Notes: • While the template offers some flexibility, we would like the name to link to the Machine name to ensure better user experience when troubleshooting • Generated names must be valid Kubernetes names as they are used to create a VSphereVM object and usually also as the name of the Node object. • Names are automatically truncated at 63 characters. Please note that this can lead to name conflicts, so we highly recommend to use a template which leads to a name shorter than 63 characters.

.spec.template.spec.network

Description

Network is the network configuration for this machine's VM.

Type

`object`

Required

`devices`

Property	Type	Description
`devices`	`array`	Devices is the list of network devices used by the virtual machine.
`preferredAPIServerCidr`	`string`	PreferredAPIServerCIDR is the preferred CIDR for the Kubernetes API server endpoint on this machine Deprecated: This field is going to be removed in a future release.
`routes`	`array`	Routes is a list of optional, static routes applied to the virtual machine.

.spec.template.spec.network.devices

Description

Devices is the list of network devices used by the virtual machine.

Type

`array`

.spec.template.spec.network.devices[]

Description

NetworkDeviceSpec defines the network configuration for a virtual machine's network device.

Type

`object`

Required

`networkName`

Property	Type	Description
`addressesFromPools`	`array`	AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.
`deviceName`	`string`	DeviceName may be used to explicitly assign a name to the network device as it exists in the guest operating system.
`dhcp4`	`boolean`	DHCP4 is a flag that indicates whether or not to use DHCP for IPv4 on this device. If true then IPAddrs should not contain any IPv4 addresses.

Property	Type	Description
`dhcp4Overrides`	`object`	DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides ↗)
`dhcp6`	`boolean`	DHCP6 is a flag that indicates whether or not to use DHCP for IPv6 on this device. If true then IPAddrs should not contain any IPv6 addresses.
`dhcp6Overrides`	`object`	DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides ↗)
`gateway4`	`string`	Gateway4 is the IPv4 gateway used by this device. Required when DHCP4 is false.
`gateway6`	`string`	Gateway6 is the IPv6 gateway used by this device. Required when DHCP6 is false.

Property	Type	Description
`ipAddrs`	`array`	IPAddrs is a list of one or more IPv4 and/or IPv6 addresses to assign to this device. IP addresses must also specify the segment length in CIDR notation. Required when DHCP4, DHCP6 and SkipIPAllocation are false.
`macAddr`	`string`	MACAddr is the MAC address used by this device. It is generally a good idea to omit this field and allow a MAC address to be generated. Please note that this value must use the VMware OUI to work with the in-tree vSphere cloud provider.
`mtu`	`integer`	MTU is the device's Maximum Transmission Unit size in bytes.
`nameservers`	`array`	Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf↗).
`networkName`	`string`	NetworkName is the name, managed object reference or the managed object ID of the vSphere network to which the device will be connected.
`routes`	`array`	Routes is a list of optional, static routes applied to the device.

Property	Type	Description
`searchDomains`	`array`	SearchDomains is a list of search domains used when resolving IP addresses with DNS.
`skipIPAllocation`	`boolean`	SkipIPAllocation allows the device to not have IP address or DHCP configured. This is suitable for devices for which IP allocation is handled externally, eg. using Multus CNI. If true, CAPV will not verify IP address allocation.

.spec.template.spec.network.devices[].addressesFromPools

Description

AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.

Type

`array`

.spec.template.spec.network.devices[].addressesFromPools[]

Description

TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.template.spec.network.devices[].dhcp4Overrides

Description

DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.

Property	Type	Description
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take

Property	Type	Description
		precedence.
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.template.spec.network.devices[].dhcp6Overrides

Description

DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.

Property	Type	Description
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take precedence.
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.template.spec.network.devices[].ipAddrs

Description

IPAddrs is a list of one or more IPv4 and/or IPv6 addresses to assign to this device. IP addresses must also specify the segment length in CIDR notation. Required when DHCP4, DHCP6 and SkipIPAllocation are false.

Type

`array`

.spec.template.spec.network.devices[].ipAddrs[]

Type

`string`

.spec.template.spec.network.devices[].nameservers

Description

Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf).

Type

`array`

.spec.template.spec.network.devices[].nameservers[]

Type

`string`

.spec.template.spec.network.devices[].routes

Description

Routes is a list of optional, static routes applied to the device.

Type

`array`

.spec.template.spec.network.devices[].routes[]

Description

NetworkRouteSpec defines a static network route.

Type

`object`

Required

`metric` `to` `via`

Property	Type	Description
`metric`	`integer`	Metric is the weight/priority of the route.
`to`	`string`	To is an IPv4 or IPv6 address.
`via`	`string`	Via is an IPv4 or IPv6 address.

.spec.template.spec.network.devices[].searchDomains

Description

SearchDomains is a list of search domains used when resolving IP addresses with DNS.

Type

`array`

.spec.template.spec.network.devices[].searchDomains[]

Type

`string`

.spec.template.spec.network.routes

Description

Routes is a list of optional, static routes applied to the virtual machine.

Type

`array`

.spec.template.spec.network.routes[]

Description

NetworkRouteSpec defines a static network route.

Type

`object`

Required

`metric` `to` `via`

Property	Type	Description
`metric`	`integer`	Metric is the weight/priority of the route.
`to`	`string`	To is an IPv4 or IPv6 address.
`via`	`string`	Via is an IPv4 or IPv6 address.

.spec.template.spec.pciDevices

Description

PciDevices is the list of pci devices used by the virtual machine.

Type

`array`

.spec.template.spec.pciDevices[]

Description

PCIDeviceSpec defines virtual machine's PCI configuration.

Type

`object`

Property	Type	Description
`customLabel`	`string`	CustomLabel is the hardware label of a virtual machine's PCI device. Defaults to the eponymous property value in the template from which the virtual machine is cloned.
`deviceId`	`integer`	DeviceID is the device ID of a virtual machine's PCI, in integer. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with VGPUProfile as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.
`vGPUProfile`	`string`	VGPUProfile is the profile name of a virtual machine's vGPU, in string. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with DeviceID and VendorID as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.

Property	Type	Description
`vendorId`	`integer`	VendorId is the vendor ID of a virtual machine's PCI, in integer. Defaults to the eponymous property value in the template from which the virtual machine is cloned. Mutually exclusive with VGPUProfile as VGPUProfile and DeviceID + VendorID are two independent ways to define PCI devices.

.spec.template.spec.tagIDs

Description

TagIDs is an optional set of tags to add to an instance. Specified tagIDs must use URN-notation instead of display names.

Type

`array`

.spec.template.spec.tagIDs[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachinetemplates`
  - `DELETE`: delete collection of VSphereMachineTemplate
  - `GET`: list objects of kind VSphereMachineTemplate
  - `POST`: create a new VSphereMachineTemplate

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachinetemplates/{name}`
  - `DELETE`: delete the specified VSphereMachineTemplate
  - `GET`: read the specified VSphereMachineTemplate
  - `PATCH`: partially update the specified VSphereMachineTemplate
  - `PUT`: replace the specified VSphereMachineTemplate

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachinetemplates

HTTP method

`DELETE`

Description

delete collection of VSphereMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineTemplateList` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineTemplate` schema
201 - Created	`VSphereMachineTemplate` schema
202 - Accepted	`VSphereMachineTemplate` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachinetemplates/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineTemplate` schema
201 - Created	`VSphereMachineTemplate` schema
401 - Unauthorized	Empty

VSphereMachineConfigPool [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereMachineConfigPool is the Schema for the vspheremachineconfigpools API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereMachineConfigPoolSpec defines the desired state of VSphereMachineConfigPool.
`status`	`object`	VSphereMachineConfigPoolStatus defines the observed state of VSphereMachineConfigPool.

.spec

Description

VSphereMachineConfigPoolSpec defines the desired state of VSphereMachineConfigPool.

Type

`object`

Required

`clusterRef` `configs`

Property	Type	Description
`clusterRef`	`object`	ClusterRef references the CAPI Cluster (in the same namespace) whose VSphereCluster provides vCenter server, thumbprint, and credential chain (IdentityRef) for this pool's vCenter operations (e.g.

Property	Type	Description
		disk reclaim). Required. Can only be changed when consumerRef is nil. The pool will not reconcile until the referenced Cluster and its VSphereCluster infrastructure are available.
`configs`	`array`	Configs is the list of pre-defined machine configuration slots.
`datacenter`	`string`	Datacenter is the default vSphere datacenter for slots in this pool. It is used when a slot does not define its own Datacenter.
`releaseDelayHours`	`integer`	ReleaseDelayHours is the time to wait before marking a released slot as "Available" for any machine. During this period, the slot can only be reused if specifically requested or via priority reuse. Default is 24.

.spec.clusterRef

Description

ClusterRef references the CAPI Cluster (in the same namespace) whose VSphereCluster provides vCenter server, thumbprint, and credential chain (IdentityRef) for this pool's vCenter operations (e.g. disk reclaim). Required. Can only be changed when consumerRef is nil. The pool will not reconcile until the referenced Cluster and its VSphereCluster infrastructure are available.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗

Property	Type	Description
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.configs

Description

Configs is the list of pre-defined machine configuration slots.

Type

`array`

.spec.configs[]

Description

MachineConfigSlot defines a single machine configuration slot in the pool.

Type

`object`

Required

`hostname` `network`

Property	Type	Description
`datacenter`	`string`	Datacenter is the vSphere datacenter for this slot. If set, it takes precedence over VSphereMachineConfigPool.spec.datacenter. If unset, the pool-level Datacenter acts as the default.
`ephemeralDisks`	`array`	EphemeralDisks are non-persistent data disks. Unlike PersistentDisks they are created fresh (as empty disks) whenever the slot's VM is (re)created and deleted together with the VM: they do not participate in detach preservation, reclaim, or slot-release gating. Their SCSI unit number is assigned by the controller at clone time and observed in status.ephemeralDiskStatuses; it is not user-declared.
`hostname`	`string`	Hostname is the unique identifier for this slot and will be assigned to the VM. It must also be a valid Kubernetes node name because CAPV uses it for kubeadm nodeRegistration.name and the kubelet serving certificate DNS SAN.
`network`	`object`	Network describes the primary and additional network configurations for this slot.
`persistentDisks`	`array`	PersistentDisks that survive VM deletion.

.spec.configs[].ephemeralDisks

Description

EphemeralDisks are non-persistent data disks. Unlike PersistentDisks they are created fresh (as empty disks) whenever the slot's VM is (re)created and deleted together with the VM: they do not participate in detach preservation, reclaim, or slot-release gating. Their SCSI unit number is assigned by the controller at clone time and observed in status.ephemeralDiskStatuses; it is not user-declared.

Type

`array`

.spec.configs[].ephemeralDisks[]

Description

EphemeralDisk defines a non-persistent data disk that is created empty with the slot's VM and deleted together with it. It mirrors the guest-facing fields of PersistentDisk (size, placement, mount, format) but omits all persistence semantics: there is no UnitNumber spec field (the SCSI unit is controller-assigned at clone time and observed in status.ephemeralDiskStatuses), no VolumePath/DiskUUID backfill (the backing vmdk is always newly created), and no WipeFilesystem (a fresh disk is always empty and formatted from scratch).

Type

`object`

Required

`name` `sizeGiB`

Property	Type	Description
`datastore`	`string`	Datastore is the vSphere datastore name.
`fsFormat`	`string`	FSFormat is the filesystem format (default "ext4").

Property	Type	Description
`mountOptions`	`array`	MountOptions for the filesystem mount.
`mountPath`	`string`	MountPath is the mount path inside the VM guest OS (e.g., "/var/lib/containerd").
`name`	`string`	Name is the disk name. It shares a namespace with the slot's persistent disks: a name must be unique across both lists within the slot.
`sizeGiB`	`integer`	SizeGiB is the disk size.
`storagePolicy`	`string`	StoragePolicy is the vSphere storage policy name.

.spec.configs[].ephemeralDisks[].mountOptions

Description

MountOptions for the filesystem mount.

Type

`array`

.spec.configs[].ephemeralDisks[].mountOptions[]

Type

`string`

.spec.configs[].network

Description

Network describes the primary and additional network configurations for this slot.

Type

`object`

Required

`primary`

Property	Type	Description
`additional`	`array`	Additional are the remaining network configurations attached after the primary device.
`primary`	`object`	Primary is the network configuration used for kubelet node IP registration.

.spec.configs[].network.additional

Description

Additional are the remaining network configurations attached after the primary device.

Type

`array`

.spec.configs[].network.additional[]

Description

NetworkConfig defines the network configuration for a slot device.

