容器服务

TKE 调度

产品文档

腾讯云

文档目录

TKE 调度
作业调度
原生节点专用调度器
产品介绍

最近更新时间：2024-04-24 15:55:45

简介

组件介绍

Kuberentes 的调度逻辑为**按照 Pod 的 Request 进行调度**。节点上的可调度资源会被 Pod 的 Request 量占用，且无法腾挪。原生节点专用调度器是容器服务 TKE 基于 Kubernetes 原生 Kube-scheduler Extender 机制实现的调度器插件，可以虚拟放大节点的容量，用来解决节点资源都被占用，但本身利用率很低的问题。

部署在集群内的 Kubernetes 对象

Kubernetes 对象名称	类型	请求资源	所属 Namespace
crane-scheduler-controller	Deployment	每个实例 CPU: 200m Memory：200Mi，共计1个实例	kube-system
crane-descheduler	Deployment	每个实例 CPU: 200m Memory：200Mi，共计1个实例	kube-system
crane-scheduler	Deployment	每个实例 CPU: 200m Memory：200Mi，共计3个实例	kube-system
crane-scheduler-controller	Service	-	kube-system
crane-scheduler	Service	-	kube-system
crane-scheduler	ClusterRole	-	kube-system
crane-descheduler	ClusterRole	-	kube-system

crane-scheduler	ClusterRoleBinding	-	kube-system
crane-descheduler	ClusterRoleBinding	-	kube-system
crane-scheduler-policy	ConfigMap	-	kube-system
crane-descheduler-policy	ConfigMap	-	kube-system
ClusterNodeResourcePolicy	CRD	-	-
CraneSchedulerConfiguration	CRD	-	-
NodeResourcePolicy	CRD	-	-
crane-scheduler-controller-mutating-webhook	MutatingWebhookConfiguration	-	-

应用场景

场景1：解决节点装箱率高但利用率低的问题

说明：

基本概念如下：

装箱率：节点上所有 Pod 的 Request 之和除以节点实际的规格。

利用率：节点上所有 Pod 的真实用量之和除以节点实际的规格。

Kubernetes 原生调度器是基于 Pod Request 资源进行调度，因此，即使此时节点上的真实用量很低，但节点上所有 Pod 的 Request 之和接近节点实际规格，也无法调度新的 Pod 上来，造成较大的资源浪费。并且，业务通常为了保证自己服务的稳定性，倾向申请过量的资源，即较大的 Request，导致节点资源被占用，无法腾挪。此时节点的装箱率很高，但实际资源利用率较低。

此时可以使用原生节点专用调度器，虚拟放大节点的 CPU 和内存的规格，使得节点可调度资源被虚拟放大，可以调度进更多的 Pod。

场景2：节点水位线的设置

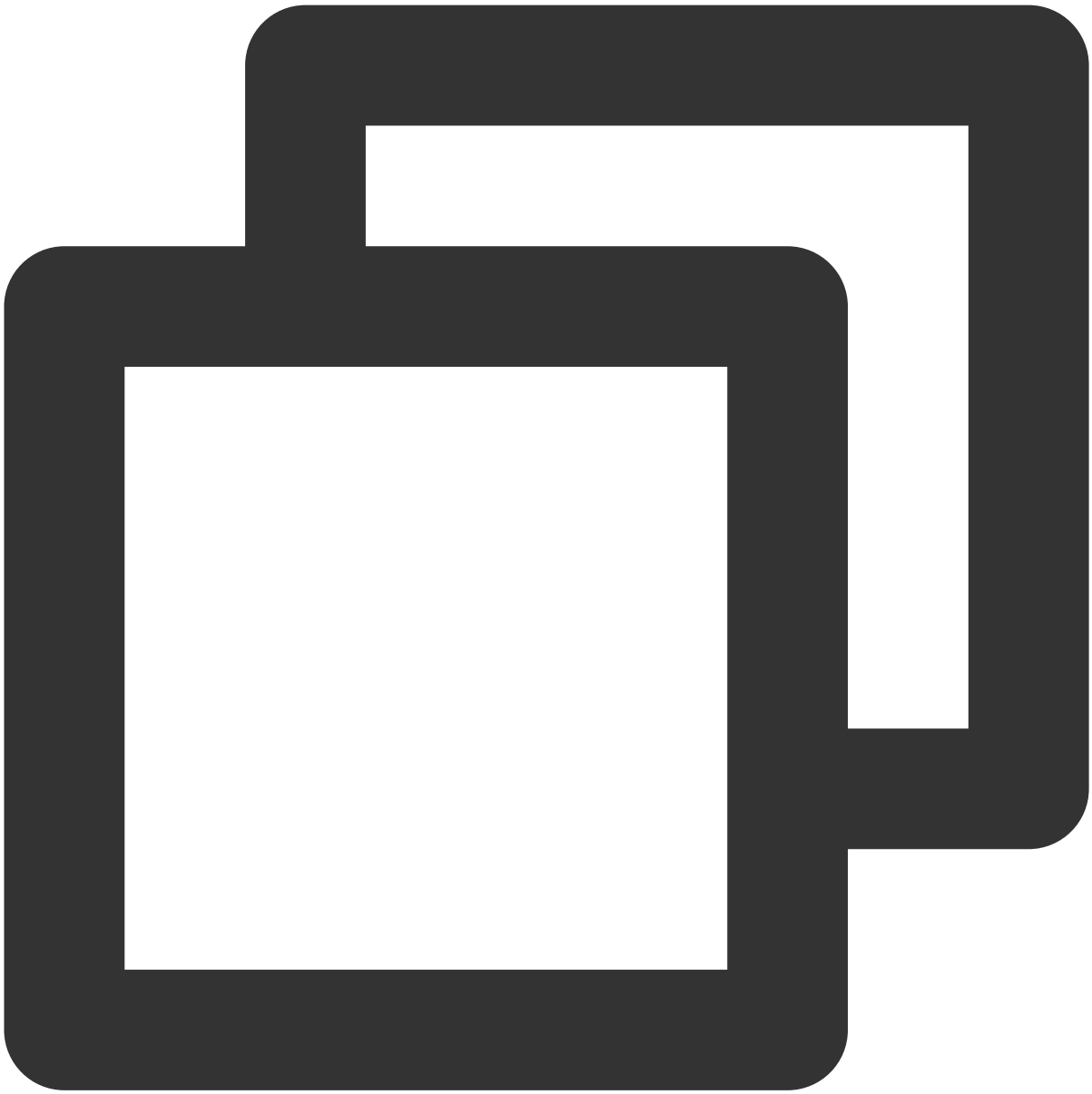
节点的水位线设置用于保障节点的稳定性的同时，设置节点的目标利用率：

调度时的水位控制：设定原生节点目标资源利用率，保障稳定性。Pod 调度时，高于该水位的节点将不被选中。并且，满足水位线的节点中，如下图所示，优先选择真实负载水位较低的节点，使得集群节点的利用率的分布更加均衡。

