容器服务

TKE Serverless 集群指南

产品文档

腾讯云

文档目录

TKE Serverless 集群指南
TKE Serverless 集群概述
TKE Serverless 集群

最近更新时间：2023-08-10 15:43:31

什么是 TKE Serverless 集群？

TKE Serverless 集群是腾讯云容器服务推出的无须用户购买节点即可部署工作负载的服务模式。TKE Serverless 集群完全兼容原生 Kubernetes，支持使用原生方式购买及管理资源，按照容器真实使用的资源量计费。TKE Serverless 集群还扩展支持腾讯云的存储及网络等产品，同时确保用户容器的安全隔离，开箱即用。

相关概念

容器和镜像

容器是进程级别的虚拟化技术，将系统资源进行隔离和控制，让原来全局的资源仅能在容器内进程使用。镜像类似于轻量化的虚拟机快照，也可理解为容器的静态形式。镜像定义了容器运行的一切文件和依赖关系，保证了容器运行的一致性。
容器技术通过把应用程序及其依赖全部打包成镜像，再使用镜像生成资源隔离的环境来运行程序，简单高效的实现了应用程序运行时的独立性和环境一致性。

Kubernetes

Kubernetes 是 Google 基于 Borg 开源的容器编排调度引擎，是 CNCF（Cloud Native Computing Foundation）最重要的组件之一。提供了生产级别的应用编排、容器调度、服务发现、自动扩缩容等功能，详情请参阅 Kubernetes 官方文档。

产品优势

原生支持

TKE Serverless 集群紧跟社区，支持最新的 Kubernetes 版本及原生的 Kubernetes 集群管理方式。以插件的形式扩展支持腾讯云系列产品，例如存储、网络、负载均衡等服务，开箱即用。

无服务器

TKE Serverless 集群是一种全托管的 Kubernetes 服务，意味着用户无须管理任何计算节点。TKE Serverless 集群以 Pod 的形式交付计算资源，支持用户使用 Kubernetes 原生的方式购买、退还及管理云资源。

安全可靠

基于腾讯云成熟的虚拟化技术和网络架构，提供99.95%以上的服务可用性。腾讯云保证用户间 TKE Serverless 集群的虚拟化隔离和网络隔离，支持用户通过安全组、网络 ACL 等产品为具体服务配置网络策略。

秒级伸缩

通过腾讯云自研的轻量虚拟化技术，确保更快的资源创建效率，用户可秒级创建或删除容器服务。TKE Serverless 集群支持设置 Kubernetes 原生 HPA 的方式，可让服务根据实际负载进行自动伸缩。

降低成本

无服务器的形态决定了 TKE Serverless 集群能为用户带来更高的资源利用率和更低的运维成本，灵活高效的弹性伸缩能力保证容器服务仅会使用当前负载需要的资源量。

服务集成

TKE Serverless 集群能够和腾讯云的大部分业务做到高度集成，例如存储产品云硬盘 CBS、文件存储 CFS 及对象存储 COS、云数据库 TencentDB 系列产品、私有网络 VPC 系列产品等，提供了满足各类业务需求的解决方案。

使用限制

详情请参考 购买限制 和 资源规格。

产品定价

TKE Serverless 集群是全托管的无服务器 Kubernetes 服务，提供三种类型的计费模式：预留券、按量计费和竞价模式。详情请参考 产品定价。

与容器服务的对比

特征	TKE 标准集群	TKE Serverless 集群
Kubernetes	原生支持	原生支持，存在由无计算节点造成的功能缺省，详情请参见 注意事项
私有网络	支持	

特征	TKE 标准集群	TKE Serverless 集群
计算节点	需要用户自主购买并管理 CVM，裸金属等计算节点	无需购买节点
管理方式	支持原生 Kubernetes API、Kubectl 等方式	
集群	支持创建、管理多个集群	
命名空间	原生支持	
工作负载	原生支持	支持除 DaemonSet 之外的其他原生工作负载类型
服务	原生支持，集成 CLB 插件	
存储	原生支持，集成 CBS、CFS 等插件	

应用场景

微服务场景

使用 TKE Serverless 集群来运行微服务，免除用户对计算节点的运维工作。服务可根据负载情况自动伸缩，使用最合理的资源量来承载应用，降低资源使用成本。

离线计算场景

使用 TKE Serverless 集群运行离线计算任务，只需准备容器镜像，即可快速部署任务负载。另外， TKE Serverless 集群仅收取任务真实运行时间所使用算力的费用，任务结束 Pod 自动释放即结束计费。

在线推理场景

TKE Serverless 集群支持使用 CPU、GPU 以及 vGPU 来运行在线推理服务，丰富的资源规格和弹性伸缩的负载，使运行服务更高效、更经济。

相关服务

- 存储：如需使用云硬盘或文件存储作为容器的持久化存储，可使用 云硬盘 CBS 和 文件存储 CFS。
- 网络：
- 如需创建并管理您的私有网络，例如创建私有网络和子网、建立对等连接、使用 NAT 网关、配置路由表、配置安全策略等，可使用 私有网络 VPC。
- 如需管理服务的内外网访问配置，可使用 负载均衡 CLB。
- API：如需使用腾讯云 API 来访问腾讯云的产品和服务，请参考 腾讯云 API 文档。

购买 TKE Serverless 集群
地域和可用区

最近更新时间：2023-03-30 16:26:40

地域

简介

地域（Region）是指物理的数据中心的地理区域。腾讯云不同地域之间完全隔离，保证不同地域间最大程度的稳定性和容错性。为了降低访问时延、提高下载速度，建议您选择最靠近您客户的地域。
您可以通过下表查看 TKE Serverless 集群当前支持的地域、可用区及资源类型，其他地域将相继开放。

相关特性

不同地域之间的网络完全隔离，不同地域之间的云产品**默认不能通过内网通信**。
不同地域之间的云产品，可以通过 公网 IP 访问 Internet 的方式进行通信。处于不同私有网络的云产品，可以通过 云联网 进行通信，此通信方式较为高速、稳定。
负载均衡 当前默认支持同地域流量转发，绑定本地域的云服务器。如果开通 跨地域绑定 功能，则可支持负载均衡跨地域绑定云服务器。

可用区

简介

可用区（Zone）是指腾讯云在同一地域内电力和网络互相独立的物理数据中心。其目标是能够保证可用区间故障相互隔离（大型灾害或者大型电力故障除外），不出现故障扩散，使得用户的业务持续在线服务。通过启动独立可用区内的实例，用户可以保护应用程序不受单一位置故障的影响。
您可以通过 API 接口 查询可用区列表 查看完整的可用区列表。

相关特性

处于相同地域不同可用区，但在同一个私有网络下的云产品之间均通过内网互通，可以直接使用 内网 IP 访问。
说明
内网互通是指同一账户下的资源互通，不同账户的资源内网完全隔离。

TKE Serverless 集群支持地域

中国

地域	可用区
华南地区（广州） ap-guangzhou	广州三区 ap-guangzhou-3
	广州四区 ap-guangzhou-4
	广州六区 ap-guangzhou-6
	广州七区 ap-guangzhou-7
华东地区（上海） ap-shanghai	上海二区 ap-shanghai-2
	上海三区 ap-shanghai-3
	上海四区 ap-shanghai-4
	上海五区 ap-shanghai-5
华东地区（南京） ap-nanjing	南京一区 ap-nanjing-1
	南京二区 ap-nanjing-2
华北地区（北京） ap-beijing	北京三区 ap-beijing-3
	北京四区 ap-beijing-4
	北京五区 ap-beijing-5
	北京六区 ap-beijing-6
	北京七区

	ap-beijing-7
西南地区（成都） ap-chengdu	成都一区 ap-chengdu-1
	成都二区 ap-chengdu-2
港澳台地区（中国香港） ap-hongkong	香港二区（中国香港节点可用于覆盖港澳台地区） ap-hongkong-2

其他国家和地区

地域	可用区
亚太东南（新加坡） ap-singapore	新加坡一区（新加坡节点可用于覆盖亚太东南地区） ap-singapore-1
	新加坡二区（新加坡节点可用于覆盖亚太东南地区） ap-singapore-2
亚太东南（雅加达） ap-jakarta	雅加达一区（雅加达节点可用于覆盖亚太东南地区） ap-jakarta-1
亚太东北（首尔） ap-seoul	首尔一区（首尔节点可用于覆盖亚太东北地区） ap-seoul-1
	首尔二区（首尔节点可用于覆盖亚太东北地区） ap-seoul-2
亚太东北（东京） ap-tokyo	东京二区（东京节点可用区覆盖亚太东北地区） ap-tokyo-2
亚太南部（孟买） ap-mumbai	孟买一区（孟买节点可用于覆盖亚太南部地区） ap-mumbai-1
	孟买二区（孟买节点可用于覆盖亚太南部地区） ap-mumbai-2
亚太东南（曼谷） ap-bangkok	曼谷一区 （曼谷节点用户覆盖亚太东南地区） ap-bangkok-1
北美地区（多伦多） na-toronto	多伦多一区（多伦多节点可用于覆盖北美地区） na-toronto-1
美国东部（弗吉尼亚） na-ashburn	弗吉尼亚一区 （弗吉尼亚节点用户覆盖美国东部地区） na-ashburn-1

	弗吉尼亚二区 （弗吉尼亚节点用户覆盖美国东部地区） na-ashburn-2
欧洲地区（法兰克福） eu-frankfurt	法兰克福一区（法兰克福节点可用于覆盖欧洲地区） eu-frankfurt-1

计费概述

最近更新时间：2022-12-08 18:03:06

计费模式

TKE Serverless 集群以超级节点维度承载资源，超级节点上提供的计费模式：**按量计费模式**。详情可参见 计费模式说明。

计费方式

TKE Serverless 集群中按量计费的超级节点会根据节点上实际调度的 Pod 规格及运行时长进行计费，具体计算方式请参见 产品定价。

其他费用

如在使用 TKE Serverless 集群时使用到 负载均衡 CLB、云硬盘 CBS、文件存储 CFS 等其他收费产品时，按原产品计费原则计费，具体细节请参考各产品购买指南。

产品定价

最近更新时间：2023-05-30 10:46:05

超级节点的新版产品定价将于2023年7月1日00:00开始生效。

按量计费定价

在按量计费场景下，**Serverless** 容器服务（EKS）服务会根据用户选择的容器资源类型计算相关费用。计费公式为：**费用 = 相关计费项配置 × 资源单位时间价格 × 运行时间（精确到秒）**。目前，超级节点支持的计费项配置详情请参见 资源规格。