Type

`object`

Required

`networkName`

Property	Type	Description
`deviceName`	`string`	DeviceName explicitly assigns a guest OS interface name to the network device.
`dns`	`array`	DNS nameservers
`gateway`	`string`	
`ip`	`string`	IPv4 configuration
`ipv6`	`string`	IPv6 configuration
`ipv6Gateway`	`string`	
`networkName`	`string`	NetworkName is the name of the vSphere network (PortGroup or DVPortGroup).

.spec.configs[].network.additional[].dns

Description

DNS nameservers

Type

`array`

.spec.configs[].network.additional[].dns[]

Type

`string`

.spec.configs[].network.primary

Description

Primary is the network configuration used for kubelet node IP registration.

Type

`object`

Required

`networkName`

Property	Type	Description
`deviceName`	`string`	DeviceName explicitly assigns a guest OS interface name to the network device.
`dns`	`array`	DNS nameservers
`gateway`	`string`	
`ip`	`string`	IPv4 configuration
`ipv6`	`string`	IPv6 configuration
`ipv6Gateway`	`string`	

Property	Type	Description
`networkName`	`string`	NetworkName is the name of the vSphere network (PortGroup or DVPortGroup).

.spec.configs[].network.primary.dns

Description

DNS nameservers

Type

`array`

.spec.configs[].network.primary.dns[]

Type

`string`

.spec.configs[].persistentDisks

Description

PersistentDisks that survive VM deletion.

Type

`array`

.spec.configs[].persistentDisks[]

Description

PersistentDisk defines a disk that survives VM deletion.

Type

`object`

Required

`name` `sizeGiB`

Property	Type	Description
`datastore`	`string`	Datastore is the vSphere datastore name.
`diskUUID`	`string`	DiskUUID is backfilled by the controller.
`fsFormat`	`string`	FSFormat is the filesystem format (default "ext4").
`mountOptions`	`array`	MountOptions for the filesystem mount.
`mountPath`	`string`	MountPath is the mount path inside the VM guest OS (e.g., "/var/lib/etcd").
`name`	`string`	Name is the disk name.
`sizeGiB`	`integer`	SizeGiB is the disk size.
`storagePolicy`	`string`	StoragePolicy is the vSphere storage policy name.

Property	Type	Description
`unitNumber`	`integer`	UnitNumber is the SCSI unit number for the disk (0-15, excluding 7). This ensures consistent disk ordering across VM recreations.
`volumePath`	`string`	VolumePath is backfilled by the controller after the disk is created. For vSphere, this is usually the datastore path to the .vmdk file.
`wipeFilesystem`	`boolean`	WipeFilesystem controls whether to wipe the filesystem content when the slot is reused by a new VM. When true, the disk content is cleared on the first boot of a new VM (reboots and manual service restarts are not affected). Defaults to false (data is preserved across VM recreations).

.spec.configs[].persistentDisks[].mountOptions

Description

MountOptions for the filesystem mount.

Type

`array`

.spec.configs[].persistentDisks[].mountOptions[]

Type

`string`

.status

Description

VSphereMachineConfigPoolStatus defines the observed state of VSphereMachineConfigPool.

Type

`object`

Property	Type	Description
`allocated`	`integer`	Allocated is the number of slots currently bound to a VSphereMachine (slots in the InUse state).
`available`	`integer`	Available is the number of slots currently free for allocation (slots in the Available state).
`conditions`	`array`	Conditions defines current state of the machine config pool.
`configStatuses`	`array`	ConfigStatuses tracks the state of each slot.
`consumerRef`	`object`	ConsumerRef is the workload controller (KubeadmControlPlane or MachineDeployment) currently bound to this pool. Set automatically by the controller when a machine allocates a slot. Cleared when the pool becomes fully reusable.
`ephemeralDiskStatuses`	`array`	EphemeralDiskStatuses records the controller-observed SCSI unit number of each

Property	Type	Description
		non-persistent disk, keyed by (hostname, disk name). The unit is assigned at clone time and reused (hydrated back onto the in-memory slot) on the next reconcile so the guest disk table can address the disk. Unlike PersistentDiskStatuses this carries no VolumePath/DiskUUID/Phase: ephemeral disks are recreated with the VM and never reclaimed.
`persistentDiskStatuses`	`array`	PersistentDiskStatuses tracks the observed state of each persistent disk, keyed by (hostname, disk name). The controller backfills the vmdk path, disk UUID, and actually-assigned SCSI unit here instead of writing them back onto spec, and drives reclaim through the per-disk Phase.
`total`	`integer`	Total is the total number of configuration slots defined in the pool.
`v1beta2`	`object`	v1beta2 groups all the fields that will be added or modified in VSphereMachineConfigPool's status with the V1Beta2 version.

.status.conditions

Description

Conditions defines current state of the machine config pool.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.

Property	Type	Description
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.configStatuses

Description

ConfigStatuses tracks the state of each slot.

Type

`array`

.status.configStatuses[]

Description

MachineConfigSlotStatus tracks the state of a single slot.

Type

`object`

Required

`hostname` `state`

Property	Type	Description
`hostname`	`string`	Hostname matches the Hostname in the Spec.
`lastReleasedTime`	`string`	LastReleasedTime is the timestamp when the slot transitioned to Released.
`machineRef`	`object`	MachineRef is the reference to the Machine currently using this slot.
`reclaimStatus`	`object`	ReclaimStatus tracks asynchronous reclaim progress for this slot.
`state`	`string`	State is the allocation state of the slot.

.status.configStatuses[].machineRef

Description

MachineRef is the reference to the Machine currently using this slot.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗

Property	Type	Description
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.status.configStatuses[].reclaimStatus

Description

ReclaimStatus tracks asynchronous reclaim progress for this slot.

Type

`object`

Property	Type	Description
`lastError`	`string`	LastError stores the latest reclaim task failure.
`retryAfter`	`string`	RetryAfter prevents tight retry loops after reclaim task failures.
`state`	`string`	State tracks the lifecycle of the reclaim task.

Property	Type	Description
`taskRef`	`string`	TaskRef tracks the in-flight vCenter reclaim task for this slot.
`volumePath`	`string`	VolumePath tracks the persistent disk currently being reclaimed.

.status.consumerRef

Description

ConsumerRef is the workload controller (KubeadmControlPlane or MachineDeployment) currently bound to this pool. Set automatically by the controller when a machine allocates a slot. Cleared when the pool becomes fully reusable.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2

Property	Type	Description
		in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.status.ephemeralDiskStatuses

Description

EphemeralDiskStatuses records the controller-observed SCSI unit number of each non-persistent disk, keyed by (hostname, disk name). The unit is assigned at clone time and reused (hydrated back onto the in-memory slot) on the next reconcile so the guest disk table can address the disk. Unlike PersistentDiskStatuses this carries no VolumePath/DiskUUID/Phase: ephemeral disks are recreated with the VM and never reclaimed.

Type

`array`

.status.ephemeralDiskStatuses[]

Description

EphemeralDiskStatus records the controller-observed SCSI unit number of a single non-persistent disk, keyed by (Hostname, Name). It deliberately omits VolumePath/DiskUUID/Phase: an ephemeral disk's backing vmdk is always newly created and deleted with the VM, so there is nothing to reclaim and no lifecycle to track. The recorded UnitNumber is hydrated back onto the in-memory slot each reconcile so the guest disk table can address the disk.

Type

`object`

Required

`hostname` `name`

Property	Type	Description
`hostname`	`string`	Hostname is the slot this disk belongs to (matches MachineConfigSlot.Hostname).
`name`	`string`	Name is the disk name (matches EphemeralDisk.Name within the slot).

Property	Type	Description
`unitNumber`	`integer`	UnitNumber is the SCSI unit number the disk was attached at, assigned by the controller at clone time and reused on the next VM recreation to keep guest disk addressing stable.

.status.persistentDiskStatuses

Description

PersistentDiskStatuses tracks the observed state of each persistent disk, keyed by (hostname, disk name). The controller backfills the vmdk path, disk UUID, and actually-assigned SCSI unit here instead of writing them back onto spec, and drives reclaim through the per-disk Phase.

Type

`array`

.status.persistentDiskStatuses[]

Description

PersistentDiskStatus tracks the observed state of a single persistent disk. It replaces the controller-backfilled VolumePath/DiskUUID on spec and the per-slot ReclaimStatus, keyed by (Hostname, Name).

Type

`object`

Required

`hostname` `name`

Property	Type	Description
`diskUUID`	`string`	DiskUUID is the disk UUID, backfilled after the disk is created. The guest uses it to identify and mount the disk.
`hostname`	`string`	Hostname is the slot this disk belongs to (matches MachineConfigSlot.Hostname).
`lastError`	`string`	LastError stores the latest reclaim failure, surfaced when Phase is Error.
`lastTransitionTime`	`string`	LastTransitionTime is the time Phase last changed.
`name`	`string`	Name is the disk name (matches PersistentDisk.Name within the slot).
`ownerMachineName`	`string`	OwnerMachineName is the name of the Machine currently owning the disk.
`ownerMachineUID`	`string`	OwnerMachineUID is the UID of the Machine currently owning the disk. It is preserved across VM delete/recreate so a rolling upgrade re-attaches the disk to the rebuilt machine.
`phase`	`string`	Phase is the observed lifecycle of the disk: Creating -> Attached while in use, Attached ->

Property	Type	Description
		Available when the VM is released, and Available -> Reclaiming -> Reclaimed as the backing is deleted (Error on a failed reclaim, retried after RetryAfter). Reclaimed is a terminal tombstone: the backing is gone but the record is kept so the release-1 migration does not re-seed the disk from spec's frozen VolumePath.
`retryAfter`	`string`	RetryAfter prevents tight retry loops after a failed reclaim task.
`taskRef`	`string`	TaskRef tracks the in-flight vCenter reclaim task for this disk. CAPV reclaims disks asynchronously, so the task moniker is polled across reconciles (DCS/HCS call their SDKs synchronously and have no equivalent).
`unitNumber`	`integer`	UnitNumber is the SCSI unit number the disk was actually attached at. When spec does not pin a UnitNumber, this observed value is reused on the next VM recreation to keep disk ordering stable.
`volumePath`	`string`	VolumePath is the datastore path to the backing .vmdk, backfilled after the disk is created.

.status.v1beta2

Description

v1beta2 groups all the fields that will be added or modified in VSphereMachineConfigPool's status with the V1Beta2 version.

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions represents the observations of a VSphereMachineConfigPool's current state. Known condition types are Ready, MembersValid, MembersUnique, SlotAvailable, PersistentDisksReady, ClusterRefReady and VCenterAvailable.

.status.v1beta2.conditions

Description

conditions represents the observations of a VSphereMachineConfigPool's current state. Known condition types are Ready, MembersValid, MembersUnique, SlotAvailable, PersistentDisksReady, ClusterRefReady and VCenterAvailable.

Type

`array`

.status.v1beta2.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.

Property	Type	Description
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachineconfigpools`
  - `DELETE` : delete collection of VSphereMachineConfigPool
  - `GET` : list objects of kind VSphereMachineConfigPool
  - `POST` : create a new VSphereMachineConfigPool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachineconfigpools/{name}`
  - `DELETE` : delete the specified VSphereMachineConfigPool
  - `GET` : read the specified VSphereMachineConfigPool
  - `PATCH` : partially update the specified VSphereMachineConfigPool
  - `PUT` : replace the specified VSphereMachineConfigPool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachineconfigpools/{name}/status`
  - `GET` : read status of the specified VSphereMachineConfigPool

- `PATCH`: partially update status of the specified VSphereMachineConfigPool
- `PUT`: replace status of the specified VSphereMachineConfigPool

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachineconfigpools

HTTP method

`DELETE`

Description

delete collection of VSphereMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPoolList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachineConfigPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPool` schema
201 - Created	`VSphereMachineConfigPool` schema
202 - Accepted	`VSphereMachineConfigPool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachineconfigpools/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachineConfigPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPool` schema
201 - Created	`VSphereMachineConfigPool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/vspheremachineconfigpools/{name}/status

HTTP method

`GET`

Description

read status of the specified VSphereMachineConfigPool

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VSphereMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VSphereMachineConfigPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The

Parameter	Type	Description
		request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereMachineConfigPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereMachineConfigPool` schema
201 - Created	`VSphereMachineConfigPool` schema
401 - Unauthorized	Empty

VSphereFailureDomain [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereFailureDomain is the Schema for the vspherefailuredomains API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereFailureDomainSpec defines the desired state of VSphereFailureDomain.