运行时的水位控制：设定当前原生节点目标资源利用率，保障稳定性。节点运行时，高于该水位的节点可能引发驱逐。因为驱逐本身是高危动作，需注意下述事项。

注意事项

1. 为避免驱逐关键的 Pod，该功能默认不驱逐 Pod。对于可以驱逐的 Pod，用户需要显示给判断 Pod 所属 workload。例如，statefulset、deployment 等对象设置可驱逐 annotation：



```
descheduler.alpha.kubernetes.io/evictable: 'true'
```

2. 建议为集群开启事件持久化，以便更好地监控组件异常以及故障定位。驱逐 Pod 时会产生对应事件，可根据 reason 为 “Descheduled” 类型的事件观察 Pod 是否被重复驱逐。
3. 驱逐动作对节点的要求：集群需要有3个及以上的低负载原生节点，其中低负载定义是 Node 负载小于运行时的水位控制。

4. 节点维度过滤后，开始驱逐 Node 上 Workload，此时为 Workload 级别的过滤，需要工作负载的副本数大于等于2或者是 Workload spec replicas 的一半及以上。
5. 在 Pod 维度上，如果 Pod 的负载大于节点的驱逐水位，则不可驱逐，以避免驱逐到其他节点造成其他节点负载过高。

场景3：指定命名空间下的 Pod 在下次调度时只调度到原生节点上

原生节点是由腾讯云 TKE 容器服务团队面向 Kubernetes 环境推出的全新节点类型，依托腾讯云千万核容器运维的技术沉淀，为用户提供原生化、高稳定、快响应的 K8s 节点管理能力。原生节点支持节点规格放大，Request 推荐等能力，因此为了充分利用原生节点的优势，建议您将工作负载调度到原生节点上。在开启原生节点调度器时，您可以选择命名空间，指定命名空间下的 Pod 在下次调度时只调度到原生节点上。

注意：

若此时原生节点资源不足，会导致 Pod Pending。

限制条件

该功能仅支持原生节点，更多请参考 原生节点概述。
需要保证 Kubernetes 版本为 v1.22.5-tke.8，v1.20.6-tke.24，v1.18.4-tke.28，v1.16.3-tke.30 及以上。集群版本升级请参考 升级集群。

风险控制

该组件卸载后，只会删除原生节点专用调度器有关调度逻辑，不会对原生 Kube-Scheduler 的调度功能有任何影响。
原生节点上存量的 Pod 因为已经调度，不受影响。但原生节点上 kubelet 重启时，可能会引发 Pod 的驱逐，因为此时可能原生节点上 Pod 的 Request 之和可能大于原生真实的规格。
当调低放大系数时，原生节点上存量的 Pod 因为已经调度，不受影响。但原生节点上 kubelet 重启时，可能会引发 Pod 的驱逐，因为此时可能原生节点上 Pod 的 Request 之和可能大于原生节点当前放大后的规格。
用户看到 Kubernetes 集群中的 Node 资源和对应的 CVM 节点资源不一致。
未来可能会发生负载过高、稳定性问题。
节点规格放大以后，节点 kubelet 层和资源 QoS 相关模块可能受到影响，例如 kubelet 绑核，原来4核的节点被当做8核调度，Pod 绑核可能受到影响。

组件权限说明

Crane scheduler 权限

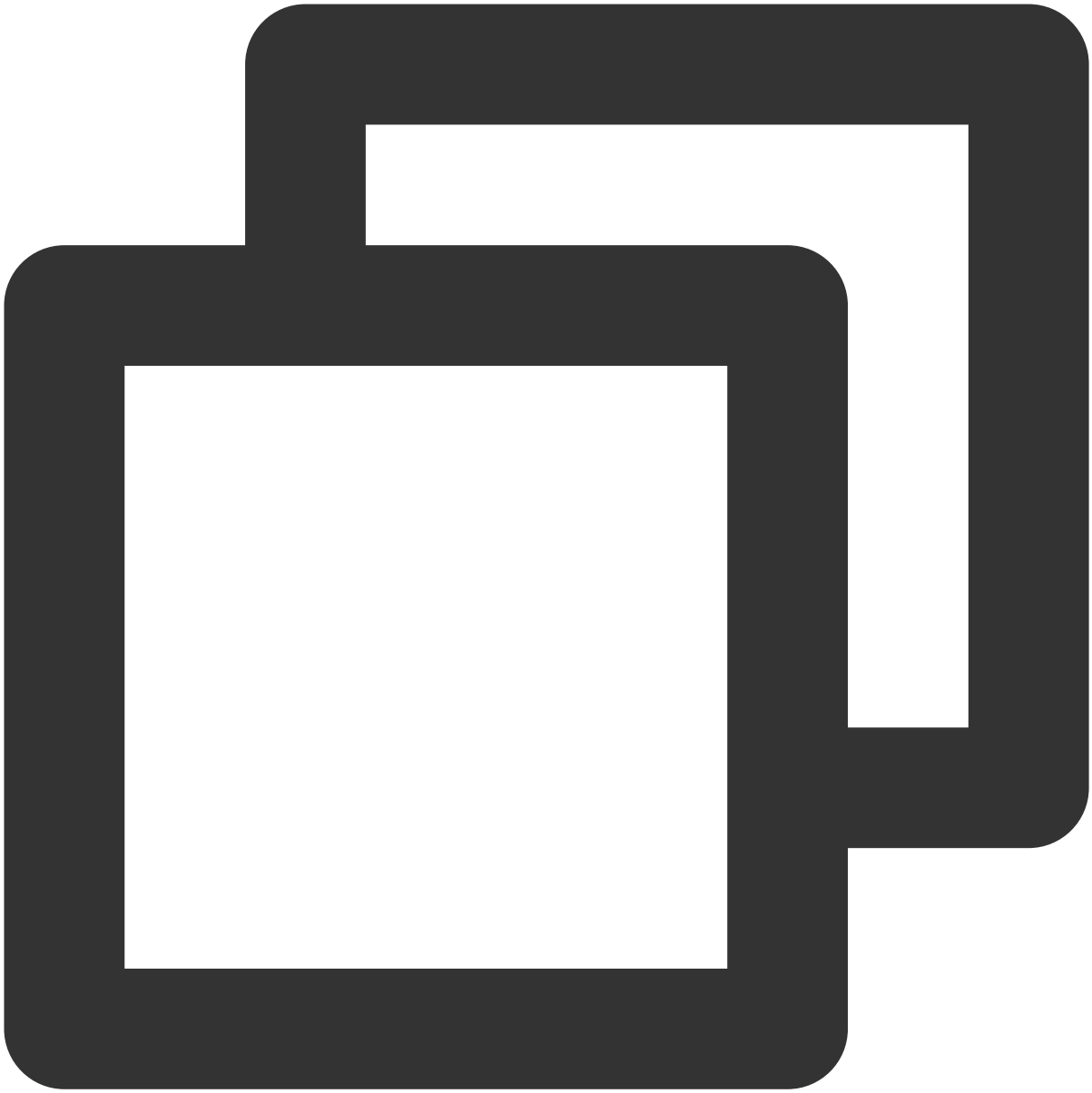
权限说明

该组件权限是当前功能实现的最小权限依赖。

权限场景

功能	涉及对象	涉及操作权限
需要跟踪节点的更新及变化，获取节点利用率。	nodes	get/watch/list
跟踪pod的更新及变化，根据集群内pod最近调度情况决定节点调度优先级。	pods/namespaces	get/watch/list
需要更新节点利用率到节点资源，实现调度及查询逻辑的解耦。	nodes/status	patch
需要支持多副本保障组件可用性 。	leases	create/get/update
需要跟踪 configmap 的更新及变化，实现指定 pod 调度到原生节点的功能。	configmap	get/list/watch