Intel Pod 按量计费

地域	计费项			
	CPU（美元/核/秒）	内存（美元/GiB/秒）	高性能磁盘（美元/GB/秒） 超过20GB部分收费	SSD 磁盘（美元/GB/秒） 超过20GB部分收费
上海、北京、广州、南京、重庆、成都、清远、武汉、长沙、郑州、西安、沈阳、福州、合肥、济南、杭州、石家庄、上海金融、深圳金融、北京金融、上海自动驾驶云、日本、泰国、美国硅谷、印度、弗吉尼亚、圣保罗	0.000004976	0.000002073	0.00000004	0.000000104
香港、台北、新加坡	0.000004976	0.000002248	0.00000002	0.000000010
首尔、法兰克福	0.000005224	0.000002612	0.00000002	0.000000010

雅加达	0.000005804	0.000002902	0.00000002	0.000000010

AMD Pod 按量计费

地域	计费项			
	CPU（美元/核/秒）	内存（美元/GiB/秒）	高性能磁盘（美元/GB/秒） 超过20GB部分收费	SSD 磁盘（美元/GB/秒） 超过20GB部分收费
上海、北京	0.00000269	0.00000133	0.00000004	0.000000104
广州、清远、武汉、长沙、郑州、西安、沈阳、福州、合肥、济南、杭州、石家庄	0.00000253	0.00000133	0.00000002	0.000000010
上海金融、深圳金融、北京金融、上海自动驾驶云	0.00000410	0.00000203	0.00000002	0.000000010
香港、台北、日本、新加坡、泰国、印度、弗吉尼亚、圣保罗、法兰克福	0.00000361	0.00000182	0.00000002	0.000000010
美国硅谷	0.00000299	0.00000149	0.00000002	0.000000010

运行时间说明

运行时间从 Pod 拉取首个容器的镜像开始计算，到 Pod 运行终止结束。此段时间为 Pod 的计费时间，以秒为单位计算。

计费示例

示例1

弗吉尼亚地域某 Deployment 的 Intel Pod 资源规格为2核4GB，副本数固定为2。假设该 Deployment 从启动到终止，共耗时5分钟，即要计算300秒的费用。

则该 Deployment 的运行费用 = 2 × (2 × 0.000004976+ 4 × 0.000002073) × 300 = 0.019495美元

示例2

美国硅谷地域某 CronJob 需要每次启动10个AMD Pod，每个 Pod 资源规格为4核8GB，运行10分钟后结束。假设该 CronJob 每天执行2次，使用弹性容器服务托管该任务。

则该任务每天收费 = 2 × 10 × (4 × 0.00000299 + 8 × 0.00000149) × 600 = 0.28656美元

购买限制

最近更新时间：2022-12-08 18:03:06

使用要求

在使用 TKE Serverless 集群前，您需要 注册腾讯云账号，并完成 实名认证。

支持地域

TKE Serverless 集群支持的地域请参见 地域和可用区，资源规格信息请参考 资源规格。

资源配额

集群与 Pod 限制

资源	限制（个）	说明
同一地域内最大集群数量	5	包括创建中、运行中状态的集群
同一集群最大 Pod 规模	100	包括所有 Namespace、所有负载、任何状态的 Pod
单个工作负载最大 Pod 副本数	100	包括负载内任何状态的 Pod
同一地域最大容器实例规模	500	包括任何状态的容器实例

其他相关限制

当使用 TKE Serverless 时，您的所有 Pod 都是云上独立的计算、网络实例，等同于一台云服务器实例，故还会受到下述约束：

1. 创建工作负载时，工作负载的每个 Pod 会默认关联一个 安全组，且同一个工作负载的所有 Pod 副本会关联同一个安全组，每一个 Pod 对安全组来说等同一台云服务器实例，此时会受 单个安全组关联的云服务器实例数 的配额约束。
2. 当使用 CLB 类型 Service 时，每一个绑定 CLB 的 Pod 对该 CLB 来说等同一台云服务器实例，此时会受 一个负载均衡实例的转发规则可绑定的服务器数量 的配额约束。

申请提升配额操作指引

若您需要超过以上配额的资源，可填写提升配额申请，由腾讯云对您的实际需求进行评估，评估通过之后将为您提升配额。

1. 请 提交工单，选择**其他问题 > 立即创建**，进入创建工单信息填写页面。
2. 在问题描述中填写“期望提升 TKE Serverless 相关配额”，注明目标地区、提高配额的对象、目标配额，并按照页面提示填写您可用的手机号等信息。
3. 填写完成后，单击**提交工单**即可。

资源规格

最近更新时间：2023-05-06 17:36:46

概述

使用 TKE Serverless 集群，您无需关心集群节点，但为了合理分配资源和准确核算，您需在部署工作负载时指定 Pod 申请的资源规格。腾讯云会根据您指定的规格为工作负载分配计算资源，也会根据此规格进行费用核算。
当您使用 Kubernetes API 或 Kubectl 创建 TKE Serverless 集群工作负载时，可以通过 Annotation 指定资源规格。如果不指定，TKE Serverless 集群会根据工作负载设置的容器 Request、Limit 等参数进行计算。详情请参考 指定资源规格。

注意

资源规格是 Pod 内容器可使用的最大资源量。
目前支持分配的 CPU、GPU 规格见下表，分配时请勿超出所支持的规格。
Pod 内所有 Container 设置的 Request 之和不可大于最大 Pod 规格。
Pod 内任何 Container 设置的 Limit 不可大于最大 Pod 规格。

CPU 规格

TKE Serverless 集群在所有支持 CPU 资源类型的地域提供以下 CPU Pod 规格。TKE Serverless 集群提供一系列 CPU 选项，不同的 CPU 大小会对应不同的内存选择区间，请在创建工作负载时根据您的实际需求选择最合适规格，并进行资源分配。

Intel

CPU/核	内存区间/GiB	内存区间粒度/GiB
0.25	0.5、1、2	-
0.5	1、2、3、4	-
1	1 - 8	1
2	2、4 - 16	1
4	8 - 32	1
8	16 - 32	1
12	24 - 48	1

16	32 - 64	1

星星海 AMD

基于腾讯云自研星星海服务器，提供可靠、安全、稳定的高性能。详情请参见 云服务器标准型 SA2 介绍。

CPU/核	内存区间/GiB	内存区间粒度/GiB
1	1 - 4	1
2	2 - 8	1
4	4 - 16	1
8	8 - 32	1
16	32 - 64	1

GPU 规格

TKE Serverless 集群提供以下型号 GPU Pod 规格，不同的 GPU 卡型号和大小会对应不同的 CPU、内存选项，请在创建工作负载时根据您的实际需求选择最合适规格，并进行资源分配。

注意

如果您需要通过 yaml 来创建、管理和使用 GPU 的工作负载，请参考 Annotation 说明。

GPU 型号	GPU/卡	CPU/核	内存/GiB
NVIDIA Tesla V100 - 1颗	1	8	40
NVIDIA Tesla V100 - 2颗	2	18	80
NVIDIA Tesla V100 - 4颗	4	36	160
NVIDIA Tesla V100 - 8颗	8	72	320
1/4 NVIDIA T4 - 1/4颗	1	4	20
1/2 NVIDIA T4 - 1/2颗	1	10	40
NVIDIA T4 - 1颗	1	8	32
NVIDIA T4 - 1颗	1	20	80
NVIDIA T4 - 1颗	1	32	128
NVIDIA T4 - 2颗	2	40	160

NVIDIA T4 - 4颗	4	80	320
NVIDIA A10 - PNV4 - 1颗	1	28	116
NVIDIA A10 - PNV4 - 2颗	2	56	232
NVIDIA A10 - PNV4 - 4颗	4	112	466
NVIDIA A10 - PNV4 - 8颗	8	224	932
NVIDIA A10 - GNV4 - 1颗	1	12	44
NVIDIA A10 - GNV4v - 1/4颗	1	6	24
NVIDIA A10 - GNV4v - 1/2颗	1	14	58
NVIDIA A10 - GNV4v - 1颗	1	28	116

竞价模式说明

最近更新时间：2022-09-13 15:00:12

竞价模式概述

TKE Serverless 集群竞价模式是一种低成本资源购置模式，其核心特点为价格低于按量付费实例，在一些场景可以极大降低运行容器的成本，但资源可能会被腾讯云中断回收。您在使用竞价模式运行时仅需支付较少的费用，并运行到容器资源被回收为止。

使用竞价模式时，您可以像使用普通按量计费资源一样在容器部署工作负载，且同样具备普通按量计费模式下的所有功能。

竞价模式策略

价格策略

目前TKE Serverless 集群采用的竞价模式为**固定折扣比例（折扣比例为20%）**，即所有规格的竞价模式将以原规格产品定价 的固定折扣出售。

> 注意：
> 该折扣仅对 Pod 资源规格计费项（CPU、内存、GPU）生效。不包括网络带宽、网络流量、持久化存储等资源的费用。

回收中断机制

竞价模式下的容器会因为腾讯云计算资源池库存不足而产生回收中断。当库存不足时，会从已分配的竞价模式容器里随机回收，容器缓存数据不会保留。

回收中断时将会产生以下中断事件：

```
EVENT REASON : “SpotPodInterruption”
EVENT MESSAGE : “Spot pod was interrupted, it will be killed and re-created”
```

适用场景

适合短时长突发、周期任务

适用于不需要长期运行的突发性、周期性短时长工作负载。例如视频转码、视频渲染、服务压测、批量计算、爬虫等。

适合可切分的计算任务

适用于可以将长时间作业按作业对象切分为细粒度任务进行计算的系统。例如 EMR 等大数据套件。

适合无状态或者支持断点续传能力的计算任务

- 适用于将计算中间结果放到持久化存储上，可接受 Pod 被回收重启后继续运算的工作负载。
- 适用于支持自动负载均衡和服务发现的无状态工作负载，可接受 Pod 被回收重启的工作负载。

竞价模式开启

您可以通过在工作负载 YAML 中定义如下 Pod template annotation 方式，为工作负载开启竞价模式。

```
eks.tke.cloud.tencent.com/spot-pod: "true"
```