.spec

Description

VSphereFailureDomainSpec defines the desired state of VSphereFailureDomain.

Type

`object`

Required

`region` `topology` `zone`

Property	Type	Description
`region`	`object`	Region defines the name and type of a region
`topology`	`object`	Topology describes a given failure domain using vSphere constructs

Property	Type	Description
`zone`	`object`	Zone defines the name and type of a zone

.spec.region

Description

Region defines the name and type of a region

Type

`object`

Required

`name` `tagCategory` `type`

Property	Type	Description
`autoConfigure`	`boolean`	AutoConfigure tags the Type which is specified in the Topology Deprecated: This field is going to be removed in a future release.
`name`	`string`	Name is the name of the tag that represents this failure domain
`tagCategory`	`string`	TagCategory is the category used for the tag

Property	Type	Description
`type`	`string`	Type is the type of failure domain, the current values are "Datacenter", "ComputeCluster" and "HostGroup"

.spec.topology

Description

Topology describes a given failure domain using vSphere constructs

Type

`object`

Required

`datacenter`

Property	Type	Description
`computeCluster`	`string`	ComputeCluster as the failure domain
`datacenter`	`string`	Datacenter as the failure domain.
`datastore`	`string`	Datastore is the name or inventory path of the datastore in which the virtual machine is created/located.
`hosts`	`object`	Hosts has information required for placement of machines on VSphere hosts.

Property	Type	Description
`networkConfigurations`	`array`	NetworkConfigurations is a list of network configurations within this failure domain.
`networks`	`array`	Networks is the list of networks within this failure domain

.spec.topology.hosts

Description

Hosts has information required for placement of machines on VSphere hosts.

Type

`object`

Required

`hostGroupName` `vmGroupName`

Property	Type	Description
`hostGroupName`	`string`	HostGroupName is the name of the Host group
`vmGroupName`	`string`	VMGroupName is the name of the VM group

.spec.topology.networkConfigurations

Description

NetworkConfigurations is a list of network configurations within this failure domain.

Type

`array`

.spec.topology.networkConfigurations[]

Description

NetworkConfiguration defines a network configuration that should be used when consuming a failure domain.

Type

`object`

Required

`networkName`

Property	Type	Description
`addressesFromPools`	`array`	AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.
`dhcp4`	`boolean`	DHCP4 is a flag that indicates whether or not to use DHCP for IPv4.
`dhcp4Overrides`	`object`	DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides↗)

Property	Type	Description
`dhcp6`	`boolean`	DHCP6 is a flag that indicates whether or not to use DHCP for IPv6.
`dhcp6Overrides`	`object`	DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides↗)
`nameservers`	`array`	Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf↗).
`networkName`	`string`	NetworkName is the network name for this machine's VM.
`searchDomains`	`array`	SearchDomains is a list of search domains used when resolving IP addresses with DNS.

.spec.topology.networkConfigurations[].addressesFromPools

Description

AddressesFromPools is a list of IPAddressPools that should be assigned to IPAddressClaims. The machine's cloud-init metadata will be populated with IPAddresses fulfilled by an IPAM provider.

Type

`array`

.spec.topology.networkConfigurations[].addressesFromPools[]

Description

TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.topology.networkConfigurations[].dhcp4Overrides

Description

DHCP4Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.

Property	Type	Description
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take precedence.
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.topology.networkConfigurations[].dhcp6Overrides

Description

DHCP6Overrides allows for the control over several DHCP behaviors. Overrides will only be applied when the corresponding DHCP flag is set. Only configured values will be sent, omitted values will default to distribution defaults. Dependent on support in the network stack for your distribution. For more information see the netplan reference (https://netplan.io/reference#dhcp-overrides)

Type

`object`

Property	Type	Description
`hostname`	`string`	Hostname is the name which will be sent to the DHCP server instead of the machine's hostname.
`routeMetric`	`integer`	RouteMetric is used to prioritize routes for devices. A lower metric for an interface will have a higher priority.
`sendHostname`	`boolean`	SendHostname when `true`, the hostname of the machine will be sent to the DHCP server.
`useDNS`	`boolean`	UseDNS when `true`, the DNS servers in the DHCP server will be used and take precedence.
`useDomains`	`string`	UseDomains can take the values `true`, `false`, or `route`. When `true`, the domain name from the DHCP server will be used as the DNS search domain for this device. When `route`, the domain name from the DHCP response will be used for routing DNS only, not for searching.
`useHostname`	`boolean`	UseHostname when `true`, the hostname from the DHCP server will be set as the transient hostname of the machine.

Property	Type	Description
`useMTU`	`boolean`	UseMTU when `true`, the MTU from the DHCP server will be set as the MTU of the device.
`useNTP`	`boolean`	UseNTP when `true`, the NTP servers from the DHCP server will be used by systemd-timesyncd and take precedence.
`useRoutes`	`string`	UseRoutes can take the values `true`, `false`, or `route`. When `true`, the routes from the DHCP server will be installed in the routing table.

.spec.topology.networkConfigurations[].nameservers

Description

Nameservers is a list of IPv4 and/or IPv6 addresses used as DNS nameservers. Please note that Linux allows only three nameservers (https://linux.die.net/man/5/resolv.conf).

Type

`array`

.spec.topology.networkConfigurations[].nameservers[]

Type

`string`

.spec.topology.networkConfigurations[].searchDomains

Description

SearchDomains is a list of search domains used when resolving IP addresses with DNS.

Type

`array`

.spec.topology.networkConfigurations[].searchDomains[]

Type

`string`

.spec.topology.networks

Description

Networks is the list of networks within this failure domain

Type

`array`

.spec.topology.networks[]

Type

`string`

.spec.zone

Description

Zone defines the name and type of a zone

Type

`object`

Required

`name` `tagCategory` `type`

Property	Type	Description
`autoConfigure`	`boolean`	AutoConfigure tags the Type which is specified in the Topology Deprecated: This field is going to be removed in a future release.
`name`	`string`	Name is the name of the tag that represents this failure domain
`tagCategory`	`string`	TagCategory is the category used for the tag
`type`	`string`	Type is the type of failure domain, the current values are "Datacenter", "ComputeCluster" and "HostGroup"

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspherefailuredomains`
  - `DELETE` : delete collection of VSphereFailureDomain
  - `GET` : list objects of kind VSphereFailureDomain
  - `POST` : create a new VSphereFailureDomain
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspherefailuredomains/{name}`
  - `DELETE` : delete the specified VSphereFailureDomain
  - `GET` : read the specified VSphereFailureDomain

- `PATCH`: partially update the specified VSphereFailureDomain
- `PUT`: replace the specified VSphereFailureDomain

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspherefailuredomains

HTTP method

`DELETE`

Description

delete collection of VSphereFailureDomain

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereFailureDomain

HTTP responses

HTTP code	Response body
200 - OK	`VSphereFailureDomainList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereFailureDomain

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereFailureDomain` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereFailureDomain` schema

HTTP code	Response body
201 - Created	`VSphereFailureDomain` schema
202 - Accepted	`VSphereFailureDomain` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspherefailuredomains/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereFailureDomain

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereFailureDomain

HTTP responses

HTTP code	Response body
200 - OK	`VSphereFailureDomain` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereFailureDomain

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereFailureDomain` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereFailureDomain

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The

Parameter	Type	Description
		request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereFailureDomain` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereFailureDomain` schema
201 - Created	`VSphereFailureDomain` schema
401 - Unauthorized	Empty

VSphereDeploymentZone [infrastructure.cluster.x-k8s.io/v1beta1]

Description

VSphereDeploymentZone is the Schema for the vspheredeploymentzones API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VSphereDeploymentZoneSpec defines the desired state of VSphereDeploymentZone.
`status`	`object`	VSphereDeploymentZoneStatus contains the status for a VSphereDeploymentZone.

.spec

Description

VSphereDeploymentZoneSpec defines the desired state of VSphereDeploymentZone.

Type

`object`

Required

`placementConstraint`

Property	Type	Description
`controlPlane`	`boolean`	ControlPlane determines if this failure domain is suitable for use by control plane machines.

Property	Type	Description
`failureDomain`	`string`	FailureDomain is the name of the VSphereFailureDomain used for this VSphereDeploymentZone
`placementConstraint`	`object`	PlacementConstraint encapsulates the placement constraints used within this deployment zone.
`server`	`string`	Server is the address of the vSphere endpoint.

.spec.placementConstraint

Description

PlacementConstraint encapsulates the placement constraints used within this deployment zone.

Type

`object`

Property	Type	Description
`folder`	`string`	Folder is the name or inventory path of the folder in which the virtual machine is created/located.
`resourcePool`	`string`	ResourcePool is the name or inventory path of the resource pool in which the virtual machine is created/located.

.status

Description

VSphereDeploymentZoneStatus contains the status for a VSphereDeploymentZone.

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions defines current service state of the VSphereDeploymentZone.
`ready`	`boolean`	Ready is true when the VSphereDeploymentZone resource is ready. If set to false, it will be ignored by VSphereClusters
`v1beta2`	`object`	v1beta2 groups all the fields that will be added or modified in VSphereDeploymentZone's status with the V1Beta2 version.

.status.conditions

Description

Conditions defines current service state of the VSphereDeploymentZone.

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
`severity`	`string`	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.

Property	Type	Description
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.v1beta2

Description

v1beta2 groups all the fields that will be added or modified in VSphereDeploymentZone's status with the V1Beta2 version.

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions represents the observations of a VSphereDeploymentZone's current state. Known condition types are Ready, VCenterAvailable, PlacementConstraintReady, FailureDomainValidated and Paused.

.status.v1beta2.conditions

Description

conditions represents the observations of a VSphereDeploymentZone's current state. Known condition types are Ready, VCenterAvailable, PlacementConstraintReady, FailureDomainValidated and Paused.

Type

`array`

.status.v1beta2.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation

Property	Type	Description
		is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspheredeploymentzones`
  - `DELETE`: delete collection of VSphereDeploymentZone
  - `GET`: list objects of kind VSphereDeploymentZone
  - `POST`: create a new VSphereDeploymentZone

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspheredeploymentzones/{name}`
  - `DELETE`: delete the specified VSphereDeploymentZone
  - `GET`: read the specified VSphereDeploymentZone
  - `PATCH`: partially update the specified VSphereDeploymentZone
  - `PUT`: replace the specified VSphereDeploymentZone
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspheredeploymentzones/{name}/status`
  - `GET`: read status of the specified VSphereDeploymentZone
  - `PATCH`: partially update status of the specified VSphereDeploymentZone
  - `PUT`: replace status of the specified VSphereDeploymentZone

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspheredeploymentzones

HTTP method

`DELETE`

Description

delete collection of VSphereDeploymentZone

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VSphereDeploymentZone

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZoneList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VSphereDeploymentZone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereDeploymentZone` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZone` schema
201 - Created	`VSphereDeploymentZone` schema
202 - Accepted	`VSphereDeploymentZone` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspheredeploymentzones/{name}

HTTP method

`DELETE`

Description

delete the specified VSphereDeploymentZone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VSphereDeploymentZone

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZone` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VSphereDeploymentZone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZone` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VSphereDeploymentZone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereDeploymentZone` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZone` schema
201 - Created	`VSphereDeploymentZone` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/vspheredeploymentzones/{name}/status

HTTP method

`GET`

Description

read status of the specified VSphereDeploymentZone

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZone` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VSphereDeploymentZone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZone` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VSphereDeploymentZone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VSphereDeploymentZone` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VSphereDeploymentZone` schema
201 - Created	`VSphereDeploymentZone` schema
401 - Unauthorized	Empty

Bare Metal Provider APIs

BaremetalCluster [infrastructur

BaremetalMachine [infrastructu

BaremetalM

MachineInventoryPool [infrastr

MachineRegistration [elementa

SeedImage

MachineInventory [elemental.cattle.io/v1beta1]

BaremetalCluster [infrastructure.cluster.x-k8s.io/v1beta1]

Description

BaremetalCluster is the Schema for the baremetalclusters API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	BaremetalClusterSpec describes the cluster-level infrastructure configuration owned by the baremetal provider. See design-doc/api/cr-reference.md §3.
`status`	`object`	BaremetalClusterStatus aggregates infra-level readiness for CAPI.

.spec

Description

BaremetalClusterSpec describes the cluster-level infrastructure configuration owned by the baremetal provider. See design-doc/api/cr-reference.md §3.

Type

`object`

Property	Type	Description
`controlPlaneEndpoint`	`object`	ControlPlaneEndpoint is the apiserver endpoint exposed to the workload plane. CAPI contract: once set, must not change.