权限定义



```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: crane-scheduler
rules:
- apiGroups:
  - ""
  resources:
  - pods
  - nodes
  - namespaces
```

```
  verbs:
  - list
  - watch
  - get
- apiGroups:
  - ""
  resources:
  - nodes/status
  verbs:
  - patch
- apiGroups:
  - ""
  resources:
  - configmaps
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - extensions
  - apps
  resources:
  - deployments/scale
  verbs:
  - get
  - update
- apiGroups:
  - coordination.k8s.io
  resources:
  - leases
  verbs:
  - create
  - get
  - update
- apiGroups:
  - "scheduling.crane.io"
  resources:
  - clusternoderesourcepolicies
  - noderesourcepolicies
  - craneschedulerconfigurations
  verbs:
  - get
  - list
  - watch
  - update
  - create
  - patch
```

Crane descheduler 权限

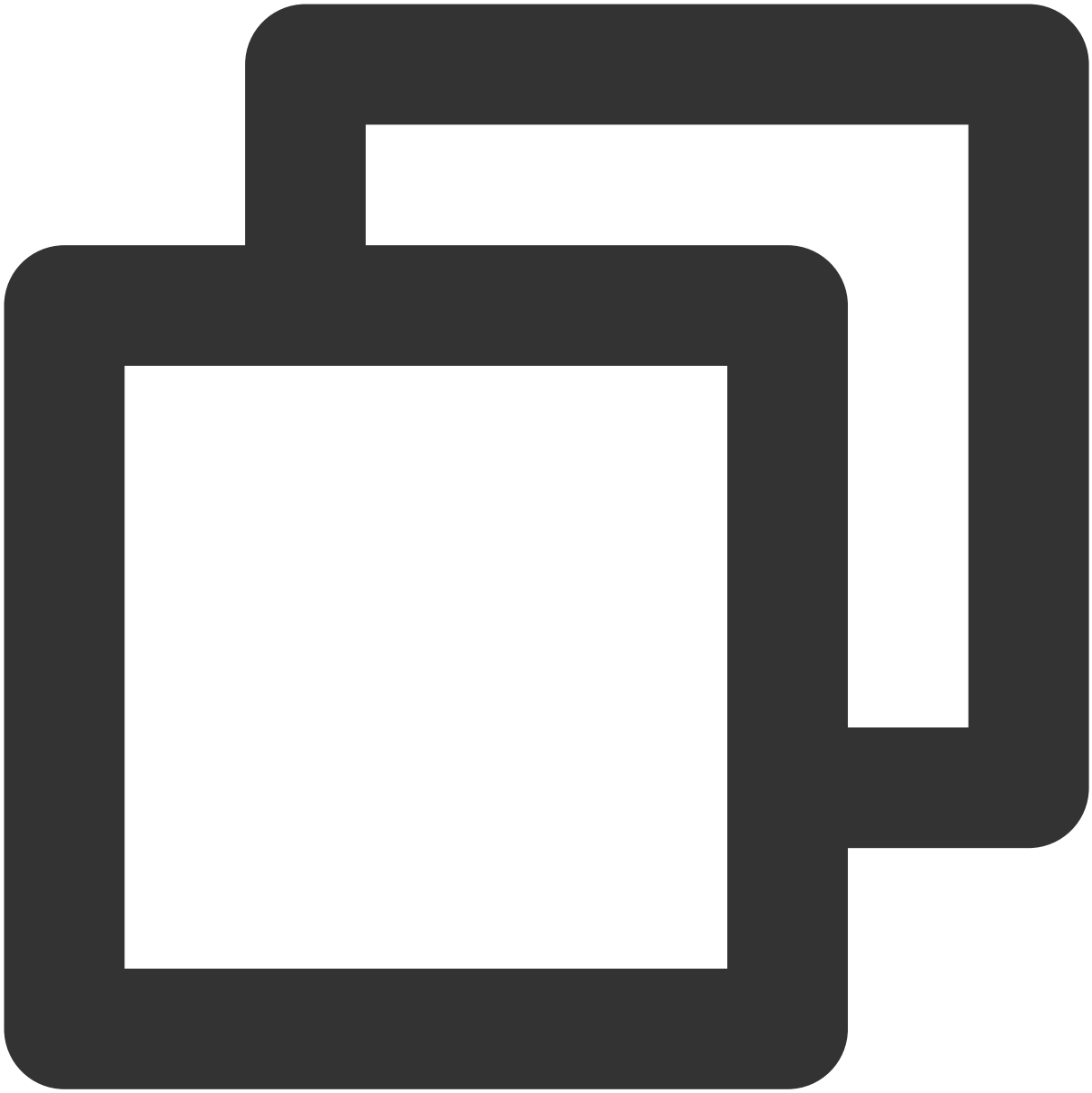
权限说明

该组件权限是当前功能实现的最小权限依赖。

权限场景

功能	涉及对象	涉及操作权限
需要跟踪节点的更新及变化，获取节点利用率。	nodes	get/watch/list
跟踪 pod 的更新及变化，根据集群内 pod 信息决定优先驱逐的 pod。	pods	get/watch/list
驱逐 pod。	pods/eviction	create
需要得知 pod 所在的 workload ready 的个数是否占总需求数的一半及以上，来决定是否驱逐 pod。	replicasets/deployments/statefulsets/statefulsetpluses/job	get
驱逐 pod 时需要上报事件。	create	events

权限定义



```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: crane-descheduler
  namespace: kube-system
rules:
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "watch", "list"]
  - apiGroups: [""]
    resources: ["pods"]
```

```
    verbs: ["get", "watch", "list"]
  - apiGroups: [""]
    resources: ["nodes/status"]
    verbs: ["patch"]
  - apiGroups: [""]
    resources: ["pods/eviction"]
    verbs: ["create"]
  - apiGroups: ["*"]
    resources: ["replicasets"]
    verbs: ["get"]
  - apiGroups: ["*"]
    resources: ["deployments"]
    verbs: ["get"]
  - apiGroups: ["apps"]
    resources: ["statefulsets"]
    verbs: ["get"]
  - apiGroups: ["platform.stke"]
    resources: ["statefulsetpluses"]
    verbs: ["get"]
  - apiGroups: [""]
    resources: ["events"]
    verbs: ["create"]
  - apiGroups: ["*"]
    resources: ["jobs"]
    verbs: ["get"]
  - apiGroups: [ "coordination.k8s.io" ]
    resources: [ "leases" ]
    verbs: [ "get", "create", "update" ]
```