其他支持 Annotation 以及范例请参见 Annotation 说明。

TKE Serverless 集群管理
创建集群

最近更新时间：2022-09-13 17:05:46

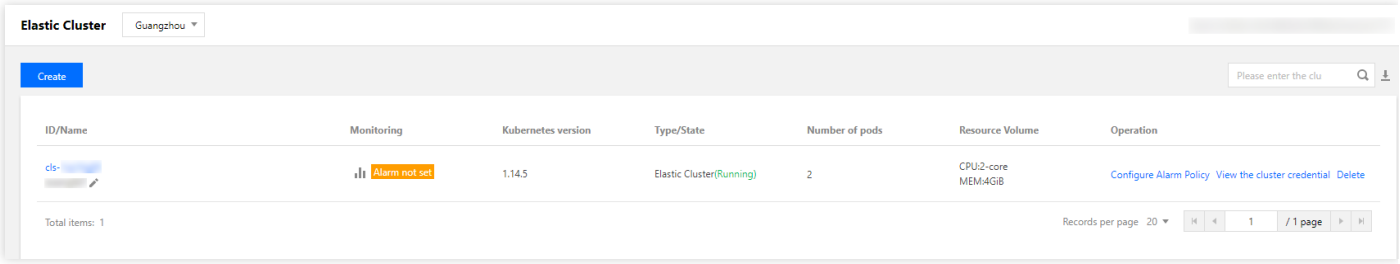
本文介绍通过腾讯云容器服务控制台创建 TKE Serverless 集群。

前提条件

请前往 CAM 管理控制台 开通相应的服务权限。

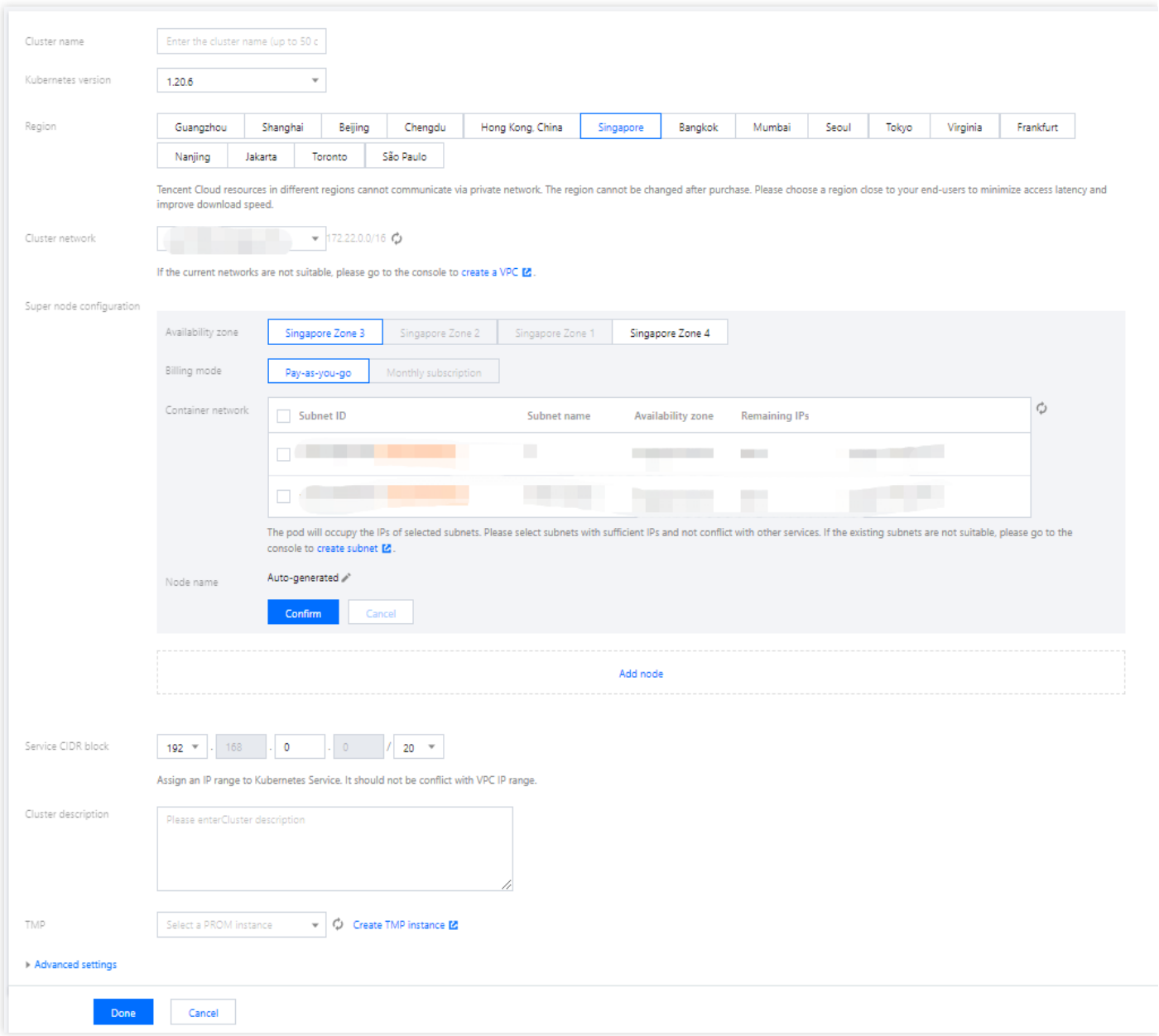
操作步骤

1. 登录容器服务控制台，选择左侧导航栏中的 **集群**。
2. 在页面上方选择需创建集群的地域，并单击**新建**。如下图所示：



3. 在“选择集群类型”弹窗中，选择 **Serverless 集群类型**，单击**创建**。

4. 在“创建集群”页面，根据以下提示设置集群信息。如下图所示：



- **集群名称**：创建的 Serverless 集群名称，不超过60个字符。
- **Kubernetes版本**：Serverless 集群支持1.16以上的多个 Kubernetes 版本选择，各版本特性对比请查看 Supported Versions of the Kubernetes Documentation。
- **所在地域**：建议您根据所在地理位置选择靠近的地域，可降低访问延迟，提高下载速度。
- **集群网络**：该字段展示所在地域下您已创建的全部 VPC 网络，请选择合适的网络作为集群网络，若无可用网络信息或现有网络不合适，可跳转至控制台新建私有网络，详情请参见 私有网络（VPC），私有网络中的子网将用于 serverless 集群的容器网络。
- **超级节点配置**：
  - **可用区**：选择超级节点所在的可用区。

- **计费模式**：全地域支持按量计费模式。
- **容器网络**：为集群内容器分配在容器网络地址范围内的 IP 地址。集群内超级节点的 Pod 会直接占用 VPC 子网 IP（剩余 IP 数将限制调度至超级节点上的 Pod 数），请尽量选择 IP 数量充足且与其他产品使用无冲突的子网。超级节点上的 Pod 会直接运行在用户已指定的 VPC 网络上，每个 Pod 在生命周期内都会绑定一个指定 VPC 内的弹性网卡。您可前往 弹性网卡列表 查看 Pod 关联的网卡。
  按量计费模式下，支持选择多个子网，每个子网对应创建一个按量计费的超级节点，按量计费的超级节点创建时不计费，当实际创建工作负载后 Pod 调度至按量计费超级节点时开始按照 Pod 规格和实际运行时长计费。

  注意
  - 建议为容器网络配置多个可用区，这样您的工作负载在部署时会自动打散分布在多个可用区，可用性更高。
  - 请确保为容器网络分配 IP 充足的子网，避免创建大规模工作负载时因为 IP 资源耗尽无法创建 Pod。

- **Service CIDR**：集群的 ClusterIP Service 默认分配在所选 VPC 子网中，请尽量选择 IP 数量充足且与其他产品使用无冲突的子网。
- **集群描述**：创建集群的相关信息，该信息将显示在“集群信息”页面。
- **高级设置**：
  - CoreDNS：会自动在集群命名空间 kube-system 中部署 2 副本的 Deployment:coredns，该服务默认不收取费用，同时不建议进行修改。
  - 标签：腾讯云标签服务，用于从不同维度对资源分类管理。

5. 单击**完成**即可开始创建，可在“集群”列表页面查看集群的创建进度。

连接集群

最近更新时间：2023-05-25 09:57:20

本文档介绍如何通过 Kubernetes 命令行工具 Kubectl 从本地客户端机器连接到 TKE Serverless 集群。

前提条件

- 请安装 curl 软件。
- 请根据操作系统的类型，选择获取 Kubectl 工具的方式：

> 注意
> 请根据实际需求，将命令行中的 “v1.18.4” 替换成业务所需的 Kubectl 版本。一般来说，客户端的 Kubctl 与服务端的 Kubernetes 的最高版本保持一致即可。您可以在**基本信息**的“基本信息”模块里查看 k8s 版本。

  - Mac OS 系统
  - Linux 系统
  - Windows 系统

执行以下命令，获取 Kubectl 工具：

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.18.4/bin/darwin/amd64/kubectl
```

操作步骤

安装 Kubectl 工具

1. 参考 Installing and Setting up kubectl，安装 Kubectl 工具。

> 说明
> - 如果您已经安装 Kubectl 工具，请忽略本步骤。
> - 此步骤以 Linux 系统为例。

2. 依次执行以下命令，添加执行权限。

```
chmod +x ./kubectl
```

```
sudo mv ./kubectl /usr/local/bin/kubectl
```

3. 执行以下命令，测试安装结果。

```
kubectl version
```

如若输出类似以下版本信息，即表示安装成功。

```
Client Version: version.Info{Major:"1", Minor:"5", GitVersion:"v1.5.2", GitComm
it:"08e099554f3c31f6e6f07b448ab3ed78d0520507", GitTreeState:"clean", BuildDat
e:"2017-01-12T04:57:25Z", GoVersion:"go1.7.4", Compiler:"gc", Platform:"linux/a
md64"}
```

配置 Kubeconfig

1. 登录容器服务控制台，选择左侧导航栏中的 **集群** 。
2. 在集群管理页，单击需连接的 Serverless 集群 ID，进入该集群的管理页面。
3. 选择左侧导航栏中的**基本信息**，进入该集群“基础信息”页面。如下图所示：
4. 在“集群APIServer信息”中，开启外网或内网访问地址，查看该集群的访问地址和 Kubeconfig 访问凭证内容等信息。

- 获取访问入口：请根据实际需求进行设置。
  - **外网访问**：默认不开启。开启外网访问会将集群 apiserver 暴露到公网，请谨慎操作。且需配置来源授权，默认全拒绝，您可配置放通单个 IP 或 CIDR ，强烈不建议配置 0.0.0.0/0 放通全部来源。
  - **内网访问**：默认不开启。开启内网访问时，您需要指定 Apiserver 的内网访问子网，开启成功后将在已配置的子网中分配 IP 地址。
- **访问地址**：集群 APIServer 地址。请注意该地址不支持复制粘贴至浏览器进行访问。
- **Kubeconfig**：该集群的访问凭证，可复制、下载。

5. 根据实际情况进行集群凭据配置。详情可参见控制台内**通过Kubectl连接Kubernetes集群操作说明**。

访问 Kubernetes 集群

1. 完成 Kubeconfig 配置后，依次执行以下命令查看并切换 context 以访问本集群。

