
Audit Annotations

This page serves as a reference for the audit annotations of the `kubernetes.io` namespace. These annotations apply to `Event` object from API group `audit.k8s.io`.

Note:

The following annotations are not used within the Kubernetes API. When you [enable auditing](#) in your cluster, audit event data is written using `Event` from API group `audit.k8s.io`. The annotations apply to audit events. Audit events are different from objects in the [Event API](#) (API group `events.k8s.io`).

`k8s.io/deprecated`

Example: `k8s.io/deprecated: "true"`

Value **must** be "true" or "false". The value "true" indicates that the request used a deprecated API version.

`k8s.io/removed-release`

Example: `k8s.io/removed-release: "1.22"`

Value **must** be in the format "`<MAJOR>.<MINOR>`". It is set to target the removal release on requests made to deprecated API versions with a target removal release.

`pod-security.kubernetes.io/exempt`

Example: `pod-security.kubernetes.io/exempt: namespace`

Value **must** be one of `user`, `namespace`, or `runtimeClass` which correspond to [Pod Security Exemption](#) dimensions. This annotation indicates on which dimension was based the exemption from the PodSecurity enforcement.

`pod-security.kubernetes.io/enforce-policy`

Example: `pod-security.kubernetes.io/enforce-policy: restricted:latest`

Value **must** be `privileged:<version>`, `baseline:<version>`, `restricted:<version>` which correspond to [Pod Security Standard](#) levels accompanied by a version which **must** be `latest` or a valid Kubernetes version in the format `v<MAJOR>.<MINOR>`. This annotations informs about the enforcement level that allowed or denied the pod during PodSecurity admission.

See [Pod Security Standards](#) for more information.

`pod-security.kubernetes.io/audit-violations`

Example: `pod-security.kubernetes.io/audit-violations: would violate PodSecurity "restricted:latest": allowPrivilegeEscalation != false (container "example" must set securityContext.allowPrivilegeEscalation=false), ...`

Value details an audit policy violation, it contains the [Pod Security Standard](#) level that was transgressed as well as the specific policies on the fields that were violated from the PodSecurity enforcement.

See [Pod Security Standards](#) for more information.

`apiserver.latency.k8s.io/etcd`

Example: `apiserver.latency.k8s.io/etcd: "4.730661757s"`

This annotation indicates the measure of latency incurred inside the storage layer, it accounts for the time it takes to send data to the etcd and get the complete response back.

The value of this audit annotation does not include the time incurred in admission, or validation.

`apiserver.latency.k8s.io/decode-response-object`

Example: `apiserver.latency.k8s.io/decode-response-object: "450.6649ns"`

This annotation records the time taken to decode the response received from the storage layer (etcd)

`apiserver.latency.k8s.io/apf-queue-wait`

Example: `apiserver.latency.k8s.io/apf-queue-wait: "100ns"`

This annotation records the time that a request spent queued due to API server priorities.

See [API Priority and Fairness](#) (APF) for more information about this mechanism.

authorization.k8s.io/decision

Example: `authorization.k8s.io/decision: "forbid"`

Value must be **forbid** or **allow**. This annotation indicates whether or not a request was authorized in Kubernetes audit logs.

See [Auditing](#) for more information.

authorization.k8s.io/reason

Example: `authorization.k8s.io/reason: "Human-readable reason for the decision"`

This annotation gives reason for the [decision](#) in Kubernetes audit logs.

See [Auditing](#) for more information.

missing-san.invalid-cert.kubernetes.io/\$hostname

Example: `missing-san.invalid-cert.kubernetes.io/example-svc.example-namespace.svc: "relies on a legacy Common Name field instead of the SAN extension for subject validation"`

Used by Kubernetes version v1.24 and later

This annotation indicates a webhook or aggregated API server is using an invalid certificate that is missing `subjectAltNames`. Support for these certificates was disabled by default in Kubernetes 1.19, and removed in Kubernetes 1.23.

Requests to endpoints using these certificates will fail. Services using these certificates should replace them as soon as possible to avoid disruption when running in Kubernetes 1.23+ environments.

There's more information about this in the Go documentation: [X.509 CommonName deprecation](#).

insecure-sha1.invalid-cert.kubernetes.io/\$hostname

Example: `insecure-sha1.invalid-cert.kubernetes.io/example-svc.example-namespace.svc: "uses an insecure SHA-1 signature"`

Used by Kubernetes version v1.24 and later

This annotation indicates a webhook or aggregated API server is using an insecure certificate signed with a SHA-1 hash. Support for these insecure certificates is disabled by default in Kubernetes 1.24, and will be removed in a future release.

Services using these certificates should replace them as soon as possible, to ensure connections are secured properly and to avoid disruption in future releases.

There's more information about this in the Go documentation: [Rejecting SHA-1 certificates](#).

validation.policy.admission.k8s.io/validation_failure

Example: `validation.policy.admission.k8s.io/validation_failure: '{"message": "Invalid value", {"policy": "policy.example.com", {"binding": "policybinding.example.com", {"expressionIndex": "1", {"validationActions": ["Audit"]}]}'`

Used by Kubernetes version v1.27 and later.

This annotation indicates that a admission policy validation evaluated to false for an API request, or that the validation resulted in an error while the policy was configured with `failurePolicy: Fail`.

The value of the annotation is a JSON object. The `message` in the JSON provides the message about the validation failure.

The `policy`, `binding` and `expressionIndex` in the JSON identifies the name of the `ValidatingAdmissionPolicy`, the name of the `ValidatingAdmissionPolicyBinding` and the index in the policy validations of the CEL expressions that failed, respectively.

The `validationActions` shows what actions were taken for this validation failure. See [Validating Admission Policy](#) for more details about `validationActions`.

Well-Known Labels, Annotations and Taints

Kubernetes reserves all labels, annotations and taints in the `kubernetes.io` and `k8s.io` namespaces.

This document serves both as a reference to the values and as a coordination point for assigning values.

Labels, annotations and taints used on API objects

apf.kubernetes.io/autoupdate-spec

Type: Annotation

Example: `apf.kubernetes.io/autoupdate-spec: "true"`

Used on: [FlowSchema and PriorityLevelConfiguration Objects](#)

If this annotation is set to true on a FlowSchema or PriorityLevelConfiguration, the `spec` for that object is managed by the kube-apiserver. If the API server does not recognize an APF object, and you annotate it for automatic update, the API server deletes the entire object. Otherwise, the API server does not manage the object spec. For more details, read [Maintenance of the Mandatory and Suggested Configuration Objects](#).

app.kubernetes.io/component

Type: Label

Example: `app.kubernetes.io/component: "database"`

Used on: All Objects (typically used on [workload resources](#)).

The component within the application architecture.

One of the [recommended labels](#).

app.kubernetes.io/created-by (deprecated)

Type: Label

Example: `app.kubernetes.io/created-by: "controller-manager"`

Used on: All Objects (typically used on [workload resources](#)).

The controller/user who created this resource.

Note:

Starting from v1.9, this label is deprecated.

app.kubernetes.io/instance

Type: Label

Example: `app.kubernetes.io/instance: "mysql-abcxyz"`

Used on: All Objects (typically used on [workload resources](#)).

A unique name identifying the instance of an application. To assign a non-unique name, use [app.kubernetes.io/name](#).

One of the [recommended labels](#).

app.kubernetes.io/managed-by

Type: Label

Example: `app.kubernetes.io/managed-by: "helm"`

Used on: All Objects (typically used on [workload resources](#)).

The tool being used to manage the operation of an application.

One of the [recommended labels](#).

app.kubernetes.io/name

Type: Label

Example: `app.kubernetes.io/name: "mysql"`

Used on: All Objects (typically used on [workload resources](#)).

The name of the application.

One of the [recommended labels](#).

app.kubernetes.io/part-of

Type: Label

Example: `app.kubernetes.io/part-of: "wordpress"`

Used on: All Objects (typically used on [workload resources](#)).

The name of a higher-level application this object is part of.

One of the [recommended labels](#).

app.kubernetes.io/version

Type: Label

Example: `app.kubernetes.io/version: "5.7.21"`

Used on: All Objects (typically used on [workload resources](#)).

The current version of the application.

Common forms of values include:

- [semantic version](#)
- the Git [revision hash](#) for the source code.

One of the [recommended labels](#).

applyset.kubernetes.io/additional-namespaces (alpha)

Type: Annotation

Example: `applyset.kubernetes.io/additional-namespaces: "namespace1,namespace2"`

Used on: Objects being used as ApplySet parents.

Use of this annotation is Alpha. For Kubernetes version 1.34, you can use this annotation on Secrets, ConfigMaps, or custom resources if the [CustomResourceDefinition](#) defining them has the `applyset.kubernetes.io/is-parent-type` label.

Part of the specification used to implement [ApplySet-based pruning in kubectl](#). This annotation is applied to the parent object used to track an ApplySet to extend the scope of the ApplySet beyond the parent object's own namespace (if any). The value is a comma-separated list of the names of namespaces other than the parent's namespace in which objects are found.

applyset.kubernetes.io/contains-group-kinds (alpha)

Type: Annotation

Example: `applyset.kubernetes.io/contains-group-kinds: "certificates.cert-manager.io,configmaps,deployments.apps,secrets,services"`

Used on: Objects being used as ApplySet parents.

Use of this annotation is Alpha. For Kubernetes version 1.34, you can use this annotation on Secrets, ConfigMaps, or custom resources if the [CustomResourceDefinition](#) defining them has the `applyset.kubernetes.io/is-parent-type` label.

Part of the specification used to implement [ApplySet-based pruning in kubectl](#). This annotation is applied to the parent object used to track an ApplySet to optimize listing of ApplySet member objects. It is optional in the ApplySet specification, as tools can perform discovery or use a different optimization. However, as of Kubernetes version 1.34, it is required by kubectl. When present, the value of this annotation must be a comma separated list of the group-kinds, in the fully-qualified name format, i.e. `<resource>.<group>`.

applyset.kubernetes.io/contains-group-resources (deprecated)

Type: Annotation

Example: `applyset.kubernetes.io/contains-group-resources: "certificates.cert-manager.io,configmaps,deployments.apps,secrets,services"`

Used on: Objects being used as ApplySet parents.

For Kubernetes version 1.34, you can use this annotation on Secrets, ConfigMaps, or custom resources if the [CustomResourceDefinition](#) defining them has the `applyset.kubernetes.io/is-parent-type` label.

Part of the specification used to implement [ApplySet-based pruning in kubectl](#). This annotation is applied to the parent object used to track an ApplySet to optimize listing of ApplySet member objects. It is optional in the ApplySet specification, as tools can perform discovery or use a different optimization. However, in Kubernetes version 1.34, it is required by kubectl. When present, the value of this annotation must be a comma separated list of the group-kinds, in the fully-qualified name format, i.e. `<resource>.<group>`.

Note:

This annotation is currently deprecated and replaced by [applyset.kubernetes.io/contains-group-kinds](#), support for this will be removed in applyset beta or GA.

applyset.kubernetes.io/id (alpha)

Type: Label

Example: `applyset.kubernetes.io/id: "applyset-0eFHV8ySqP7XoShsGvyWfQD3s96yqWHzc4e0HR1dsY-v1"`

Used on: Objects being used as ApplySet parents.