Property	Type	Description
`controlPlaneLoadBalancer`	`object`	ControlPlaneLoadBalancer describes the VIP / LB fronting the control plane. When set, the reconciler backfills ControlPlaneEndpoint from this value.
`networkType`	`string`	NetworkType identifies the CNI implementation the provider should expect. Phase 1 treats this as metadata only.

.spec.controlPlaneEndpoint

Description

ControlPlaneEndpoint is the apiserver endpoint exposed to the workload plane. CAPI contract: once set, must not change.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	
`port`	`integer`	

.spec.controlPlaneLoadBalancer

Description

ControlPlaneLoadBalancer describes the VIP / LB fronting the control plane. When set, the reconciler backfills ControlPlaneEndpoint from this value.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	
`port`	`integer`	
`type`	`string`	
`vrid`	`integer`	VRID is the keepalived/vrrp identifier used by the internal load balancer.

.status

Description

BaremetalClusterStatus aggregates infra-level readiness for CAPI.

Type

`object`

Property	Type	Description
`conditions`	`array`	
`failureDomains`	`object`	FailureDomains is reserved for future Phase 3 work.

Property	Type	Description
`ready`	`boolean`	Ready follows the CAPI infrastructure contract (required).

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.

Property	Type	Description
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.failureDomains

Description

FailureDomains is reserved for future Phase 3 work.

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalclusters`
  - `DELETE`: delete collection of BaremetalCluster
  - `GET`: list objects of kind BaremetalCluster
  - `POST`: create a new BaremetalCluster
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalclusters/{name}`
  - `DELETE`: delete the specified BaremetalCluster
  - `GET`: read the specified BaremetalCluster
  - `PATCH`: partially update the specified BaremetalCluster
  - `PUT`: replace the specified BaremetalCluster
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalclusters/{name}/status`
  - `GET`: read status of the specified BaremetalCluster
  - `PATCH`: partially update status of the specified BaremetalCluster
  - `PUT`: replace status of the specified BaremetalCluster

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalcluste

rs

HTTP method

`DELETE`

Description

delete collection of BaremetalCluster

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind BaremetalCluster

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalClusterList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new BaremetalCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalCluster` schema
201 - Created	`BaremetalCluster` schema

HTTP code	Response body
202 - Accepted	`BaremetalCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalclusters/{name}

HTTP method

`DELETE`

Description

delete the specified BaremetalCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified BaremetalCluster

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified BaremetalCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified BaremetalCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The

Parameter	Type	Description
		request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalCluster` schema
201 - Created	`BaremetalCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalclusters/{name}/status

HTTP method

`GET`

Description

read status of the specified BaremetalCluster

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified BaremetalCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified BaremetalCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalCluster` schema
201 - Created	`BaremetalCluster` schema
401 - Unauthorized	Empty

BaremetalMachine [infrastructure.cluster.x-k8s.io/v1beta1]

Description

BaremetalMachine is the Schema for the baremetalmachines API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	BaremetalMachineSpec follows design-doc/api/resource-model.md §8. Role / version / upgrade image deliberately absent: role is inferred from the owning KubeadmControlPlane / MachineDeployment, version comes from the owner Machine, and the elemental upgrade image is resolved at reprovision time via the provider image catalog.
`status`	`object`	BaremetalMachineStatus tracks provider-internal allocation and the reprovision pipeline.

.spec

Description

BaremetalMachineSpec follows design-doc/api/resource-model.md §8. Role / version / upgrade image deliberately absent: role is inferred from the owning KubeadmControlPlane / MachineDeployment, version comes from the owner Machine, and the elemental upgrade image is resolved at reprovision time via the provider image catalog.

Type

`object`

Required

`machineInventoryPoolRef`

Property	Type	Description
`machineInventoryPoolRef`	`object`	MachineInventoryPoolRef pins which pool this machine draws from. The field is immutable once set (see webhook).
`providerID`	`string`	ProviderID is the CAPI contract field populated once the node is bound to a MachineInventory. Format: baremetal:///

.spec.machineInventoryPoolRef

Description

MachineInventoryPoolRef pins which pool this machine draws from. The field is immutable once set (see webhook).

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	

.status

Description

BaremetalMachineStatus tracks provider-internal allocation and the reprovision pipeline.

Type

`object`

Property	Type	Description
`addresses`	`array`	Addresses surfaces addresses reported by the MachineInventory (Phase 1: unused).
`conditions`	`array`	
`failureMessage`	`string`	FailureMessage is a CAPI-contract human-readable message.
`failureReason`	`string`	FailureReason is a CAPI-contract terse reason for a terminal failure.
`lastAppliedBootstrapHash`	`string`	LastAppliedBootstrapHash tracks the sha256 of the bootstrap user-data that was last rendered into a reprovision plan. It prevents redundant patches.
`lastAppliedPlanChecksum`	`string`	LastAppliedPlanChecksum tracks the sha256 of the last plan body we wrote, used to detect Applied/Failed transitions from MachineInventory.status.plan.
`machineInventoryRef`	`object`	MachineInventoryRef is the inventory selected from the pool.

Property	Type	Description
`phase`	`string`	Phase is the coarse state; see BaremetalMachinePhase constants.
`planSecretRef`	`object`	PlanSecretRef is the elemental plan secret currently driven by the provider.
`ready`	`boolean`	Ready indicates CAPI-contract readiness (node joined and healthy).

.status.addresses

Description

Addresses surfaces addresses reported by the MachineInventory (Phase 1: unused).

Type

`array`

.status.addresses[]

Description

NodeAddress contains information for the node's address.

Type

`object`

Required

`address` `type`

Property	Type	Description
`address`	`string`	The node address.
`type`	`string`	Node address type, one of Hostname, ExternalIP or InternalIP.

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.

Property	Type	Description
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.machineInventoryRef

Description

MachineInventoryRef is the inventory selected from the pool.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	

.status.planSecretRef

Description

PlanSecretRef is the elemental plan secret currently driven by the provider.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachines`
  - `DELETE`: delete collection of BaremetalMachine

- `GET` : list objects of kind BaremetalMachine
- `POST` : create a new BaremetalMachine

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachines/{name}`
  - `DELETE` : delete the specified BaremetalMachine
  - `GET` : read the specified BaremetalMachine
  - `PATCH` : partially update the specified BaremetalMachine
  - `PUT` : replace the specified BaremetalMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachines/{name}/status`
  - `GET` : read status of the specified BaremetalMachine
  - `PATCH` : partially update status of the specified BaremetalMachine
  - `PUT` : replace status of the specified BaremetalMachine

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachines

HTTP method

`DELETE`

Description

delete collection of BaremetalMachine

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind BaremetalMachine

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachineList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new BaremetalMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachine` schema
201 - Created	`BaremetalMachine` schema
202 - Accepted	`BaremetalMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachines/{name}

HTTP method

`DELETE`

Description

delete the specified BaremetalMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified BaremetalMachine

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified BaremetalMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified BaremetalMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachine` schema

HTTP code	Response body
201 - Created	`BaremetalMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachines/{name}/status

HTTP method

`GET`

Description

read status of the specified BaremetalMachine

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified BaremetalMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified BaremetalMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachine` schema
201 - Created	`BaremetalMachine` schema

HTTP code	Response body
401 - Unauthorized	Empty

BaremetalMachineTemplate [infrastructure.cluster.x-k8s.io/v1beta1]

Description

BaremetalMachineTemplate lets KubeadmControlPlane and MachineDeployment bind a pool.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In

Property	Type	Description
		CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	BaremetalMachineTemplateSpec is the template for BaremetalMachine.

.spec

Description

BaremetalMachineTemplateSpec is the template for BaremetalMachine.

Type

`object`

Required

`template`

Property	Type	Description
`template`	`object`	BaremetalMachineTemplateResource is the templated spec embedded in a template.

.spec.template

Description

BaremetalMachineTemplateResource is the templated spec embedded in a template.

Type

`object`

Required

`spec`

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create. This is a copy of customizable fields from metav1.ObjectMeta. ObjectMeta is embedded in `Machine.Spec`, `MachineDeployment.Template` and `MachineSet.Template`, which are not top-level Kubernetes objects. Given that metav1.ObjectMeta has lots of special cases and read-only fields which end up in the generated CRD validation, having it as a subset simplifies the API and some issues that can impact user experience. During the upgrade to controller-tools@v2 ↗ for v1alpha2, we noticed a failure would occur running Cluster API test suite against the new CRDs, specifically `spec.metadata.creationTimestamp in body must be of type string: "null"`. The investigation showed that `controller-tools@v2` behaves differently than its previous version when handling types from metav1 package. In more details, we found that embedded (non-top level) types that embedded `metav1.ObjectMeta` had validation properties, including for `creationTimestamp` (metav1.Time). The `metav1.Time` type specifies a custom json marshaller that, when IsZero() is true, returns

Property	Type	Description
		`null` which breaks validation because the field isn't marked as nullable. In future versions, controller-tools@v2 might allow overriding the type and validation for embedded types. When that happens, this hack should be revisited.
`spec`	`object`	BaremetalMachineSpec follows design-doc/api/resource-model.md §8. Role / version / upgrade image deliberately absent: role is inferred from the owning KubeadmControlPlane / MachineDeployment, version comes from the owner Machine, and the elemental upgrade image is resolved at reprovision time via the provider image catalog.

.spec.template.spec

Description

BaremetalMachineSpec follows design-doc/api/resource-model.md §8. Role / version / upgrade image deliberately absent: role is inferred from the owning KubeadmControlPlane / MachineDeployment, version comes from the owner Machine, and the elemental upgrade image is resolved at reprovision time via the provider image catalog.

Type

`object`

Required

`machineInventoryPoolRef`

Property	Type	Description
`machineInventoryPoolRef`	`object`	MachineInventoryPoolRef pins which pool this machine draws from. The field is immutable once set (see webhook).
`providerID`	`string`	ProviderID is the CAPI contract field populated once the node is bound to a MachineInventory. Format: baremetal:///

.spec.template.spec.machineInventoryPoolRef

Description

MachineInventoryPoolRef pins which pool this machine draws from. The field is immutable once set (see webhook).

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachinetemplates`

- `DELETE`: delete collection of BaremetalMachineTemplate
- `GET`: list objects of kind BaremetalMachineTemplate
- `POST`: create a new BaremetalMachineTemplate

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachinetemplates/{name}`
  - `DELETE`: delete the specified BaremetalMachineTemplate
  - `GET`: read the specified BaremetalMachineTemplate
  - `PATCH`: partially update the specified BaremetalMachineTemplate
  - `PUT`: replace the specified BaremetalMachineTemplate

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachinetemplates

HTTP method

`DELETE`

Description

delete collection of BaremetalMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind BaremetalMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachineTemplateList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new BaremetalMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachineTemplate` schema
201 - Created	`BaremetalMachineTemplate` schema
202 - Accepted	`BaremetalMachineTemplate` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/baremetalmachinetemplates/{name}

HTTP method

`DELETE`

Description

delete the specified BaremetalMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified BaremetalMachineTemplate

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified BaremetalMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachineTemplate` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified BaremetalMachineTemplate

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BaremetalMachineTemplate` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BaremetalMachineTemplate` schema
201 - Created	`BaremetalMachineTemplate` schema
401 - Unauthorized	Empty

MachineInventoryPool [infrastructure.cluster.x-k8s.io/v1beta1]

Description

MachineInventoryPool is the Schema for the machineinventorypools API.

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	MachineInventoryPoolSpec declares which MachineInventory objects may back a particular KubeadmControlPlane / MachineDeployment. See design.md §“MachineInventoryPool”.
`status`	`object`	MachineInventoryPoolStatus summarises pool capacity from the provider's allocation annotations plus the upstream MachineInventory.status.plan.

.spec

Description

MachineInventoryPoolSpec declares which MachineInventory objects may back a particular KubeadmControlPlane / MachineDeployment. See design.md §“MachineInventoryPool”.

Type

`object`

Required

`clusterName` `machineInventories`

Property	Type	Description
`clusterName`	`string`	ClusterName binds this pool to a specific Cluster. A MachineInventory referenced here must not belong to another active pool.
`machineInventories`	`array`	MachineInventories enumerates the MachineInventory names allowed to back BaremetalMachines that reference this pool.

.spec.machineInventories

Description

MachineInventories enumerates the MachineInventory names allowed to back BaremetalMachines that reference this pool.

Type

`array`

.spec.machineInventories[]

Description

PoolInventoryRef references a MachineInventory and optionally overrides the hostname used when reprovisioning it.