Request 智能推荐

最近更新时间：2023-07-26 10:12:11

简介

组件介绍

Kubernetes 可以有效的提升业务编排能力和资源利用率，但如果没有额外的能力支撑，提升的能力十分有限，根据 TKE 团队之前统计的数据，TKE 节点的资源平均利用率也只有14%左右。
Kubernetes 集群的资源利用率不高的主要原因是根据 Kubernetes 的资源调度逻辑，在创建 Kubernetes 工作负载时，通常需要为工作负载配置合适的资源 Request 和 Limit，表示对资源的占用和限制，其中对利用率影响最大的是 Request。为防止自己的工作负载所用的资源被别的工作负载所占用，或者是为了应对高峰流量时的资源消耗诉求，用户习惯于为 Request 设置较大的数值，Request 和实际使用资源之间的差值，是不能被其它工作负载所使用的，因此造成了浪费。Request 数值设置不合理，造成了 Kubernetes 集群资源利用率低。
容器服务 TKE 支持在集群中安装 **Request 智能推荐**组件。Request 智能推荐可以为 Kubernetes 的 Workload 推荐容器级别资源的 Request/Limit 数值，减少资源浪费。

部署在集群内的资源对象

开启集群的 Request 智能推荐 , 将在集群内部署以下 Kubernetes 对象：

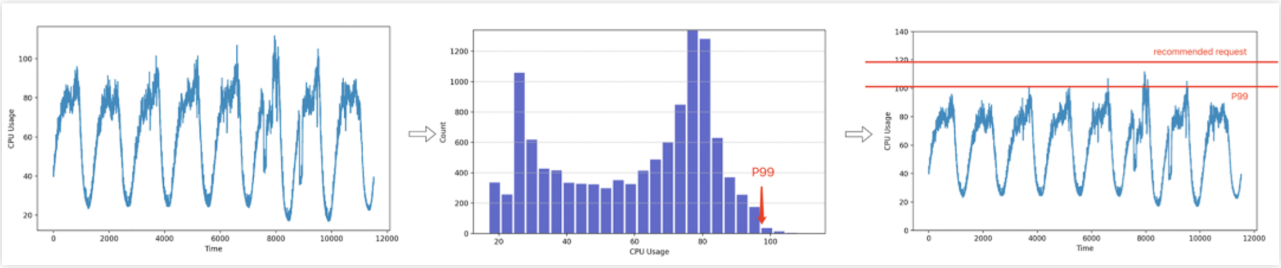
Kubernetes 对象名称	类型	默认占用资源	所属 Namespaces
analytics.analysis.crane.io	CustomResourceDefinition	-	-
recommendations.analysis.crane.io	CustomResourceDefinition	-	-
crane-system	Namespace	-	-
housekeeper-default	Analytics	-	crane-system
recommendation-config	ConfigMap	-	crane-system
craned	ClusterRole	-	-
craned	ClusterRoleBinding	-	-
craned	Service	-	crane-system
craned	ServiceAccount	-	crane-system
craned	Deployment	-	crane-system

功能说明

支持为 Deployment、StatefulSet、DaemonSet 中的每一个 Container 智能推荐合适的资源 Request/Limit。
支持一键更新：使用推荐值一键更新 Workload 中 Container 的资源值。
支持维持 Request/Limit 比例：推荐的 Request/Limit 会维持初始 Workload 中 Container 设置的 Reqeust/Limit 之间的比例，若 Limit 在创建 Workload 时没有设置，则不会推荐 Limit。
控制台的 Request 推荐一键更新能力会默认给工作负载加上 nodeSelector 的属性，Workload 在更新时， Pod 将只能调度到原生节点上，若原生节点资源不足，会引发 Pod 的 Pending。

Request 推荐原理

组件在 crane-system 命名空间下创建 Analytics CR 对象，覆盖所有集群中的所有 Kubernetes 原生工作负载（Deployment、DaemonSet、StatefulSet），会分析工作负载最长 14 天的监控数据数据，12 小时更新一次推荐值。
然后根据 Analytics 生成集群内每个工作负载的 Recommendation CR 对象，用于存储推荐的数据。
Recommendation CR 如果产生了推荐数据，就会把推荐数据写入到对应工作负载的 Annotation 里。



注意事项

环境要求

Kubernetes 版本：1.10+

节点要求

容器服务控制台中**一键更新 Workload Request** 功能会将工作负载迁移至 原生节点，若您的集群原生节点上资源不足，会导致 Pod 发生 Pending。

被控资源要求

支持 Deployment、StatefulSet、DaemonSet。

不支持 Job、CronJob，不支持不是由 Workload 管理的 Pod。

推荐阈值

推荐最小值：单个容器推荐的 CPU 最小值是0.125核，即125m；内存的最小值是125Mi。

使用说明

安装组件

1. 登录 容器服务控制台。
2. 在左侧选择 **TKE Insight > Node Map**。

说明：

您也可以在 **TKE Insight > Workload Map** 中进行安装。

3. 在 Node Map 页面中，鼠标悬浮到页面下方某一个 Node 上，单击**详情**。
4. 在该 Node 的详情页的右上角，打开“Request 推荐” 开关，进行调度器的参数配置。

注意：

该功能是集群级别的全局开关，开启后，会自动分析工作负载历史的监控数据，推荐合适的 Request 数值。
开启后非立即生效，为准确计算推荐值，需要分析该 Workload 的历史资源使用数据。
不同的 Workload 的计算时间长度可能不一致，集群中不同的 Workload 之间互相可能会有影响。
开启该功能后，对至少运行一天的 Workload 产生推荐数据。
对于开启功能后新建的 Workload，一般情况下，也需要一天的时间才会产生 Workload 的推荐数据。
建议工作负载稳定运行一段时间之后，再使用推荐值更新 Workload。