Use of this label is Alpha. For Kubernetes version 1.34, you can use this label on Secrets, ConfigMaps, or custom resources if the [CustomResourceDefinition](#) defining them has the `applyset.kubernetes.io/is-parent-type` label.

Part of the specification used to implement [ApplySet-based pruning in kubectl](#). This label is what makes an object an ApplySet parent object. Its value is the unique ID of the ApplySet, which is derived from the identity of the parent object itself. This ID **must** be the base64 encoding (using the URL safe encoding

of RFC4648) of the hash of the group-kind-name-namespace of the object it is on, in the form: `<base64(sha256(<name>.<namespace>.<kind>.<group>))>`. There is no relation between the value of this label and object UID.

applyset.kubernetes.io/is-parent-type (alpha)

Type: Label

Example: `applyset.kubernetes.io/is-parent-type: "true"`

Used on: Custom Resource Definition (CRD)

Use of this label is Alpha. Part of the specification used to implement [ApplySet-based pruning in kubectrl](#). You can set this label on a CustomResourceDefinition (CRD) to identify the custom resource type it defines (not the CRD itself) as an allowed parent for an ApplySet. The only permitted value for this label is `"true"`; if you want to mark a CRD as not being a valid parent for ApplySets, omit this label.

applyset.kubernetes.io/part-of (alpha)

Type: Label

Example: `applyset.kubernetes.io/part-of: "applyset-0eFHV8ySqP7XoShsGvyWfQD3s96yqwHmzc4e0HR1dsY-v1"`

Used on: All objects.

Use of this label is Alpha. Part of the specification used to implement [ApplySet-based pruning in kubectrl](#). This label is what makes an object a member of an ApplySet. The value of the label **must** match the value of the `applyset.kubernetes.io/id` label on the parent object.

applyset.kubernetes.io/tooling (alpha)

Type: Annotation

Example: `applyset.kubernetes.io/tooling: "kubectrl/v1.34"`

Used on: Objects being used as ApplySet parents.

Use of this annotation is Alpha. For Kubernetes version 1.34, you can use this annotation on Secrets, ConfigMaps, or custom resources if the CustomResourceDefinition defining them has the `applyset.kubernetes.io/is-parent-type` label.

Part of the specification used to implement [ApplySet-based pruning in kubectrl](#). This annotation is applied to the parent object used to track an ApplySet to indicate which tooling manages that ApplySet. Tooling should refuse to mutate ApplySets belonging to other tools. The value must be in the format `<toolname>/<semver>`.

apps.kubernetes.io/pod-index (beta)

Type: Label

Example: `apps.kubernetes.io/pod-index: "0"`

Used on: Pod

When a StatefulSet controller creates a Pod for the StatefulSet, it sets this label on that Pod. The value of the label is the ordinal index of the pod being created.

See [Pod Index Label](#) in the StatefulSet topic for more details. Note the [PodIndexLabel](#) feature gate must be enabled for this label to be added to pods.

resource.kubernetes.io/pod-claim-name

Type: Annotation

Example: `resource.kubernetes.io/pod-claim-name: "my-pod-claim"`

Used on: ResourceClaim

This annotation is assigned to generated ResourceClaims. Its value corresponds to the name of the resource claim in the `.spec` of any Pod(s) for which the ResourceClaim was created. This annotation is an internal implementation detail of [dynamic resource allocation](#). You should not need to read or modify the value of this annotation.

cluster-autoscaler.kubernetes.io/safe-to-evict

Type: Annotation

Example: `cluster-autoscaler.kubernetes.io/safe-to-evict: "true"`

Used on: Pod

When this annotation is set to `"true"`, the cluster autoscaler is allowed to evict a Pod even if other rules would normally prevent that. The cluster autoscaler never evicts Pods that have this annotation explicitly set to `"false"`; you could set that on an important Pod that you want to keep running. If this annotation is not set then the cluster autoscaler follows its Pod-level behavior.

config.kubernetes.io/local-config

Type: Annotation

Example: `config.kubernetes.io/local-config: "true"`

Used on: All objects

This annotation is used in manifests to mark an object as local configuration that should not be submitted to the Kubernetes API.

A value of `"true"` for this annotation declares that the object is only consumed by client-side tooling and should not be submitted to the API server.

A value of `"false"` can be used to declare that the object should be submitted to the API server even when it would otherwise be assumed to be local.

This annotation is part of the Kubernetes Resource Model (KRM) Functions Specification, which is used by Kustomize and similar third-party tools. For example, Kustomize removes objects with this annotation from its final build output.

container.apparmor.security.beta.kubernetes.io/* (deprecated)

Type: Annotation

Example: `container.apparmor.security.beta.kubernetes.io/my-container: my-custom-profile`

Used on: Pods

This annotation allows you to specify the AppArmor security profile for a container within a Kubernetes pod. As of Kubernetes v1.30, this should be set with the `appArmorProfile` field instead. To learn more, see the [AppArmor](#) tutorial. The tutorial illustrates using AppArmor to restrict a container's abilities and access.

The profile specified dictates the set of rules and restrictions that the containerized process must adhere to. This helps enforce security policies and isolation for your containers.

internal.config.kubernetes.io/* (reserved prefix)

Type: Annotation

Used on: All objects

This prefix is reserved for internal use by tools that act as orchestrators in accordance with the Kubernetes Resource Model (KRM) Functions Specification. Annotations with this prefix are internal to the orchestration process and are not persisted to the manifests on the filesystem. In other words, the orchestrator tool should set these annotations when reading files from the local filesystem and remove them when writing the output of functions back to the filesystem.

A KRM function **must not** modify annotations with this prefix, unless otherwise specified for a given annotation. This enables orchestrator tools to add additional internal annotations, without requiring changes to existing functions.

internal.config.kubernetes.io/path

Type: Annotation

Example: `internal.config.kubernetes.io/path: "relative/file/path.yaml"`

Used on: All objects

This annotation records the slash-delimited, OS-agnostic, relative path to the manifest file the object was loaded from. The path is relative to a fixed location on the filesystem, determined by the orchestrator tool.

This annotation is part of the Kubernetes Resource Model (KRM) Functions Specification, which is used by Kustomize and similar third-party tools.

A KRM Function **should not** modify this annotation on input objects unless it is modifying the referenced files. A KRM Function **may** include this annotation on objects it generates.

internal.config.kubernetes.io/index

Type: Annotation

Example: `internal.config.kubernetes.io/index: "2"`

Used on: All objects

This annotation records the zero-indexed position of the YAML document that contains the object within the manifest file the object was loaded from. Note that YAML documents are separated by three dashes (`---`) and can each contain one object. When this annotation is not specified, a value of 0 is implied.

This annotation is part of the Kubernetes Resource Model (KRM) Functions Specification, which is used by Kustomize and similar third-party tools.

A KRM Function **should not** modify this annotation on input objects unless it is modifying the referenced files. A KRM Function **may** include this annotation on objects it generates.

kube-scheduler-simulator.sigs.k8s.io/bind-result

Type: Annotation

Example: `kube-scheduler-simulator.sigs.k8s.io/bind-result: '{"DefaultBinder":"success"}'`

Used on: Pod

This annotation records the result of bind scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/filter-result

Type: Annotation

Example:

```
kube-scheduler-simulator.sigs.k8s.io/filter-result: >-  
  {"node-282x7":{"AzureDiskLimits":"passed","EBSLimits":"passed","GCEPDLimits":"passed","InterPodAffinity":"passed","NodeAffin.
```

Used on: Pod

This annotation records the result of filter scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/finalscore-result

Type: Annotation

Example:

```
kube-scheduler-simulator.sigs.k8s.io/finalscore-result: >-  
  {"node-282x7":{"ImageLocality":"0","InterPodAffinity":"0","NodeAffinity":"0","NodeNumber":"0","NodeResourcesBalancedAllocati
```

Used on: Pod

This annotation records the final scores that the scheduler calculates from the scores from score scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/permit-result

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/permit-result: '{"CustomPermitPlugin":"success"}'

Used on: Pod

This annotation records the result of permit scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/permit-result-timeout

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/permit-result-timeout: '{"CustomPermitPlugin":"10s"}'

Used on: Pod

This annotation records the timeouts returned from permit scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/postfilter-result

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/postfilter-result: '{"DefaultPreemption":"success"}'

Used on: Pod

This annotation records the result of postfilter scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/prebind-result

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/prebind-result: '{"VolumeBinding":"success"}'

Used on: Pod

This annotation records the result of prebind scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/prefilter-result

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/prebind-result: '{"NodeAffinity":["\node-a"]}'

Used on: Pod

This annotation records the PreFilter result of prefilter scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/prefilter-result-status

Type: Annotation

Example:

```
kube-scheduler-simulator.sigs.k8s.io/prefilter-result-status: >-  
  {"InterPodAffinity": "success", "NodeAffinity": "success", "NodePorts": "success", "NodeResourcesFit": "success", "PodTopologySpread
```

Used on: Pod

This annotation records the result of prefilter scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/prescore-result

Type: Annotation

Example:

```
kube-scheduler-simulator.sigs.k8s.io/prescore-result: >-  
  {"InterPodAffinity": "success", "NodeAffinity": "success", "NodeNumber": "success", "PodTopologySpread": "success", "TaintToleration
```

Used on: Pod

This annotation records the result of prefilter scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/reserve-result

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/reserve-result: '{"VolumeBinding": "success"}'

Used on: Pod

This annotation records the result of reserve scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/result-history

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/result-history: '[]'

Used on: Pod

This annotation records all the past scheduling results from scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/score-result

Type: Annotation

```
kube-scheduler-simulator.sigs.k8s.io/score-result: >-  
  {"node-282x7": {"ImageLocality": "0", "InterPodAffinity": "0", "NodeAffinity": "0", "NodeNumber": "0", "NodeResourcesBalancedAllocati
```

Used on: Pod

This annotation records the result of score scheduler plugins, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kube-scheduler-simulator.sigs.k8s.io/selected-node

Type: Annotation

Example: kube-scheduler-simulator.sigs.k8s.io/selected-node: node-282x7

Used on: Pod

This annotation records the node that is selected by the scheduling cycle, used by <https://sigs.k8s.io/kube-scheduler-simulator>.

kubernetes.io/arch

Type: Label

Example: kubernetes.io/arch: "amd64"

Used on: Node

The Kubelet populates this with `runtime.GOARCH` as defined by Go. This can be handy if you are mixing ARM and x86 nodes.

kubernetes.io/os

Type: Label

Example: kubernetes.io/os: "linux"

Used on: Node, Pod

For nodes, the kubelet populates this with `runtime.GOOS` as defined by Go. This can be handy if you are mixing operating systems in your cluster (for example: mixing Linux and Windows nodes).

You can also set this label on a Pod. Kubernetes allows you to set any value for this label; if you use this label, you should nevertheless set it to the `runtime.GOOS` string for the operating system that this Pod actually works with.