Type

`object`

Required

`name`

Property	Type	Description
`hostname`	`string`	Hostname is used as the node hostname when this inventory is selected. When empty, the MachineInventory name is used.
`name`	`string`	

.status

Description

MachineInventoryPoolStatus summarises pool capacity from the provider's allocation annotations plus the upstream MachineInventory.status.plan.

Type

`object`

Property	Type	Description
`allocated`	`integer`	
`available`	`integer`	
`conditions`	`array`	
`preparing`	`integer`	
`reprovisioning`	`integer`	
`total`	`integer`	
`unavailable`	`integer`	

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.

Property	Type	Description
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/machineinventorypools`
  - `DELETE`: delete collection of MachineInventoryPool
  - `GET`: list objects of kind MachineInventoryPool
  - `POST`: create a new MachineInventoryPool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/machineinventorypools/{name}`
  - `DELETE`: delete the specified MachineInventoryPool

- `GET` : read the specified MachineInventoryPool
- `PATCH` : partially update the specified MachineInventoryPool
- `PUT` : replace the specified MachineInventoryPool

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/machineinventorypools/{name}/status`
  - `GET` : read status of the specified MachineInventoryPool
  - `PATCH` : partially update status of the specified MachineInventoryPool
  - `PUT` : replace status of the specified MachineInventoryPool

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/machineinventorypools

HTTP method

`DELETE`

Description

delete collection of MachineInventoryPool

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind MachineInventoryPool

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPoolList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new MachineInventoryPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineInventoryPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPool` schema
201 - Created	`MachineInventoryPool` schema
202 - Accepted	`MachineInventoryPool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/machineinventorypools/{name}

HTTP method

`DELETE`

Description

delete the specified MachineInventoryPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified MachineInventoryPool

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified MachineInventoryPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified MachineInventoryPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineInventoryPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPool` schema
201 - Created	`MachineInventoryPool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/machineinventorypools/{name}/status

HTTP method

`GET`

Description

read status of the specified MachineInventoryPool

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified MachineInventoryPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the

Parameter	Type	Description
		default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified MachineInventoryPool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineInventoryPool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryPool` schema
201 - Created	`MachineInventoryPool` schema
401 - Unauthorized	Empty

MachineRegistration [elemental.cattle.io/v1beta1]

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	
`status`	`object`	

.spec

Type

`object`

Property	Type	Description
`config`	`object`	Config the cloud config that will be used to provision the node.
`machineInventoryAnnotations`	`object`	MachineInventoryAnnotations annotations to be added to the created MachineInventory object.

Property	Type	Description
`machineInventoryLabels`	`object`	MachineInventoryLabels label to be added to the created MachineInventory object.
`machineName`	`string`	

.spec.config

Description

Config the cloud config that will be used to provision the node.

Type

`object`

Property	Type	Description
`cloud-config`		
`elemental`	`object`	
`network`	`object`	NetworkTemplate contains a map of IPAddressPools and a schemaless network config template.

.spec.config.elemental

Type

`object`

Property	Type	Description
`install`	`object`	
`registration`	`object`	
`reset`	`object`	
`system-agent`	`object`	

.spec.config.elemental.install

Type

`object`

Property	Type	Description
`config-dir`	`string`	
`config-urls`	`array`	
`debug`	`boolean`	
`device`	`string`	
`device-selector`	`array`	
`disable-boot-entry`	`boolean`	
`eject-cd`	`boolean`	
`firmware`	`string`	
`iso`	`string`	
`no-format`	`boolean`	
`poweroff`	`boolean`	

Property	Type	Description
`reboot`	`boolean`	
`snapshotter`	`object`	
`system-uri`	`string`	
`tty`	`string`	

.spec.config.elemental.install.config-urls

Type

`array`

.spec.config.elemental.install.config-urls[]

Type

`string`

.spec.config.elemental.install.device-selector

Type

`array`

.spec.config.elemental.install.device-selector[]

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	
`operator`	`string`	
`values`	`array`	

.spec.config.elemental.install.device-selector[].values

Type

`array`

.spec.config.elemental.install.device-selector[].values[]

Type

`string`

.spec.config.elemental.install.snapshotter

Type

`object`

Property	Type	Description
`maxSnaps`	`integer`	MaxSnaps sets the maximum amount of snapshots to keep
`type`	`string`	Type sets the snapshotter type for a new installation, available options are 'loopdevice' and 'btrfs'

.spec.config.elemental.registration

Type

`object`

Property	Type	Description
`auth`	`string`	
`ca-cert`	`string`	
`emulate-tpm`	`boolean`	
`emulated-tpm-seed`	`integer`	
`no-smbios`	`boolean`	
`no-toolkit`	`boolean`	
`url`	`string`	

.spec.config.elemental.reset

Type

`object`

Property	Type	Description
`config-urls`	`array`	
`debug`	`boolean`	
`disable-boot-entry`	`boolean`	
`enabled`	`boolean`	
`poweroff`	`boolean`	

Property	Type	Description
`reboot`	`boolean`	
`reset-oem`	`boolean`	
`reset-persistent`	`boolean`	
`system-uri`	`string`	

.spec.config.elemental.reset.config-urls

Type

`array`

.spec.config.elemental.reset.config-urls[]

Type

`string`

.spec.config.elemental.system-agent

Type

`object`

Property	Type	Description
`secret-name`	`string`	
`secret-namespace`	`string`	
`strictTLSMode`	`boolean`	
`token`	`string`	
`url`	`string`	

.spec.config.network

Description

NetworkTemplate contains a map of IPAddressPools and a schemaless network config template.

Type

`object`

Property	Type	Description
`config`		Config contains the network config template (nmc, nmstate, or nmconnections formats)
`configurator`	`string`	Configurator
`ipAddresses`	`object`	IPAddresses contains a map of IPPools references

.spec.config.network.ipAddresses

Description

IPAddresses contains a map of IPPools references

Type

`object`

.spec.machineInventoryAnnotations

Description

MachineInventoryAnnotations annotations to be added to the created MachineInventory object.

Type

`object`

.spec.machineInventoryLabels

Description

MachineInventoryLabels label to be added to the created MachineInventory object.

Type

`object`

.status

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions describe the state of the machine registration object.
`registrationToken`	`string`	RegistrationToken a token for registering a machine.
`registrationURL`	`string`	RegistrationURL is the URL for registering a new machine.
`serviceAccountRef`	`object`	ServiceAccountRef a reference to the service account created by the machine registration.

.status.conditions

Description

Conditions describe the state of the machine registration object.

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation

Property	Type	Description
		is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.serviceAccountRef

Description

ServiceAccountRef a reference to the service account created by the machine registration.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗

Property	Type	Description
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

API Endpoints

The following API endpoints are available:

- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineregistrations`
  - `DELETE`: delete collection of MachineRegistration
  - `GET`: list objects of kind MachineRegistration
  - `POST`: create a new MachineRegistration
- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineregistrations/{name}`
  - `DELETE`: delete the specified MachineRegistration
  - `GET`: read the specified MachineRegistration
  - `PATCH`: partially update the specified MachineRegistration
  - `PUT`: replace the specified MachineRegistration

- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineregistrations/{name}/status`
  - `GET`: read status of the specified MachineRegistration
  - `PATCH`: partially update status of the specified MachineRegistration
  - `PUT`: replace status of the specified MachineRegistration

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineregistrations

HTTP method

`DELETE`

Description

delete collection of MachineRegistration

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind MachineRegistration

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistrationList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new MachineRegistration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineRegistration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistration` schema
201 - Created	`MachineRegistration` schema
202 - Accepted	`MachineRegistration` schema
401 - Unauthorized	Empty

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineregistrations/{name}

HTTP method

`DELETE`

Description

delete the specified MachineRegistration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified MachineRegistration

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistration` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified MachineRegistration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified MachineRegistration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineRegistration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistration` schema
201 - Created	`MachineRegistration` schema
401 - Unauthorized	Empty

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineregistrations/{name}/status

HTTP method

`GET`

Description

read status of the specified MachineRegistration

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistration` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified MachineRegistration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified MachineRegistration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineRegistration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineRegistration` schema
201 - Created	`MachineRegistration` schema
401 - Unauthorized	Empty

SeedImage [elemental.cattle.io/v1beta1]

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	
`status`	`object`	

.spec

Type

`object`

Required

`registrationRef` `type`

Property	Type	Description
`baseImage`	`string`	BaseImg the base elemental image used to build the seed image.
`buildContainer`	`object`	BuildContainer settings for a custom container used to generate the downloadable image.
`cleanupAfterMinutes`	`integer`	LifetimeMinutes the time at which the built seed image will be cleaned up. If when the lifetime elapses the built image is being downloaded, the active download will be completed before removing the built image. Default is 60 minutes, set to 0 to disable.

Property	Type	Description
`cloud-config`		CloudConfig contains cloud-config data to be put in the generated iso.
`registrationRef`	`object`	MachineRegistrationRef a reference to the related MachineRegistration.
`retriggerBuild`	`boolean`	RetriggerBuild triggers to build again a cleaned up seed image.
`size`	`integer|string`	Size specifies the size of the volume used to store the image. Defaults to 6Gi
`targetPlatform`	`string`	Platform specifies the target platform for the built image. Example: linux/amd64
`type`	`string`	Type specifies the type of seed image to built. Valid values are iso|raw Defaults to "iso"

.spec.buildContainer

Description

BuildContainer settings for a custom container used to generate the downloadable image.

Type

`object`

Property	Type	Description
`args`	`array`	Args same as corev1.Container.Args
`command`	`array`	Command same as corev1.Container.Command
`image`	`string`	Image container image to run
`imagePullPolicy`	`string`	Args same as corev1.Container.ImagePullPolicy
`name`	`string`	Name of the spawned container

.spec.buildContainer.args

Description

Args same as corev1.Container.Args

Type

`array`

.spec.buildContainer.args[]

Type

`string`

.spec.buildContainer.command

Description

Command same as corev1.Container.Command

Type

`array`

.spec.buildContainer.command[]

Type

`string`

.spec.registrationRef

Description

MachineRegistrationRef a reference to the related MachineRegistration.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.

Property	Type	Description
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.status

Type

`object`

Property	Type	Description
`checksumURL`	`string`	ChecksumURL the URL from which the SeedImage checksum can be downloaded once the image is built.
`conditions`	`array`	Conditions describe the state of the seedimage object.
`downloadToken`	`string`	DownloadToken a token to identify the seed image to download.
`downloadURL`	`string`	DownloadURL the URL from which the SeedImage can be downloaded once built.
`state`	`string`	State reflect the state of the seed image build process.

.status.conditions

Description

Conditions describe the state of the seedimage object.

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.

Property	Type	Description
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/seedimages`
  - `DELETE`: delete collection of SeedImage
  - `GET`: list objects of kind SeedImage
  - `POST`: create a new SeedImage
- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/seedimages/{name}`
  - `DELETE`: delete the specified SeedImage
  - `GET`: read the specified SeedImage
  - `PATCH`: partially update the specified SeedImage

- `PUT`: replace the specified SeedImage

- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/seedimages/{name}/status`
  - `GET`: read status of the specified SeedImage
  - `PATCH`: partially update status of the specified SeedImage
  - `PUT`: replace status of the specified SeedImage

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/seedimages

HTTP method

`DELETE`

Description

delete collection of SeedImage

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind SeedImage

HTTP responses

HTTP code	Response body
200 - OK	`SeedImageList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new SeedImage

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`SeedImage` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`SeedImage` schema
201 - Created	`SeedImage` schema
202 - Accepted	`SeedImage` schema
401 - Unauthorized	Empty

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/seedimages/{name}

HTTP method

`DELETE`

Description

delete the specified SeedImage

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified SeedImage

HTTP responses

HTTP code	Response body
200 - OK	`SeedImage` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified SeedImage

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`SeedImage` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified SeedImage

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`SeedImage` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`SeedImage` schema
201 - Created	`SeedImage` schema
401 - Unauthorized	Empty

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/seedimages/{name}/status

HTTP method

`GET`

Description

read status of the specified SeedImage

HTTP responses

HTTP code	Response body
200 - OK	`SeedImage` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified SeedImage

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`SeedImage` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified SeedImage

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`SeedImage` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`SeedImage` schema
201 - Created	`SeedImage` schema
401 - Unauthorized	Empty

MachineInventory [elemental.cattle.io/v1beta1]

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	
`status`	`object`	

.spec

Type

`object`

Property	Type	Description
`ipAddressClaims`	`object`	IPAddressClaims is a map of IPAddressClaim associated to this machine. The map key is the ipAddressPool.name.
`ipAddressPools`	`object`	IPAddressPools is a list of IPAddressPool associated to this machine.