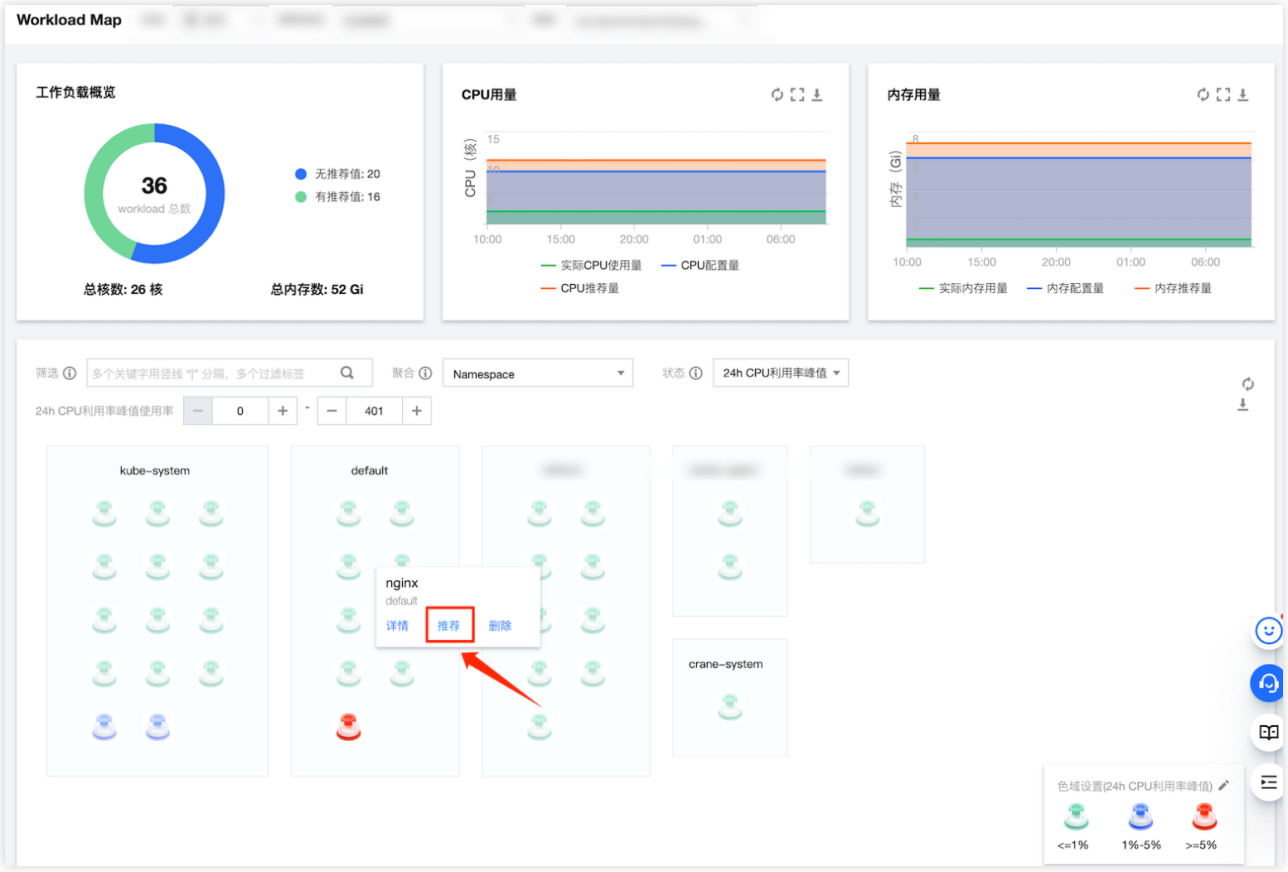
使用组件

1. 登录 容器服务控制台。
2. 在左侧选择 **TKE Insight > Workload Map**。

说明：

Workload Map 主要通过可视化的页面展示工作负载的各项状态和指标，帮助用户了解当前工作负载的配置量和实际使用情况，辅助用户分析工作负载可能存在的问题。更多可参考文档 Workload Map。

3. 在 Workload Map 页面，鼠标悬浮到页面下方某一个 Workload 上，单击**推荐**。



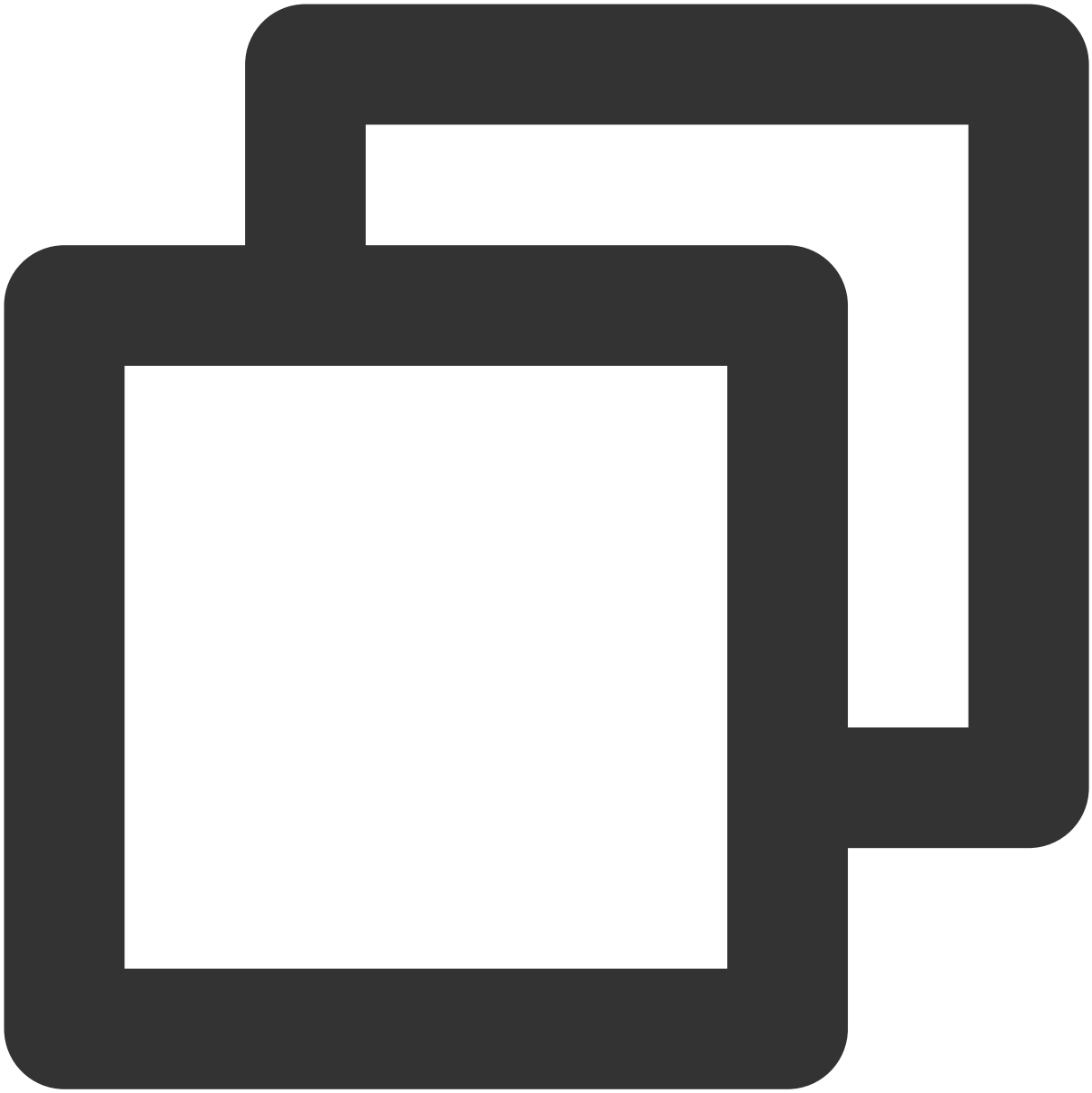
4. 在弹窗中，单击**确认**，即可使用推荐的 Request 数值更新原始 Workload 里面的数值。

说明：

容器服务控制台中**一键更新 Workload Request** 功能会将工作负载迁移至 原生节点，若您的集群原生节点上资源不足，会导致 Pod 发生 Pending。

后台获取推荐数值

Request 智能推荐组件会将每个工作负载的推荐值保存在该工作负载的 YAML 里，您可以通过标准的 Kuberentes API 获取每个工作负载的推荐值，然后集成到业务的发布系统中。如下所示查看工作负载下每个容器的 Request 推荐量：



```
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    analysis.crane.io/resource-recommendation: |
      containers:
      # 若一个 Pod 里有多个容器，每个容器都有 CPU 和 Memory 的 Request 的推荐值
      - containerName: nginx
        target:
          cpu: 125m
          memory: 125Mi #若这里缺少单位，显示的是字符串"58243235"，省略的单位是byte
```

