OpenShift Container Platform 4.18

Node APIs

Reference guide for node APIs

Last Updated: 2025-05-15

OpenShift Container Platform 4.18 Node APIs

Reference guide for node APIs

Abstract

This document describes the OpenShift Container Platform node API objects and their detailed specifications.

Table of Contents

CHAPTER 1. NODE APIS

1.1. NODE [V1]

Description

Node is a worker node in Kubernetes. Each node will have a unique identifier in the cache (i.e. in etcd).

Type

object

1.2. PERFORMANCEPROFILE [PERFORMANCE.OPENSHIFT.IO/V2]

Description

PerformanceProfile is the Schema for the performanceprofiles API

Type

object

1.3. PROFILE [TUNED.OPENSHIFT.IO/V1]

Description

Profile is a specification for a Profile resource.

Type

object

1.4. RUNTIMECLASS [NODE.K8S.IO/V1]

Description

RuntimeClass defines a class of container runtime supported in the cluster. The RuntimeClass is used to determine which container runtime is used to run all containers in a pod. RuntimeClasses are manually defined by a user or cluster provisioner, and referenced in the PodSpec. The Kubelet is responsible for resolving the RuntimeClassName reference before running the pod. For more details, see https://kubernetes.io/docs/concepts/containers/runtime-class/

Type

object

1.5. TUNED [TUNED.OPENSHIFT.IO/V1]

Description

Tuned is a collection of rules that allows cluster-wide deployment of node-level sysctls and more flexibility to add custom tuning specified by user needs. These rules are translated and passed to all containerized Tuned daemons running in the cluster in the format that the daemons understand. The responsibility for applying the node-level tuning then lies with the containerized Tuned daemons. More info: https://github.com/openshift/cluster-node-tuning-operator

Type

object

CHAPTER 2. NODE [V1]

Description

Node is a worker node in Kubernetes. Each node will have a unique identifier in the cache (i.e. in etcd).

Type

object

2.1. 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
metadata	**ObjectMeta**	Standard object’s metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	NodeSpec describes the attributes that a node is created with.
status	**object**	NodeStatus is information about the current status of a node.

2.1.1. .spec

Description

NodeSpec describes the attributes that a node is created with.

Type

object

Property	Type	Description
configSource	**object**	NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22
externalID	**string**	Deprecated. Not all kubelets will set this field. Remove field after 1.13. see: https://issues.k8s.io/61966
podCIDR	**string**	PodCIDR represents the pod IP range assigned to the node.
podCIDRs	**array (string)**	podCIDRs represents the IP ranges assigned to the node for usage by Pods on that node. If this field is specified, the 0th entry must match the podCIDR field. It may contain at most 1 value for each of IPv4 and IPv6.
providerID	**string**	ID of the node assigned by the cloud provider in the format: <ProviderName>://<ProviderSpecificNodeID>
taints	**array**	If specified, the node's taints.
taints[]	**object**	The node this Taint is attached to has the "effect" on any pod that does not tolerate the Taint.
unschedulable	**boolean**	Unschedulable controls node schedulability of new pods. By default, node is schedulable. More info: https://kubernetes.io/docs/concepts/nodes/node/#manual-node-administration

2.1.2. .spec.configSource

Description

NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22

Type

object

Property	Type	Description
configMap	**object**	ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

2.1.3. .spec.configSource.configMap

Description

ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

Type

object

Required

- **namespace**
- **name**
- **kubeletConfigKey**

Property	Type	Description
kubeletConfigKey	**string**	KubeletConfigKey declares which key of the referenced ConfigMap corresponds to the KubeletConfiguration structure This field is required in all cases.
name	**string**	Name is the metadata.name of the referenced ConfigMap. This field is required in all cases.
namespace	**string**	Namespace is the metadata.namespace of the referenced ConfigMap. This field is required in all cases.

Property	Type	Description
resourceVersion	**string**	ResourceVersion is the metadata.ResourceVersion of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.
uid	**string**	UID is the metadata.UID of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.

2.1.4. .spec.taints

Description

If specified, the node's taints.

Type

array

2.1.5. .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

- **key**
- **effect**

Property	Type	Description

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. Possible enum values: - **"NoExecute"** Evict any already-running pods that do not tolerate the taint. Currently enforced by NodeController. - **"NoSchedule"** Do not allow new pods to schedule onto the node unless they tolerate the taint, but allow all pods submitted to Kubelet without going through the scheduler to start, and allow all already-running pods to continue running. Enforced by the scheduler. - **"PreferNoSchedule"** Like TaintEffectNoSchedule, but the scheduler tries not to schedule new pods onto the node, rather than prohibiting new pods from scheduling onto the node entirely. Enforced by the scheduler.
key	**string**	Required. The taint key to be applied to a node.
timeAdded	**Time**	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.

2.1.6. .status

Description

NodeStatus is information about the current status of a node.

Type

object

Property	Type	Description

Property	Type	Description
addresses	**array**	List of addresses reachable to the node. Queried from cloud provider, if available. More info: https://kubernetes.io/docs/concepts/nodes/node/#addresses Note: This field is declared as mergeable, but the merge key is not sufficiently unique, which can cause data corruption when it is merged. Callers should instead use a full-replacement patch. See https://pr.k8s.io/79391 for an example. Consumers should assume that addresses can change during the lifetime of a Node. However, there are some exceptions where this may not be possible, such as Pods that inherit a Node's address in its own status or consumers of the downward API (status.hostIP).
addresses[]	**object**	NodeAddress contains information for the node's address.
allocatable	**object (Quantity)**	Allocatable represents the resources of a node that are available for scheduling. Defaults to Capacity.
capacity	**object (Quantity)**	Capacity represents the total resources of a node. More info: https://kubernetes.io/docs/reference/node/node-status/#capacity
conditions	**array**	Conditions is an array of current observed node conditions. More info: https://kubernetes.io/docs/concepts/nodes/node/#condition
conditions[]	**object**	NodeCondition contains condition information for a node.