When the `kubernetes.io/os` label value for a Pod does not match the label value on a Node, the kubelet on the node will not admit the Pod. However, this is not taken into account by the kube-scheduler. Alternatively, the kubelet refuses to run a Pod where you have specified a Pod OS, if this isn't the same as the operating system for the node where that kubelet is running. Just look for [Pods OS](#) for more details.

kubernetes.io/metadata.name

Type: Label

Example: `kubernetes.io/metadata.name: "mynamespace"`

Used on: Namespaces

The Kubernetes API server (part of the [control plane](#)) sets this label on all namespaces. The label value is set to the name of the namespace. You can't change this label's value.

This is useful if you want to target a specific namespace with a label [selector](#).

kubernetes.io/limit-ranger

Type: Annotation

Example: `kubernetes.io/limit-ranger: "LimitRanger plugin set: cpu, memory request for container nginx; cpu, memory limit for container nginx"`

Used on: Pod

Kubernetes by default doesn't provide any resource limit, that means unless you explicitly define limits, your container can consume unlimited CPU and memory. You can define a default request or default limit for pods. You do this by creating a `LimitRange` in the relevant namespace. Pods deployed after you define a `LimitRange` will have these limits applied to them. The annotation `kubernetes.io/limit-ranger` records that resource defaults were specified for the Pod, and they were applied successfully. For more details, read about [LimitRanges](#).

kubernetes.io/config.hash

Type: Annotation

Example: `kubernetes.io/config.hash: "df7cc47f8477b6b1226d7d23a904867b"`

Used on: Pod

When the kubelet creates a static Pod based on a given manifest, it attaches this annotation to the static Pod. The value of the annotation is the UID of the Pod. Note that the kubelet also sets the `.spec.nodeName` to the current node name as if the Pod was scheduled to the node.

kubernetes.io/config.mirror

Type: Annotation

Example: `kubernetes.io/config.mirror: "df7cc47f8477b6b1226d7d23a904867b"`

Used on: Pod

For a static Pod created by the kubelet on a node, a [mirror Pod](#) is created on the API server. The kubelet adds an annotation to indicate that this Pod is actually a mirror Pod. The annotation value is copied from the [kubernetes.io/config.hash](#) annotation, which is the UID of the Pod.

When updating a Pod with this annotation set, the annotation cannot be changed or removed. If a Pod doesn't have this annotation, it cannot be added during a Pod update.

kubernetes.io/config.source

Type: Annotation

Example: `kubernetes.io/config.source: "file"`

Used on: Pod

This annotation is added by the kubelet to indicate where the Pod comes from. For static Pods, the annotation value could be one of `file` or `http` depending on where the Pod manifest is located. For a Pod created on the API server and then scheduled to the current node, the annotation value is `api`.

kubernetes.io/config.seen

Type: Annotation

Example: `kubernetes.io/config.seen: "2023-10-27T04:04:56.011314488Z"`

Used on: Pod

When the kubelet sees a Pod for the first time, it may add this annotation to the Pod with a value of current timestamp in the RFC3339 format.

addonmanager.kubernetes.io/mode

Type: Label

Example: `addonmanager.kubernetes.io/mode: "Reconcile"`

Used on: All objects

To specify how an add-on should be managed, you can use the `addonmanager.kubernetes.io/mode` label. This label can have one of three values: `Reconcile`, `EnsureExists`, or `Ignore`.

- **Reconcile:** Addon resources will be periodically reconciled with the expected state. If there are any differences, the add-on manager will recreate, reconfigure or delete the resources as needed. This is the default mode if no label is specified.
- **EnsureExists:** Addon resources will be checked for existence only but will not be modified after creation. The add-on manager will create or re-create the resources when there is no instance of the resource with that name.
- **Ignore:** Addon resources will be ignored. This mode is useful for add-ons that are not compatible with the add-on manager or that are managed by another controller.

For more details, see [Addon-manager](#).

beta.kubernetes.io/arch (deprecated)

Type: Label

This label has been deprecated. Please use [kubernetes.io/arch](#) instead.

beta.kubernetes.io/os (deprecated)

Type: Label

This label has been deprecated. Please use [kubernetes.io/os](#) instead.

kube-aggregator.kubernetes.io/automanaged

Type: Label

Example: `kube-aggregator.kubernetes.io/automanaged: "onstart"`

Used on: APIService

The `kube-apiserver` sets this label on any APIService object that the API server has created automatically. The label marks how the control plane should manage that APIService. You should not add, modify, or remove this label by yourself.

Note:

Automanaged APIService objects are deleted by `kube-apiserver` when it has no built-in or custom resource API corresponding to the API group/version of the APIService.

There are two possible values:

- **onstart:** The APIService should be reconciled when an API server starts up, but not otherwise.
- **true:** The API server should reconcile this APIService continuously.

service.alpha.kubernetes.io/tolerate-unready-endpoints (deprecated)

Type: Annotation

Used on: StatefulSet

This annotation on a Service denotes if the Endpoints controller should go ahead and create Endpoints for unready Pods. Endpoints of these Services retain their DNS records and continue receiving traffic for the Service from the moment the kubelet starts all containers in the pod and marks it *Running*, til the kubelet stops all containers and deletes the pod from the API server.

autoscaling.alpha.kubernetes.io/behavior (deprecated)

Type: Annotation

Used on: HorizontalPodAutoscaler

This annotation was used to configure the scaling behavior for a HorizontalPodAutoscaler (HPA) in earlier Kubernetes versions. It allowed you to specify how the HPA should scale pods up or down, including setting stabilization windows and scaling policies. Setting this annotation has no effect in any supported release of Kubernetes.

kubernetes.io/hostname

Type: Label

Example: `kubernetes.io/hostname: "ip-172-20-114-199.ec2.internal"`

Used on: Node

The Kubelet populates this label with the hostname of the node. Note that the hostname can be changed from the "actual" hostname by passing the `--hostname-override` flag to the kubelet.

This label is also used as part of the topology hierarchy. See [topology.kubernetes.io/zone](https://kubernetes.io/docs/concepts/scheduling-eviction/topology.html) for more information.

kubernetes.io/change-cause

Type: Annotation

Example: `kubernetes.io/change-cause: "kubectl edit --record deployment foo"`

Used on: All Objects

This annotation is a best guess at why something was changed.

It is populated when adding `--record` to a `kubectl` command that may change an object.

kubernetes.io/description

Type: Annotation

Example: `kubernetes.io/description: "Description of K8s object."`

Used on: All Objects

This annotation is used for describing specific behaviour of given object.

kubernetes.io/enforce-mountable-secrets (deprecated)

Type: Annotation

Example: `kubernetes.io/enforce-mountable-secrets: "true"`

Used on: ServiceAccount

Note:

`kubernetes.io/enforce-mountable-secrets` is deprecated since Kubernetes v1.32. Use separate namespaces to isolate access to mounted secrets.

The value for this annotation must be **true** to take effect. When you set this annotation to "true", Kubernetes enforces the following rules for Pods running as this ServiceAccount:

1. Secrets mounted as volumes must be listed in the ServiceAccount's `secrets` field.
2. Secrets referenced in `envFrom` for containers (including sidecar containers and init containers) must also be listed in the ServiceAccount's `secrets` field. If any container in a Pod references a Secret not listed in the ServiceAccount's `secrets` field (and even if the reference is marked as `optional`), then the Pod will fail to start, and an error indicating the non-compliant secret reference will be generated.
3. Secrets referenced in a Pod's `imagePullSecrets` must be present in the ServiceAccount's `imagePullSecrets` field, the Pod will fail to start, and an error indicating the non-compliant image pull secret reference will be generated.

When you create or update a Pod, these rules are checked. If a Pod doesn't follow them, it won't start and you'll see an error message. If a Pod is already running and you change the `kubernetes.io/enforce-mountable-secrets` annotation to `true`, or you edit the associated ServiceAccount to remove the reference to a Secret that the Pod is already using, the Pod continues to run.

node.kubernetes.io/exclude-from-external-load-balancers

Type: Label

Example: `node.kubernetes.io/exclude-from-external-load-balancers`

Used on: Node

You can add labels to particular worker nodes to exclude them from the list of backend servers used by external load balancers. The following command can be used to exclude a worker node from the list of backend servers in a backend set:

```
kubectl label nodes <node-name> node.kubernetes.io/exclude-from-external-load-balancers=true
```

controller.kubernetes.io/pod-deletion-cost

Type: Annotation

Example: `controller.kubernetes.io/pod-deletion-cost: "10"`

Used on: Pod

This annotation is used to set [Pod Deletion Cost](#) which allows users to influence ReplicaSet downscaling order. The annotation value parses into an `int32` type.

cluster-autoscaler.kubernetes.io/enable-ds-eviction

Type: Annotation

Example: `cluster-autoscaler.kubernetes.io/enable-ds-eviction: "true"`

Used on: Pod

This annotation controls whether a DaemonSet pod should be evicted by a ClusterAutoscaler. This annotation needs to be specified on DaemonSet pods in a DaemonSet manifest. When this annotation is set to `"true"`, the ClusterAutoscaler is allowed to evict a DaemonSet Pod, even if other rules would normally prevent that. To disallow the ClusterAutoscaler from evicting DaemonSet pods, you can set this annotation to `"false"` for important DaemonSet pods. If this annotation is not set, then the ClusterAutoscaler follows its overall behavior (i.e evict the DaemonSets based on its configuration).

Note:

This annotation only impacts DaemonSet Pods.

kubernetes.io/ingress-bandwidth

Type: Annotation

Example: `kubernetes.io/ingress-bandwidth: 10M`

Used on: Pod

You can apply quality-of-service traffic shaping to a pod and effectively limit its available bandwidth. Ingress traffic to a Pod is handled by shaping queued packets to effectively handle data. To limit the bandwidth on a Pod, write an object definition JSON file and specify the data traffic speed using `kubernetes.io/ingress-bandwidth` annotation. The unit used for specifying ingress rate is bits per second, as a [Quantity](#). For example, `10M` means 10 megabits per second.

Note:

Ingress traffic shaping annotation is an experimental feature. If you want to enable traffic shaping support, you must add the `bandwidth` plugin to your CNI configuration file (default `/etc/cni/net.d`) and ensure that the binary is included in your CNI bin dir (default `/opt/cni/bin`).

kubernetes.io/egress-bandwidth

Type: Annotation

Example: `kubernetes.io/egress-bandwidth: 10M`

Used on: Pod

Egress traffic from a Pod is handled by policing, which simply drops packets in excess of the configured rate. The limits you place on a Pod do not affect the bandwidth of other Pods. To limit the bandwidth on a Pod, write an object definition JSON file and specify the data traffic speed using `kubernetes.io/egress-bandwidth` annotation. The unit used for specifying egress rate is bits per second, as a [Quantity](#). For example, `10M` means 10 megabits per second.