注意：

组件本身不会推荐 Limit，在控制台使用 Request 推荐值更新 Workload 时，会维持该 Workload Request 和 Limit 的比值以保证 QoS 不会发生变化。您如果在后台获取到 Request 推荐值，可以作为参考更新原始 Workload 的资源配置量。

组件权限说明

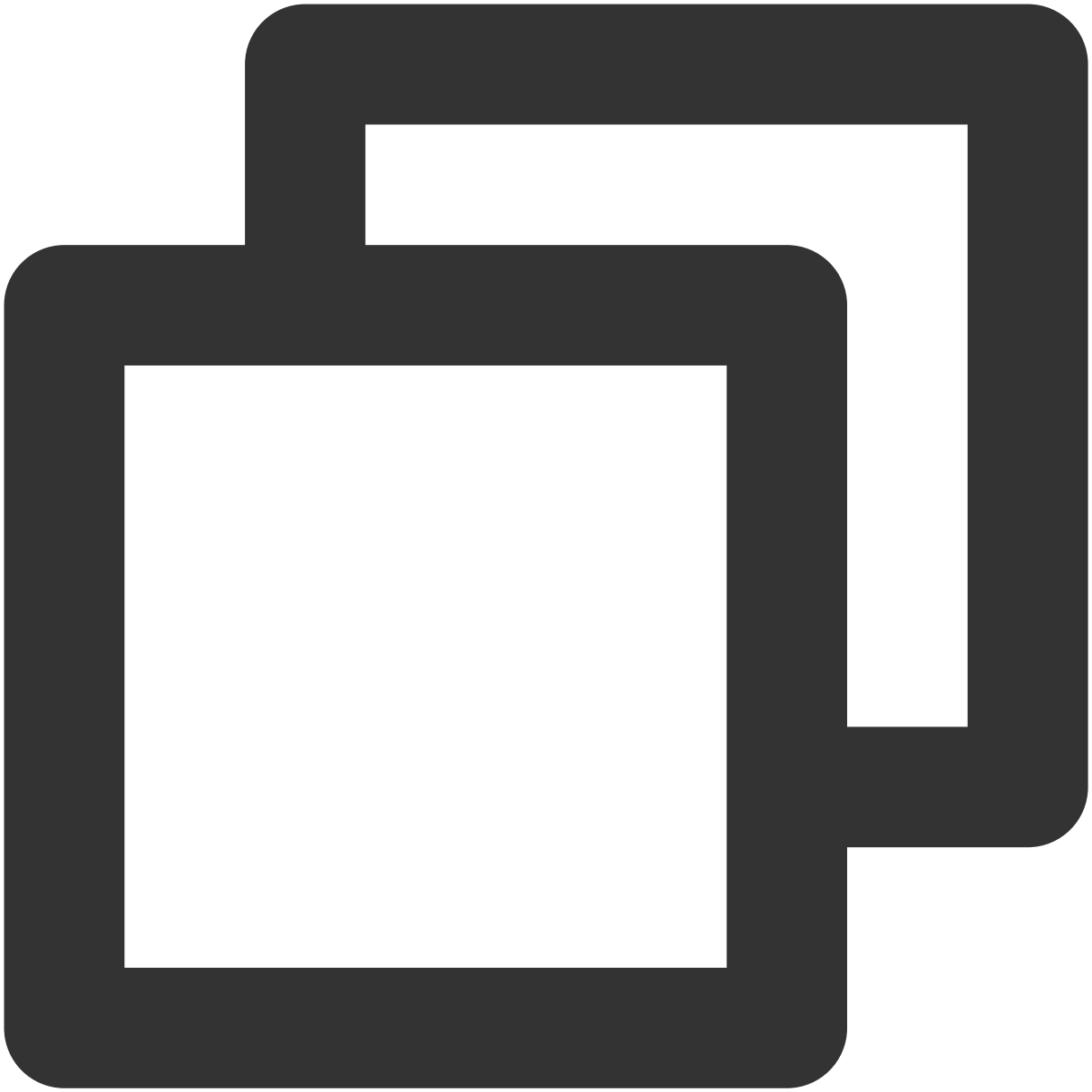
权限说明

该组件权限是当前功能实现的最小权限依赖。

权限场景

功能	涉及对象	涉及操作权限
记录 pod 的 oom 记录	pod	get/list/watch
根据 node 查找空闲节点并进行推荐	node	get/list/watch
需要更新 workload 的 annotations 记录推荐相关的信息	workload	get/list/watch
需要将异常信息以事件进行记录	event	create/patch/update
监听推荐相关资源的变更，进行资源推荐	analysis.crane.io	所有权限

权限定义



```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: craned
rules:
- apiGroups:
  - ""
  resources:
  - configmaps
  - pods
  - nodes
```

```
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - analysis.crane.io
  resources:
  - "*"
  verbs:
  - "*"
- apiGroups:
  - apps
  resources:
  - daemonsets
  - deployments
  - deployments/scale
  - statefulsets
  - statefulsets/scale
  verbs:
  - get
  - list
  - watch
- apiGroups:
    - apps
  resources:
    - daemonsets/status
    - deployments/status
    - deployments/scale
    - statefulsets/status
    - statefulsets/scale
  verbs:
    - update
- apiGroups:
  - autoscaling
  resources:
  - horizontalpodautoscalers
  verbs:
  - '*'
- apiGroups:
  - autoscaling.crane.io
  resources:
  - '*'
  verbs:
  - '*'
- apiGroups:
  - ""
  resources:
```

```
  - events
  verbs:
  - create
  - patch
  - update
- apiGroups:
  - prediction.crane.io
  resources:
  - '*'
  verbs:
  - '*'
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: craned
  namespace: crane-system
rules:
  - apiGroups:
      - ""
    resources:
      - configmaps
      - secrets
    verbs:
      - create
  - apiGroups:
      - ""
    resourceNames:
      - craned
    resources:
      - configmaps
    verbs:
      - get
      - patch
      - update
  - apiGroups:
      - ""
    resourceNames:
      - clusters-secret-store
    resources:
      - secrets
    verbs:
      - get
  - apiGroups:
      - coordination.k8s.io
    resources:
      - leases
```

```
    verbs:
      - get
      - patch
      - update
      - create
```

精细调度

QoSAgent

最近更新时间：2024-02-05 16:29:25

QoS Agent 是腾讯云基于服务质量增强的扩展组件。提供丰富的能力，在提升集群资源利用率的同时，提供稳定性质量保障。

注意：

QoS 相关的能力仅支持在 原生节点 上使用，若您的节点不是原生节点，或工作负载不在原生节点上，相关能力无法生效。

部署在集群内的 Kubernetes 对象

Kubernetes 对象名称	类型	默认占用资源	所属 Namespaces
avoidanceactions.ensurance.crane.io	CustomResourceDefinition	-	-
nodeqoss.ensurance.crane.io	CustomResourceDefinition	-	-
podqoss.ensurance.crane.io	CustomResourceDefinition	-	-
timeseriespredictions.prediction.crane.io	CustomResourceDefinition	-	-
kube-system	Namespace	-	-
all-be-pods	PodQOS	-	kube-system
qos-agent	ClusterRole	-	-
qos-agent	ClusterRoleBinding	-	-
crane-agent	Service	-	kube-system
qos-agent	ServiceAccount	-	kube-system
qos-agent	Daemonset	-	kube-system