Property	Type	Description
config	**object**	NodeConfigStatus describes the status of the config assigned by Node.Spec.ConfigSource.
daemonEndpoints	**object**	NodeDaemonEndpoints lists ports opened by daemons running on the Node.
features	**object**	NodeFeatures describes the set of features implemented by the CRI implementation. The features contained in the NodeFeatures should depend only on the cri implementation independent of runtime handlers.
images	**array**	List of container images on this node
images[]	**object**	Describe a container image
nodeInfo	**object**	NodeSystemInfo is a set of ids/uuids to uniquely identify the node.
phase	**string**	NodePhase is the recently observed lifecycle phase of the node. More info: https://kubernetes.io/docs/concepts/nodes/node/#phase The field is never populated, and now is deprecated. Possible enum values: - **"Pending"** means the node has been created/added by the system, but not configured. - **"Running"** means the node has been configured and has Kubernetes components running. - **"Terminated"** means the node has been removed from the cluster.
runtimeHandlers	**array**	The available runtime handlers.
runtimeHandlers[]	**object**	NodeRuntimeHandler is a set of runtime handler information.

Property	Type	Description
volumesAttached	**array**	List of volumes that are attached to the node.
volumesAttached[]	**object**	AttachedVolume describes a volume attached to a node
volumesInUse	**array (string)**	List of attachable volumes in use (mounted) by the node.

2.1.7. .status.addresses

Description

List of addresses reachable to the node. Queried from cloud provider, if available. More info: https://kubernetes.io/docs/concepts/nodes/node/#addresses Note: This field is declared as mergeable, but the merge key is not sufficiently unique, which can cause data corruption when it is merged. Callers should instead use a full-replacement patch. See https://pr.k8s.io/79391 for an example. Consumers should assume that addresses can change during the lifetime of a Node. However, there are some exceptions where this may not be possible, such as Pods that inherit a Node's address in its own status or consumers of the downward API (status.hostIP).

Type

array

2.1.8. .status.addresses[]

Description

NodeAddress contains information for the node's address.

Type

object

Required

- **type**
- **address**

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

2.1.9. .status.conditions

Description

Conditions is an array of current observed node conditions. More info: https://kubernetes.io/docs/concepts/nodes/node/#condition

Type

array

2.1.10. .status.conditions[]

Description

NodeCondition contains condition information for a node.

Type

object

Required

- **type**
- **status**

Property	Type	Description
lastHeartbeatTime	Time	Last time we got an update on a given condition.
lastTransitionTime	Time	Last time the condition transit from one status to another.
message	**string**	Human readable message indicating details about last transition.
reason	**string**	(brief) reason for the condition's last transition.
status	**string**	Status of the condition, one of True, False, Unknown.
type	**string**	Type of node condition.

2.1.11. .status.config

Description

NodeConfigStatus describes the status of the config assigned by Node.Spec.ConfigSource.

Type

object

Property	Type	Description
active	**object**	NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22
assigned	**object**	NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22
error	**string**	Error describes any problems reconciling the Spec.ConfigSource to the Active config. Errors may occur, for example, attempting to checkpoint Spec.ConfigSource to the local Assigned record, attempting to checkpoint the payload associated with Spec.ConfigSource, attempting to load or validate the Assigned config, etc. Errors may occur at different points while syncing config. Earlier errors (e.g. download or checkpointing errors) will not result in a rollback to LastKnownGood, and may resolve across Kubelet retries. Later errors (e.g. loading or validating a checkpointed config) will result in a rollback to LastKnownGood. In the latter case, it is usually possible to resolve the error by fixing the config assigned in Spec.ConfigSource. You can find additional information for debugging by searching the error message in the Kubelet log. Error is a human-readable description of the error state; machines can check whether or not Error is empty, but should not rely on the stability of the Error text across Kubelet versions.

Property	Type	Description
lastKnownGood	**object**	NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22

2.1.12. .status.config.active

Description

NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22

Type

object

Property	Type	Description
configMap	**object**	ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

2.1.13. .status.config.active.configMap

Description

ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

Type

object

Required

- **namespace**
- **name**
- **kubeletConfigKey**

Property	Type	Description

Property	Type	Description
kubeletConfigKey	**string**	KubeletConfigKey declares which key of the referenced ConfigMap corresponds to the KubeletConfiguration structure This field is required in all cases.
name	**string**	Name is the metadata.name of the referenced ConfigMap. This field is required in all cases.
namespace	**string**	Namespace is the metadata.namespace of the referenced ConfigMap. This field is required in all cases.
resourceVersion	**string**	ResourceVersion is the metadata.ResourceVersion of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.
uid	**string**	UID is the metadata.UID of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.

2.1.14. .status.config.assigned

Description

NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22

Type

object

Property	Type	Description
configMap	**object**	ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

2.1.15. .status.config.assigned.configMap

Description

ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

Type

object

Required

- **namespace**
- **name**
- **kubeletConfigKey**

Property	Type	Description
kubeletConfigKey	**string**	KubeletConfigKey declares which key of the referenced ConfigMap corresponds to the KubeletConfiguration structure This field is required in all cases.
name	**string**	Name is the metadata.name of the referenced ConfigMap. This field is required in all cases.
namespace	**string**	Namespace is the metadata.namespace of the referenced ConfigMap. This field is required in all cases.
resourceVersion	**string**	ResourceVersion is the metadata.ResourceVersion of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.
uid	**string**	UID is the metadata.UID of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.

2.1.16. .status.config.lastKnownGood

Description

NodeConfigSource specifies a source of node configuration. Exactly one subfield (excluding metadata) must be non-nil. This API is deprecated since 1.22

Type

object

Property	Type	Description
configMap	**object**	ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

2.1.17. .status.config.lastKnownGood.configMap

Description

ConfigMapNodeConfigSource contains the information to reference a ConfigMap as a config source for the Node. This API is deprecated since 1.22: https://git.k8s.io/enhancements/keps/sig-node/281-dynamic-kubelet-configuration

Type

object

Required

- **namespace**
- **name**
- **kubeletConfigKey**

Property	Type	Description
kubeletConfigKey	**string**	KubeletConfigKey declares which key of the referenced ConfigMap corresponds to the KubeletConfiguration structure This field is required in all cases.
name	**string**	Name is the metadata.name of the referenced ConfigMap. This field is required in all cases.
namespace	**string**	Namespace is the metadata.namespace of the referenced ConfigMap. This field is required in all cases.
resourceVersion	**string**	ResourceVersion is the metadata.ResourceVersion of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.