```
kubectl config get-contexts
```

```
kubectl config use-context cls-3jju4zdc-context-default
```

2. 执行以下命令， 测试是否可正常访问集群。

```
kubectl get pod
```

如果无法连接请查看是否已经开启公网访问或内网访问入口，并确保访问客户端在指定的网络环境内。

相关说明

Kubectl 命令行介绍

Kubectl 是一个用于 Kubernetes 集群操作的命令行工具。本文涵盖 kubectl 语法、常见命令操作并提供常见示例。有关每个命令（包括所有主命令和子命令）的详细信息，请参阅 kubectl 参考文档 或使用 `kubectl help` 命令查看详细帮助，kubectl 安装说明请参见 安装 Kubectl 工具。

集群生命周期

最近更新时间：2023-03-27 11:08:16

集群生命周期说明

状态	说明
创建中	TKE Serverless 集群正在创建，正在申请云资源。
运行中	集群正常运行。
闲置中	集群处于闲置状态。
激活中	集群从闲置状态中调整到运行中状态的过程。
删除中	集群在删除中，将同步销毁和释放云资源。
异常	集群中存在异常，如网络不可达等。

集群闲置状态说明

为了减少资源空置成本，引导用户更合理地规划和管理集群资源，TKE Serverless 上线了闲置集群回收功能，将没有存量 Pod，且连续7天没有创建/删除 Pod 的集群视为闲置集群，被标记为闲置集群的 TKE Serverless 集群将展示为“闲置中”状态。

闲置集群定义条件：

没有存量的由用户侧创建的 Pod。

连续7天没有 Pod 创建和删除。

距离上次集群闲置变正常状态时间间隔至少3天。

距离新建集群时间超过7天。

状态为“闲置中”的集群，会保留原集群的所有信息，但集群为不可用状态，不支持查看集群监控，不支持查看集群详情，并限制了 APIServer 相关操作，您可通过单击操作栏中**更多** > **激活**唤醒集群，也可以单击**删除**删除集群。

闲置集群激活后集群与原集群功能和配置完全一致。需要注意的是，如果重新被激活的集群连续3天保持没有 Pod 的状态，且没有创建/删除 Pod 操作，会再次被标记为闲置集群。

注意事项

最近更新时间：2022-09-14 10:09:36

计费方式

TKE Serverless集群是付费产品，采取按量计费模式。计费详情请参见 计费概述、产品定价、购买限制。

Pod 规格配置

Pod 的规格配置是容器运行时可用资源和使用服务计费的依据，请务必了解 TKE Serverless 集群支持的 Pod 规格配置和指定方法，详情请参见 资源规格 和 指定资源规格。

Pod 临时存储

每个 Pod 创建时会分配不超过20GiB的临时镜像存储。

注意：

- 临时镜像存储将于 Pod 生命周期结束时删除，请勿用于存储重要数据。
- 由于存储镜像，实际可用空间小于20GiB。
- 重要数据、超大文件等推荐挂载 Volume 持久化存储。

Kubernetes 版本

不支持 1.14 以下版本。

其他说明

TKE Serverless 集群没有 Node，所以不支持部分依赖 Node 组件，例如 Kubelet、Kube-proxy 的功能。

Node

- 暂不支持添加、管理物理节点。

- TKE Serverless 集群的节点会被“超级节点”替代，每个超级节点对应一个容器网络中指定的 VPC 子网。
- 超级节点也支持节点亲和性、污点、封锁等调度操作。
- 超级节点上可调度的 Pod 数只受对应 VPC 子网的 IP 数限制。

容器网络

- TKE Serverless 集群的容器网络采用的是与云服务器、云数据库等云产品平级的 VPC 网络，集群中的每个 Pod 都会占用一个 VPC 子网 IP。
- Pod 与 Pod、Pod 与其他同 VPC 云产品间直接通过 VPC 网络通信，没有性能损耗。

Pod 隔离性

TKE Serverless 集群中的 Pod 拥有与云服务器完全一致的安全隔离性。Pod 在腾讯云底层物理服务器上调度创建，创建时会通过虚拟化技术保证 Pod 间的资源隔离。

节点内核

仅支持自定义 net 开头的内核参数。

Workload

不支持部署 DaemonSet 类型的工作负载。

Service

不支持部署 NodePort 类型的服务。

此外，普通 Kubernetes 集群 ClusterIP 类型的 Service 依赖节点转发流量，TKE Serverless 集群为了兼容 ClusterIP 类型的 Service ，需用户单独指定一个 VPC 子网作为集群 Service CIDR。所有 ClusterIP Service 会在此子网中创建内网负载均衡实现，并占用子网一个 IP，请确保子网具备充足剩余 IP。

Volume

支持 Hostpath 类型的数据卷，详情请参见 存储卷说明。
由于 TKE Serverless 集群没有节点，虽然集群依旧兼容 Pod 中与 Host 相关的参数（例如 Hostpath、Hostnetwork: true、DnsPolicy: ClusterFirstWithHostNet 等），但不一定能满足预期效果。例如您希望使用 Hostpath 共享数据，但可能调度到同一个超级节点上的两个 Pod 实际为不同子机的 Hostpath。因此建议您在 TKE Serverless 集群运行的任务，不要强依赖与 Host 相关的参数。

端口限制

在 TKE Serverless集群中，9100端口被保留，不可使用。

超级节点管理
创建超级节点

最近更新时间：2023-02-02 17:05:22

超级节点简介

超级节点是 TKE Serverless 集群提供的一种调度能力，在创建 TKE Serverless 集群时会对应在容器网络所在的每个子网中创建一个超级节点。

当您遇到如下场景时，可通过创建超级节点来解决相应的问题：

- 在 TKE Serverless 集群中运行大规模工作负载时，因为服务所在可用区子网的 IP 资源耗尽而无法创建 Pod。这种情况下建议您在新的子网中创建超级节点来拓展 IP 数量。
- 随着服务规模的扩大需要将服务自动打散分布在多个可用区。这种情况下建议您在新的子网中创建超级节点以拓展资源可用区。
- 在创建工作负载时由于可用区资源紧张而导致的 Pod pending，具体表现为如下集群事件：

