MachineConfiguration APIs

MachineConfig [machineconfig MachineConfigPool [machinecc MachineCo

MachineConfig [machineconfiguration.alauda.io/v1alpha1]

Description

MachineConfig defines the configuration for a machine Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

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`	MachineConfigSpec is the spec for MachineConfig

.spec

Description

MachineConfigSpec is the spec for MachineConfig

Type

`object`

Property	Type	Description
`config`	`object`	Config is a Ignition Config object.
`kernelArguments`	`array`	kernelArguments contains a list of kernel arguments to be added

.spec.config

Description

Config is a Ignition Config object.

Type

`object`

.spec.kernelArguments

Description

kernelArguments contains a list of kernel arguments to be added

Type

`array`

.spec.kernelArguments[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigs`
  - `DELETE`: delete collection of MachineConfig
  - `GET`: list objects of kind MachineConfig
  - `POST`: create a new MachineConfig
- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigs/{name}`
  - `DELETE`: delete the specified MachineConfig
  - `GET`: read the specified MachineConfig
  - `PATCH`: partially update the specified MachineConfig

- `PUT`: replace the specified MachineConfig

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigs

HTTP method

`DELETE`

Description

delete collection of MachineConfig

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind MachineConfig

HTTP responses

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

HTTP method

`POST`

Description

create a new MachineConfig

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`	`MachineConfig` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MachineConfig` schema

HTTP code	Response body
201 - Created	`MachineConfig` schema
202 - Accepted	`MachineConfig` schema
401 - Unauthorized	Empty

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigs/{name}

HTTP method

`DELETE`

Description

delete the specified MachineConfig

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 MachineConfig

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified MachineConfig

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	`MachineConfig` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified MachineConfig

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`	`MachineConfig` schema	`application/json` formatted

HTTP responses

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

MachineConfigPool [machineconfiguration.alauda.io/v1alpha1]

Description

MachineConfigPool describes a pool of MachineConfigs. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

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`	MachineConfigPoolSpec is the spec for MachineConfigPool resource.
`status`	`object`	MachineConfigPoolStatus is the status for MachineConfigPool resource.

.spec

Description

MachineConfigPoolSpec is the spec for MachineConfigPool resource.

Type

`object`

Property	Type	Description
`configuration`	`object`	The targeted MachineConfig object for the machi config pool.
`machineConfigSelector`	`object`	machineConfigSelector specifies a label selector MachineConfigs. Refer https://kubernetes.io/docs/concepts/overview/wo with-objects/labels/ ↗ on how label and selectors work.
`maxUnavailable`	`integer|string`	maxUnavailable defines either an integer numbe percentage of nodes in the pool that can go Unavailable during an update. This includes node Unavailable for any reason, including user initiate cordons, failing nodes, etc. The default value is 1 A value larger than 1 will mean multiple nodes go unavailable during the update, which may affect y workload stress on the remaining nodes. You car set this value to 0 to stop updates (it will default t to 1); to stop updates, use the 'paused' property instead. Drain will respect Pod Disruption Budget (PDBs) such as etcd quorum guards, even if maxUnavailable is greater than one.
`nodeSelector`	`object`	nodeSelector specifies a label selector for Machi
`paused`	`boolean`	paused specifies whether or not changes to this machine config pool should be stopped. This incl

Property	Type	Description
		generating new desiredMachineConfig and upda machines.

.spec.configuration

Description

The targeted MachineConfig object for the machine config pool.

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. TODO: this design is not final and this field is subject to change in the future.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-

Property	Type	Description
		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 ↗
`source`	`array`	source is the list of MachineConfig objects that were used to generate the single MachineConfig object specified in `content`.
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.configuration.source

Description

source is the list of MachineConfig objects that were used to generate the single MachineConfig object specified in `content`.