功能说明

功能	说明
CPU 使用优先级	CPU 使用优先级的功能可以通过对工作负载设置优先级，保证高优先级业务在发生资源竞争时的资源供给量，并压制低优先级业务。
CPU Burst	CPU Burst 可以临时给延迟敏感型应用提供超过 Limit 数量的资源，保证其稳定性。
CPU 超线程隔离	避免高优先级容器线程的 L2 Cache 受到运行在同一个 CPU 物理核上的低优先级线程的影响。
内存 QoS 增强	全方位提升内存表现，以及灵活限制容器对内存的使用。
网络 QoS 增强	全方位提升网络表现，以及灵活限制容器对网络的使用。
磁盘 IO QoS 增强	全方位提升磁盘表现，以及灵活限制容器对磁盘的使用。

QoS Agent 权限

说明：

权限场景章节中仅列举了组件核心功能涉及到的相关权限，完整权限列表请参考**权限定义**章节。

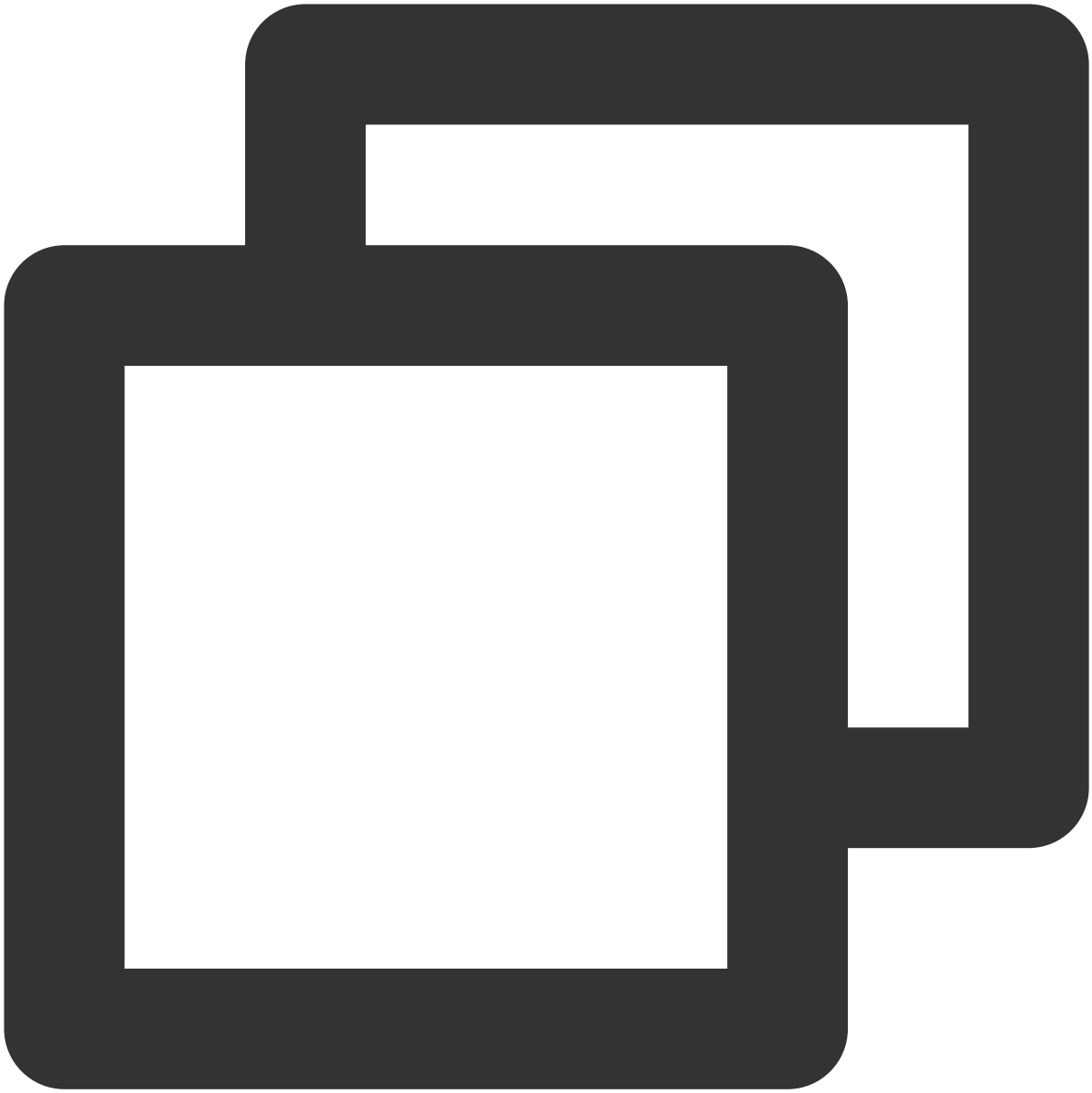
权限说明

该组件权限是当前功能实现的最小权限依赖。

权限场景

功能	涉及对象	涉及操作权限
读取 podqos、nodeqos、时间序列等配置。	podqoss / nodeqoss / avoidanceactions	get/list/watch/update
查看当前节点的 pod 信息。	pod	get/list/watch
根据 Podqos 开启隔离能力 / 修改 node resource 以增加离线资源。	pod status	update/patch
给 node 添加污点。	node	get/list/watch/update
根据隔离开启情况、资源干扰情况发送 event。	event	所有权限

权限定义



```
rules:
 - apiGroups:
     - ""
   resources:
     - pods
   verbs:
     - get
     - list
     - watch
 - apiGroups:
     - ""
```

```
    resources:
      - pods/status
    verbs:
      - update
      - patch
  - apiGroups:
      - ""
    resources:
      - nodes
    verbs:
      - get
      - list
      - watch
      - update
  - apiGroups:
      - ""
    resources:
      - nodes/status
      - nodes/finalizers
    verbs:
      - update
      - patch
  - apiGroups:
      - ""
    resources:
      - pods/eviction
    verbs:
      - create
  - apiGroups:
      - ""
    resources:
      - configmaps
    verbs:
      - get
      - list
      - watch
  - apiGroups:
      - ""
    resources:
      - events
    verbs:
      - "*"
  - apiGroups:
      - "ensurance.crane.io"
    resources:
      - podqoss
      - nodeqoss
```

```
      - avoidanceactions
    verbs:
      - get
      - list
      - watch
      - update
  - apiGroups:
      - "prediction.crane.io"
    resources:
      - timeseriespredictions
      - timeseriespredictions/finalizers
    verbs:
      - get
      - list
      - watch
      - create
      - update
      - patch
  - apiGroups:
      - "topology.crane.io"
    resources:
      - "noderesourcetopologies"
    verbs:
      - get
      - list
      - watch
      - create
      - update
      - patch
```