Property	Type	Description
`machineHash`	`string`	MachineHash the hash of the identifier used by the host to identify to the operator. This is used when the host authenticates without TPM. Both the authentication method and the identifier used to derive the hash depend upon the MachineRegistration spec.config.elemental.registration.auth value.
`network`	`object`	NetworkConfig is the final NetworkConfig.
`observedNetwork`	`object`	ObservedNetwork is a live-ISO-side snapshot of the host's actual network configuration uploaded by elemental-register during initial registration (Alauda fork). It is informational only and is intentionally kept separate from Network (the target/template network). Consumers such as baremetal providers can read this to know what the host looked like before install-time network reconfiguration.
`storage`	`object`	Storage is the explicit desired state for Inventory-owned data volumes. Devices omitted from this list remain unmanaged.
`tpmHash`	`string`	TPMHash the hash of the TPM EK public key. This is used if you are using TPM2 to identifiy nodes. You can obtain the TPM by running `rancherd get-tpm-hash` on the node. Or nodes can report their TPM hash by using the MachineRegister.

.spec.ipAddressClaims

Description

IPAddressClaims is a map of IPAddressClaim associated to this machine. The map key is the ipAddressPool.name.

Type

`object`

.spec.ipAddressPools

Description

IPAddressPools is a list of IPAddressPool associated to this machine.

Type

`object`

.spec.network

Description

NetworkConfig is the final NetworkConfig.

Type

`object`

Property	Type	Description
`config`		Config contains the network config template (nmc, nmstate, or nmconnections formats)
`configurator`	`string`	Configurator

Property	Type	Description
`ipAddresses`	`object`	IPAddresses contains a map of claimed IPAddresses

.spec.network.ipAddresses

Description

IPAddresses contains a map of claimed IPAddresses

Type

`object`

.spec.observedNetwork

Description

ObservedNetwork is a live-ISO-side snapshot of the host's actual network configuration uploaded by elemental-register during initial registration (Alauda fork). It is informational only and is intentionally kept separate from Network (the target/template network). Consumers such as baremetal providers can read this to know what the host looked like before install-time network reconfiguration.

Type

`object`

Property	Type	Description
`dnsServers`	`array`	DNSServers observed (typically from /etc/resolv.conf).
`interfaces`	`array`	Interfaces observed on the host.

Property	Type	Description
`routes`	`array`	Routes observed in the host's routing table.
`searchDomains`	`array`	SearchDomains observed.

.spec.observedNetwork.dnsServers

Description

DNSServers observed (typically from /etc/resolv.conf).

Type

`array`

.spec.observedNetwork.dnsServers[]

Type

`string`

.spec.observedNetwork.interfaces

Description

Interfaces observed on the host.

Type

`array`

.spec.observedNetwork.interfaces[]

Description

ObservedInterface describes a single network interface on the host.

Type

`object`

Required

`name`

Property	Type	Description
`addresses`	`array`	Addresses assigned to the interface, in CIDR form (e.g. 10.0.0.5/24, fe80::1/64). Both IPv4 and IPv6 may appear.
`mac`	`string`	MAC is the hardware address.
`mtu`	`integer`	MTU in bytes.
`name`	`string`	Name is the OS-level interface name (e.g. eth0, ens3).

.spec.observedNetwork.interfaces[].addresses

Description

Addresses assigned to the interface, in CIDR form (e.g. 10.0.0.5/24, fe80::1/64). Both IPv4 and IPv6 may appear.

Type

`array`

.spec.observedNetwork.interfaces[].addresses[]

Type

`string`

.spec.observedNetwork.routes

Description

Routes observed in the host's routing table.

Type

`array`

.spec.observedNetwork.routes[]

Description

ObservedRoute describes a single routing table entry.

Type

`object`

Required

`destination`

Property	Type	Description
`destination`	`string`	Destination CIDR, or the literal string "default" for the default route.
`gateway`	`string`	Gateway is the next-hop IP, if any.
`interface`	`string`	Interface is the outgoing device name, if any.
`metric`	`integer`	Metric of the route.

.spec.observedNetwork.searchDomains

Description

SearchDomains observed.

Type

`array`

.spec.observedNetwork.searchDomains[]

Type

`string`

.spec.storage

Description

Storage is the explicit desired state for Inventory-owned data volumes. Devices omitted from this list remain unmanaged.

Type

`object`

Property	Type	Description
`retryNonce`	`integer`	RetryNonce is a monotonic operator-controlled retry counter. It is not part of the normalized storage content hash.
`volumes`	`array`	Volumes are the only block devices the platform may manage.

.spec.storage.volumes

Description

Volumes are the only block devices the platform may manage.

Type

`array`

.spec.storage.volumes[]

Description

MachineInventoryStorageVolume declares one filesystem and its business mount point on this Inventory.

Type

`object`

Required

`filesystem` `mount` `name` `source`

Property	Type	Description
`filesystem`	`object`	StorageVolumeFilesystem declares filesystem adoption/initialization intent.
`mount`	`object`	StorageVolumeMount declares the business mount path and readiness policy.
`name`	`string`	Name is the stable logical identity inside this Inventory.
`source`	`object`	StorageVolumeSource selects one observer-reported logical block device.

.spec.storage.volumes[].filesystem

Description

StorageVolumeFilesystem declares filesystem adoption/initialization intent.

Type

`object`

Required

`policy` `type`

Property	Type	Description
`expectedUUID`	`string`	ExpectedUUID is mandatory for Adopt and forbidden for a new blank initialization.
`policy`	`string`	StorageFilesystemPolicy declares whether an existing filesystem is adopted or an entirely blank logical device may be initialized.
`type`	`string`	StorageFilesystemType is a supported ordinary single-host filesystem.

.spec.storage.volumes[].mount

Description

StorageVolumeMount declares the business mount path and readiness policy.

Type

`object`

Required

`path`

Property	Type	Description
`options`	`array`	Options contains individual, admission-whitelisted mount options.
`path`	`string`	Path is a normalized absolute path accepted by provider admission.
`required`	`boolean`	Required defaults to true. Optional volumes are still prepared and activated; only Machine readiness is relaxed.

.spec.storage.volumes[].mount.options

Description

Options contains individual, admission-whitelisted mount options.

Type

`array`

.spec.storage.volumes[].mount.options[]

Type

`string`

.spec.storage.volumes[].source

Description

StorageVolumeSource selects one observer-reported logical block device.

Type

`object`

Required

`deviceID`

Property	Type	Description
`deviceID`	`string`	DeviceID must exactly reference status.observedStorage.devices[].id. Dynamic kernel paths such as /dev/sdb are not accepted.
`minimumSize`	`integer|string`	MinimumSize is a non-destructive capacity assertion.
`multipath`	`object`	Multipath is valid only for a wwid: aggregate map.

.spec.storage.volumes[].source.multipath

Description

Multipath is valid only for a wwid: aggregate map.

Type

`object`

Property	Type	Description
`minimumActivePaths`	`integer`	MinimumActivePaths defaults to one.

.status

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions describe the state of the machine inventory object.
`observedStorage`	`object`	ObservedStorage is objective host state written only by the authenticated periodic observer channel.
`plan`	`object`	PlanStatus reflect the status of the plan owned by the machine inventory object.
`storage`	`object`	Storage is desired-state convergence written only by the baremetal provider Inventory Storage Reconciler.

.status.conditions

Description

Conditions describe the state of the machine inventory object.

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.

Property	Type	Description
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.observedStorage

Description

ObservedStorage is objective host state written only by the authenticated periodic observer channel.

Type

`object`

Property	Type	Description
`bootID`	`string`	BootID is the host's current Linux boot ID.
`devices`	`array`	Devices contains facts for all discovered logical block devices.
`observedAt`	`string`	ObservedAt is set by the operator when it accepts the report.

Property	Type	Description
`reportEpoch`	`integer`	ReportEpoch is allocated by the operator for an observer session.
`sequence`	`integer`	Sequence strictly increases within ReportEpoch.

.status.observedStorage.devices

Description

Devices contains facts for all discovered logical block devices.

Type

`array`

.status.observedStorage.devices[]

Description

ObservedStorageDevice describes one logical block device and its topology.

Type

`object`

Required

`id` `kind` `systemRole`

Property	Type	Description
`consumers`	`array`	Consumers contains descendant block devices layered on this logical device (for example partitions, dm-crypt, LVM or RAID). These are objective topology facts; provider policy decides

Property	Type	Description
		whether they make the device ineligible for management.
`filesystem`	`object`	ObservedStorageFilesystem describes an existing filesystem.
`health`	`object`	ObservedStorageMultipathHealth reports aggregate path health.
`id`	`string`	ID is the canonical management identity (wwn:, nvme-eui:, nvme-nguid:, serial:, partuuid:, or wwid:).
`kind`	`string`	ObservedStorageDeviceKind identifies the logical object represented by a report record.
`memberIDs`	`array`	MemberIDs lists Multipath member-path diagnostic identities. Members are never valid desired device IDs.
`model`	`string`	
`mounts`	`array`	
`parentID`	`string`	ParentID links a partition to its direct parent disk.
`partitionTableType`	`string`	

Property	Type	Description
`path`	`string`	Path is diagnostic only and must never be persisted as desired identity.
`readOnly`	`boolean`	
`removable`	`boolean`	
`rotational`	`boolean`	
`serial`	`string`	
`signatures`	`array`	
`sizeBytes`	`integer`	
`stablePaths`	`array`	StablePaths are current udev aliases for diagnostics/resolution.
`startBytes`	`integer`	StartBytes is populated for partitions.
`systemEvidence`	`array`	
`systemRole`	`string`	ObservedStorageSystemRole is an objective system-disk classification. Unknown is intentionally fail-closed in provider policy.
`transport`	`string`	Transport and identity evidence are informational and support safe serial fallback classification.
`wwn`	`string`	

.status.observedStorage.devices[].consumers

Description

Consumers contains descendant block devices layered on this logical device (for example partitions, dm-crypt, LVM or RAID). These are objective topology facts; provider policy decides whether they make the device ineligible for management.

Type

`array`

.status.observedStorage.devices[].consumers[]

Description

ObservedStorageConsumer describes one descendant block consumer. Path is diagnostic only; Type and FilesystemType are the kernel/util-linux facts.

Type

`object`

Required

`path` `type`

Property	Type	Description
`filesystemType`	`string`	
`mounts`	`array`	
`path`	`string`	
`type`	`string`	

.status.observedStorage.devices[].consumers[].mounts

Type

`array`

.status.observedStorage.devices[].consumers[].mounts[]

Type

`string`

.status.observedStorage.devices[].filesystem

Description

ObservedStorageFilesystem describes an existing filesystem.

Type

`object`

Required

`type`

Property	Type	Description
`type`	`string`	
`uuid`	`string`	

.status.observedStorage.devices[].health

Description

ObservedStorageMultipathHealth reports aggregate path health.

Type

`object`

Property	Type	Description
`activePaths`	`integer`	
`totalPaths`	`integer`	

.status.observedStorage.devices[].memberIDs

Description

MemberIDs lists Multipath member-path diagnostic identities. Members are never valid desired device IDs.

Type

`array`

.status.observedStorage.devices[].memberIDs[]

Type

`string`

.status.observedStorage.devices[].mounts

Type

`array`

.status.observedStorage.devices[].mounts[]

Description

ObservedStorageMount describes a current mount of this exact logical device.

Type

`object`

Required

`path`

Property	Type	Description
`options`	`array`	
`path`	`string`	

.status.observedStorage.devices[].mounts[].options

Type

`array`

.status.observedStorage.devices[].mounts[].options[]

Type

`string`

.status.observedStorage.devices[].signatures

Type

`array`

.status.observedStorage.devices[].signatures[]

Description

ObservedStorageSignature is one wipefs/lsblk signature fact.

Type

`object`

Required

`type` `value`

Property	Type	Description
`type`	`string`	
`value`	`string`	

.status.observedStorage.devices[].stablePaths

Description

StablePaths are current udev aliases for diagnostics/resolution.

Type

`array`

.status.observedStorage.devices[].stablePaths[]

Type

`string`

.status.observedStorage.devices[].systemEvidence

Type

`array`

.status.observedStorage.devices[].systemEvidence[]

Type

`string`

.status.plan

Description

PlanStatus reflect the status of the plan owned by the machine inventory object.