Note:

Egress traffic shaping annotation is an experimental feature. If you want to enable traffic shaping support, you must add the `bandwidth` plugin to your CNI configuration file (default `/etc/cni/net.d`) and ensure that the binary is included in your CNI bin dir (default `/opt/cni/bin`).

beta.kubernetes.io/instance-type (deprecated)

Type: Label

Note:

Starting in v1.17, this label is deprecated in favor of [node.kubernetes.io/instance-type](#).

node.kubernetes.io/instance-type

Type: Label

Example: `node.kubernetes.io/instance-type: "m3.medium"`

Used on: Node

The Kubelet populates this with the instance type as defined by the cloud provider. This will be set only if you are using a cloud provider. This setting is handy if you want to target certain workloads to certain instance types, but typically you want to rely on the Kubernetes scheduler to perform resource-based scheduling. You should aim to schedule based on properties rather than on instance types (for example: require a GPU, instead of requiring a `g2.xlarge`).

failure-domain.beta.kubernetes.io/region (deprecated)

Type: Label

Note:

Starting in v1.17, this label is deprecated in favor of [topology.kubernetes.io/region](#).

failure-domain.beta.kubernetes.io/zone (deprecated)

Type: Label

Note:

Starting in v1.17, this label is deprecated in favor of [topology.kubernetes.io/zone](https://kubernetes.io/docs/concepts/storage/persistentvolumes/#topology).

pv.kubernetes.io/bind-completed

Type: Annotation

Example: `pv.kubernetes.io/bind-completed: "yes"`

Used on: PersistentVolumeClaim

When this annotation is set on a PersistentVolumeClaim (PVC), that indicates that the lifecycle of the PVC has passed through initial binding setup. When present, that information changes how the control plane interprets the state of PVC objects. The value of this annotation does not matter to Kubernetes.

pv.kubernetes.io/bound-by-controller

Type: Annotation

Example: `pv.kubernetes.io/bound-by-controller: "yes"`

Used on: PersistentVolume, PersistentVolumeClaim

If this annotation is set on a PersistentVolume or PersistentVolumeClaim, it indicates that a storage binding (PersistentVolume → PersistentVolumeClaim, or PersistentVolumeClaim → PersistentVolume) was installed by the [controller](#). If the annotation isn't set, and there is a storage binding in place, the absence of that annotation means that the binding was done manually. The value of this annotation does not matter.

pv.kubernetes.io/provisioned-by

Type: Annotation

Example: `pv.kubernetes.io/provisioned-by: "kubernetes.io/rbd"`

Used on: PersistentVolume

This annotation is added to a PersistentVolume(PV) that has been dynamically provisioned by Kubernetes. Its value is the name of volume plugin that created the volume. It serves both users (to show where a PV comes from) and Kubernetes (to recognize dynamically provisioned PVs in its decisions).

pv.kubernetes.io/migrated-to

Type: Annotation

Example: `pv.kubernetes.io/migrated-to: pd.csi.storage.gke.io`

Used on: PersistentVolume, PersistentVolumeClaim

It is added to a PersistentVolume(PV) and PersistentVolumeClaim(PVC) that is supposed to be dynamically provisioned/deleted by its corresponding CSI driver through the `CSIMigration` feature gate. When this annotation is set, the Kubernetes components will "stand-down" and the `external-provisioner` will act on the objects.

statefulset.kubernetes.io/pod-name

Type: Label

Example: `statefulset.kubernetes.io/pod-name: "mystatefulset-7"`

Used on: Pod

When a StatefulSet controller creates a Pod for the StatefulSet, the control plane sets this label on that Pod. The value of the label is the name of the Pod being created.

See [Pod Name Label](#) in the StatefulSet topic for more details.

scheduler.alpha.kubernetes.io/node-selector

Type: Annotation

Example: `scheduler.alpha.kubernetes.io/node-selector: "name-of-node-selector"`

Used on: Namespace

The [PodNodeSelector](#) uses this annotation key to assign node selectors to pods in namespaces.

topology.kubernetes.io/region

Type: Label

Example: `topology.kubernetes.io/region: "us-east-1"`

Used on: Node, PersistentVolume

See [topology.kubernetes.io/zone](https://kubernetes.io/docs/concepts/storage/persistentvolumes/#topology).

topology.kubernetes.io/zone

Type: Label

Example: topology.kubernetes.io/zone: "us-east-1c"

Used on: Node, PersistentVolume

On Node: The kubelet or the external cloud-controller-manager populates this with the information from the cloud provider. This will be set only if you are using a cloud provider. However, you can consider setting this on nodes if it makes sense in your topology.

On PersistentVolume: topology-aware volume provisioners will automatically set node affinity constraints on a PersistentVolume.

A zone represents a logical failure domain. It is common for Kubernetes clusters to span multiple zones for increased availability. While the exact definition of a zone is left to infrastructure implementations, common properties of a zone include very low network latency within a zone, no-cost network traffic within a zone, and failure independence from other zones. For example, nodes within a zone might share a network switch, but nodes in different zones should not.

A region represents a larger domain, made up of one or more zones. It is uncommon for Kubernetes clusters to span multiple regions. While the exact definition of a zone or region is left to infrastructure implementations, common properties of a region include higher network latency between them than within them, non-zero cost for network traffic between them, and failure independence from other zones or regions. For example, nodes within a region might share power infrastructure (e.g. a UPS or generator), but nodes in different regions typically would not.

Kubernetes makes a few assumptions about the structure of zones and regions:

1. regions and zones are hierarchical: zones are strict subsets of regions and no zone can be in 2 regions
2. zone names are unique across regions; for example region "africa-east-1" might be comprised of zones "africa-east-1a" and "africa-east-1b"

It should be safe to assume that topology labels do not change. Even though labels are strictly mutable, consumers of them can assume that a given node is not going to be moved between zones without being destroyed and recreated.

Kubernetes can use this information in various ways. For example, the scheduler automatically tries to spread the Pods in a ReplicaSet across nodes in a single-zone cluster (to reduce the impact of node failures, see [kubernetes.io/hostname](https://kubernetes.io/docs/concepts/scheduling/kubernetes-scheduler.html#hostname)). With multiple-zone clusters, this spreading behavior also applies to zones (to reduce the impact of zone failures). This is achieved via *SelectorSpreadPriority*.

SelectorSpreadPriority is a best effort placement. If the zones in your cluster are heterogeneous (for example: different numbers of nodes, different types of nodes, or different pod resource requirements), this placement might prevent equal spreading of your Pods across zones. If desired, you can use homogeneous zones (same number and types of nodes) to reduce the probability of unequal spreading.

The scheduler (through the *VolumeZonePredicate* predicate) also will ensure that Pods, that claim a given volume, are only placed into the same zone as that volume. Volumes cannot be attached across zones.

If *PersistentVolumeLabel* does not support automatic labeling of your PersistentVolumes, you should consider adding the labels manually (or adding support for *PersistentVolumeLabel*). With *PersistentVolumeLabel*, the scheduler prevents Pods from mounting volumes in a different zone. If your infrastructure doesn't have this constraint, you don't need to add the zone labels to the volumes at all.

volume.beta.kubernetes.io/storage-provisioner (deprecated)

Type: Annotation

Example: volume.beta.kubernetes.io/storage-provisioner: "k8s.io/minikube-hostpath"

Used on: PersistentVolumeClaim

This annotation has been deprecated since v1.23. See [volume.kubernetes.io/storage-provisioner](https://kubernetes.io/docs/concepts/storage/storage-provisioners.html).

volume.beta.kubernetes.io/storage-class (deprecated)

Type: Annotation

Example: volume.beta.kubernetes.io/storage-class: "example-class"

Used on: PersistentVolume, PersistentVolumeClaim

This annotation can be used for PersistentVolume(PV) or PersistentVolumeClaim(PVC) to specify the name of [StorageClass](https://kubernetes.io/docs/concepts/storage/storage-classes.html). When both the *storageClassName* attribute and the *volume.beta.kubernetes.io/storage-class* annotation are specified, the annotation *volume.beta.kubernetes.io/storage-class* takes precedence over the *storageClassName* attribute.

This annotation has been deprecated. Instead, set the [storageClassName field](https://kubernetes.io/docs/concepts/storage/persistentvolumes.html#storage-class) for the PersistentVolumeClaim or PersistentVolume.

volume.beta.kubernetes.io/mount-options (deprecated)

Type: Annotation

Example : volume.beta.kubernetes.io/mount-options: "ro,soft"

Used on: PersistentVolume

A Kubernetes administrator can specify additional [mount options](https://kubernetes.io/docs/concepts/storage/persistentvolumes.html#mount-options) for when a PersistentVolume is mounted on a node.

volume.kubernetes.io/storage-provisioner

Type: Annotation

Used on: PersistentVolumeClaim

This annotation is added to a PVC that is supposed to be dynamically provisioned. Its value is the name of a volume plugin that is supposed to provision a volume for this PVC.

volume.kubernetes.io/selected-node

Type: Annotation

Used on: PersistentVolumeClaim

This annotation is added to a PVC that is triggered by a scheduler to be dynamically provisioned. Its value is the name of the selected node.

volumes.kubernetes.io/controller-managed-attach-detach

Type: Annotation

Used on: Node

If a node has the annotation `volumes.kubernetes.io/controller-managed-attach-detach`, its storage attach and detach operations are being managed by the *volume attach/detach controller*.

The value of the annotation isn't important.

node.kubernetes.io/windows-build

Type: Label

Example: `node.kubernetes.io/windows-build: "10.0.17763"`

Used on: Node

When the kubelet is running on Microsoft Windows, it automatically labels its Node to record the version of Windows Server in use.

The label's value is in the format "MajorVersion.MinorVersion.BuildNumber".

storage.alpha.kubernetes.io/migrated-plugins

Type: Annotation

Example: `storage.alpha.kubernetes.io/migrated-plugins: "kubernetes.io/cinder"`

Used on: CSINode (an extension API)

This annotation is automatically added for the CSINode object that maps to a node that installs CSIDriver. This annotation shows the in-tree plugin name of the migrated plugin. Its value depends on your cluster's in-tree cloud provider storage type.

For example, if the in-tree cloud provider storage type is `CSIMigrationvSphere`, the CSINodes instance for the node should be updated with:
`storage.alpha.kubernetes.io/migrated-plugins: "kubernetes.io/vsphere-volume"`

service.kubernetes.io/headless

Type: Label

Example: `service.kubernetes.io/headless: ""`

Used on: Endpoints

The control plane adds this label to an Endpoints object when the owning Service is headless. To learn more, read [Headless Services](#).

service.kubernetes.io/topology-aware-hints (deprecated)

Example: `service.kubernetes.io/topology-aware-hints: "Auto"`

Used on: Service

This annotation was used for enabling *topology aware hints* on Services. Topology aware hints have since been renamed: the concept is now called [topology aware routing](#). Setting the annotation to `Auto`, on a Service, configured the Kubernetes control plane to add topology hints on EndpointSlices associated with that Service. You can also explicitly set the annotation to `Disabled`.

If you are running a version of Kubernetes older than 1.34, check the documentation for that Kubernetes version to see how topology aware routing works in that release.