Property	Type	Description
uid	**string**	UID is the metadata.UID of the referenced ConfigMap. This field is forbidden in Node.Spec, and required in Node.Status.

2.1.18. .status.daemonEndpoints

Description

NodeDaemonEndpoints lists ports opened by daemons running on the Node.

Type

object

Property	Type	Description
kubeletEndpoint	**object**	DaemonEndpoint contains information about a single Daemon endpoint.

2.1.19. .status.daemonEndpoints.kubeletEndpoint

Description

DaemonEndpoint contains information about a single Daemon endpoint.

Type

object

Required

- **Port**

Property	Type	Description
Port	**integer**	Port number of the given endpoint.

2.1.20. .status.features

Description

NodeFeatures describes the set of features implemented by the CRI implementation. The features contained in the NodeFeatures should depend only on the cri implementation independent of runtime handlers.

Type

object

Property	Type	Description
supplementalGroupsPolicy	**boolean**	SupplementalGroupsPolicy is set to true if the runtime supports SupplementalGroupsPolicy and ContainerUser.

2.1.21. .status.images

Description

List of container images on this node

Type

array

2.1.22. .status.images[]

Description

Describe a container image

Type

object

Property	Type	Description
names	**array (string)**	Names by which this image is known. e.g. ["kubernetes.example/hyperkube:v1.0.7", "cloud-vendor.registry.example/cloud-vendor/hyperkube:v1.0.7"]
sizeBytes	**integer**	The size of the image in bytes.

2.1.23. .status.nodeInfo

Description

NodeSystemInfo is a set of ids/uuids to uniquely identify the node.

Type

object

Required

- **machineID**
- **systemUUID**
- **bootID**
- **kernelVersion**

- **osImage**
- **containerRuntimeVersion**
- **kubeletVersion**
- **kubeProxyVersion**
- **operatingSystem**
- **architecture**

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://1.4.2).
kernelVersion	**string**	Kernel Version reported by the node from 'uname -r' (e.g. 3.16.0-0.bpo.4-amd64).
kubeProxyVersion	**string**	Deprecated: KubeProxy Version reported by the node.
kubeletVersion	**string**	Kubelet Version reported by the node.
machineID	**string**	MachineID reported by the node. For unique machine identification in the cluster this field is preferred. Learn 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
osImage	**string**	OS Image reported by the node from /etc/os-release (e.g. Debian GNU/Linux 7 (wheezy)).

Property	Type	Description
systemUUID	**string**	SystemUUID reported by the node. For unique machine 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/rhsm/uuid

2.1.24. .status.runtimeHandlers

Description

The available runtime handlers.

Type

array

2.1.25. .status.runtimeHandlers[]

Description

NodeRuntimeHandler is a set of runtime handler information.

Type

object

Property	Type	Description
features	**object**	NodeRuntimeHandlerFeatures is a set of features implemented by the runtime handler.
name	**string**	Runtime handler name. Empty for the default runtime handler.

2.1.26. .status.runtimeHandlers[].features

Description

NodeRuntimeHandlerFeatures is a set of features implemented by the runtime handler.

Type

object

Property	Type	Description
recursiveReadOnlyMounts	**boolean**	RecursiveReadOnlyMounts is set to true if the runtime handler supports RecursiveReadOnlyMounts.
userNamespaces	**boolean**	UserNamespaces is set to true if the runtime handler supports UserNamespaces, including for volumes.

2.1.27. .status.volumesAttached

Description

List of volumes that are attached to the node.

Type

array

2.1.28. .status.volumesAttached[]

Description

AttachedVolume describes a volume attached to a node

Type

object

Required

- **name**
- **devicePath**

Property	Type	Description
devicePath	**string**	DevicePath represents the device path where the volume should be available
name	**string**	Name of the attached volume

2.2. API ENDPOINTS

The following API endpoints are available:

- **/api/v1/nodes**
  - **DELETE**: delete collection of Node

  - **GET**: list or watch objects of kind Node
  - **POST**: create a Node
- **/api/v1/watch/nodes**
  - **GET**: watch individual changes to a list of Node. deprecated: use the 'watch' parameter with a list operation instead.
- **/api/v1/nodes/{name}**
  - **DELETE**: delete a Node
  - **GET**: read the specified Node
  - **PATCH**: partially update the specified Node
  - **PUT**: replace the specified Node
- **/api/v1/watch/nodes/{name}**
  - **GET**: watch changes to an object of kind Node. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/api/v1/nodes/{name}/status**
  - **GET**: read status of the specified Node
  - **PATCH**: partially update status of the specified Node
  - **PUT**: replace status of the specified Node

2.2.1. /api/v1/nodes

HTTP method

DELETE

Description

delete collection of Node

Table 2.1. 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

Table 2.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind Node

Table 2.3. HTTP responses

HTTP code	Reponse body
200 - OK	**NodeList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Node

Table 2.4. 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.

Table 2.5. Body parameters

Parameter	Type	Description
body	Node schema	

Table 2.6. HTTP responses

HTTP code	Reponse body
200 - OK	Node schema
201 - Created	Node schema
202 - Accepted	Node schema
401 - Unauthorized	Empty

2.2.2. /api/v1/watch/nodes

HTTP method

GET

Description

watch individual changes to a list of Node. deprecated: use the 'watch' parameter with a list operation instead.

Table 2.7. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

2.2.3. /api/v1/nodes/{name}

Table 2.8. Global path parameters

Parameter	Type	Description
name	**string**	name of the Node

HTTP method

DELETE

Description

delete a Node

Table 2.9. 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

Table 2.10. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Node

Table 2.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Node** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Node

Table 2.12. 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.

Table 2.13. HTTP responses

HTTP code	Reponse body
200 - OK	**Node** schema
201 - Created	**Node** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Node

Table 2.14. 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.