部署方式

1. 登录 容器服务控制台，在左侧导航栏中选择**集群**。
2. 在集群列表中，单击目标集群 ID，进入集群详情页。
3. 选择左侧菜单栏中的**组件管理**，在组件管理页面单击**新建**。
4. 在**新建组件管理**页面中勾选 **QoS Agent**。
5. 单击**完成**即可安装组件。

注意：

在部署完成之后，因为集群的 cgroup 驱动可能不同，需要您手动选择对应的驱动。操作方式如下：

1. 在集群里的**扩展组件**里，找到部署成功的 QoS Agent，单击右侧的**更新配置**。
2. 在修改 QoS Agent 的组件配置页面，选择 cgroupDrive 选项右侧的下拉框，选择与自己集群匹配的 cgroupDrive。
3. 单击**完成**。

常见问题

如何确认自己集群的 cgroupDrive？

集群的 cgroupDrive 只可能是 cgroupfs 或 systemd。确认方式如下：
首先查看集群的运行时，可以在集群的“基本信息”页里的“运行时组件”判断当前集群是 docker 还是 containerd。
如果运行时是 docker 的话，在集群的任意节点上执行 `docker info` 查看 `Cgroup Driver` 的字段内容。
如果运行时是 containerd，在集群的任意节点的 /etc/containerd/config.toml 文件里，如果字段：`SystemdCgroup = true`，代表是 systemd，否则是 cgroup。

如何选择作用的业务或者节点？

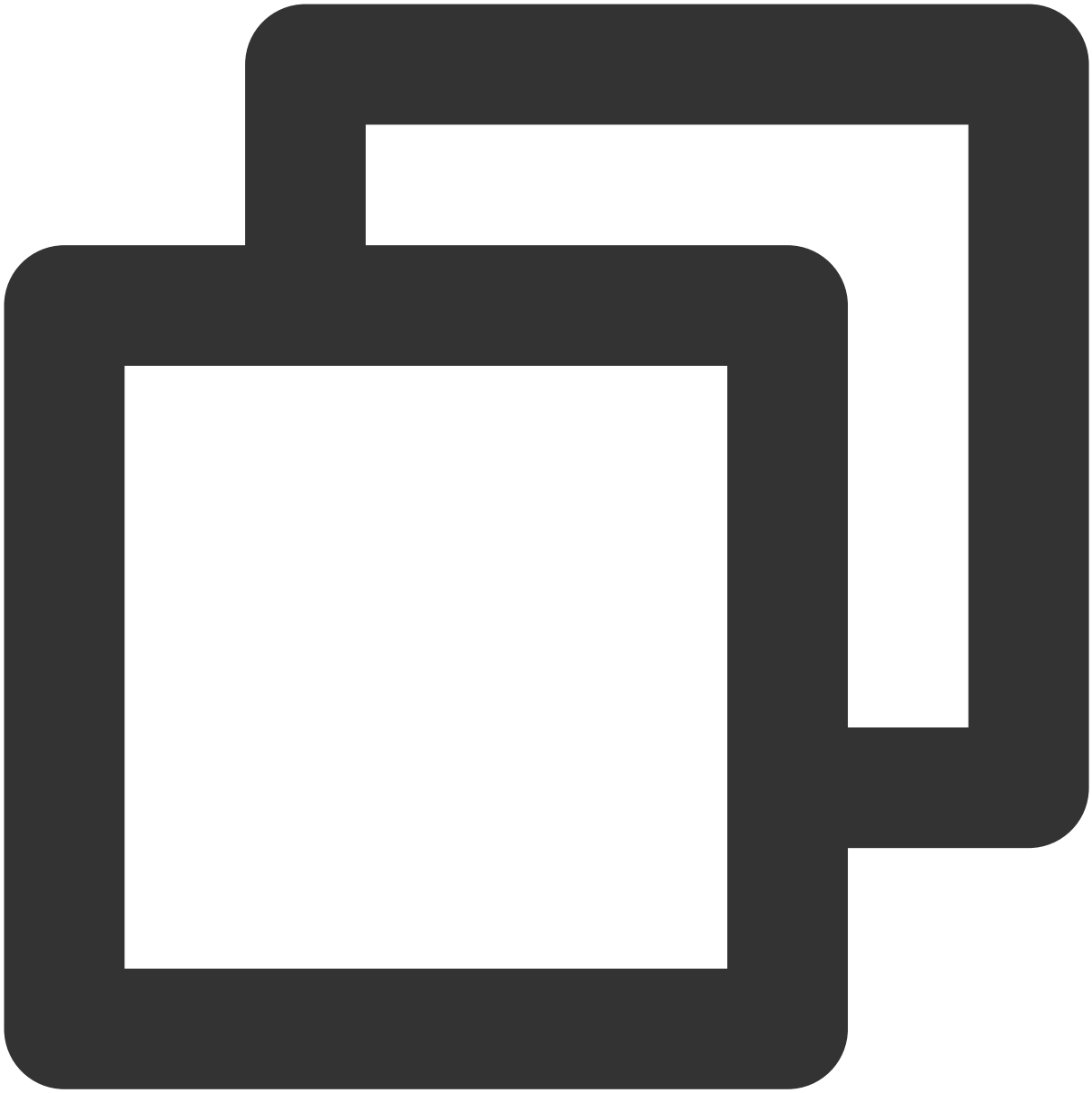
支持通过 `label` 或者 `scope` 选择某种资源对象。
注意：
当同时存在下面两种 selector 的时候，会取“与”，也即满足所有条件。

labelSelector

labelSelector 通过关联资源对象的 label 对资源对象进行筛选，常用的使用方式是，业务侧在指定的工作负载上打上特定的标签，并将该标签提供给运维侧，运维在创建 PodQOS 时通过 labelSelector 字段关联该标签，即可赋予不同的业务不同的 QoS 能力。

scopeSelector

scopeSelector 由多个 MatchExpressions 组成，这些 MatchExpressions 之间是“与”的关系，MatchExpressions 中有三个字段，分别是 ScopeName，Operator 和 ScopeName 对应的 Values；
其中 ScopeName 包括 QOSClass，Priority，Namespace 三种；
QOSClass 是指希望关联具有特定的 QOSClass 的 Workload，Values 可以填：Guaranteed，Burstable，BestEffort 中的一种或多种；
Priority 是指希望关联具有特定的 Priority 的 workload，Values 可以填特定的 priority 数值，如["1000", "2000-3000"]，支持 priority 范围；
Namespace 是指希望关联特定的 Namespace 的 Workload，Values 可以填一个或多个。
Operator 包含两种，分别是 In 和 NotIn，不填默认为 In。
如下述示例，表示将满足 app-type=offline 的 BestEffortPod，CPU 优先级设置为7：



```
apiVersion: ensurance.crane.io/v1alpha1
kind: PodQOS
metadata:
  name: offline-task
spec:
  allowedActions:
  - eviction
  resourceQOS:
    cpuQOS:
      cpuPriority: 7
  scopeSelector:
```

```
    matchExpressions:
    - operator: In
      scopeName: QOSClass
      values:
      - BestEffort
  labelSelector:
    matchLabels:
      app-type: offline
```