There are no other valid values for this annotation. If you don't want topology aware hints for a Service, don't add this annotation.

service.kubernetes.io/topology-mode

Type: Annotation

Example: `service.kubernetes.io/topology-mode: Auto`

Used on: Service

This annotation provides a way to define how Services handle network topology; for example, you can configure a Service so that Kubernetes prefers keeping traffic between a client and server within a single topology zone. In some cases this can help reduce costs or improve network performance.

See [Topology Aware Routing](#) for more details.

kubernetes.io/service-name

Type: Label

Example: `kubernetes.io/service-name: "my-website"`

Used on: EndpointSlice

Kubernetes associates [EndpointSlices](#) with [Services](#) using this label.

This label records the [name](#) of the Service that the EndpointSlice is backing. All EndpointSlices should have this label set to the name of their associated Service.

kubernetes.io/service-account.name

Type: Annotation

Example: `kubernetes.io/service-account.name: "sa-name"`

Used on: Secret

This annotation records the [name](#) of the ServiceAccount that the token (stored in the Secret of type `kubernetes.io/service-account-token`) represents.

kubernetes.io/service-account.uid

Type: Annotation

Example: `kubernetes.io/service-account.uid: da68f9c6-9d26-11e7-b84e-002dc52800da`

Used on: Secret

This annotation records the [unique ID](#) of the ServiceAccount that the token (stored in the Secret of type `kubernetes.io/service-account-token`) represents.

kubernetes.io/legacy-token-last-used

Type: Label

Example: `kubernetes.io/legacy-token-last-used: 2022-10-24`

Used on: Secret

The control plane only adds this label to Secrets that have the type `kubernetes.io/service-account-token`. The value of this label records the date (ISO 8601 format, UTC time zone) when the control plane last saw a request where the client authenticated using the service account token.

If a legacy token was last used before the cluster gained the feature (added in Kubernetes v1.26), then the label isn't set.

kubernetes.io/legacy-token-invalid-since

Type: Label

Example: `kubernetes.io/legacy-token-invalid-since: 2023-10-27`

Used on: Secret

The control plane automatically adds this label to auto-generated Secrets that have the type `kubernetes.io/service-account-token`. This label marks the Secret-based token as invalid for authentication. The value of this label records the date (ISO 8601 format, UTC time zone) when the control plane detects that the auto-generated Secret has not been used for a specified duration (defaults to one year).

endpoints.kubernetes.io/managed-by (deprecated)

Type: Label

Example: `endpoints.kubernetes.io/managed-by: endpoint-controller`

Used on: Endpoints

This label is used internally to mark Endpoints objects that were created by Kubernetes (as opposed to Endpoints created by users or external controllers).

Note:

The [Endpoints](#) API is deprecated in favor of [EndpointSlice](#).

endpointslice.kubernetes.io/managed-by

Type: Label

Example: `endpointslice.kubernetes.io/managed-by: endpointslice-controller.k8s.io`

Used on: EndpointSlices

The label is used to indicate the controller or entity that manages the EndpointSlice. This label aims to enable different EndpointSlice objects to be managed by different controllers or entities within the same cluster. The value `endpointslice-controller.k8s.io` indicates an EndpointSlice object that was created automatically by Kubernetes for a Service with a [selectors](#).

endpointslice.kubernetes.io/skip-mirror

Type: Label

Example: `endpointslice.kubernetes.io/skip-mirror: "true"`

Used on: Endpoints

The label can be set to `"true"` on an Endpoints resource to indicate that the EndpointSliceMirroring controller should not mirror this resource with EndpointSlices.

service.kubernetes.io/service-proxy-name

Type: Label

Example: `service.kubernetes.io/service-proxy-name: "foo-bar"`

Used on: Service

Setting a value for this label tells kube-proxy to ignore this service for proxying purposes. This allows for use of alternative proxy implementations for this service (e.g. running a DaemonSet that manages nftables its own way). Multiple alternative proxy implementations could be active simultaneously using this field, e.g. by having a value unique to each alternative proxy implementation to be responsible for their respective services.

experimental.windows.kubernetes.io/isolation-type (deprecated)

Type: Annotation

Example: `experimental.windows.kubernetes.io/isolation-type: "hyperv"`

Used on: Pod

The annotation is used to run Windows containers with Hyper-V isolation.

Note:

Starting from v1.20, this annotation is deprecated. Experimental Hyper-V support was removed in 1.21.

ingressclass.kubernetes.io/is-default-class

Type: Annotation

Example: `ingressclass.kubernetes.io/is-default-class: "true"`

Used on: IngressClass

When a IngressClass resource has this annotation set to `"true"`, new Ingress resource without a class specified will be assigned this default class.

nginx.ingress.kubernetes.io/configuration-snippet

Type: Annotation

Example: `nginx.ingress.kubernetes.io/configuration-snippet: " more_set_headers \"Request-Id: $req_id\";\nmore_set_headers \"Example: 42\";\n"`

Used on: Ingress

You can use this annotation to set extra configuration on an Ingress that uses the [NGINX Ingress Controller](#). The `configuration-snippet` annotation is ignored by default since version 1.9.0 of the ingress controller. The NGINX ingress controller setting `allow-snippet-annotations` has to be explicitly enabled to use this annotation. Enabling the annotation can be dangerous in a multi-tenant cluster, as it can lead people with otherwise limited permissions being able to retrieve all Secrets in the cluster.

kubernetes.io/ingress.class (deprecated)

Type: Annotation

Used on: Ingress

Note:

Starting in v1.18, this annotation is deprecated in favor of `spec.ingressClassName`.

kubernetes.io/cluster-service (deprecated)

Type: Label

Example: `kubernetes.io/cluster-service: "true"`

Used on: Service

This label indicates that the Service provides a service to the cluster, if the value is set to true. When you run `kubectl cluster-info`, the tool queries for Services with this label set to true.

However, setting this label on any Service is deprecated.

storageclass.kubernetes.io/is-default-class

Type: Annotation

Example: `storageclass.kubernetes.io/is-default-class: "true"`

Used on: StorageClass

When a single StorageClass resource has this annotation set to "true", new PersistentVolumeClaim resource without a class specified will be assigned this default class.

alpha.kubernetes.io/provided-node-ip (alpha)

Type: Annotation

Example: `alpha.kubernetes.io/provided-node-ip: "10.0.0.1"`

Used on: Node

The kubelet can set this annotation on a Node to denote its configured IPv4 and/or IPv6 address.

When kubelet is started with the `--cloud-provider` flag set to any value (includes both external and legacy in-tree cloud providers), it sets this annotation on the Node to denote an IP address set from the command line flag (`--node-ip`). This IP is verified with the cloud provider as valid by the cloud-controller-manager.

batch.kubernetes.io/job-completion-index

Type: Annotation, Label

Example: `batch.kubernetes.io/job-completion-index: "3"`

Used on: Pod

The Job controller in the kube-controller-manager sets this as a label and annotation for Pods created with Indexed [completion mode](#).

Note the [PodIndexLabel](#) feature gate must be enabled for this to be added as a pod **label**, otherwise it will just be an annotation.

batch.kubernetes.io/cronjob-scheduled-timestamp

Type: Annotation

Example: `batch.kubernetes.io/cronjob-scheduled-timestamp: "2016-05-19T03:00:00-07:00"`

Used on: Jobs and Pods controlled by CronJobs

This annotation is used to record the original (expected) creation timestamp for a Job, when that Job is part of a CronJob. The control plane sets the value to that timestamp in RFC3339 format. If the Job belongs to a CronJob with a timezone specified, then the timestamp is in that timezone. Otherwise, the timestamp is in controller-manager's local time.

kubectl.kubernetes.io/default-container

Type: Annotation

Example: `kubectl.kubernetes.io/default-container: "front-end-app"`

The value of the annotation is the container name that is default for this Pod. For example, `kubectl logs` or `kubectl exec` without `-c` or `--container` flag will use this default container.

kubectl.kubernetes.io/default-logs-container (deprecated)

Type: Annotation

Example: `kubectl.kubernetes.io/default-logs-container: "front-end-app"`

The value of the annotation is the container name that is the default logging container for this Pod. For example, `kubectl logs` without `-c` or `--container` flag will use this default container.

Note:

This annotation is deprecated. You should use the [kubectl.kubernetes.io/default-container](https://kubernetes.io/docs/reference/labels-annotations/#kubectlkubernetesio/default-container) annotation instead. Kubernetes versions 1.25 and newer ignore this annotation.

kubectl.kubernetes.io/last-applied-configuration

Type: Annotation

Example: *see following snippet*

```
kubectl.kubernetes.io/last-applied-configuration: >
{"apiVersion": "apps/v1", "kind": "Deployment", "metadata": {"annotations": {}, "name": "example", "namespace": "default"}, "spec": {"se.
```

Used on: all objects

The kubectl command line tool uses this annotation as a legacy mechanism to track changes. That mechanism has been superseded by [Server-side apply](#).

kubectl.kubernetes.io/restartedAt

Type: Annotation

Example: kubectl.kubernetes.io/restartedAt: "2024-06-21T17:27:41Z"

Used on: Deployment, ReplicaSet, StatefulSet, DaemonSet, Pod

This annotation contains the latest restart time of a resource (Deployment, ReplicaSet, StatefulSet or DaemonSet), where kubectl triggered a rollout in order to force creation of new Pods. The command `kubectl rollout restart <RESOURCE>` triggers a restart by patching the template metadata of all the pods of resource with this annotation. In above example the latest restart time is shown as 21st June 2024 at 17:27:41 UTC.

You should not assume that this annotation represents the date / time of the most recent update; a separate change could have been made since the last manually triggered rollout.

If you manually set this annotation on a Pod, nothing happens. The restarting side effect comes from how workload management and Pod templating works.

endpoints.kubernetes.io/over-capacity

Type: Annotation

Example: endpoints.kubernetes.io/over-capacity:truncated

Used on: Endpoints

The [control plane](#) adds this annotation to an [Endpoints](#) object if the associated [Service](#) has more than 1000 backing endpoints. The annotation indicates that the Endpoints object is over capacity and the number of endpoints has been truncated to 1000.

If the number of backend endpoints falls below 1000, the control plane removes this annotation.

endpoints.kubernetes.io/last-change-trigger-time

Type: Annotation

Example: endpoints.kubernetes.io/last-change-trigger-time: "2023-07-20T04:45:21Z"

Used on: Endpoints

This annotation set to an [Endpoints](#) object that represents the timestamp (The timestamp is stored in RFC 3339 date-time string format. For example, '2018-10-22T19:32:52.1Z'). This is timestamp of the last change in some Pod or Service object, that triggered the change to the Endpoints object.

control-plane.alpha.kubernetes.io/leader (deprecated)

Type: Annotation

```
Example: control-plane.alpha.kubernetes.io/leader={"holderIdentity":"controller-0","leaseDurationSeconds":15,"acquireTime":"2023-01-19T13:12:57Z","renewTime":"2023-01-19T13:13:54Z","leaderTransitions":1}
```

Used on: Endpoints

The [control plane](#) previously set annotation on an [Endpoints](#) object. This annotation provided the following detail:

- Who is the current leader.
- The time when the current leadership was acquired.
- The duration of the lease (of the leadership) in seconds.
- The time the current lease (the current leadership) should be renewed.
- The number of leadership transitions that happened in the past.