Type

`array`

.spec.configuration.source[]

Description

ObjectReference contains enough information to let you inspect or modify the referred object. --- New uses of this type are discouraged because of difficulty describing its usage when embedded in APIs. 1. Ignored fields. It includes many fields which are not generally honored. For instance, ResourceVersion and FieldPath are both very rarely valid in actual usage. 2. Invalid usage help. It is impossible to add specific help for individual usage. In most embedded usages, there are particular restrictions like, "must refer only to types A and B" or "UID not honored" or "name must be restricted". Those cannot be well described when embedded. 3. Inconsistent validation. Because the usages are different, the validation rules are different by usage, which makes it hard for users to predict what will happen. 4. The fields are both imprecise and overly precise. Kind is not a precise mapping to a URL. This can produce ambiguity during interpretation and require a REST mapping. In most cases, the dependency is on the group,resource tuple and the version of the actual struct is irrelevant. 5. We cannot easily change it. Because this type is embedded in many locations, updates to this type will affect numerous schemas. Don't make new APIs embed an underspecified API type they do not control. Instead of using this type, create a locally provided and used type that is well-focused on your reference. For example, ServiceReferences for admission registration: https://github.com/kubernetes/api/blob/release-1.17/admissionregistration/v1/types.go#L533 .

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. TODO: this design is not final and this field is subject to change in the future.
`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:

Property	Type	Description
		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.machineConfigSelector

Description

machineConfigSelector specifies a label selector for MachineConfigs. Refer https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/ on how label and selectors work.

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.machineConfigSelector.matchExpressions

Description

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

Type

`array`

.spec.machineConfigSelector.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.machineConfigSelector.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.machineConfigSelector.matchExpressions[].values[]

Type

`string`

.spec.machineConfigSelector.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.nodeSelector

Description

nodeSelector specifies a label selector for Machines

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.nodeSelector.matchExpressions

Description

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

Type

`array`

.spec.nodeSelector.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.nodeSelector.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.nodeSelector.matchExpressions[].values[]

Type

`string`

.spec.nodeSelector.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

MachineConfigPoolStatus is the status for MachineConfigPool resource.

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions represents the latest available observations of current state.
`configuration`	`object`	configuration represents the current MachineConfig object for the machine config pool.
`degradedMachineCount`	`integer`	degradedMachineCount represents the total number of machines marked degraded (or unreconcilable). A node is marked degraded if applying a configuration failed..
`machineCount`	`integer`	machineCount represents the total number of machines in the machine config pool.

Property	Type	Description
`observedGeneration`	`integer`	observedGeneration represents the generation observed by the controller.
`poolSynchronizersStatus`	`array`	poolSynchronizersStatus is the status of the machines managed by the pool synchronizers.
`readyMachineCount`	`integer`	readyMachineCount represents the total number of ready machines targeted by the pool.
`unavailableMachineCount`	`integer`	unavailableMachineCount represents the total number of unavailable (non-ready) machines targeted by the pool. A node is marked unavailable if it is in updating state or NodeReady condition is false.
`updatedMachineCount`	`integer`	updatedMachineCount represents the total number of machines targeted by the pool that have the CurrentMachineConfig as their config.

.status.conditions

Description

conditions represents the latest available observations of current state.

Type

`array`

.status.conditions[]

Description

MachineConfigPoolCondition contains condition information for an MachineConfigPool.

Type

`object`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the timestamp corresponding to the last status change of this condition.
`message`	`string`	message is a human readable description of the details of the last transition, complementing reason.
`reason`	`string`	reason is a brief machine readable explanation for the condition's last transition.
`status`	`string`	status of the condition, one of ('True', 'False', 'Unknown').
`type`	`string`	type of the condition, currently ('Done', 'Updating', 'Failed').

.status.configuration

Description

configuration represents the current MachineConfig object for the machine config pool.

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. TODO: this design is not final and this field is subject to change in the future.
`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 ↗
`source`	`array`	source is the list of MachineConfig objects that were used to generate the single MachineConfig object specified in `content`.
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.status.configuration.source

Description

source is the list of MachineConfig objects that were used to generate the single MachineConfig object specified in `content`.