Table 2.15. Body parameters

Parameter	Type	Description
body	Node schema	

Table 2.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Node** schema
201 - Created	**Node** schema
401 - Unauthorized	Empty

2.2.4. /api/v1/watch/nodes/{name}

Table 2.17. Global path parameters

Parameter	Type	Description
name	**string**	name of the Node

HTTP method

GET

Description

watch changes to an object of kind Node. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 2.18. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

2.2.5. /api/v1/nodes/{name}/status

Table 2.19. Global path parameters

Parameter	Type	Description
name	**string**	name of the Node

HTTP method

GET

Description

read status of the specified Node

Table 2.20. HTTP responses

HTTP code	Reponse body
200 - OK	**Node** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Node

Table 2.21. 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.

Table 2.22. HTTP responses

HTTP code	Reponse body
200 - OK	**Node** schema
201 - Created	**Node** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Node

Table 2.23. 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.

Table 2.24. Body parameters

Parameter	Type	Description
body	Node schema	

Table 2.25. HTTP responses

HTTP code	Reponse body
200 - OK	Node schema
201 - Created	Node schema
401 - Unauthorized	Empty

CHAPTER 3. PERFORMANCEPROFILE [PERFORMANCE.OPENSHIFT.IO/V2]

Description

PerformanceProfile is the Schema for the performanceprofiles API

Type

object

3.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	PerformanceProfileSpec defines the desired state of PerformanceProfile.
status	**object**	PerformanceProfileStatus defines the observed state of PerformanceProfile.

3.1.1. .spec

Description

PerformanceProfileSpec defines the desired state of PerformanceProfile.

Type

object

Required

- **cpu**
- **nodeSelector**

Property	Type	Description
additionalKernelArgs	**array (string)**	Additional kernel arguments.
cpu	**object**	CPU defines a set of CPU related parameters.
globallyDisableIrqLoadBalancing	**boolean**	GloballyDisableIrqLoadBalancing toggles whether IRQ load balancing will be disabled for the Isolated CPU set. When the option is set to "true" it disables IRQs load balancing for the Isolated CPU set. Setting the option to "false" allows the IRQs to be balanced across all CPUs, however the IRQs load balancing can be disabled per pod CPUs when using irq-load-balancing.crio.io/cpu-quota.crio.io annotations. Defaults to "false"
hardwareTuning	**object**	HardwareTuning defines a set of CPU frequencies for isolated and reserved cpus.

Property	Type	Description
hugepages	**object**	HugePages defines a set of huge pages related parameters. It is possible to set huge pages with multiple size values at the same time. For example, hugepages can be set with 1G and 2M, both values will be set on the node by the Performance Profile Controller. It is important to notice that setting hugepages default size to 1G will remove all 2M related folders from the node and it will be impossible to configure 2M hugepages under the node.
machineConfigLabel	**object (string)**	MachineConfigLabel defines the label to add to the MachineConfigs the operator creates. It has to be used in the MachineConfigSelector of the MachineConfigPool which targets this performance profile. Defaults to "machineconfiguration.openshift.io/role=<same role as in NodeSelector label key>"
machineConfigPoolSelector	**object (string)**	MachineConfigPoolSelector defines the MachineConfigPool label to use in the MachineConfigPoolSelector of resources like KubeletConfigs created by the operator. Defaults to "machineconfiguration.openshift.io/role=<same role as in NodeSelector label key>"
net	**object**	Net defines a set of network related features

Property	Type	Description
nodeSelector	**object (string)**	NodeSelector defines the Node label to use in the NodeSelectors of resources like Tuned created by the operator. It most likely should, but does not have to match the node label in the NodeSelector of the MachineConfigPool which targets this performance profile. In the case when machineConfigLabels or machineConfigPoolSelector are not set, we are expecting a certain NodeSelector format <domain>/<role>: "" in order to be able to calculate the default values for the former mentioned fields.
numa	**object**	NUMA defines options related to topology aware affinities
realTimeKernel	**object**	RealTimeKernel defines a set of real time kernel related parameters. RT kernel won't be installed when not set.
workloadHints	**object**	WorkloadHints defines hints for different types of workloads. It will allow defining exact set of tuned and kernel arguments that should be applied on top of the node.

3.1.2. .spec.cpu

Description

CPU defines a set of CPU related parameters.

Type

object

Required

- **isolated**
- **reserved**

Property	Type	Description
balanceIsolated	**boolean**	BalanceIsolated toggles whether or not the Isolated CPU set is eligible for load balancing work loads. When this option is set to "false", the Isolated CPU set will be static, meaning workloads have to explicitly assign each thread to a specific cpu in order to work across multiple CPUs. Setting this to "true" allows workloads to be balanced across CPUs. Setting this to "false" offers the most predictable performance for guaranteed workloads, but it offloads the complexity of cpu load balancing to the application. Defaults to "true"
isolated	**string**	Isolated defines a set of CPUs that will be used to give to application threads the most execution time possible, which means removing as many extraneous tasks off a CPU as possible. It is important to notice the CPU manager can choose any CPU to run the workload except the reserved CPUs. In order to guarantee that your workload will run on the isolated CPU: 1. The union of reserved CPUs and isolated CPUs should include all online CPUs 2. The isolated CPUs field should be the complementary to reserved CPUs field
offlined	**string**	Offline defines a set of CPUs that will be unused and set offline
reserved	**string**	Reserved defines a set of CPUs that will not be used for any container workloads initiated by kubelet.

Property	Type	Description
shared	**string**	Shared defines a set of CPUs that will be shared among guaranteed workloads that needs additional cpus which are not exclusive, alongside the isolated, exclusive resources that are being used already by those workloads.

3.1.3. .spec.hardwareTuning

Description

HardwareTuning defines a set of CPU frequencies for isolated and reserved cpus.

Type

object

Property	Type	Description
isolatedCpuFreq	**integer**	IsolatedCpuFreq defines a minimum frequency to be set across isolated cpus
reservedCpuFreq	**integer**	ReservedCpuFreq defines a maximum frequency to be set across reserved cpus

3.1.4. .spec.hugepages

Description

HugePages defines a set of huge pages related parameters. It is possible to set huge pages with multiple size values at the same time. For example, hugepages can be set with 1G and 2M, both values will be set on the node by the Performance Profile Controller. It is important to notice that setting hugepages default size to 1G will remove all 2M related folders from the node and it will be impossible to configure 2M hugepages under the node.