```
EVENT REASON : “FailedCreatePodSandBox”
EVENT MESSAGE : “Failed to create pod sandbox in underlay (will retry): insuffi
cient resource”
```

这种情况可以创建关联其他可用区的超级节点来拓展集群可用资源。

计费方式

超级节点本身不收取费用，实际会根据工作负载申请的 CPU、GPU、内存数值以及工作负载的运行时间来核算费用，计费详情请参见 TKE Serverless 集群计费概述、TKE Serverless 集群产品定价、TKE Serverless 集群购买限制。

操作场景

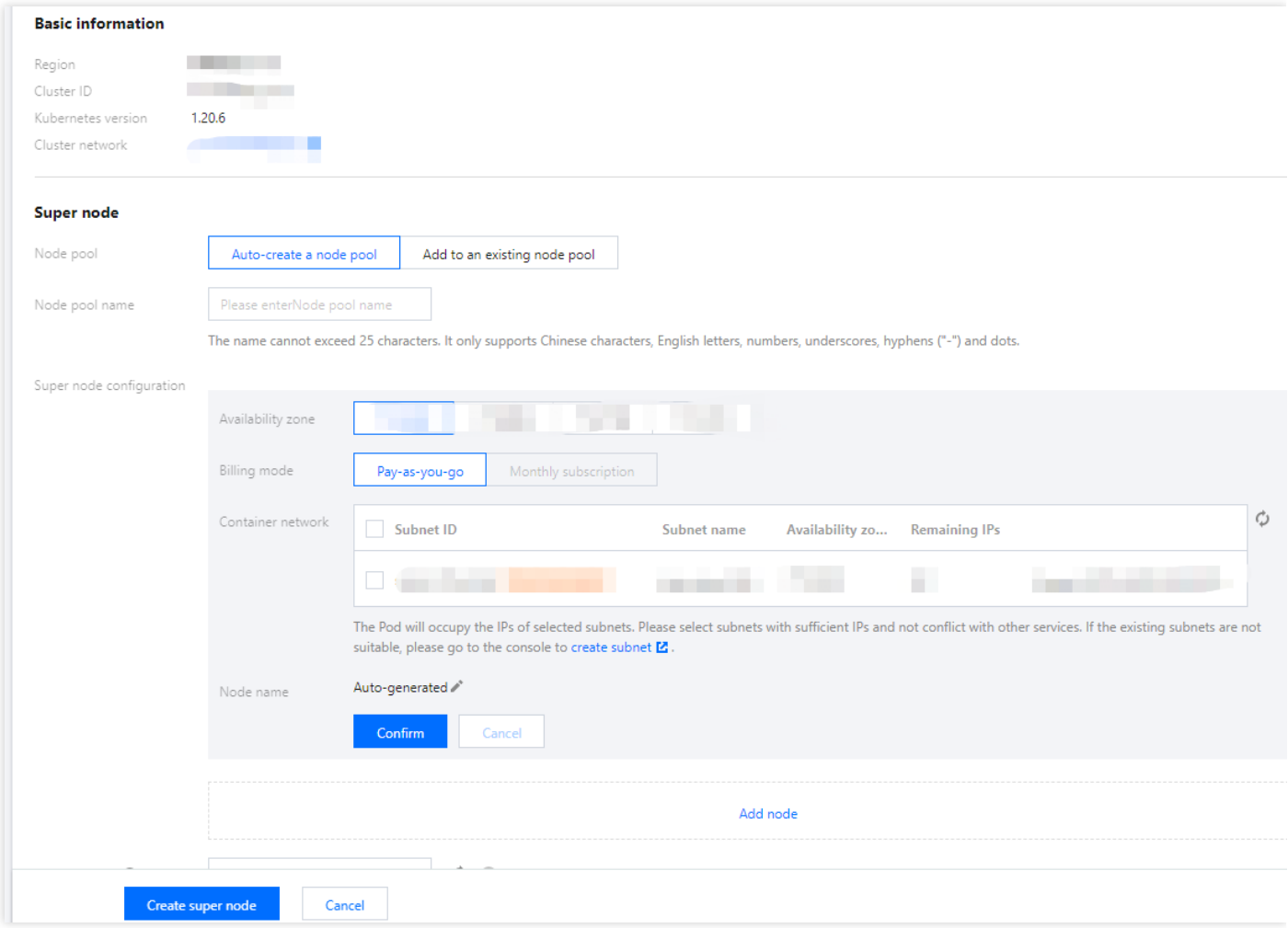
本文介绍如何通过容器服务控制台在 TKE Serverless 集群中新建超级节点。

前提条件

- 请确保已经创建 TKE Serverless 集群。操作详情见 创建 TKE Serverless 集群。
- 建议阅读 超级节点 Pod 调度说明。

操作步骤

1. 登录 容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，选择 Serverless 集群 ID，进入该集群基本信息页面。
3. 选择左侧菜单栏中的**超级节点**，在超级节点页面单击**新建节点**。
4. 在“新建超级节点”页面，参考以下提示进行设置。如下图所示：



- **计费模式**：提供按量计费的计费模式。
- **操作系统类型**：该子网支持的操作系统类型，支持选择 Linux 和 Windows。当选择 Windows 时，支持在该子网下运行 Windows 容器，默认选择 Linux 操作系统。

- **容器网络**：指定 Pod 调度到超级节点时所分配的网络。调度到超级节点上的 Pod 采用的是与云服务器、云数据库等云产品平级的 VPC 网络，每个 Pod 都会占用一个 VPC 子网 IP。可以选择 Serverless 集群所在VPC的任意子网，请根据实际需求选择合适的可用子网并保证所选的子网可用 IP 数量充足。
- **安全组**：安全组具有防火墙的功能，用于设置云服务器的网络访问控制。详情请参见 容器服务安全组设置。

5. 单击**确定**，创建超级节点。

相关操作

超级节点创建完成之后，您可参考 管理超级节点 进行后续的节点管理。

管理超级节点

最近更新时间：2022-09-14 16:00:59

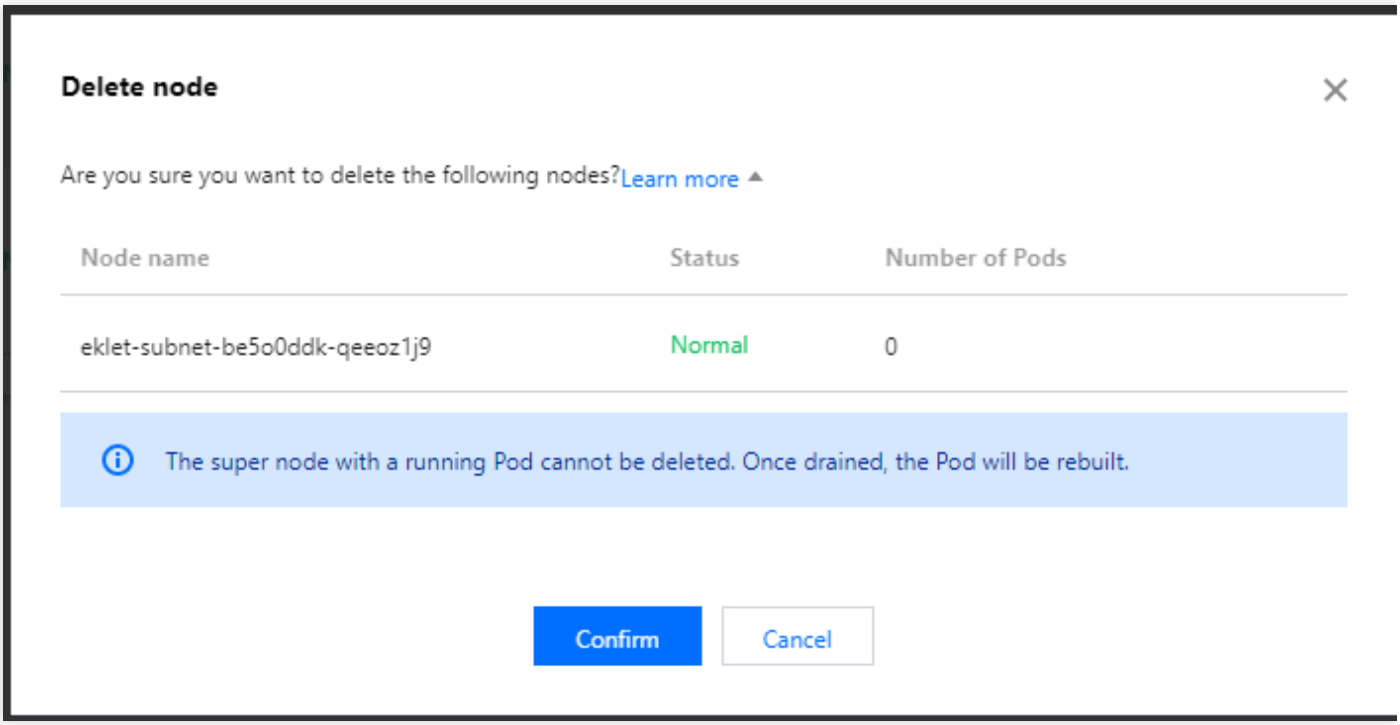
本文介绍如何在容器服务控制台管理 Serverless 集群中的超级节点。

前提条件

- 请确保已经创建 Serverless 集群。
- 建议阅读 超级节点 Pod 调度说明。

删除超级节点

1. 登录 容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，选择 Serverless 集群 ID，进入该集群基本信息页面。
3. 选择左侧菜单栏中的**超级节点**，进入“超级节点”列表页面。
4. 在列表页中，单击选定的超级节点右侧的**移出**。
5. 在“删除节点”弹窗确认页面，单击**确认**删除节点。如下图所示：



说明

- 集群中至少存在一个可用的超级节点，当集群中只有一个超级节点时不支持删除该节点。

- 超级节点上如果有运行中的 Pod 则不可删除，若要移出请先进行封锁和驱逐操作，驱逐会重建 Pod，请谨慎操作。

管理超级节点

1. 登录 容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，选择 Serverless 集群 ID，进入该集群基本信息页面。
3. 选择左侧菜单栏中的**超级节点**，进入“超级节点”列表页面。
4. 单击节点名称，进入节点详情页。支持管理超级节点上的 Pod、查看超级节点事件、Yaml 等信息。

修改超级节点配置

1. 登录 容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，选择 Serverless 集群 ID，进入该集群基本信息页面。
3. 选择左侧菜单栏中的**超级节点**，进入“超级节点”列表页面。
4. 选择需要修改配置的节点，单击选定的超级节点右侧的**编辑Label & Taint**。
5. 在“修改超级节点Label&Taint”弹窗中，修改超级节点Label&Taint配置。
6. 单击**确定**完成修改超级节点配置。

Kubernetes 对象管理
工作负载管理

最近更新时间：2023-05-23 11:24:32

本文介绍如何在 TKE Serverless 集群中选择多种工作负载形式来运行您的服务。

注意：

- Serverless 集群没有 Node，工作负载在创建时会根据参数设置为每个 Pod 分配实际的资源。详情请参见指定资源规格。
- 如果您需要通过 yaml 来创建、管理您的 Kubernetes 对象，可通过指定 annotation 完成。详情请参见 Annotation 说明。

工作负载类型介绍

Deployment

Deployment 声明了 Pod 的模板和控制 Pod 的运行策略，适用于部署无状态的应用程序。您可以根据业务需求，对 Deployment 中运行的 Pod 的副本数、调度策略、更新策略等进行声明。

StatefulSet

StatefulSet 主要用于管理有状态的应用，可以创建具有持久性标识符的 Pod。Pod 迁移或销毁重启后，标识符仍会保留。在需要持久化存储时，您可以通过标识符对存储卷进行一一对应。如果应用程序不需要持久的标识符，建议您使用 Deployment 部署应用程序。

Job

Job 控制器会创建 1 - N 个 Pod，这些 Pod 按照运行规则运行，直至运行结束。Job 可用于批量计算和数据分析等场景，它可以通过设置重复执行次数、并行度、重启策略等参数来满足业务需求。
Job 运行完毕后，将不再创建新的 Pod，仅存在原 Pod 运行记录，但底层真实资源已释放，因此运行结束后无法在控制台 “日志” 中查询到对应 Pod 的日志。

CronJob

一个 CronJob 对象类似于 crontab（cron table）文件中的一行。它根据指定的预定计划周期性地运行一个 Job，格式可以参考 Cron。
Cron 格式说明如下：

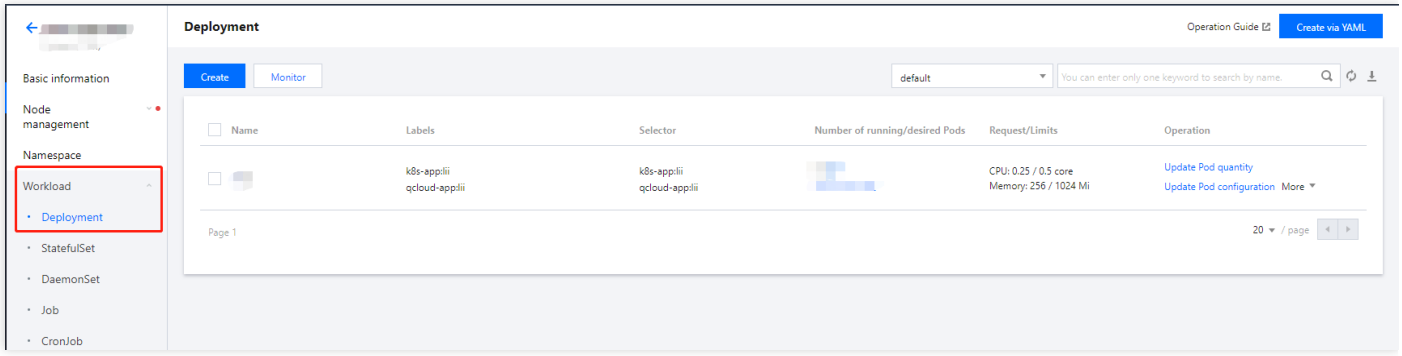
```
# 文件格式说明
# ——分钟（0 – 59）
# | ——小时（0 – 23）
# | | ——日（1 – 31）
# | | | ——月（1 – 12）
# | | | | ——星期（0 – 6）
# | | | | |
# * * * * *
```

前提条件

- 已创建状态为“运行中”的 Serverless 集群，详情请参见 创建集群。
- 集群有合适的且为 Active 状态的命名空间。

操作步骤

1. 登录 容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，选择 Serverless 集群 ID，进入该集群基本信息页面。
3. 在左侧导航中选择工作负载类型，单击**新建**。如下图所示：



4. 在“新建Workload” 页面。填写工作负载名，并选择要创建的工作负载类型。

- 各类型工作负载的具体参数设置请参考：
  - Deployment 管理
  - StatefulSet 管理
  - CronJob 管理
  - Job 管理
- 其他操作指引请参考：
  - 设置工作负载的资源限制

- 设置工作负载的调度规则
- 设置工作负载的健康检查
- 设置工作负载的运行命令和参数

服务管理

最近更新时间：2022-09-14 10:18:54

操作场景

本文档介绍工作负载如何通过 Service 和 Ingress 对外暴露服务入口。

服务类型介绍

Service

Service 定义访问后端 Pod 的访问策略，提供固定的虚拟访问 IP。您可以通过 Service 负载均衡地访问到后端的 Pod。

Service 支持以下类型：

- 公网访问：使用 Service 的 Loadbalance 模式，自动创建公网 CLB。公网 IP 可直接访问到后端的 Pod。
- 集群内访问：使用 Service 的 ClusterIP 模式，用于集群内访问。
- VPC 内网访问：使用 Service 的 Loadbalance 模式，自动创建内网 CLB。指定 `annotations:service.kubernetes.io/qcloud-loadbalancer-internal-subnetid: subnet-xxxxxxxx`，VPC 内网即可通过内网 IP 直接访问到后端的 Pod。

> 注意：
> 当 Service 类型为公网访问时，默认不为该 Service 开启 ClusterIP。可通过在 yaml 中增加如下 annotations 开启 ClusterIP：