Type

`array`

.status.configuration.source[]

Description

ObjectReference contains enough information to let you inspect or modify the referred object. --- New uses of this type are discouraged because of difficulty describing its usage when embedded in APIs. 1. Ignored fields. It includes many fields which are not generally honored. For instance, ResourceVersion and FieldPath are both very rarely valid in actual usage. 2. Invalid usage help. It is impossible to add specific help for individual usage. In most embedded usages, there are particular restrictions like, "must refer only to types A and B" or "UID not honored" or "name must be restricted". Those cannot be well described when embedded. 3. Inconsistent validation. Because the usages are different, the validation rules are different by usage, which makes it hard for users to predict what will happen. 4. The fields are both imprecise and overly precise. Kind is not a precise mapping to a URL. This can produce ambiguity during interpretation and require a REST mapping. In most cases, the dependency is on the group,resource tuple and the version of the actual struct is irrelevant. 5. We cannot easily change it. Because this type is embedded in many locations, updates to this type will affect numerous schemas. Don't make new APIs embed an underspecified API type they do not control. Instead of using this type, create a locally provided and used type that is well-focused on your reference. For example, ServiceReferences for admission registration: https://github.com/kubernetes/api/blob/release-1.17/admissionregistration/v1/types.go#L533 .

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

Property	Type	Description
		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. TODO: this design is not final and this field is subject to change in the future.
`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.poolSynchronizersStatus

Description

poolSynchronizersStatus is the status of the machines managed by the pool synchronizers.

Type

`array`

.status.poolSynchronizersStatus[]

Type

`object`

Required

`availableMachineCount` `machineCount` `poolSynchronizerType` `readyMachineCount` `unavailableMachineCount` `updatedMachineCount`

Property	Type	Description
`availableMachineCount`	`integer`	availableMachineCount is the number of machines managed by the node synchronizer which are available.
`machineCount`	`integer`	machineCount is the number of machines that are managed by the node synchronizer.
`observedGeneration`	`integer`	observedGeneration is the last generation change that has been applied.

Property	Type	Description
`poolSynchronizerType`	`string`	poolSynchronizerType describes the type of the pool synchronizer.
`readyMachineCount`	`integer`	readyMachineCount is the number of machines managed by the node synchronizer that are in a ready state.
`unavailableMachineCount`	`integer`	unavailableMachineCount is the number of machines managed by the node synchronizer but are unavailable.
`updatedMachineCount`	`integer`	updatedMachineCount is the number of machines that have been updated by the node synchronizer.

API Endpoints

The following API endpoints are available:

- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigpools`
  - `DELETE`: delete collection of MachineConfigPool
  - `GET`: list objects of kind MachineConfigPool
  - `POST`: create a new MachineConfigPool
- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigpools/{name}`

- `DELETE`: delete the specified MachineConfigPool
- `GET`: read the specified MachineConfigPool
- `PATCH`: partially update the specified MachineConfigPool
- `PUT`: replace the specified MachineConfigPool

- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigpools/{name}/status`
  - `GET`: read status of the specified MachineConfigPool
  - `PATCH`: partially update status of the specified MachineConfigPool
  - `PUT`: replace status of the specified MachineConfigPool

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigpools

HTTP method

`DELETE`

Description

delete collection of MachineConfigPool

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind MachineConfigPool

HTTP responses

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

HTTP method

`POST`

Description

create a new MachineConfigPool

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`	`MachineConfigPool` schema	`application/json` formatted

HTTP responses

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

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigpools/{name}

HTTP method

`DELETE`

Description

delete the specified MachineConfigPool

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 MachineConfigPool

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified MachineConfigPool

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	`MachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified MachineConfigPool

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`	`MachineConfigPool` schema	`application/json` formatted