Type

object

Property	Type	Description
defaultHugepagesSize	**string**	DefaultHugePagesSize defines huge pages default size under kernel boot parameters.
pages	**array**	Pages defines huge pages that we want to allocate at boot time.

Property	Type	Description
pages[]	**object**	HugePage defines the number of allocated huge pages of the specific size.

3.1.5. .spec.hugepages.pages

Description

Pages defines huge pages that we want to allocate at boot time.

Type

array

3.1.6. .spec.hugepages.pages[]

Description

HugePage defines the number of allocated huge pages of the specific size.

Type

object

Property	Type	Description
count	**integer**	Count defines amount of huge pages, maps to the 'hugepages' kernel boot parameter.
node	**integer**	Node defines the NUMA node where hugepages will be allocated, if not specified, pages will be allocated equally between NUMA nodes
size	**string**	Size defines huge page size, maps to the 'hugepagesz' kernel boot parameter.

3.1.7. .spec.net

Description

Net defines a set of network related features

Type

object

Property	Type	Description
devices	**array**	Devices contains a list of network device representations that will be set with a netqueue count equal to CPU.Reserved . If no devices are specified then the default is all devices.
devices[]	**object**	Device defines a way to represent a network device in several options: device name, vendor ID, model ID, PCI path and MAC address
userLevelNetworking	**boolean**	UserLevelNetworking when enabled - sets either all or specified network devices queue size to the amount of reserved CPUs. Defaults to "false".

3.1.8. .spec.net.devices

Description

Devices contains a list of network device representations that will be set with a netqueue count equal to CPU.Reserved . If no devices are specified then the default is all devices.

Type

array

3.1.9. .spec.net.devices[]

Description

Device defines a way to represent a network device in several options: device name, vendor ID, model ID, PCI path and MAC address

Type

object

Property	Type	Description
deviceID	**string**	Network device ID (model) represnted as a 16 bit hexmadecimal number.
interfaceName	**string**	Network device name to be matched. It uses a syntax of shell-style wildcards which are either positive or negative.

Property	Type	Description
vendorID	**string**	Network device vendor ID represnted as a 16 bit Hexmadecimal number.

3.1.10. .spec.numa

Description

NUMA defines options related to topology aware affinities

Type

object

Property	Type	Description
topologyPolicy	**string**	Name of the policy applied when TopologyManager is enabled Operator defaults to "best-effort"

3.1.11. .spec.realTimeKernel

Description

RealTimeKernel defines a set of real time kernel related parameters. RT kernel won't be installed when not set.

Type

object

Property	Type	Description
enabled	**boolean**	Enabled defines if the real time kernel packages should be installed. Defaults to "false"

3.1.12. .spec.workloadHints

Description

WorkloadHints defines hints for different types of workloads. It will allow defining exact set of tuned and kernel arguments that should be applied on top of the node.

Type

object

Property	Type	Description

Property	Type	Description
highPowerConsumption	**boolean**	HighPowerConsumption defines if the node should be configured in high power consumption mode. The flag will affect the power consumption but will improve the CPUs latency. Defaults to false.
mixedCpus	**boolean**	MixedCpus enables the mixed-cpu-node-plugin on the node. Defaults to false.
perPodPowerManagement	**boolean**	PerPodPowerManagement defines if the node should be configured in per pod power management. PerPodPowerManagement and HighPowerConsumption hints can not be enabled together. Defaults to false.
realTime	**boolean**	RealTime defines if the node should be configured for the real time workload. Defaults to true.

3.1.13. .status

Description

PerformanceProfileStatus defines the observed state of PerformanceProfile.

Type

object

Property	Type	Description
conditions	**array**	Conditions represents the latest available observations of current state.
conditions[]	**object**	Condition represents the state of the operator's reconciliation functionality.
runtimeClass	**string**	RuntimeClass contains the name of the RuntimeClass resource created by the operator.

Property	Type	Description
tuned	**string**	Tuned points to the Tuned custom resource object that contains the tuning values generated by this operator.

3.1.14. .status.conditions

Description

Conditions represents the latest available observations of current state.

Type

array

3.1.15. .status.conditions[]

Description

Condition represents the state of the operator's reconciliation functionality.

Type

object

Required

- **status**
- **type**

Property	Type	Description
lastHeartbeatTime	**string**	
lastTransitionTime	**string**	
message	**string**	
reason	**string**	
status	**string**	
type	**string**	ConditionType is the state of the operator's reconciliation functionality.

3.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/performance.openshift.io/v2/performanceprofiles**
  - **DELETE**: delete collection of PerformanceProfile
  - **GET**: list objects of kind PerformanceProfile
  - **POST**: create a PerformanceProfile
- **/apis/performance.openshift.io/v2/performanceprofiles/{name}**
  - **DELETE**: delete a PerformanceProfile
  - **GET**: read the specified PerformanceProfile
  - **PATCH**: partially update the specified PerformanceProfile
  - **PUT**: replace the specified PerformanceProfile
- **/apis/performance.openshift.io/v2/performanceprofiles/{name}/status**
  - **GET**: read status of the specified PerformanceProfile
  - **PATCH**: partially update status of the specified PerformanceProfile
  - **PUT**: replace status of the specified PerformanceProfile

3.2.1. /apis/performance.openshift.io/v2/performanceprofiles

HTTP method

DELETE

Description

delete collection of PerformanceProfile

Table 3.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind PerformanceProfile

Table 3.2. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfileList** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

POST

Description

create a PerformanceProfile

Table 3.3. 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.

Table 3.4. Body parameters

Parameter	Type	Description
body	**PerformanceProfile** schema	

Table 3.5. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfile** schema
201 - Created	**PerformanceProfile** schema
202 - Accepted	**PerformanceProfile** schema
401 - Unauthorized	Empty

3.2.2. /apis/performance.openshift.io/v2/performanceprofiles/{name}

Table 3.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the PerformanceProfile

HTTP method

DELETE

Description

delete a PerformanceProfile

Table 3.7. 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

Table 3.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified PerformanceProfile

Table 3.9. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfile** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified PerformanceProfile

Table 3.10. 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.