Kubernetes now uses [Leases](#) to manage leader assignment for the Kubernetes control plane.

batch.kubernetes.io/job-tracking (deprecated)

Type: Annotation

Example: `batch.kubernetes.io/job-tracking: ""`

Used on: Jobs

The presence of this annotation on a Job used to indicate that the control plane is [tracking the Job status using finalizers](#). Adding or removing this annotation no longer has an effect (Kubernetes v1.27 and later) All Jobs are tracked with finalizers.

job-name (deprecated)

Type: Label

Example: `job-name: "pi"`

Used on: Jobs and Pods controlled by Jobs

Note:

Starting from Kubernetes 1.27, this label is deprecated. Kubernetes 1.27 and newer ignore this label and use the prefixed `job-name` label.

controller-uid (deprecated)

Type: Label

Example: `controller-uid: "$UID"`

Used on: Jobs and Pods controlled by Jobs

Note:

Starting from Kubernetes 1.27, this label is deprecated. Kubernetes 1.27 and newer ignore this label and use the prefixed `controller-uid` label.

batch.kubernetes.io/job-name

Type: Label

Example: `batch.kubernetes.io/job-name: "pi"`

Used on: Jobs and Pods controlled by Jobs

This label is used as a user-friendly way to get Pods corresponding to a Job. The `job-name` comes from the name of the Job and allows for an easy way to get Pods corresponding to the Job.

batch.kubernetes.io/controller-uid

Type: Label

Example: `batch.kubernetes.io/controller-uid: "$UID"`

Used on: Jobs and Pods controlled by Jobs

This label is used as a programmatic way to get all Pods corresponding to a Job.

The `controller-uid` is a unique identifier that gets set in the `selector` field so the Job controller can get all the corresponding Pods.

scheduler.alpha.kubernetes.io/defaultTolerations

Type: Annotation

Example: `scheduler.alpha.kubernetes.io/defaultTolerations: '[{"operator": "Equal", "value": "value1", "effect": "NoSchedule", "key": "dedicated-node"}]'`

Used on: Namespace

This annotation requires the [PodTolerationRestriction](#) admission controller to be enabled. This annotation key allows assigning tolerations to a namespace and any new pods created in this namespace would get these tolerations added.

scheduler.alpha.kubernetes.io/tolerationsWhitelist

Type: Annotation

Example: `scheduler.alpha.kubernetes.io/tolerationsWhitelist: '[{"operator": "Exists", "effect": "NoSchedule", "key": "dedicated-node"}]'`

Used on: Namespace

This annotation is only useful when the (Alpha) [PodTolerationRestriction](#) admission controller is enabled. The annotation value is a JSON document that defines a list of allowed tolerations for the namespace it annotates. When you create a Pod or modify its tolerations, the API server checks the tolerations to see if they are mentioned in the allow list. The pod is admitted only if the check succeeds.

scheduler.alpha.kubernetes.io/preferAvoidPods (deprecated)

Type: Annotation

Used on: Node

This annotation requires the [NodePreferAvoidPods scheduling plugin](#) to be enabled. The plugin is deprecated since Kubernetes 1.22. Use [Taints and Tolerations](#) instead.

node.kubernetes.io/not-ready

Type: Taint

Example: `node.kubernetes.io/not-ready: "NoExecute"`

Used on: Node

The Node controller detects whether a Node is ready by monitoring its health and adds or removes this taint accordingly.

node.kubernetes.io/unreachable

Type: Taint

Example: `node.kubernetes.io/unreachable: "NoExecute"`

Used on: Node

The Node controller adds the taint to a Node corresponding to the [NodeCondition](#) Ready being Unknown.

node.kubernetes.io/unschedulable

Type: Taint

Example: `node.kubernetes.io/unschedulable: "NoSchedule"`

Used on: Node

The taint will be added to a node when initializing the node to avoid race condition.

node.kubernetes.io/memory-pressure

Type: Taint

Example: `node.kubernetes.io/memory-pressure: "NoSchedule"`

Used on: Node

The kubelet detects memory pressure based on `memory.available` and `allocatableMemory.available` observed on a Node. The observed values are then compared to the corresponding thresholds that can be set on the kubelet to determine if the Node condition and taint should be added/removed.

node.kubernetes.io/disk-pressure

Type: Taint

Example: `node.kubernetes.io/disk-pressure : "NoSchedule"`

Used on: Node

The kubelet detects disk pressure based on `imagefs.available`, `imagefs.inodesFree`, `nodefs.available` and `nodefs.inodesFree` (Linux only) observed on a Node. The observed values are then compared to the corresponding thresholds that can be set on the kubelet to determine if the Node condition and taint should be added/removed.

node.kubernetes.io/network-unavailable

Type: Taint

Example: `node.kubernetes.io/network-unavailable: "NoSchedule"`

Used on: Node

This is initially set by the kubelet when the cloud provider used indicates a requirement for additional network configuration. Only when the route on the cloud is configured properly will the taint be removed by the cloud provider.

node.kubernetes.io/pid-pressure

Type: Taint

Example: `node.kubernetes.io/pid-pressure: "NoSchedule"`

Used on: Node

The kubelet checks D-value of the size of `/proc/sys/kernel/pid_max` and the PIDs consumed by Kubernetes on a node to get the number of available PIDs that referred to as the `pid.available` metric. The metric is then compared to the corresponding threshold that can be set on the kubelet to determine if the

node condition and taint should be added/removed.

node.kubernetes.io/out-of-service

Type: Taint

Example: `node.kubernetes.io/out-of-service:NoExecute`

Used on: Node

A user can manually add the taint to a Node marking it out-of-service. If a Node is marked out-of-service with this taint, the Pods on the node will be forcefully deleted if there are no matching tolerations on it and volume detach operations for the Pods terminating on the node will happen immediately. This allows the Pods on the out-of-service node to recover quickly on a different node.

Caution:

Refer to [Non-graceful node shutdown](#) for further details about when and how to use this taint.

node.cloudprovider.kubernetes.io/uninitialized

Type: Taint

Example: `node.cloudprovider.kubernetes.io/uninitialized: "NoSchedule"`

Used on: Node

Sets this taint on a Node to mark it as unusable, when kubelet is started with the "external" cloud provider, until a controller from the cloud-controller-manager initializes this Node, and then removes the taint.

node.cloudprovider.kubernetes.io/shutdown

Type: Taint

Example: `node.cloudprovider.kubernetes.io/shutdown: "NoSchedule"`

Used on: Node

If a Node is in a cloud provider specified shutdown state, the Node gets tainted accordingly with `node.cloudprovider.kubernetes.io/shutdown` and the taint effect of `NoSchedule`.

feature.node.kubernetes.io/*

Type: Label

Example: `feature.node.kubernetes.io/network-sriov.capable: "true"`

Used on: Node

These labels are used by the Node Feature Discovery (NFD) component to advertise features on a node. All built-in labels use the `feature.node.kubernetes.io` label namespace and have the format `feature.node.kubernetes.io/<feature-name>: "true"`. NFD has many extension points for creating vendor and application-specific labels. For details, see the [customization guide](#).

nfd.node.kubernetes.io/master.version

Type: Annotation

Example: `nfd.node.kubernetes.io/master.version: "v0.6.0"`

Used on: Node

For node(s) where the Node Feature Discovery (NFD) [master](#) is scheduled, this annotation records the version of the NFD master. It is used for informative use only.

nfd.node.kubernetes.io/worker.version

Type: Annotation

Example: `nfd.node.kubernetes.io/worker.version: "v0.4.0"`

Used on: Nodes

This annotation records the version for a Node Feature Discovery's [worker](#) if there is one running on a node. It's used for informative use only.

nfd.node.kubernetes.io/feature-labels

Type: Annotation

Example: `nfd.node.kubernetes.io/feature-labels: "cpu-cpuid.ADX,cpu-cpuid.AESNI,cpu-hardware_multithreading,kernel-version.full"`

Used on: Nodes

This annotation records a comma-separated list of node feature labels managed by [Node Feature Discovery](#) (NFD). NFD uses this for an internal mechanism. You should not edit this annotation yourself.

nfd.node.kubernetes.io/extended-resources

Type: Annotation

Example: `nfd.node.kubernetes.io/extended-resources: "accelerator.acme.example/q500,example.com/coprocessor-fx5"`

Used on: Nodes

This annotation records a comma-separated list of [extended resources](#) managed by [Node Feature Discovery](#) (NFD). NFD uses this for an internal mechanism. You should not edit this annotation yourself.

nfd.node.kubernetes.io/node-name

Type: Label

Example: `nfd.node.kubernetes.io/node-name: node-1`

Used on: Nodes

It specifies which node the NodeFeature object is targeting. Creators of NodeFeature objects must set this label and consumers of the objects are supposed to use the label for filtering features designated for a certain node.

Note:

These Node Feature Discovery (NFD) labels or annotations only apply to the nodes where NFD is running. To learn more about NFD and its components go to its official [documentation](#).

service.beta.kubernetes.io/aws-load-balancer-access-log-emit-interval (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-access-log-emit-interval: "5"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures the load balancer for a Service based on this annotation. The value determines how often the load balancer writes log entries. For example, if you set the value to 5, the log writes occur 5 seconds apart.

service.beta.kubernetes.io/aws-load-balancer-access-log-enabled (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-access-log-enabled: "false"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures the load balancer for a Service based on this annotation. Access logging is enabled if you set the annotation to "true".

service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-name (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-name: example`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures the load balancer for a Service based on this annotation. The load balancer writes logs to an S3 bucket with the name you specify.

service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-prefix (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-prefix: "/example"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures the load balancer for a Service based on this annotation. The load balancer writes log objects with the prefix that you specify.

service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags: "Environment=demo,Project=example"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures tags (an AWS concept) for a load balancer based on the comma-separated key/value pairs in the value of this annotation.

service.beta.kubernetes.io/aws-load-balancer-alpn-policy (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-alpn-policy: HTTP2Optional`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-attributes (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-attributes: "deletion_protection.enabled=true"`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-backend-protocol (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-backend-protocol: tcp`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures the load balancer listener based on the value of this annotation.

service.beta.kubernetes.io/aws-load-balancer-connection-draining-enabled (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-connection-draining-enabled: "false"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures the load balancer based on this annotation. The load balancer's connection draining setting depends on the value you set.

service.beta.kubernetes.io/aws-load-balancer-connection-draining-timeout (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-connection-draining-timeout: "60"`

Used on: Service

If you configure [connection draining](#) for a Service of type: `LoadBalancer`, and you use the AWS cloud, the integration configures the draining period based on this annotation. The value you set determines the draining timeout in seconds.

service.beta.kubernetes.io/aws-load-balancer-ip-address-type (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-ip-address-type: ipv4`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-connection-idle-timeout (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-connection-idle-timeout: "60"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The load balancer has a configured idle timeout period (in seconds) that applies to its connections. If no data has been sent or received by the time that the idle timeout period elapses, the load balancer closes the connection.

service.beta.kubernetes.io/aws-load-balancer-cross-zone-load-balancing-enabled (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-cross-zone-load-balancing-enabled: "true"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. If you set this annotation to "true", each load balancer node distributes requests evenly across the registered targets in all enabled [availability zones](#). If you disable cross-zone load balancing, each load balancer node distributes requests evenly across the registered targets in its availability zone only.

service.beta.kubernetes.io/aws-load-balancer-eip-allocations (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-eip-allocations: "eipalloc-01bcdef23bcdef456,eipalloc-def1234abc4567890"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The value is a comma-separated list of elastic IP address allocation IDs.