Type

`object`

Property	Type	Description
`checksum`	`string`	Checksum checksum of the created plan.

Property	Type	Description
`secretRef`	`object`	PlanSecretRef a reference to the created plan secret.
`state`	`string`	State reflect state of the plan that belongs to the machine inventory.

.status.plan.secretRef

Description

PlanSecretRef a reference to the created plan secret.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.

Property	Type	Description
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.status.storage

Description

Storage is desired-state convergence written only by the baremetal provider Inventory Storage Reconciler.

Type

`object`

Property	Type	Description
`appliedSpecHash`	`string`	
`appliedVolumes`	`array`	
`conditions`	`array`	
`hashVersion`	`string`	
`lastConsumedInitializationApprovalCounter`	`integer`	
`lastConsumedInitializationApprovalHash`	`string`	
`lastConsumedRetryNonce`	`integer`	
`operation`	`object`	StorageOperationStatus is the durable controller journal projection for a logical operation and its current execution attempt.
`pendingSpecHash`	`string`	
`phase`	`string`	StorageLifecyclePhase is the Inventory-wide managed-storage phase.

.status.storage.appliedVolumes

Type

`array`

.status.storage.appliedVolumes[]

Description

ManagedStorageVolumeStatus is a complete cleanup-capable snapshot of one managed volume, not merely a diagnostic summary.

Type

`object`

Required

`deviceID` `effectiveRequired` `filesystemPolicy` `filesystemType` `mountPath` `name` `required`

Property	Type	Description
`active`	`boolean`	
`deviceID`	`string`	
`effectiveRequired`	`boolean`	
`expectedFilesystemUUID`	`string`	
`filesystemPolicy`	`string`	StorageFilesystemPolicy declares whether an existing filesystem is adopted or an entirely blank logical device may be initialized.
`filesystemType`	`string`	StorageFilesystemType is a supported ordinary single-host filesystem.
`filesystemUUID`	`string`	

Property	Type	Description
`lastOperationID`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`minimumActivePaths`	`integer`	
`minimumSizeBytes`	`integer`	
`mountOptions`	`array`	
`mountPath`	`string`	
`mutationState`	`string`	StorageMutationState records whether durable host state may have changed.
`name`	`string`	
`phase`	`string`	
`reason`	`string`	
`required`	`boolean`	
`resolvedPath`	`string`	

.status.storage.appliedVolumes[].mountOptions

Type

`array`

.status.storage.appliedVolumes[].mountOptions[]

Type

`string`

.status.storage.conditions

Type

`array`

.status.storage.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation

Property	Type	Description
		is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.storage.operation

Description

StorageOperationStatus is the durable controller journal projection for a logical operation and its current execution attempt.

Type

`object`

Required

`attemptID` `mutationState` `operationID` `phase` `planOwnerKind`

`planOwnerUID` `retryNonce` `startedAt` `targetSpecHash` `type` `updatedAt`

Property	Type	Description
`attemptID`	`string`	
`initializationApprovalCounter`	`integer`	
`initializationApprovalHash`	`string`	
`mutationState`	`string`	StorageMutationState records whether durable host state may have changed.
`observationBootID`	`string`	ObservationBootID is the host boot ID captured before a Machine-owned lifecycle operation. Activate uses it to prove that reprovision crossed a boot before accepting a later mounted observation. It is a fact, not an ordering key.
`observationReportEpoch`	`integer`	ObservationReportEpoch and ObservationSequence capture the host report seen when the operation was claimed. The controller waits for a strictly newer accepted report before writing a host plan.
`observationSequence`	`integer`	
`operationID`	`string`	

Property	Type	Description
`phase`	`string`	StorageOperationPhase is the controller-side state of an operation.
`planChecksum`	`string`	
`planOwnerKind`	`string`	
`planOwnerUID`	`string`	
`retryNonce`	`integer`	
`startedAt`	`string`	
`targetSpecHash`	`string`	
`type`	`string`	StorageOperationType identifies one logical convergence operation.
`updatedAt`	`string`	
`volumes`	`array`	

.status.storage.operation.volumes

Type

`array`

.status.storage.operation.volumes[]

Description

ManagedStorageVolumeStatus is a complete cleanup-capable snapshot of one managed volume, not merely a diagnostic summary.

Type

`object`

Required

`deviceID` `effectiveRequired` `filesystemPolicy` `filesystemType` `mountPath` `name` `required`

Property	Type	Description
`active`	`boolean`	
`deviceID`	`string`	
`effectiveRequired`	`boolean`	
`expectedFilesystemUUID`	`string`	
`filesystemPolicy`	`string`	StorageFilesystemPolicy declares whether an existing filesystem is adopted or an entirely blank logical device may be initialized.
`filesystemType`	`string`	StorageFilesystemType is a supported ordinary single-host filesystem.
`filesystemUUID`	`string`	
`lastOperationID`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`minimumActivePaths`	`integer`	
`minimumSizeBytes`	`integer`	

Property	Type	Description
`mountOptions`	`array`	
`mountPath`	`string`	
`mutationState`	`string`	StorageMutationState records whether durable host state may have changed.
`name`	`string`	
`phase`	`string`	
`reason`	`string`	
`required`	`boolean`	
`resolvedPath`	`string`	

.status.storage.operation.volumes[].mountOptions

Type

`array`

.status.storage.operation.volumes[].mountOptions[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineinventories`

- `DELETE`: delete collection of MachineInventory
- `GET`: list objects of kind MachineInventory
- `POST`: create a new MachineInventory

- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineinventories/{name}`
  - `DELETE`: delete the specified MachineInventory
  - `GET`: read the specified MachineInventory
  - `PATCH`: partially update the specified MachineInventory
  - `PUT`: replace the specified MachineInventory
- `/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineinventories/{name}/status`
  - `GET`: read status of the specified MachineInventory
  - `PATCH`: partially update status of the specified MachineInventory
  - `PUT`: replace status of the specified MachineInventory

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineinventories

HTTP method

`DELETE`

Description

delete collection of MachineInventory

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind MachineInventory

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventoryList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new MachineInventory

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineInventory` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventory` schema
201 - Created	`MachineInventory` schema
202 - Accepted	`MachineInventory` schema
401 - Unauthorized	Empty

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineinventories/{name}

HTTP method

`DELETE`

Description

delete the specified MachineInventory

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified MachineInventory

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventory` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified MachineInventory

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventory` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified MachineInventory

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineInventory` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventory` schema

HTTP code	Response body
201 - Created	`MachineInventory` schema
401 - Unauthorized	Empty

/apis/elemental.cattle.io/v1beta1/namespaces/{namespace}/machineinventories/{name}/status

HTTP method

`GET`

Description

read status of the specified MachineInventory

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventory` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified MachineInventory

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventory` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified MachineInventory

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MachineInventory` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineInventory` schema
201 - Created	`MachineInventory` schema
401 - Unauthorized	Empty

User-Provisioned Infrastructure Provider APIs

UserProvisionedCluster UserProvisionedMachine UserProvisi

UserProvisionedCluster [infrastructure.cluster.x-k8s.io/v1beta1]

Description

UserProvisionedCluster is the Schema for the baremetalclusters API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	UserProvisionedClusterSpec defines the desired state of UserProvisionedCluster
`status`	`object`	UserProvisionedSpecClusterStatus defines the observed state of UserProvisionedCluster

.spec

Description

UserProvisionedClusterSpec defines the desired state of UserProvisionedCluster

Type

`object`

Required

`clusterCIDR` `credentialRef` `dnsDomain` `ha` `hostnameAsNodeName` `kubeProxy` `machinePoolRef` `networkConfig` `networkType` `serviceCIDR`

Property	Type	Description
`apiserverExtraArgs`	`object`	

Property	Type	Description
`clusterCIDR`	`object`	NetworkRanges represents ranges of network addresses.
`controlPlaneEndpoint`	`object`	APIEndpoint represents a reachable Kubernetes API endpoint.
`controllerManagerExtraArgs`	`object`	
`credentialRef`	`object`	LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.
`dnsDomain`	`string`	
`dockerExtraArgs`	`object`	
`etcd`	`object`	
`ha`	`object`	
`hostnameAsNodeName`	`boolean`	
`kubeProxy`	`object`	
`kubeletExtraArgs`	`object`	
`machinePoolRef`	`object`	LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Property	Type	Description
`maxNodePodNum`	`integer`	
`networkConfig`	`object`	
`networkType`	`string`	
`publicAlternaticenames`	`array`	
`schedulerExtraArgs`	`object`	
`serviceCIDR`	`object`	NetworkRanges represents ranges of network addresses.
`version`	`string`	

.spec.apiserverExtraArgs

Type

`object`

.spec.clusterCIDR

Description

NetworkRanges represents ranges of network addresses.

Type

`object`

Required

`cidrBlocks`

Property	Type	Description
`cidrBlocks`	`array`	

.spec.clusterCIDR.cidrBlocks

Type

`array`

.spec.clusterCIDR.cidrBlocks[]

Type

`string`

.spec.controlPlaneEndpoint

Description

APIEndpoint represents a reachable Kubernetes API endpoint.

Type

`object`

Required

`host` `port`

Property	Type	Description
`host`	`string`	The hostname on which the API server is serving.
`port`	`integer`	The port on which the API server is serving.

.spec.controllerManagerExtraArgs

Type

`object`

.spec.credentialRef

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.dockerExtraArgs

Type

`object`

.spec.etcd

Type

`object`

Property	Type	Description
`local`	`object`	

.spec.etcd.local

Type

`object`

Property	Type	Description
`dataDir`	`string`	DataDir is the directory etcd will place its data. Defaults to "/var/lib/etcd".
`extraArgs`	`object`	ExtraArgs are extra arguments provided to the etcd binary when run inside a static pod.
`peerCertSANs`	`array`	PeerCertSANs sets extra Subject Alternative Names for the etcd peer signing cert.
`serverCertSANs`	`array`	ServerCertSANs sets extra Subject Alternative Names for the etcd server signing cert.

.spec.etcd.local.extraArgs

Description

ExtraArgs are extra arguments provided to the etcd binary when run inside a static pod.

Type

`object`

.spec.etcd.local.peerCertSANs

Description

PeerCertSANs sets extra Subject Alternative Names for the etcd peer signing cert.

Type

`array`

.spec.etcd.local.peerCertSANs[]

Type

`string`

.spec.etcd.local.serverCertSANs

Description

ServerCertSANs sets extra Subject Alternative Names for the etcd server signing cert.

Type

`array`

.spec.etcd.local.serverCertSANs[]

Type

`string`

.spec.ha

Type

`object`

Required

`config` `type`

Property	Type	Description
`config`	`object`	
`type`	`string`	

.spec.ha.config

Type

`object`

Required

`vip` `vport` `vrid`

Property	Type	Description
`vip`	`string`	
`vport`	`integer`	
`vrid`	`integer`	

.spec.kubeProxy

Type

`object`

Required

`ipvs`

Property	Type	Description
`ipvs`	`boolean`	

.spec.kubeletExtraArgs

Type

`object`

.spec.machinePoolRef

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.networkConfig

Type

`object`

Required

`device` `stack`

Property	Type	Description
`device`	`string`	
`stack`	`string`	

.spec.publicAlternaticenames

Type

`array`

.spec.publicAlternaticenames[]

Type

`string`

.spec.schedulerExtraArgs

Type

`object`

.spec.serviceCIDR

Description

NetworkRanges represents ranges of network addresses.

Type

`object`

Required

`cidrBlocks`

Property	Type	Description
`cidrBlocks`	`array`	

.spec.serviceCIDR.cidrBlocks

Type

`array`

.spec.serviceCIDR.cidrBlocks[]

Type

`string`

.status

Description

UserProvisionedSpecClusterStatus defines the observed state of UserProvisionedCluster

Type

`object`

Property	Type	Description
`addresses`	`array`	
`clusterCIDR`	`string`	
`conditions`	`array`	
`dnsIP`	`string`	
`initialized`	`boolean`	
`nodeCIDRMaskSize`	`integer`	
`nodeCIDRMaskSizeIPv4`	`integer`	
`nodeCIDRMaskSizeIPv6`	`integer`	
`phase`	`string`	
`ready`	`boolean`	
`secondaryClusterCIDR`	`string`	
`secondaryServiceCIDR`	`string`	
`serviceCIDR`	`string`	
`version`	`string`	

.status.addresses

Type

`array`

.status.addresses[]

Description

ClusterAddress contains information for the cluster's address.