Table 3.11. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfile** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified PerformanceProfile

Table 3.12. 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.

Table 3.13. Body parameters

Parameter	Type	Description
body	**PerformanceProfile** schema	

Table 3.14. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfile** schema
201 - Created	**PerformanceProfile** schema
401 - Unauthorized	Empty

3.2.3. /apis/performance.openshift.io/v2/performanceprofiles/{name}/status

Table 3.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the PerformanceProfile

HTTP method

GET

Description

read status of the specified PerformanceProfile

Table 3.16. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfile** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified PerformanceProfile

Table 3.17. 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.

Table 3.18. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfile** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified PerformanceProfile

Table 3.19. 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.

Table 3.20. Body parameters

Parameter	Type	Description
body	**PerformanceProfile** schema	

Table 3.21. HTTP responses

HTTP code	Reponse body
200 - OK	**PerformanceProfile** schema
201 - Created	**PerformanceProfile** schema
401 - Unauthorized	Empty

CHAPTER 4. PROFILE [TUNED.OPENSHIFT.IO/V1]

Description

Profile is a specification for a Profile resource.

Type

object

4.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	
status	**object**	ProfileStatus is the status for a Profile resource; the status is for internal use only and its fields may be changed/removed in the future.

4.1.1. .spec

Description

Type

object

Required

- **config**

Property	Type	Description
config	**object**	
profile	**array**	Tuned profiles.
profile[]	**object**	A Tuned profile.

4.1.2. .spec.config

Description

Type

object

Required

- **tunedProfile**

Property	Type	Description
debug	**boolean**	option to debug TuneD daemon execution
providerName	**string**	Name of the cloud provider as taken from the Node providerID: <ProviderName>://<ProviderSpecificNodeID>
tunedConfig	**object**	Global configuration for the TuneD daemon as defined in tuned-main.conf
tunedProfile	**string**	TuneD profile to apply
verbosity	**integer**	klog logging verbosity

4.1.3. .spec.config.tunedConfig

Description

Global configuration for the TuneD daemon as defined in tuned-main.conf

Type

object

Property	Type	Description
reapply_sysctl	**boolean**	turn reapply_sysctl functionality on/off for the TuneD daemon: true/false

4.1.4. .spec.profile

Description

Tuned profiles.

Type

array

4.1.5. .spec.profile[]

Description

A Tuned profile.

Type

object

Required

- **data**
- **name**

Property	Type	Description
data	**string**	Specification of the Tuned profile to be consumed by the Tuned daemon.
name	**string**	Name of the Tuned profile to be used in the recommend section.

4.1.6. .status

Description

ProfileStatus is the status for a Profile resource; the status is for internal use only and its fields may be changed/removed in the future.

Type

object

Required

- **tunedProfile**

Property	Type	Description
conditions	**array**	conditions represents the state of the per-node Profile application
conditions[]	**object**	ProfileStatusCondition represents a partial state of the per-node Profile application.
tunedProfile	**string**	the current profile in use by the Tuned daemon

4.1.7. .status.conditions

Description

conditions represents the state of the per-node Profile application

Type

array

4.1.8. .status.conditions[]

Description

ProfileStatusCondition represents a partial state of the per-node Profile application.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the time of the last update to the current status property.
message	**string**	message provides additional information about the current condition. This is only to be consumed by humans.

Property	Type	Description
reason	**string**	reason is the CamelCase reason for the condition’s current status.
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type specifies the aspect reported by this condition.

4.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/tuned.openshift.io/v1/profiles**
  - **GET**: list objects of kind Profile
- **/apis/tuned.openshift.io/v1/namespaces/{namespace}/profiles**
  - **DELETE**: delete collection of Profile
  - **GET**: list objects of kind Profile
  - **POST**: create a Profile
- **/apis/tuned.openshift.io/v1/namespaces/{namespace}/profiles/{name}**
  - **DELETE**: delete a Profile
  - **GET**: read the specified Profile
  - **PATCH**: partially update the specified Profile
  - **PUT**: replace the specified Profile
- **/apis/tuned.openshift.io/v1/namespaces/{namespace}/profiles/{name}/status**
  - **GET**: read status of the specified Profile
  - **PATCH**: partially update status of the specified Profile
  - **PUT**: replace status of the specified Profile

4.2.1. /apis/tuned.openshift.io/v1/profiles

HTTP method

GET

Description

list objects of kind Profile

Table 4.1. HTTP responses

HTTP code	Reponse body
200 - OK	**ProfileList** schema
401 - Unauthorized	Empty

4.2.2. /apis/tuned.openshift.io/v1/namespaces/{namespace}/profiles

HTTP method

DELETE

Description

delete collection of Profile

Table 4.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Profile

Table 4.3. HTTP responses

HTTP code	Reponse body
200 - OK	**ProfileList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Profile

Table 4.4. Query parameters

Parameter	Type	Description

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.

Table 4.5. Body parameters

Parameter	Type	Description
body	**Profile** schema	

Table 4.6. HTTP responses

HTTP code	Reponse body
200 - OK	**Profile** schema
201 - Created	**Profile** schema
202 - Accepted	**Profile** schema
401 - Unauthorized	Empty

4.2.3. /apis/tuned.openshift.io/v1/namespaces/{namespace}/profiles/{name}

Table 4.7. Global path parameters

Parameter	Type	Description
name	**string**	name of the Profile

HTTP method

DELETE

Description

delete a Profile

Table 4.8. 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

Table 4.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Profile

Table 4.10. HTTP responses

HTTP code	Reponse body
200 - OK	**Profile** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Profile

Table 4.11. 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.

Table 4.12. HTTP responses

HTTP code	Reponse body
200 - OK	**Profile** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Profile

Table 4.13. 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.