HTTP responses

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

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigpools/{name}/status

HTTP method

`GET`

Description

read status of the specified MachineConfigPool

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified MachineConfigPool

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	`MachineConfigPool` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified MachineConfigPool

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`	`MachineConfigPool` schema	`application/json` formatted

HTTP responses

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

MachineConfiguration [machineconfiguration.alauda.io/v1alpha1]

Description

MachineConfiguration provides information to configure an operator to manage Machine Configuration. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

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 is the specification of the desired behavior of the Machine Config Operator
`status`	`object`	status is the most recently observed status of the Machine Config Operator

.spec

Description

spec is the specification of the desired behavior of the Machine Config Operator

Type

`object`

Property	Type	Description
`defaultNodeDisruptionPolicySpecAction`	`object`	DefaultNodeDisruptionPolicy is the default disruption

Property	Type	Description
		policy for files/units/sshkeys
`nodeDisruptionPolicy`	`object`	nodeDisruptionPolicy allows an admin to set granular node disruption actions for MachineConfig-based updates, such as drains, service reloads, etc. Specifying this will allow for less downtime when doing small configuration updates to the cluster. This configuration has no effect on cluster upgrades which will still incur node disruption where required.

.spec.defaultNodeDisruptionPolicySpecAction

Description

DefaultNodeDisruptionPolicy is the default disruption policy for files/units/sshkeys

Type

`object`

Required

`files` `units`

Property	Type	Description
`files`	`array`	files is the default node desruption policy for files This list supports a maximum of 10 entries.
`units`	`array`	units is the default node desruption policy for units This list supports a maximum of 10 entries.

.spec.defaultNodeDisruptionPolicySpecAction.files

Description

files is the default node desruption policy for files This list supports a maximum of 10 entries.

Type

`array`

.spec.defaultNodeDisruptionPolicySpecAction.files[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload

Property	Type	Description
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

.spec.defaultNodeDisruptionPolicySpecAction.files[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service",

Property	Type	Description
		".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.defaultNodeDisruptionPolicySpecAction.files[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.defaultNodeDisruptionPolicySpecAction.units

Description

units is the default node desruption policy for units This list supports a maximum of 10 entries.

Type

`array`

.spec.defaultNodeDisruptionPolicySpecAction.units[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

.spec.defaultNodeDisruptionPolicySpecAction.units[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.defaultNodeDisruptionPolicySpecAction.units[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255

Property	Type	Description
		characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.nodeDisruptionPolicy

Description

nodeDisruptionPolicy allows an admin to set granular node disruption actions for MachineConfig-based updates, such as drains, service reloads, etc. Specifying this will allow for less downtime when doing small configuration updates to the cluster. This configuration has no effect on cluster upgrades which will still incur node disruption where required.

Type

`object`

Property	Type	Description
`files`	`array`	files is a list of MachineConfig file definitions and actions to take to changes on those paths This list supports a maximum of 50 entries.
`sshkey`	`object`	sshkey maps to the ignition.sshkeys field in the MachineConfig object, definition an action for this will apply to all sshkey changes in the cluster

Property	Type	Description
`units`	`array`	units is a list MachineConfig unit definitions and actions to take on changes to those services This list supports a maximum of 50 entries.

.spec.nodeDisruptionPolicy.files

Description

files is a list of MachineConfig file definitions and actions to take to changes on those paths This list supports a maximum of 50 entries.

Type

`array`

.spec.nodeDisruptionPolicy.files[]

Description

NodeDisruptionPolicySpecFile is a file entry and corresponding actions to take and is used in the NodeDisruptionPolicyConfig object

Type

`object`

Required

`actions` `path`

Property	Type	Description
`actions`	`array`	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a

Property	Type	Description
		reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
`path`	`string`	path is the location of a file being managed through a MachineConfig. The Actions in the policy will apply to changes to the file at this path.