```
service.kubernetes.io/qcloud-clusterip-loadbalancer-subnetid: #Service CIDR使用的子网ID
```

Ingress

Ingress 是允许访问到集群内 Service 的规则的集合，您可以通过配置转发规则，实现不同 URL 可以访问到集群内不同的 Service。

为了使 Ingress 资源正常工作，集群必须运行 Ingress-controller。腾讯云容器服务在集群内默认启用了基于腾讯云负载均衡器实现的 `l7-lb-controller`，支持 HTTP、HTTPS 及 nginx-ingress 类型，您可以根据您的业务需要选

择不同的 Ingress 类型。

前提条件

- 已创建状态为“运行中”的 Serverless 集群，详情请参见 创建集群。
- 集群有合适的且为 Active 状态的命名空间。

操作步骤

具体操作请参考 Service 管理 和 Ingress 管理。

注意事项

- Serverless 集群内创建 ClusterIP Service 会占用 Service CIDR 所选子网 IP，请确保该子网剩余 IP 足够。
- Serverless 集群 Service 创建的 CLB 会直接绑定 Endpoint 内所有 Pod 的弹性网卡。
- Serverless 集群 Service 仅支持 应用型负载均衡。
- 使用已有 CLB 创建 Service 时，仅支持使用当前未创建监听器的 CLB。

指定资源规格

最近更新时间：2022-09-13 16:49:33

TKE Serverless 集群支持 通过 Annotation 指定 及 通过 Request、Limit 自动计算 两种方式，指定为 Pod 分配的资源上限。您可选择其中一种方式进行配置。

通过 Annotation 指定

TKE Serverless 集群支持在工作负载 yaml 中以添加 `template annotation` 的方式，显式的指定 Pod 资源规格。详情请参见 Annotation 说明。

通过 Request、Limit 自动计算

TKE Serverless 集群支持对工作负载设置的 Request 及 Limit 进行计算，自动判断 Pod 运行所需的资源量。根据 Pod 资源类型的不同，其计算方法也略有差异。请参考 CPU Pod 规格计算方法 及 GPU Pod 规格计算方法，进一步了解如何通过 Request、Limit 自动计算指定资源规格。

注意：

- 如指定了工作负载 `template annotation`，则以 Annotation 配置为准，不进行 Request 及 Limit 核算。
- Request 及 Limit 的分配请参考 资源规格 中的 CPU、GPU 支持规格，若设定值与支持规格差别过大可能会导致某项资源分配超预期，造成资源浪费。
- 无论如何设置 Request 及 Limit，其最终计算结果都会与 资源规格 进行匹配，且最终 Pod 分配的实际资源一定不会超出其中允许的规格。
- 如 Pod 内有容器未设置 Request 及 Limit，则未设置项作为0运算。
- 如 Pod 内所有容器都未设置 Request 及 Limit，则默认使用 Pod 规格为1核2GiB。
- Initcontainer 和 Container 分别按下述方法计算，最终取大者。

CPU Pod 规格计算方法

步骤1：分别计算 Pod 的 CPU、Memory 的合计数值。

合计数值分别为 **Pod 内所有容器的 Request 之和、Pod 内所有容器的 Limit 中的最大值**中的较大者。

步骤2：按以下情况匹配 Pod 资源规格：

CPU 及 Memory 合计数值	Pod 资源选择规则
合计数值均为0	选择规格为1核2GiB。
任一合计数值为0	按非0项的合计数值进行最小匹配。 例如，CPU 合计数值为0核，Memory 合计数值为8GiB，则在 Memory 为8GiB的允许规格中进行 CPU 最小匹配，最终选择规格为1核8GiB。
合计数值均不为0	与 资源规格 进行匹配。首先选择与 CPU 合计数值一致或相近的较大规格（A 规格），然后再选择与 Memory 的相近较大规格： • 如 Memory 合计数值 < A 规格的 Memory 区间最小值，则选择 A 规格的 Memory 区间的最小值。 • 如 Memory 合计数值 > A 规格的 Memory 区间最大值，则选择与 Memory 相近的较大规格（B 规格），并将 CPU 合计数改为 B 规格 CPU。 • 如 Memory 合计数值在 A 规格 Memory 区间之内，则选择最相近较大双数值。
任一合计数值超过允许的最大规格	出现错误，无法进行匹配。

示例

请结合以下示例，进一步了解 CPU Pod 规格计算方法：

- 示例1
- 示例2

```
resources:
limits:
cpu: "1"
memory: 2Gi
requests:
cpu: "1"
memory: 2Gi
```

结果：选择 Pod 规格为1核2GiB。

GPU Pod 规格计算方法

说明：

- GPU 和 vGPU 的 Request 及 Limit 参数 `“nvidia.com/gpu”` 通常相等且仅支持整数。

- vGPU 可以看作是一种独立的 GPU 类型。例如 1/4*V100，是将一张 V100 GPU 卡 1/4 的算力虚拟为一张完整的卡进行分配，故在请求资源分配时依然应该是申请1卡 GPU，即 `“nvidia.com/GPU”=1` 。

步骤1：计算 Pod 的 GPU 的合计数值。

GPU 合计数值为 **Pod 内所有容器的 Request 之和**。

步骤2：根据以下情况匹配 Pod 资源规格：

CPU、Memory 及 GPU 合计数值	Pod 资源匹配规则
符合规格要求（例如1、2、4、8等）	首先选择与 GPU 合计数值一致或最相近的较大规格（A 规格），再按照 CPU Pod 规格计算方法 对 CPU 及 Memory 进行计算，得出 CPU 规格（B 规格）： • 如 A 规格的 CPU 及 Memory ≥ B 规格，则选择 A 规格 GPU。 • 如 A 规格的 CPU 及 Memory ＜ B 规格，则选择与 B 规格的 CPU 及 Memory 最相近且较大的 GPU 规格（C 规格）。此时实际分配的 GPU 卡数会比实际所需更多，为了防止浪费，**尽量避免出现此情况**，请尽量调小 CPU 及 Memory 的请求数值。
任一合计数值如果超过允许的最大规格	出现错误，无法进行匹配。

示例

请结合以下示例，进一步了解 GPU Pod 规格计算方法：

- 示例1
- 示例2

```
## eks.tke.cloud.tencent.com/gpu-type：V100
resources:
limits:
cpu: "8"
memory: 32Gi
nvidia.com/gpu: "1"
requests:
cpu: "4"
memory: 16Gi
nvidia.com/gpu: "1"
```

说明：

GPU 合计数值为：1

CPU 合计数值为：max(4,8) = 8核

Memory 合计数值为：max(16,32) = 32GiB

结果：8核32GiB小于 资源规格 中 V100 GPU 规格（1卡）对应的 CPU 及 Memory 规格（8核40GiB）。最终选择 Pod 规格为8核40GiB 1*V100。

其他资源管理

最近更新时间：2022-09-14 10:29:28

本文档介绍通过腾讯云容器服务控制台管理其他 Kubernetes 资源，例如命名空间、配置、存储等。

前提条件

已创建状态为“运行中”的 Serverless 集群，详情请参见 创建集群。

操作步骤

1. 登录容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，单击 Serverless 集群 ID。
3. 在集群详情页面，您可参考以下文档进行其他资源管理：

- 命名空间的操作管理请参考 Namespaces。
- 自动伸缩的操作管理请参考 自动伸缩 和 自动伸缩指标说明。
- 配置资源的操作管理请参考 ConfigMap 管理 和 Secret 管理。
- 存储资源的操作管理请参考 存储管理概述。

Annotation 说明

最近更新时间：2023-02-01 16:10:50

本文介绍超级节点特有的 Annotation 与示例，该 Annotation 针对 TKE 标准集群和 TKE Serverless 集群内超级节点上运行的 Pod 生效。

Annotation 使用方法

工作负载里添加 Pod 注解

本文所说明的注解均为在 Pod 级别添加注解，通常用户使用的是工作负载而不是裸 Pod。本文以在 Deployment 上添加 Pod 注解为例，需注意，工作负载里添加 Pod 注解是在 `.spec.template.metadata.annotations` 字段。

```
apiVersion: apps/v1
kind: Deployment
metadata:
name: nginx
spec:
replicas: 1
selector:
matchLabels:
app: nginx
template:
metadata:
labels:
app: nginx
annotations:
eks.tke.cloud.tencent.com/retain-ip: 'true' # 工作负载里添加 Pod 注解是在 .spec.temp
late.metadata.annotations 字段
spec:
containers:
- name: nginx
image: nginx
```

全局配置

如果用户希望注解默认对集群里所有 Pod 生效，可以通过修改全局配置实现。需注意，全局配置在 kube-system 命名空间下名为 eks-config 的 configmap，如没有该资源则需要自行新建。