This annotation is only relevant for Services of type: `LoadBalancer`, where the load balancer is an AWS Network Load Balancer.

service.beta.kubernetes.io/aws-load-balancer-extra-security-groups (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-extra-security-groups: "sg-12abcd3456,sg-34dcba6543"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value is a comma-separated list of extra AWS VPC security groups to configure for the load balancer.

service.beta.kubernetes.io/aws-load-balancer-healthcheck-healthy-threshold (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-healthcheck-healthy-threshold: "3"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value specifies the number of successive successful health checks required for a backend to be considered healthy for traffic.

service.beta.kubernetes.io/aws-load-balancer-healthcheck-interval (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-healthcheck-interval: "30"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value specifies the interval, in seconds, between health check probes made by the load balancer.

service.beta.kubernetes.io/aws-load-balancer-healthcheck-path (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-healthcheck-path: /healthcheck`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value determines the path part of the URL that is used for HTTP health checks.

service.beta.kubernetes.io/aws-load-balancer-healthcheck-port (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-healthcheck-port: "24"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value determines which port the load balancer connects to when performing health checks.

service.beta.kubernetes.io/aws-load-balancer-healthcheck-protocol (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-healthcheck-protocol: TCP`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value determines how the load balancer checks the health of backend targets.

service.beta.kubernetes.io/aws-load-balancer-healthcheck-timeout (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-healthcheck-timeout: "3"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value specifies the number of seconds before a probe that hasn't yet succeeded is automatically treated as having failed.

service.beta.kubernetes.io/aws-load-balancer-healthcheck-unhealthy-threshold (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-healthcheck-unhealthy-threshold: "3"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. The annotation value specifies the number of successive unsuccessful health checks required for a backend to be considered unhealthy for traffic.

service.beta.kubernetes.io/aws-load-balancer-internal (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-internal: "true"`

Used on: Service

The cloud controller manager integration with AWS elastic load balancing configures a load balancer based on this annotation. When you set this annotation to "true", the integration configures an internal load balancer.

If you use the [AWS load balancer controller](#), see [service.beta.kubernetes.io/aws-load-balancer-scheme](#).

service.beta.kubernetes.io/aws-load-balancer-manage-backend-security-group-rules (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-manage-backend-security-group-rules: "true"`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-name (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-name: my-elb`

Used on: Service

If you set this annotation on a Service, and you also annotate that Service with `service.beta.kubernetes.io/aws-load-balancer-type: "external"`, and you use the [AWS load balancer controller](#) in your cluster, then the AWS load balancer controller sets the name of that load balancer to the value you set for *this* annotation.

See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-nlb-target-type (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "true"`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-private-ipv4-addresses (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-private-ipv4-addresses: "198.51.100.0,198.51.100.64"`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-proxy-protocol (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-proxy-protocol: "*"`

Used on: Service

The official Kubernetes integration with AWS elastic load balancing configures a load balancer based on this annotation. The only permitted value is `"*"`, which indicates that the load balancer should wrap TCP connections to the backend Pod with the PROXY protocol.

service.beta.kubernetes.io/aws-load-balancer-scheme (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-scheme: internal`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-security-groups (deprecated)

Example: `service.beta.kubernetes.io/aws-load-balancer-security-groups: "sg-53fae93f,sg-8725gr62r"`

Used on: Service

The AWS load balancer controller uses this annotation to specify a comma separated list of security groups you want to attach to an AWS load balancer. Both name and ID of security are supported where name matches a Name tag, not the groupName attribute.

When this annotation is added to a Service, the load-balancer controller attaches the security groups referenced by the annotation to the load balancer. If you omit this annotation, the AWS load balancer controller automatically creates a new security group and attaches it to the load balancer.

Note:

Kubernetes v1.27 and later do not directly set or read this annotation. However, the AWS load balancer controller (part of the Kubernetes project) does still use the `service.beta.kubernetes.io/aws-load-balancer-security-groups` annotation.

service.beta.kubernetes.io/load-balancer-source-ranges (deprecated)

Example: `service.beta.kubernetes.io/load-balancer-source-ranges: "192.0.2.0/25"`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. You should set `.spec.loadBalancerSourceRanges` for the Service instead.

service.beta.kubernetes.io/aws-load-balancer-ssl-cert (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-ssl-cert: "arn:aws:acm:us-east-1:123456789012:certificate/12345678-1234-1234-1234-123456789012"`

Used on: Service

The official integration with AWS elastic load balancing configures TLS for a Service of type: `LoadBalancer` based on this annotation. The value of the annotation is the AWS Resource Name (ARN) of the X.509 certificate that the load balancer listener should use.

(The TLS protocol is based on an older technology that abbreviates to SSL.)

service.beta.kubernetes.io/aws-load-balancer-ssl-negotiation-policy (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-ssl-negotiation-policy: ELBSecurityPolicy-TLS-1-2-2017-01`

The official integration with AWS elastic load balancing configures TLS for a Service of type: `LoadBalancer` based on this annotation. The value of the annotation is the name of an AWS policy for negotiating TLS with a client peer.

service.beta.kubernetes.io/aws-load-balancer-ssl-ports (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-ssl-ports: "*"`

The official integration with AWS elastic load balancing configures TLS for a Service of type: `LoadBalancer` based on this annotation. The value of the annotation is either `"*"`, which means that all the load balancer's ports should use TLS, or it is a comma separated list of port numbers.

service.beta.kubernetes.io/aws-load-balancer-subnets (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-subnets: "private-a,private-b"`

Kubernetes' official integration with AWS uses this annotation to configure a load balancer and determine in which AWS availability zones to deploy the managed load balancing service. The value is either a comma separated list of subnet names, or a comma separated list of subnet IDs.

service.beta.kubernetes.io/aws-load-balancer-target-group-attributes (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-target-group-attributes: "stickiness.enabled=true,stickiness.type=source_ip"`

Used on: Service

The [AWS load balancer controller](#) uses this annotation. See [annotations](#) in the AWS load balancer controller documentation.

service.beta.kubernetes.io/aws-load-balancer-target-node-labels (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-target-node-labels: "kubernetes.io/os=Linux,topology.kubernetes.io/region=us-east-2"`

Kubernetes' official integration with AWS uses this annotation to determine which nodes in your cluster should be considered as valid targets for the load balancer.

service.beta.kubernetes.io/aws-load-balancer-type (beta)

Example: `service.beta.kubernetes.io/aws-load-balancer-type: external`

Kubernetes' official integrations with AWS use this annotation to determine whether the AWS cloud provider integration should manage a Service of type: `LoadBalancer`.

There are two permitted values:

`nlb`
the cloud controller manager configures a Network Load Balancer
`external`
the cloud controller manager does not configure any load balancer

If you deploy a Service of type: `LoadBalancer` on AWS, and you don't set any `service.beta.kubernetes.io/aws-load-balancer-type` annotation, the AWS integration deploys a classic Elastic Load Balancer. This behavior, with no annotation present, is the default unless you specify otherwise.

When you set this annotation to `external` on a Service of type: `LoadBalancer`, and your cluster has a working deployment of the AWS Load Balancer controller, then the AWS Load Balancer controller attempts to deploy a load balancer based on the Service specification.

Caution:

Do not modify or add the `service.beta.kubernetes.io/aws-load-balancer-type` annotation on an existing Service object. See the AWS documentation on this topic for more details.

service.beta.kubernetes.io/azure-load-balancer-disable-tcp-reset (deprecated)

Example: `service.beta.kubernetes.io/azure-load-balancer-disable-tcp-reset: "false"`

Used on: Service

This annotation only works for Azure standard load balancer backed service. This annotation is used on the Service to specify whether the load balancer should disable or enable TCP reset on idle timeout. If enabled, it helps applications to behave more predictably, to detect the termination of a connection, remove expired connections and initiate new connections. You can set the value to be either `true` or `false`.

See [Load Balancer TCP Reset](#) for more information.

Note:

This annotation is deprecated.

pod-security.kubernetes.io/enforce

Type: Label

Example: pod-security.kubernetes.io/enforce: "baseline"

Used on: Namespace

Value **must** be one of `privileged`, `baseline`, or `restricted` which correspond to [Pod Security Standard](#) levels. Specifically, the `enforce` label *prohibits* the creation of any Pod in the labeled Namespace which does not meet the requirements outlined in the indicated level.

See [Enforcing Pod Security at the Namespace Level](#) for more information.

pod-security.kubernetes.io/enforce-version

Type: Label

Example: pod-security.kubernetes.io/enforce-version: "1.34"

Used on: Namespace

Value **must** be `latest` or a valid Kubernetes version in the format `v<major>.<minor>`. This determines the version of the [Pod Security Standard](#) policies to apply when validating a Pod.

See [Enforcing Pod Security at the Namespace Level](#) for more information.

pod-security.kubernetes.io/audit

Type: Label

Example: pod-security.kubernetes.io/audit: "baseline"

Used on: Namespace

Value **must** be one of `privileged`, `baseline`, or `restricted` which correspond to [Pod Security Standard](#) levels. Specifically, the `audit` label does not prevent the creation of a Pod in the labeled Namespace which does not meet the requirements outlined in the indicated level, but adds an this annotation to the Pod.

See [Enforcing Pod Security at the Namespace Level](#) for more information.

pod-security.kubernetes.io/audit-version

Type: Label

Example: pod-security.kubernetes.io/audit-version: "1.34"

Used on: Namespace

Value **must** be `latest` or a valid Kubernetes version in the format `v<major>.<minor>`. This determines the version of the [Pod Security Standard](#) policies to apply when validating a Pod.

See [Enforcing Pod Security at the Namespace Level](#) for more information.

pod-security.kubernetes.io/warn

Type: Label

Example: pod-security.kubernetes.io/warn: "baseline"

Used on: Namespace

Value **must** be one of `privileged`, `baseline`, or `restricted` which correspond to [Pod Security Standard](#) levels. Specifically, the `warn` label does not prevent the creation of a Pod in the labeled Namespace which does not meet the requirements outlined in the indicated level, but returns a warning to the user after doing so. Note that warnings are also displayed when creating or updating objects that contain Pod templates, such as Deployments, Jobs, StatefulSets, etc.