.spec.nodeDisruptionPolicy.files[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

`array`

.spec.nodeDisruptionPolicy.files[].actions[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

.spec.nodeDisruptionPolicy.files[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long

Property	Type	Description
		and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.nodeDisruptionPolicy.files[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.nodeDisruptionPolicy.sshkey

Description

sshkey maps to the ignition.sshkeys field in the MachineConfig object, definition an action for this will apply to all sshkey changes in the cluster

Type

`object`

Required

`actions`

Property	Type	Description
`actions`	`array`	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

.spec.nodeDisruptionPolicy.sshkey.actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

`array`

.spec.nodeDisruptionPolicy.sshkey.actions[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

.spec.nodeDisruptionPolicy.sshkey.actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.nodeDisruptionPolicy.sshkey.actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.nodeDisruptionPolicy.units

Description

units is a list MachineConfig unit definitions and actions to take on changes to those services This list supports a maximum of 50 entries.

Type

`array`

.spec.nodeDisruptionPolicy.units[]

Description

NodeDisruptionPolicySpecUnit is a systemd unit name and corresponding actions to take and is used in the NodeDisruptionPolicyConfig object

Type

`object`

Required

`actions` `name`

Property	Type	Description
`actions`	`array`	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Property	Type	Description
`name`	`string`	name represents the service name of a systemd service managed through a MachineConfig Actions specified will be applied for changes to the named service. Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.nodeDisruptionPolicy.units[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

`array`

.spec.nodeDisruptionPolicy.units[].actions[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

.spec.nodeDisruptionPolicy.units[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long

Property	Type	Description
		and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.spec.nodeDisruptionPolicy.units[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.status

Description

status is the most recently observed status of the Machine Config Operator

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions is a list of conditions and their status
`nodeDisruptionPolicyStatus`	`object`	nodeDisruptionPolicyStatus status reflects what the latest cluster-validated policies are, and will be used by the Machine Config Daemon during future node updates.
`observedGeneration`	`integer`	observedGeneration is the last generation change you've dealt with

.status.conditions

Description

conditions is a list of conditions and their status

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource. --- This struct is intended for direct use as an array at the field path .status.conditions. For example, type FooStatus struct{ // Represents the observations of a foo's current state. // Known

.status.conditions.type are: "Available", "Progressing", and "Degraded" // +patchMergeKey=type // +patchStrategy=merge // +listType=map // +listMapKey=type Conditions []metav1.Condition `json:"conditions,omitempty" patchStrategy:"merge" patchMergeKey:"type" protobuf:"bytes,1,rep,name=conditions"` // other fields }

Type

`object`

Required

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

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitio from one status to another. This should be when the und condition changed. If that is not known, then using the ti when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating detai the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generatio the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condi out of date with respect to the current state of the instan

Property	Type	Description
`reason`	`string`	reason contains a programmatic identifier indicating the for the condition's last transition. Producers of specific c types may define expected values and meanings for this and whether the values are considered a guaranteed AP value should be a CamelCase string. This field may not empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	# type of condition in CamelCase or in foo.example.com/CamelCa Many .condition.type values are consistent across resou like Available, but because arbitrary conditions can be u (see .node.status.conditions), the ability to deconflict is important. The regex it matches is (dns1123Subdomain (qualifiedNameFmt)

.status.nodeDisruptionPolicyStatus

Description

nodeDisruptionPolicyStatus status reflects what the latest cluster-validated policies are, and will be used by the Machine Config Daemon during future node updates.

Type

`object`

Property	Type	Description
`clusterPolicies`	`object`	clusterPolicies is a merge of cluster default and user provided node disruption policies.

.status.nodeDisruptionPolicyStatus.clusterPolicies

Description

clusterPolicies is a merge of cluster default and user provided node disruption policies.