```
apiVersion: v1
kind: ConfigMap
metadata:
name: eks-config
namespace: kube-system
data:
pod.annotations: |
eks.tke.cloud.tencent.com/resolv-conf: |
nameserver 183.60.83.19
eks.tke.cloud.tencent.com/host-sysctls: '[{"name": "net.core.rmem_max","value":
"26214400"}]'
```

说明：
添加在 Pod 上的注解优先级高于全局配置。

资源与规格

指定 CPU 与内存

超级节点上运行的 Pod 默认会根据 request 与 limit 自动计算出底层资源的规格，用户可以参考官方文档 指定资源规格，也可以通过给 Pod 添加注解去指定 Pod 需要的计算资源规格。示例如下：

```
eks.tke.cloud.tencent.com/cpu: '8'
eks.tke.cloud.tencent.com/mem: '16Gi' # 内存一定要以 Gi 为单位，以 G 为单位则会报参数错误
```

指定系统盘大小

超级节点上运行的 Pod 默认免费提供20G系统盘，系统盘生命周期与 Pod 的生命周期一致，可满足通用场景的系统盘资源诉求，若特殊场景下用户需要额外扩容系统盘大小，支持通过 Annotation 来声明所需系统盘的大小，注意，此时超过20G的部分将按照 CBS 高性能云盘按量计费的刊例价进行计费，计费详情见 云硬盘定价说明。系统盘大小注解示例如下：

```
eks.tke.cloud.tencent.com/root-cbs-size: '50' # 指定系统盘大小，超过20 Gi 额外计费
```

说明：

若使用镜像缓存时需要调整盘大小，请参考 镜像缓存Annotation。

规格自动升配

超级节点上运行的 Pod 会按照 request 与 limit 自动计算出匹配的底层资源规格。如果对应的规格底层无相应资源，则会创建失败并在事件中提示资源不足 (insufficient resource)。如果用户接受使用更高规格的资源来创建 Pod，则可在 Pod 添加如下注解，开启自动升配，计费将按照升配后的规格计算。

```
eks.tke.cloud.tencent.com/spec-auto-upgrade: 'true' # 遇到资源不足时，开启自动升配，只按 CPU 规格向上升配一次
```

指定 GPU

如需指定 GPU，可在 Pod 上添加如下注解：

```
eks.tke.cloud.tencent.com/gpu-type: 'T4,V100' # 指定 GPU 型号，支持优先级顺序写法，若使用1/4卡的 T4 vGPU 则指定 GPU 型号为1/4*T4。
```

说明：
GPU 的卡数和规格需要您在 Request 和 Limit 中自行设置，建议参考对应型号的 GPU 规格设置合适的值。
GPU 规格详情可参考 资源规格。

指定 CPU 类型

如需指定 CPU 类型，可在 Pod 上添加如下注解：

```
eks.tke.cloud.tencent.com/cpu-type: 'amd,intel' # 表示优先创建 amd 资源 Pod，如果所选地域可用区 amd 资源不足，则会创建 intel 资源 Pod
```

开启竞价模式

如需开启 竞价模式，可在 Pod 上添加如下注解：

```
eks.tke.cloud.tencent.com/spot-pod: 'true'
```

IP 保留与 EIP

StatefulSet 固定 IP

使用固定 IP 的前提是工作负载为 StatefulSet，或者直接使用裸 Pod（需注意 Pod 名称不能改变），用户可在 Pod 级别添加注解启用固定 IP：

```
eks.tke.cloud.tencent.com/retain-ip: 'true' # 置为 true 启用固定 IP。
eks.tke.cloud.tencent.com/retain-ip-hours: '48' # 保留 IP 的最大时长（小时），Pod 销毁之后超过这个时长没有创建回来，IP 将被释放。
```

> 说明：
> TKE 集群超级节点场景下的固定 IP，用法保持跟 TKE 一致，无需添加上述注解。

绑定 EIP

如需绑定 EIP，可在 Pod 上添加如下注解：

```
eks.tke.cloud.tencent.com/eip-attributes: '{"InternetMaxBandwidthOut":50, "InternetChargeType":"TRAFFIC_POSTPAID_BY_HOUR"}' # 值可以为空串，表示启用 EIP 并使用默认配置；也可以用创建 EIP 接口的 json 参数，详细参数列表参考 https://cloud.tencent.com/document/api/215/16699#2.-.E8.BE.93.E5.85.A5.E5.8F.82.E6.95.B0，本例中的参数表示 EIP 是按量付费，且带宽上限为 50M。
```

> 注意：
> 非带宽上移的账号（传统账户），不支持绑定 EIP，若为非带宽上移账户，可 提交工单 进行账户升级。

StatefulSet 固定 EIP

如需在 StatefulSet 固定 EIP，可在 Pod 级别添加如下注解：

```
eks.tke.cloud.tencent.com/eip-attributes: '{}' # 启用 EIP 并使用默认配置。
eks.tke.cloud.tencent.com/eip-claim-delete-policy: 'Never' # Pod 删除后，EIP 是否自动回收，默认回收。使用 "Never" 不回收，即下次同名 Pod 创建出来仍然会绑定此 EIP，实现固定 EIP。
```

> 注意：
> Deployment 类型的工作负载使用 `Never` 不回收策略时，Pod 删除后 EIP 不会回收，滚动更新后的 Pod 也不会使用原本的 EIP。

StatefulSet 使用已有 EIP

如需将已有的 EIP 绑定到 Pod 而不是自动创建，可以给 Pod 指定要绑定的 EIP 实例 ID。注解如下：

```
eks.tke.cloud.tencent.com/eip-id-list: 'eip-xx1,eip-xx2' # 这里指定已有的 EIP 实例列表，确保 StatefulSet 的 Pod 副本数小于等于这里的 EIP 实例数。
```

> 注意：
> 指定 EIP 场景下，请勿配置 eip-attributes 注解。

镜像与仓库

忽略证书校验

如果自建镜像仓库的证书是自签的，拉取镜像会失败 (ErrImagePull)，可以给 Pod 添加以下注解指定拉取此镜像仓库的镜像时不校验证书：

```
eks.tke.cloud.tencent.com/registry-insecure-skip-verify: 'harbor.example.com' # 也可以写多个，逗号隔开
```

使用 HTTP 协议

如果自建镜像仓库使用的 HTTP 而不是 HTTPS，拉取镜像会失败 (ErrImagePull)，可以给 Pod 添加以下注解指定拉取此镜像仓库的镜像时使用 HTTP 协议：

```
eks.tke.cloud.tencent.com/registry-http-endpoint: 'harbor.example.com' # 也可以写多个，逗号隔开
```

镜像复用

超级节点上默认会复用系统盘以加快启动速度，复用的是同一工作负载下相同可用区 Pod 且在缓存时间内（销毁后2小时内）的系统盘。如需复用不同工作负载的 Pod 的镜像，可以在不同工作负载的 Pod 上打上相同的 `cbs-reuse-key` 的注解：

```
eks.tke.cloud.tencent.com/cbs-reuse-key: 'image-name'
```

镜像缓存

超级节点上提供 镜像缓存能力，即提前创建好镜像缓存实例，自动将需要的镜像下载下来并创建云盘快照，后续创建 Pod 开启镜像缓存，这样就会根据镜像名自动匹配镜像缓存实例的快照，直接使用快照里面的镜像内容，避免重复下载，加快 Pod 启动速度。

启用镜像缓存的注解：

```
eks.tke.cloud.tencent.com/use-image-cache: 'auto'
```

指定镜像缓存创建出来的盘类型：

```
eks.tke.cloud.tencent.com/image-cache-disk-type: 'CLOUD_SSD' # 指定镜像缓存创建出来的盘类型，可取值如下：CLOUD_BASIC为普通云硬盘，CLOUD_PREMIUM为高性能云硬盘（默认），CLOUD_SSD为SSD云硬盘，CLOUD_HSSD为增强型SSD云硬盘，CLOUD_TSSD为极速型SSD云硬盘
```

指定镜像缓存创建出来的盘的大小：

```
eks.tke.cloud.tencent.com/image-cache-disk-size: '50' # 指定镜像缓存创建出来的盘的大小，默认是镜像缓存创建时设置的大小，可以调大，不能调小
```

用户也可以手动指定镜像缓存实例，不使用自动匹配：

```
eks.tke.cloud.tencent.com/use-image-cache: 'imc-xxx'
```

镜像缓存创建出来的数据盘，在 Pod 删除后会立即销毁。如果需额外保留镜像缓存创建的数据盘，可以通过注解指定保留时间。开启数据盘保留后，再次创建 Pod 的时候会直接复用保留中的盘，可节省数据再重新从快照回滚至新云硬盘的时间：

```
eks.tke.cloud.tencent.com/image-cache-disk-retain-minute: '10' # 设定镜像缓存创建的数据盘，在销毁 Pod 后保留 10 分钟
```

绑定安全组

超级节点默认会为 Pod 绑定同地域默认项目下的 `default` 安全组，也可以通过给 Pod 加注解绑定指定安全组：

```
eks.tke.cloud.tencent.com/security-group-id: 'sg-id1,sg-id2' # 填本地域存在的安全组 id，多个用逗号隔开，网络策略按安全组顺序生效，安全组默认最多只能绑定 2000 个 Pod，如需更多请提工单提升配额。
```

绑定角色

超级节点上默认会为 Pod 关联 `TKE_QCSLinkedRoleInEKSLog` 角色，授予 Pod 里日志采集组件上报日志的权限。用户也可通过注解为 Pod 关联其他 CAM 角色，方便从 Pod 内获取操作云上资源的权限。

```
eks.tke.cloud.tencent.com/role-name: 'TKE_QCSLinkedRoleInEKSLog'
```

设置宿主机内核参数

有些内核参数以网络命名空间隔离，无法在容器内查看或设置，超级节点内 Pod 虽然是独占虚拟机，但不代表就能在容器内直接设置宿主机上的内核参数。

可以通过给 Pod 加注解来实现设置 Pod 宿主机内核参数：

```
eks.tke.cloud.tencent.com/host-sysctls: '[{"name": "net.core.rmem_max","value":
"26214400"},{"name": "net.core.wmem_max","value": "26214400"},{"name": "net.core.
rmem_default","value": "26214400"},{"name": "net.core.wmem_default","value": "262
14400"}]'
```

加载内核模块

如需加载 TOA 内核模块，可在 Pod 上添加如下注解：

```
eks.tke.cloud.tencent.com/host-modprobe: 'toa'
```

自动重建自愈

超级节点所在集群虚拟机内的 agent 会上报心跳给控制面，如果上报超时（默认 5min），一般说明 Pod 内进程已经无法正常工作了，故障原因通常是高负载，这时集群默认会自动迁移虚拟机（对当前虚拟机关机并自动创建新虚拟机，让 Pod 迁移到新虚拟机里去运行），从而实现自愈。

如果用户不希望自动重建（如用于保留现场），可以在 Pod 上添加如下注解：