Table 4.14. Body parameters

Parameter	Type	Description
body	**Profile** schema	

Table 4.15. HTTP responses

HTTP code	Reponse body
200 - OK	**Profile** schema
201 - Created	**Profile** schema
401 - Unauthorized	Empty

4.2.4. /apis/tuned.openshift.io/v1/namespaces/{namespace}/profiles/{name}/status

Table 4.16. Global path parameters

Parameter	Type	Description
name	**string**	name of the Profile

HTTP method

GET

Description

read status of the specified Profile

Table 4.17. HTTP responses

HTTP code	Reponse body
200 - OK	**Profile** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Profile

Table 4.18. 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.

Table 4.19. HTTP responses

HTTP code	Reponse body
200 - OK	**Profile** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Profile

Table 4.20. 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.

Table 4.21. Body parameters

Parameter	Type	Description
body	**Profile** schema	

Table 4.22. HTTP responses

HTTP code	Reponse body
200 - OK	**Profile** schema
201 - Created	**Profile** schema
401 - Unauthorized	Empty

CHAPTER 5. RUNTIMECLASS [NODE.K8S.IO/V1]

Description

RuntimeClass defines a class of container runtime supported in the cluster. The RuntimeClass is used to determine which container runtime is used to run all containers in a pod. RuntimeClasses are manually defined by a user or cluster provisioner, and referenced in the PodSpec. The Kubelet is responsible for resolving the RuntimeClassName reference before running the pod. For more details, see https://kubernetes.io/docs/concepts/containers/runtime-class/

Type

object

Required

- **handler**

5.1. 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
handler	**string**	handler specifies the underlying runtime and configuration that the CRI implementation will use to handle pods of this class. The possible values are specific to the node & CRI configuration. It is assumed that all handlers are available on every node, and handlers of the same name are equivalent on every node. For example, a handler called "runc" might specify that the runc OCI runtime (using native Linux containers) will be used to run the containers in a pod. The Handler must be lowercase, conform to the DNS Label (RFC 1123) requirements, and is immutable.

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**	More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
overhead	**object**	Overhead structure represents the resource overhead associated with running a pod.
scheduling	**object**	Scheduling specifies the scheduling constraints for nodes supporting a RuntimeClass.

5.1.1. .overhead

Description

Overhead structure represents the resource overhead associated with running a pod.

Type

object

Property	Type	Description
podFixed	**object (Quantity)**	podFixed represents the fixed resource overhead associated with running a pod.

5.1.2. .scheduling

Description

Scheduling specifies the scheduling constraints for nodes supporting a RuntimeClass.

Type

object

Property	Type	Description
nodeSelector	**object (string)**	nodeSelector lists labels that must be present on nodes that support this RuntimeClass. Pods using this RuntimeClass can only be scheduled to a node matched by this selector. The RuntimeClass nodeSelector is merged with a pod's existing nodeSelector. Any conflicts will cause the pod to be rejected in admission.
tolerations	**array (Toleration)**	tolerations are appended (excluding duplicates) to pods running with this RuntimeClass during admission, effectively unioning the set of nodes tolerated by the pod and the RuntimeClass.

5.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/node.k8s.io/v1/runtimeclasses**
  - **DELETE**: delete collection of RuntimeClass
  - **GET**: list or watch objects of kind RuntimeClass
  - **POST**: create a RuntimeClass
- **/apis/node.k8s.io/v1/watch/runtimeclasses**
  - **GET**: watch individual changes to a list of RuntimeClass. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/node.k8s.io/v1/runtimeclasses/{name}**
  - **DELETE**: delete a RuntimeClass
  - **GET**: read the specified RuntimeClass
  - **PATCH**: partially update the specified RuntimeClass
  - **PUT**: replace the specified RuntimeClass
- **/apis/node.k8s.io/v1/watch/runtimeclasses/{name}**
  - **GET**: watch changes to an object of kind RuntimeClass. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

5.2.1. /apis/node.k8s.io/v1/runtimeclasses

HTTP method

DELETE

Description

delete collection of RuntimeClass

Table 5.1. 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

Table 5.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind RuntimeClass

Table 5.3. HTTP responses

HTTP code	Reponse body
200 - OK	**RuntimeClassList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a RuntimeClass

Table 5.4. 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.

Table 5.5. Body parameters

Parameter	Type	Description
body	**RuntimeClass** schema	

Table 5.6. HTTP responses

HTTP code	Reponse body
200 - OK	**RuntimeClass** schema
201 - Created	**RuntimeClass** schema
202 - Accepted	**RuntimeClass** schema
401 - Unauthorized	Empty

5.2.2. /apis/node.k8s.io/v1/watch/runtimeclasses

HTTP method

GET

Description

watch individual changes to a list of RuntimeClass. deprecated: use the 'watch' parameter with a list operation instead.

Table 5.7. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

5.2.3. /apis/node.k8s.io/v1/runtimeclasses/{name}

Table 5.8. Global path parameters

Parameter	Type	Description
name	**string**	name of the RuntimeClass

HTTP method

DELETE

Description

delete a RuntimeClass

Table 5.9. 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

Table 5.10. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified RuntimeClass

Table 5.11. HTTP responses

HTTP code	Reponse body
200 - OK	RuntimeClass schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified RuntimeClass

Table 5.12. 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.

Table 5.13. HTTP responses

HTTP code	Reponse body
200 - OK	**RuntimeClass** schema
201 - Created	**RuntimeClass** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified RuntimeClass

Table 5.14. 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.

Table 5.15. Body parameters

Parameter	Type	Description
body	**RuntimeClass** schema	

Table 5.16. HTTP responses

HTTP code	Reponse body
200 - OK	**RuntimeClass** schema
201 - Created	**RuntimeClass** schema
401 - Unauthorized	Empty

5.2.4. /apis/node.k8s.io/v1/watch/runtimeclasses/{name}

Table 5.17. Global path parameters

Parameter	Type	Description
name	**string**	name of the RuntimeClass

HTTP method

GET

Description

watch changes to an object of kind RuntimeClass. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 5.18. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

CHAPTER 6. TUNED [TUNED.OPENSHIFT.IO/V1]

Description

Tuned is a collection of rules that allows cluster-wide deployment of node-level sysctls and more flexibility to add custom tuning specified by user needs. These rules are translated and passed to all containerized Tuned daemons running in the cluster in the format that the daemons understand. The responsibility for applying the node-level tuning then lies with the containerized Tuned daemons. More info: https://github.com/openshift/cluster-node-tuning-operator

Type

object

6.1. 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
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 Tuned. More info: https://git.k8s.io/community/contributors/devel/api-conventions.md#spec-and-status

Property	Type	Description
status	**object**	TunedStatus is the status for a Tuned resource.