Type

`object`

Property	Type	Description
`files`	`array`	files is a list of MachineConfig file definitions and actions to take to changes on those paths
`sshkey`	`object`	sshkey is the overall sshkey MachineConfig definition
`units`	`array`	units is a list MachineConfig unit definitions and actions to take on changes to those services

.status.nodeDisruptionPolicyStatus.clusterPolicies.files

Description

files is a list of MachineConfig file definitions and actions to take to changes on those paths

Type

`array`

.status.nodeDisruptionPolicyStatus.clusterPolicies.files[]

Description

NodeDisruptionPolicyStatusFile is a file entry and corresponding actions to take and is used in the NodeDisruptionPolicyClusterStatus object

Type

`object`

Required

`actions` `path`

Property	Type	Description
`actions`	`array`	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
`path`	`string`	path is the location of a file being managed through a MachineConfig. The Actions in the policy will apply to changes to the file at this path.

.status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

`array`

.status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicyStatusActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload, None and Special. reload/restart requires a corresponding service target specified in the

Property	Type	Description
		reload/restart field. Other values require no further configuration

.status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey

Description

sshkey is the overall sshkey MachineConfig definition

Type

`object`

Required

`actions`

Property	Type	Description
`actions`	`array`	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this

Property	Type	Description
		list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

.status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

`array`

.status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicyStatusActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload, None and Special. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

.status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.status.nodeDisruptionPolicyStatus.clusterPolicies.units

Description

units is a list MachineConfig unit definitions and actions to take on changes to those services

Type

`array`

.status.nodeDisruptionPolicyStatus.clusterPolicies.units[]

Description

NodeDisruptionPolicyStatusUnit is a systemd unit name and corresponding actions to take and is used in the NodeDisruptionPolicyClusterStatus object

Type

`object`

Required

`actions` `name`

Property	Type	Description
`actions`	`array`	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Property	Type	Description
`name`	`string`	name represents the service name of a systemd service managed through a MachineConfig Actions specified will be applied for changes to the named service. Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

`array`

.status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[]

Type

`object`

Required

`type`

Property	Type	Description
`reload`	`object`	reload specifies the service to reload, only valid if type is reload
`restart`	`object`	restart specifies the service to restart, only valid if type is restart
`type`	`string`	type represents the commands that will be carried out if this NodeDisruptionPolicyStatusActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload, None and Special. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

.status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

.status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

`object`

Required

`serviceName`

Property	Type	Description
`serviceName`	`string`	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

API Endpoints

The following API endpoints are available:

- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigurations`
  - `DELETE`: delete collection of MachineConfiguration
  - `GET`: list objects of kind MachineConfiguration
  - `POST`: create a new MachineConfiguration
- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigurations/{name}`
  - `DELETE`: delete the specified MachineConfiguration
  - `GET`: read the specified MachineConfiguration
  - `PATCH`: partially update the specified MachineConfiguration
  - `PUT`: replace the specified MachineConfiguration
- `/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigurations/{name}/status`
  - `GET`: read status of the specified MachineConfiguration
  - `PATCH`: partially update status of the specified MachineConfiguration
  - `PUT`: replace status of the specified MachineConfiguration

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigurations

HTTP method

`DELETE`

Description

delete collection of MachineConfiguration

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind MachineConfiguration

HTTP responses

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

HTTP method

`POST`

Description

create a new MachineConfiguration

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`	`MachineConfiguration` schema	`application/json` formatted

HTTP responses

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

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigurations/{name}

HTTP method

`DELETE`

Description

delete the specified MachineConfiguration

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 MachineConfiguration

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified MachineConfiguration

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	`MachineConfiguration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified MachineConfiguration

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`	`MachineConfiguration` schema	`application/json` formatted

HTTP responses

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

/kubernetes/{cluster}/apis/machineconfiguration.alauda.io/v1alpha1/machineconfigurations/{name}/status

HTTP method

`GET`

Description

read status of the specified MachineConfiguration

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified MachineConfiguration

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	`MachineConfiguration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified MachineConfiguration

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`	`MachineConfiguration` schema	`application/json` formatted

HTTP responses

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

HTTP code	Response body
401 - Unauthorized	Empty