Type

`object`

Required

`host` `path` `port` `type`

Property	Type	Description
`host`	`string`	The cluster address.
`path`	`string`	path
`port`	`integer`	port
`type`	`string`	Cluster address type, one of Public, ExternalIP or InternalIP.

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	Last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	A human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	The reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may not be empty.
`severity`	`string`	Severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.

Property	Type	Description
`status`	`string`	Status of the condition, one of True, False, Unknown.
`type`	`string`	Type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedclusters`
  - `DELETE`: delete collection of UserProvisionedCluster
  - `GET`: list objects of kind UserProvisionedCluster
  - `POST`: create a new UserProvisionedCluster
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedclusters/{name}`
  - `DELETE`: delete the specified UserProvisionedCluster
  - `GET`: read the specified UserProvisionedCluster
  - `PATCH`: partially update the specified UserProvisionedCluster
  - `PUT`: replace the specified UserProvisionedCluster

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedclusters/{name}/status`
  - `GET`: read status of the specified UserProvisionedCluster
  - `PATCH`: partially update status of the specified UserProvisionedCluster
  - `PUT`: replace status of the specified UserProvisionedCluster

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedclusters

HTTP method

`DELETE`

Description

delete collection of UserProvisionedCluster

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind UserProvisionedCluster

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedClusterList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new UserProvisionedCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedCluster` schema
201 - Created	`UserProvisionedCluster` schema
202 - Accepted	`UserProvisionedCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedclusters/{name}

HTTP method

`DELETE`

Description

delete the specified UserProvisionedCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema

HTTP code	Response body
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified UserProvisionedCluster

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified UserProvisionedCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified UserProvisionedCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing

Parameter	Type	Description
		unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedCluster` schema
201 - Created	`UserProvisionedCluster` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedclusters/{name}/status

HTTP method

`GET`

Description

read status of the specified UserProvisionedCluster

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedCluster` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified UserProvisionedCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified UserProvisionedCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedCluster` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedCluster` schema
201 - Created	`UserProvisionedCluster` schema
401 - Unauthorized	Empty

UserProvisionedMachine [infrastructure.cluster.x-k8s.io/v1beta1]

Description

UserProvisionedMachine is the Schema for the baremetalmachines API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	UserProvisionedMachineSpec defines the desired state of UserProvisionedMachine
`status`	`object`	UserProvisionedMachineStatus defines the observed state of UserProvisionedMachine

.spec

Description

UserProvisionedMachineSpec defines the desired state of UserProvisionedMachine

Type

`object`

Required

`credentialRef` `cri` `port` `role` `version`

Property	Type	Description
`annotations`	`object`	
`clusterName`	`string`	

Property	Type	Description
`credentialRef`	`object`	LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.
`cri`	`object`	
`ip`	`string`	
`ipv6`	`string`	
`labels`	`object`	
`networkDevice`	`string`	
`port`	`integer`	
`providerID`	`string`	
`publicIp`	`string`	
`role`	`string`	
`taints`	`array`	
`version`	`string`	

.spec.annotations

Type

`object`

.spec.credentialRef

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.cri

Type

`object`

Required

`type` `version`

Property	Type	Description
`type`	`string`	
`version`	`string`	

.spec.labels

Type

`object`

.spec.taints

Type

`array`

.spec.taints[]

Description

The node this Taint is attached to has the "effect" on any pod that does not tolerate the Taint.

Type

`object`

Required

`effect` `key`

Property	Type	Description
`effect`	`string`	Required. The effect of the taint on pods that do not tolerate the taint. Valid effects are NoSchedule, PreferNoSchedule and NoExecute.
`key`	`string`	Required. The taint key to be applied to a node.
`timeAdded`	`string`	TimeAdded represents the time at which the taint was added. It is only written for NoExecute taints.
`value`	`string`	The taint value corresponding to the taint key.

.status

Description

UserProvisionedMachineStatus defines the observed state of UserProvisionedMachine

Type

`object`

Required

`phase`

Property	Type	Description
`addresses`	`array`	
`conditions`	`array`	
`cri`	`object`	
`nodeInfo`	`object`	NodeSystemInfo is a set of ids/uuids to uniquely identify the node.
`nodeRef`	`object`	ObjectReference contains enough information to let you inspect or modify the referred object.
`phase`	`string`	
`ready`	`boolean`	
`version`	`string`	

.status.addresses

Type

`array`

.status.addresses[]

Description

MachineAddress contains information for the node's address.

Type

`object`

Required

`address` `type`

Property	Type	Description
`address`	`string`	The machine address.
`type`	`string`	Machine address type, one of Hostname, ExternalIP, InternalIP, ExternalDNS or InternalDNS.

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	Last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	A human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	The reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may not be empty.
`severity`	`string`	Severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	Status of the condition, one of True, False, Unknown.
`type`	`string`	Type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary

Property	Type	Description
		conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.cri

Type

`object`

Required

`type` `version`

Property	Type	Description
`type`	`string`	
`version`	`string`	

.status.nodeInfo

Description

NodeSystemInfo is a set of ids/uuids to uniquely identify the node.

Type

`object`

Required

`architecture` `bootID` `containerRuntimeVersion` `kernelVersion` `kubeletVersion` `machineID` `operatingSystem` `osImage` `systemUUID`

Property	Type	Description
`architecture`	`string`	The Architecture reported by the node
`bootID`	`string`	Boot ID reported by the node.
`containerRuntimeVersion`	`string`	ContainerRuntime Version reported by the node through runtime remote API (e.g. containerd://1.4.
`kernelVersion`	`string`	Kernel Version reported by the node from 'uname (e.g. 3.16.0-0.bpo.4-amd64).
`kubeProxyVersion`	`string`	KubeProxy Version reported by the node.
`kubeletVersion`	`string`	Kubelet Version reported by the node.
`machineID`	`string`	MachineID reported by the node. For unique macl identification in the cluster this field is preferred. L more from man(5) machine-id: http://man7.org/linux/man-pages/man5/machine-id.5.html ↗
`operatingSystem`	`string`	The Operating System reported by the node

Property	Type	Description
`osImage`	`string`	OS Image reported by the node from /etc/os-relea (e.g. Debian GNU/Linux 7 (wheezy)).
`systemUUID`	`string`	SystemUUID reported by the node. For unique ma identification MachineID is preferred. This field is specific to Red Hat hosts https://access.redhat.com/documentation/en-us/red_hat_subscription_management/1/html/rhsr ↗

.status.nodeRef

Description

ObjectReference contains enough information to let you inspect or modify the referred object.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}"

Property	Type	Description
		(where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachines`
  - `DELETE`: delete collection of UserProvisionedMachine
  - `GET`: list objects of kind UserProvisionedMachine
  - `POST`: create a new UserProvisionedMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachines/{name}`
  - `DELETE`: delete the specified UserProvisionedMachine
  - `GET`: read the specified UserProvisionedMachine
  - `PATCH`: partially update the specified UserProvisionedMachine
  - `PUT`: replace the specified UserProvisionedMachine
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachines/{name}/status`
  - `GET`: read status of the specified UserProvisionedMachine
  - `PATCH`: partially update status of the specified UserProvisionedMachine
  - `PUT`: replace status of the specified UserProvisionedMachine

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachines

HTTP method

`DELETE`

Description

delete collection of UserProvisionedMachine

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind UserProvisionedMachine

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachineList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new UserProvisionedMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachine` schema
201 - Created	`UserProvisionedMachine` schema
202 - Accepted	`UserProvisionedMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachines/{name}

HTTP method

`DELETE`

Description

delete the specified UserProvisionedMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified UserProvisionedMachine

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified UserProvisionedMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified UserProvisionedMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachine` schema
201 - Created	`UserProvisionedMachine` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachines/{name}/status

HTTP method

`GET`

Description

read status of the specified UserProvisionedMachine

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified UserProvisionedMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified UserProvisionedMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachine` schema

HTTP code	Response body
201 - Created	`UserProvisionedMachine` schema
401 - Unauthorized	Empty

UserProvisionedMachinePool [infrastructure.cluster.x-k8s.io/v1beta1]

Description

UserProvisionedMachinePool is the Schema for the userprovisionedmachinepools API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	UserProvisionedMachinePoolSpec defines the desired state of UserProvisionedMachinePool
`status`	`object`	UserProvisionedMachinePoolStatus defines the observed state of UserProvisionedMachinePool

.spec

Description

UserProvisionedMachinePoolSpec defines the desired state of UserProvisionedMachinePool

Type

`object`

Required

`cri` `version`

Property	Type	Description
`clusterName`	`string`	
`cri`	`object`	

Property	Type	Description
`pool`	`array`	
`role`	`string`	
`version`	`string`	

.spec.cri

Type

`object`

Required

`type` `version`

Property	Type	Description
`type`	`string`	
`version`	`string`	

.spec.pool

Type

`array`

.spec.pool[]

Type

`object`

Required

`credentialRef` `port`

Property	Type	Description
`annotations`	`object`	
`credentialRef`	`object`	LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.
`ip`	`string`	
`ipv6`	`string`	
`labels`	`object`	
`networkDevice`	`string`	
`port`	`integer`	
`publicIp`	`string`	
`taints`	`array`	

.spec.pool[].annotations

Type

`object`

.spec.pool[].credentialRef

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.pool[].labels

Type

`object`

.spec.pool[].taints

Type

`array`

.spec.pool[].taints[]

Description

The node this Taint is attached to has the "effect" on any pod that does not tolerate the Taint.

Type

`object`

Required

`effect` `key`

Property	Type	Description
`effect`	`string`	Required. The effect of the taint on pods that do not tolerate the taint. Valid effects are NoSchedule, PreferNoSchedule and NoExecute.
`key`	`string`	Required. The taint key to be applied to a node.
`timeAdded`	`string`	TimeAdded represents the time at which the taint was added. It is only written for NoExecute taints.
`value`	`string`	The taint value corresponding to the taint key.

.status

Description

UserProvisionedMachinePoolStatus defines the observed state of UserProvisionedMachinePool

Type

`object`

Required

`phase`

Property	Type	Description
`conditions`	`array`	
`machines`	`array`	

Property	Type	Description
`phase`	`string`	

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

Type

`object`

Required

`lastTransitionTime` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	Last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	A human readable message indicating details about the transition. This field may be empty.
`reason`	`string`	The reason for the condition's last transition in CamelCase. The specific API may choose whether

Property	Type	Description
		or not this field is considered a guaranteed API. This field may not be empty.
`severity`	`string`	Severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
`status`	`string`	Status of the condition, one of True, False, Unknown.
`type`	`string`	Type of condition in CamelCase or in foo.example.com/CamelCase. Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important.

.status.machines

Type

`array`

.status.machines[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachinepools`
  - `DELETE`: delete collection of UserProvisionedMachinePool
  - `GET`: list objects of kind UserProvisionedMachinePool
  - `POST`: create a new UserProvisionedMachinePool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachinepools/{name}`
  - `DELETE`: delete the specified UserProvisionedMachinePool
  - `GET`: read the specified UserProvisionedMachinePool
  - `PATCH`: partially update the specified UserProvisionedMachinePool
  - `PUT`: replace the specified UserProvisionedMachinePool
- `/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachinepools/{name}/status`
  - `GET`: read status of the specified UserProvisionedMachinePool
  - `PATCH`: partially update status of the specified UserProvisionedMachinePool
  - `PUT`: replace status of the specified UserProvisionedMachinePool

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachinepools

HTTP method

`DELETE`

Description

delete collection of UserProvisionedMachinePool

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind UserProvisionedMachinePool

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePoolList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new UserProvisionedMachinePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedMachinePool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePool` schema
201 - Created	`UserProvisionedMachinePool` schema
202 - Accepted	`UserProvisionedMachinePool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisionedmachinepools/{name}

HTTP method

`DELETE`

Description

delete the specified UserProvisionedMachinePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified UserProvisionedMachinePool

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified UserProvisionedMachinePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified UserProvisionedMachinePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedMachinePool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePool` schema
201 - Created	`UserProvisionedMachinePool` schema
401 - Unauthorized	Empty

/apis/infrastructure.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/userprovisioned

machinepools/{name}/status

HTTP method

`GET`

Description

read status of the specified UserProvisionedMachinePool

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePool` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified UserProvisionedMachinePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified UserProvisionedMachinePool

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the

Parameter	Type	Description
		default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`UserProvisionedMachinePool` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`UserProvisionedMachinePool` schema
201 - Created	`UserProvisionedMachinePool` schema
401 - Unauthorized	Empty