6.1.1. .spec

Description

spec is the specification of the desired behavior of Tuned. More info: https://git.k8s.io/community/contributors/devel/api-conventions.md#spec-and-status

Type

object

Property	Type	Description
managementState	**string**	managementState indicates whether the registry instance represented by this config instance is under operator management or not. Valid values are Force, Managed, Unmanaged, and Removed.
profile	**array**	Tuned profiles.
profile[]	**object**	A Tuned profile.
recommend	**array**	Selection logic for all Tuned profiles.
recommend[]	**object**	Selection logic for a single Tuned profile.

6.1.2. .spec.profile

Description

Tuned profiles.

Type

array

6.1.3. .spec.profile[]

Description

A Tuned profile.

Type

object

Required

- **data**
- **name**

Property	Type	Description
data	**string**	Specification of the Tuned profile to be consumed by the Tuned daemon.
name	**string**	Name of the Tuned profile to be used in the recommend section.

6.1.4. .spec.recommend

Description

Selection logic for all Tuned profiles.

Type

array

6.1.5. .spec.recommend[]

Description

Selection logic for a single Tuned profile.

Type

object

Required

- **priority**
- **profile**

Property	Type	Description

Property	Type	Description
machineConfigLabels	**object (string)**	MachineConfigLabels specifies the labels for a MachineConfig. The MachineConfig is created automatically to apply additional host settings (e.g. kernel boot parameters) profile 'Profile' needs and can only be applied by creating a MachineConfig. This involves finding all MachineConfigPools with machineConfigSelector matching the MachineConfigLabels and setting the profile 'Profile' on all nodes that match the MachineConfigPools' nodeSelectors.
match	**array**	Rules governing application of a Tuned profile connected by logical OR operator.
match[]	**object**	Rules governing application of a Tuned profile.
operand	**object**	Optional operand configuration.
priority	**integer**	Tuned profile priority. Highest priority is 0.
profile	**string**	Name of the Tuned profile to recommend.

6.1.6. .spec.recommend[].match

Description

Rules governing application of a Tuned profile connected by logical OR operator.

Type

array

6.1.7. .spec.recommend[].match[]

Description

Rules governing application of a Tuned profile.

Type

object

Required

- **label**

Property	Type	Description
label	**string**	Node or Pod label name.
match	**array (undefined)**	Additional rules governing application of the tuned profile connected by logical AND operator.
type	**string**	Match type: [node/pod]. If omitted, "node" is assumed.
value	**string**	Node or Pod label value. If omitted, the presence of label name is enough to match.

6.1.8. .spec.recommend[].operand

Description

Optional operand configuration.

Type

object

Property	Type	Description
debug	**boolean**	turn debugging on/off for the TuneD daemon: true/false (default is false)
tunedConfig	**object**	Global configuration for the TuneD daemon as defined in tuned-main.conf
verbosity	**integer**	klog logging verbosity

6.1.9. .spec.recommend[].operand.tunedConfig

Description

Global configuration for the TuneD daemon as defined in tuned-main.conf

Type

object

Property	Type	Description
reapply_sysctl	**boolean**	turn reapply_sysctl functionality on/off for the TuneD daemon: true/false

6.1.10. .status

Description

TunedStatus is the status for a Tuned resource.

Type

object

6.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/tuned.openshift.io/v1/tuneds**
  - **GET**: list objects of kind Tuned
- **/apis/tuned.openshift.io/v1/namespaces/{namespace}/tuneds**
  - **DELETE**: delete collection of Tuned
  - **GET**: list objects of kind Tuned
  - **POST**: create a Tuned
- **/apis/tuned.openshift.io/v1/namespaces/{namespace}/tuneds/{name}**
  - **DELETE**: delete a Tuned
  - **GET**: read the specified Tuned
  - **PATCH**: partially update the specified Tuned
  - **PUT**: replace the specified Tuned

6.2.1. /apis/tuned.openshift.io/v1/tuneds

HTTP method

GET

Description

list objects of kind Tuned

Table 6.1. HTTP responses

HTTP code	Reponse body
200 - OK	**TunedList** schema
401 - Unauthorized	Empty

6.2.2. /apis/tuned.openshift.io/v1/namespaces/{namespace}/tuneds

HTTP method

DELETE

Description

delete collection of Tuned

Table 6.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Tuned

Table 6.3. HTTP responses

HTTP code	Reponse body
200 - OK	**TunedList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Tuned

Table 6.4. Query parameters

Parameter	Type	Description

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.

Table 6.5. Body parameters

Parameter	Type	Description
body	**Tuned** schema	

Table 6.6. HTTP responses

HTTP code	Reponse body
200 - OK	**Tuned** schema
201 - Created	**Tuned** schema
202 - Accepted	**Tuned** schema
401 - Unauthorized	Empty

6.2.3. /apis/tuned.openshift.io/v1/namespaces/{namespace}/tuneds/{name}

Table 6.7. Global path parameters

Parameter	Type	Description
name	**string**	name of the Tuned

HTTP method

DELETE

Description

delete a Tuned

Table 6.8. 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

Table 6.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Tuned

Table 6.10. HTTP responses

HTTP code	Reponse body
200 - OK	**Tuned** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Tuned

Table 6.11. 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.

Table 6.12. HTTP responses

HTTP code	Reponse body
200 - OK	**Tuned** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Tuned

Table 6.13. Query parameters

Parameter	Type	Description

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.

Table 6.14. Body parameters

Parameter	Type	Description
body	**Tuned** schema	

Table 6.15. HTTP responses

HTTP code	Reponse body
200 - OK	**Tuned** schema
201 - Created	**Tuned** schema
401 - Unauthorized	Empty