See [Enforcing Pod Security at the Namespace Level](#) for more information.

pod-security.kubernetes.io/warn-version

Type: Label

Example: pod-security.kubernetes.io/warn-version: "1.34"

Used on: Namespace

Value **must** be `latest` or a valid Kubernetes version in the format `v<major>.<minor>`. This determines the version of the [Pod Security Standard](#) policies to apply when validating a submitted Pod. Note that warnings are also displayed when creating or updating objects that contain Pod templates, such as

Deployments, Jobs, StatefulSets, etc.

See [Enforcing Pod Security at the Namespace Level](#) for more information.

rbac.authorization.kubernetes.io/autoupdate

Type: Annotation

Example: `rbac.authorization.kubernetes.io/autoupdate: "false"`

Used on: ClusterRole, ClusterRoleBinding, Role, RoleBinding

When this annotation is set to `"true"` on default RBAC objects created by the API server, they are automatically updated at server start to add missing permissions and subjects (extra permissions and subjects are left in place). To prevent autoupdating a particular role or rolebinding, set this annotation to `"false"`. If you create your own RBAC objects and set this annotation to `"false"`, `kubectl auth reconcile` (which allows reconciling arbitrary RBAC objects in a [manifest](#)) respects this annotation and does not automatically add missing permissions and subjects.

kubernetes.io/psp (deprecated)

Type: Annotation

Example: `kubernetes.io/psp: restricted`

Used on: Pod

This annotation was only relevant if you were using [PodSecurityPolicy](#) objects. Kubernetes v1.34 does not support the PodSecurityPolicy API.

When the PodSecurityPolicy admission controller admitted a Pod, the admission controller modified the Pod to have this annotation. The value of the annotation was the name of the PodSecurityPolicy that was used for validation.

seccomp.security.alpha.kubernetes.io/pod (non-functional)

Type: Annotation

Used on: Pod

Kubernetes before v1.25 allowed you to configure seccomp behavior using this annotation. See [Restrict a Container's Syscalls with seccomp](#) to learn the supported way to specify seccomp restrictions for a Pod.

container.seccomp.security.alpha.kubernetes.io/[NAME] (non-functional)

Type: Annotation

Used on: Pod

Kubernetes before v1.25 allowed you to configure seccomp behavior using this annotation. See [Restrict a Container's Syscalls with seccomp](#) to learn the supported way to specify seccomp restrictions for a Pod.

snapshot.storage.kubernetes.io/allow-volume-mode-change

Type: Annotation

Example: `snapshot.storage.kubernetes.io/allow-volume-mode-change: "true"`

Used on: VolumeSnapshotContent

Value can either be `true` or `false`. This determines whether a user can modify the mode of the source volume when a PersistentVolumeClaim is being created from a VolumeSnapshot.

Refer to [Converting the volume mode of a Snapshot](#) and the [Kubernetes CSI Developer Documentation](#) for more information.

scheduler.alpha.kubernetes.io/critical-pod (deprecated)

Type: Annotation

Example: `scheduler.alpha.kubernetes.io/critical-pod: ""`

Used on: Pod

This annotation lets Kubernetes control plane know about a Pod being a critical Pod so that the descheduler will not remove this Pod.

Note:

Starting in v1.16, this annotation was removed in favor of [Pod Priority](#).

jobset.sigs.k8s.io/jobset-name

Type: Label, Annotation

Example: `jobset.sigs.k8s.io/jobset-name: "my-jobset"`

Used on: Jobs, Pods

This label/annotation is used to store the name of the JobSet that a Job or Pod belongs to. [JobSet](#) is an extension API that you can deploy into your Kubernetes cluster.

jobset.sigs.k8s.io/replicatedjob-replicas

Type: Label, Annotation

Example: `jobset.sigs.k8s.io/replicatedjob-replicas: "5"`

Used on: Jobs, Pods

This label/annotation specifies the number of replicas for a ReplicatedJob.

jobset.sigs.k8s.io/replicatedjob-name

Type: Label, Annotation

Example: `jobset.sigs.k8s.io/replicatedjob-name: "my-replicatedjob"`

Used on: Jobs, Pods

This label or annotation stores the name of the replicated job that this Job or Pod is part of.

jobset.sigs.k8s.io/job-index

Type: Label, Annotation

Example: `jobset.sigs.k8s.io/job-index: "0"`

Used on: Jobs, Pods

This label/annotation is set by the JobSet controller on child Jobs and Pods. It contains the index of the Job replica within its parent ReplicatedJob.

jobset.sigs.k8s.io/job-key

Type: Label, Annotation

Example: `jobset.sigs.k8s.io/job-key: "0f1e93893c4cb372080804ddb9153093cb0d20ceffd37f653e739c232d363feb"`

Used on: Jobs, Pods

The JobSet controller sets this label (and also an annotation with the same key) on child Jobs and Pods of a JobSet. The value is the SHA256 hash of the namespaced Job name.

alpha.jobset.sigs.k8s.io/exclusive-topology

Type: Annotation

Example: `alpha.jobset.sigs.k8s.io/exclusive-topology: "zone"`

Used on: JobSets, Jobs

You can set this label/annotation on a [JobSet](#) to ensure exclusive Job placement per topology group. You can also define this label or annotation on a replicated job template. Read the documentation for JobSet to learn more.

alpha.jobset.sigs.k8s.io/node-selector

Type: Annotation

Example: `alpha.jobset.sigs.k8s.io/node-selector: "true"`

Used on: Jobs, Pods

This label/annotation can be applied to a JobSet. When it's set, the JobSet controller modifies the Jobs and their corresponding Pods by adding node selectors and tolerations. This ensures exclusive job placement per topology domain, restricting the scheduling of these Pods to specific nodes based on the strategy.

alpha.jobset.sigs.k8s.io/namespaced-job

Type: Label

Example: `alpha.jobset.sigs.k8s.io/namespaced-job: "default_myjobset-replicatedjob-0"`

Used on: Nodes

This label is either set manually or automatically (for example, a cluster autoscaler) on the nodes. When `alpha.jobset.sigs.k8s.io/node-selector` is set to `"true"`, the JobSet controller adds a nodeSelector to this node label (along with the toleration to the taint `alpha.jobset.sigs.k8s.io/no-schedule` discussed next).

alpha.jobset.sigs.k8s.io/no-schedule

Type: Taint

Example: `alpha.jobset.sigs.k8s.io/no-schedule: "NoSchedule"`

Used on: Nodes

This taint is either set manually or automatically (for example, a cluster autoscaler) on the nodes. When `alpha.jobset.sigs.k8s.io/node-selector` is set to `"true"`, the JobSet controller adds a toleration to this node taint (along with the node selector to the label `alpha.jobset.sigs.k8s.io/namespace-job` discussed previously).

jobset.sigs.k8s.io/coordinator

Type: Annotation, Label

Example: `jobset.sigs.k8s.io/coordinator: "myjobset-workers-0-0.headless-svc"`

Used on: Jobs, Pods

This annotation/label is used on Jobs and Pods to store a stable network endpoint where the coordinator pod can be reached if the [JobSet](#) spec defines the `.spec.coordinator` field.

Annotations used for audit

- [authorization.k8s.io/decision](#)
- [authorization.k8s.io/reason](#)
- [insecure-shal.invalid-cert.kubernetes.io/\\$hostname](#)
- [missing-san.invalid-cert.kubernetes.io/\\$hostname](#)
- [pod-security.kubernetes.io/audit-violations](#)
- [pod-security.kubernetes.io/enforce-policy](#)
- [pod-security.kubernetes.io/exempt](#)
- [validation.policy.admission.k8s.io/validation_failure](#)

See more details on [Audit Annotations](#).

kubeadm

kubeadm.alpha.kubernetes.io/cri-socket (deprecated)

Type: Annotation

Example: `kubeadm.alpha.kubernetes.io/cri-socket: unix:///run/containerd/container.sock`

Used on: Node

Note:

Starting from v1.34, this annotation is deprecated, kubeadm will no longer actively set and use it.

kubeadm.kubernetes.io/etcd.advertise-client-urls

Type: Annotation

Example: `kubeadm.kubernetes.io/etcd.advertise-client-urls: https://172.17.0.18:2379`

Used on: Pod

Annotation that kubeadm places on locally managed etcd Pods to keep track of a list of URLs where etcd clients should connect to. This is used mainly for etcd cluster health check purposes.

kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint

Type: Annotation

Example: `kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: https://172.17.0.18:6443`

Used on: Pod

Annotation that kubeadm places on locally managed kube-apiserver Pods to keep track of the exposed advertise address/port endpoint for that API server instance.

kubeadm.kubernetes.io/component-config.hash

Type: Annotation

Example: `kubeadm.kubernetes.io/component-config.hash: 2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98a5e886266e7ae`

Used on: ConfigMap

Annotation that kubeadm places on ConfigMaps that it manages for configuring components. It contains a hash (SHA-256) used to determine if the user has applied settings different from the kubeadm defaults for a particular component.

node-role.kubernetes.io/control-plane

Type: Label

Used on: Node

A marker label to indicate that the node is used to run control plane components. The kubeadm tool applies this label to the control plane nodes that it manages. Other cluster management tools typically also set this taint.

You can label control plane nodes with this label to make it easier to schedule Pods only onto these nodes, or to avoid running Pods on the control plane. If this label is set, the [EndpointSlice controller](#) ignores that node while calculating Topology Aware Hints.

node-role.kubernetes.io/*

Type: Label

Example: `node-role.kubernetes.io/gpu: gpu`

Used on: Node

This optional label is applied to a node when you want to mark a node role. The node role (text following / in the label key) can be set, as long as the overall key follows the [syntax](#) rules for object labels.

Kubernetes defines one specific node role, **control-plane**. A label you can use to mark that node role is [node-role.kubernetes.io/control-plane](#).

node-role.kubernetes.io/control-plane

Type: Taint

Example: `node-role.kubernetes.io/control-plane:NoSchedule`

Used on: Node

Taint that kubeadm applies on control plane nodes to restrict placing Pods and allow only specific pods to schedule on them.

If this Taint is applied, control plane nodes allow only critical workloads to be scheduled onto them. You can manually remove this taint with the following command on a specific node.

```
kubectl taint nodes <node-name> node-role.kubernetes.io/control-plane:NoSchedule-
```

node-role.kubernetes.io/master (deprecated)

Type: Taint

Used on: Node

Example: `node-role.kubernetes.io/master:NoSchedule`

Taint that kubeadm previously applied on control plane nodes to allow only critical workloads to schedule on them. Replaced by the [node-role.kubernetes.io/control-plane](#) taint. kubeadm no longer sets or uses this deprecated taint.

resource.kubernetes.io/admin-access {resource-kubernetes-io-admin-access}

Type: Label

Example: `resource.kubernetes.io/admin-access: "true"`

Used on: Namespace

Used to grant administrative access to certain resource.k8s.io API types within a namespace. When this label is set on a namespace with the value "true" (case-sensitive), it allows the use of `adminAccess: true` in any namespaced `resource.k8s.io` API types. Currently, this permission applies to `ResourceClaim` and `ResourceClaimTemplate` objects.

See [Dynamic Resource Allocation Admin access](#) for more information.