```
eks.tke.cloud.tencent.com/recreate-node-lost-pod: "false"
```

如果用户认为默认的 5min 上报超时时间过长，希望及时摘除异常 Pod 的流量，可以在 Pod 上添加如下注解来自定义心跳超时时间：

```
eks.tke.cloud.tencent.com/heartbeat-lost-period: 1m
```

磁盘清理

当超级节点内 Pod 磁盘空间紧张的时候，会自动触发清理流程以释放空间，通过 `df -h` 可以确认磁盘使用情况。

常见磁盘空间不足的原因有：

- 业务有大量临时输出。您可以通过 du 命令确认。
- 业务持有已删除的文件描述符，导致磁盘空间未释放。您可以通过 lsof 命令确认。

清理容器镜像

默认情况下，磁盘空间达 80% 以上会自动清理容器镜像来释放空间，如果已经没有容器镜像可以释放，则会报告如下事件：

```
failed to garbage collect required amount of images. Wanted to free 7980402688 by
tes, but freed 0 bytes
```

如需自定义清理的阈值，可以在 Pod 上添加如下注解：

```
eks.tke.cloud.tencent.com/image-gc-high-threshold: '80' # 表示磁盘使用空间达 80% 触发清理容器镜像
eks.tke.cloud.tencent.com/image-gc-low-threshold: '75' # 触发清理容器镜像后，默认释放 5% (high-threshold - low-threshold) 的磁盘空间，便停止清理
eks.tke.cloud.tencent.com/image-gc-period: '3m' # 磁盘空间检查的间隔，默认 3 分钟
```

清理已退出的容器

如果业务原地升级过，或者容器异常退出过，已退出的容器仍会保留，直到磁盘空间达到 85% 时才会清理已退出的容器。清理阈值可以使用如下 Annotation 调整：

```
eks.tke.cloud.tencent.com/container-gc-threshold: "85"
```

如果已退出的容器不想被自动清理（例如需要退出的信息进一步排障），可以通过如下 Annotation 关闭容器的自动清理，但副作用是磁盘空间无法自动释放：

```
eks.tke.cloud.tencent.com/must-keep-last-container: "true"
```

重启磁盘用量高的 Pod

业务需要在容器的系统盘用量超过某个百分比后直接重启 Pod，可以通过 Annotation 配置：

```
eks.tke.cloud.tencent.com/pod-eviction-threshold: "85" # 此功能只在设置后开启，默认不开启
```

只重启 Pod，不会重建子机，退出和启动都会进行正常的 gracestop、prestop、健康检查。

> 注意：
> 此特性上线时间为 2022-04-27，故在此时间前创建的 Pod，需要重建 Pod 来开启特性。

监控指标

9100 端口问题

超级节点上 的 pod 默认会通过 9100 端口对外暴露监控数据，用户可以执行以下命令访问 9100/metrics 获取数据：

- 获取全部指标：

  ```
  curl -g "http://<pod-ip>:9100/metrics"
  ```

- 大集群建议去掉 ipvs 指标：

  ```
  curl -g "http://<pod-ip>:9100/metrics?collect[]=ipvs"
  ```

如果业务本身直接监听了 9100 端口，将会有如下报错，提醒用户 9100 端口已经被使用：

```
listen() to 0.0.0.0:9100, backlog 511 failed (1: Operation not permitted)
```

您可以自定义暴露监控数据的端口号，避免与业务冲突。配置方式如下：

```
eks.tke.cloud.tencent.com/metrics-port: "9110"
```

> 说明：
> 如果 Pod 带有公网 EIP，则需要设置安全组，注意 9100 端口问题，并放通需要的端口。

监控数据上报频率

超级节点上暴露的 cAdvisor 监控数据，默认 30 秒刷新一次，可以通过如下注解调整刷新频率：

```
eks.tke.cloud.tencent.com/cri-stats-interval: '30s'
```

自定义监控指标路径

监控数据默认通过 `/metrics` 路径暴露监控数据，无需改动。如有自定义的需求，可以用下面的注解：

```
eks.tke.cloud.tencent.com/custom-metrics-url: '/metrics'
```

设置 Pod 网卡名称

Pod 默认使用 `eth0` 网卡，如果需更改网卡名称，需要添加如下注解：

```
internal.eks.tke.cloud.tencent.com/pod-eth-idx: '1' # 设置网卡名为 eth1
```

自定义 DNS

Pod 所在宿主机默认使用 VPC 内的 DNS：183.60.83.19，183.60.82.98。如果需要更改宿主机上的 DNS，可以配置如下 annotation：

```
eks.tke.cloud.tencent.com/resolv-conf: |
nameserver 4.4.4.4
nameserver 8.8.8.8
```

> 注意：
> 配置了自定义 DNS 后，则宿主机上的 DNS 以配置的为准。

日志采集到 kafka

超级节点支持通过开源的日志采集组件将日志采集到 kafka。如果采集的是容器内文件，或者采集时进行了 多行合并，则需合理配置单行大小。默认设置的单行大小是 2M，超过 2M 则会丢弃该行日志。日志的单行大小可通过如下注解调整：

```
internal.eks.tke.cloud.tencent.com/tail-buffer-max-size: '2M' # 默认支持最大 2M 的单行日志
```

采集到 kafka 时，日志内容的字段为 `log`，如需将字段名配置为 `message`，则可通过如下注解调整：

```
eks.tke.cloud.tencent.com/log-key-as-message: 'true'
```

日志上报时会额外携带 Pod 的 metadata 信息，信息以 `string` 的格式放在 `kubernetes` 字段，如果需要将 `kubernetes` 字段设置成 `json` 格式，需配置如下注解：

```
eks.tke.cloud.tencent.com/filebeat-metadata-format: 'true'
```

延迟销毁

超级节点上运行的 Job 任务结束后，底层资源就会被销毁；相应的，运行时产生的临时输出，也会被销毁，此时执行 `kubectl logs <pod>` 会提示 `can not found connection to pod`。如果需要保留底层资源定位问题，可以设置延迟销毁底层资源：

```
eks.tke.cloud.tencent.com/reserve-sandbox-duration: '1m' # 开启延迟 1 分钟销毁的功能，Failed 状态的 Pod 最后一个容器退出后，额外保留底层资源 1 分钟
eks.tke.cloud.tencent.com/reserve-succeeded-sandbox: 'false' # 延迟销毁默认只对 Failed 状态的 Pod 生效，变更此字段可将延迟销毁也应用于 Succeeded 状态的 Pod
eks.tke.cloud.tencent.com/reserve-task-shorter-than: '5s' # 如果只关心运行时间较短的 Job，可以配置此参数，只有 Pod 内有任何一个容器的运行时间低于指定值，才会触发延迟销毁。默认不开启。
```

Service 规则同步

关闭 Service 规则同步

超级节点所在集群通过 IPVS 生成 K8s service 规则。 如果无需生成 service 规则，可通过如下注解关闭：

```
eks.tke.cloud.tencent.com/cluster-ip-switch: 'disable'
```

注意：

关闭 service 规则后，则集群内的服务发现都会失效。Pod 将无法使用 `ClusterFirst` 的 `dnsPolicy`，需改成其他类型，例如 `Default`。

等待 Service 规则同步

集群在 Pod 启动的同时同步 service 规则，如果需要在 Pod 启动前先等待 service 规则同步完成，则可以配置：

```
eks.tke.cloud.tencent.com/duration-to-wait-service-rules: '30s' # 启动 Pod 前先等待 service 规则同步完成，此处设置最多等待的时间为 30s
```

设置 IPVS 参数

IPVS 规则可通过如下注解控制：

```
eks.tke.cloud.tencent.com/ipvs-scheduler: 'sh' # 调度算法，sh 是 source hash，按源地址 hash 进行转发，更利于分布式全局的负载均衡。默认为 rr （轮询 round robin）
eks.tke.cloud.tencent.com/ipvs-sh-port: "true" # 按端口进行 source hash，仅在 ipvs-scheduler 为 sh 有效。
eks.tke.cloud.tencent.com/ipvs-sync-period: '30s' # 规则刷新的最大间隔，默认 30s 刷新一次。
eks.tke.cloud.tencent.com/ipvs-min-sync-period: '2s' # 规则刷新的最小间隔，默认 service 一有变更就刷新。调整此参数可避免刷新过于频繁。
```

- 设置集群内访问 CLB 的 VIP 时流量不走 IPVS
  适用于希望集群内访问不走 ipvs 而走 clb，通过给 service 上配置该 annotation 后，不会再生成对应的 ipvs 规则：

  ```
  service.cloud.tencent.com/discard-loadbalancer-ip: 'true' # 该 annotation 配置在 service 上，无需重建 Pod 即可即时生效
  ```

自定义 Pod 时区

超级节点上的 Pod 默认为 UTC 时间，若需要调整 Pod 时区为东8区，可添加如下 Annotation：

```
eks.tke.cloud.tencent.com/host-timezone: 'Asia/Shanghai' # 该 annotation 用于设置 Pod 时区为东8区
```

Serverless 集群全局配置说明

最近更新时间：2022-10-13 10:11:54

操作场景

TKE Serverless 集群支持通过 configmap 进行全局配置。在 TKE 弹超级节点场景以及纯 TKE Serverless 集群场景下，如果用户需要批量对每个超级节点或每个 Pod 设置 annotation，此时在超级节点维度或 Pod 维度进行配置会相对繁琐，对业务 yaml 的侵入性也较大，因此 TKE Serverless 集群提供全局配置的能力，用户可以通过 configmap 进行全局配置来实现对集群内每个 Pod 注入 Annotation 的能力。

操作步骤

1. 在 kube-system 下新建一个 eks-config 的 configmap。
2. 填入相应的参数设置，使其对所有 TKE Serverless 集群的 Pod 生效。
   全局配置参考如下：

```
apiVersion: v1
kind: ConfigMap
metadata:
name: eks-config
namespace: kube-system
data:
pod.annotations: |
eks.tke.cloud.tencent.com/resolv-conf: |
nameserver 183.60.83.19
eks.tke.cloud.tencent.com/host-sysctls: "[{"name": "net.core.rmem_max","value":
"26214400"}]"
```

配置优先级说明

全局配置的优先级最低，其次是超级节点维度的配置，优先级最高的为 Pod 本身的配置，若配置冲突时，按照优先级生效。

镜像缓存

最近更新时间：2023-05-19 16:26:37

镜像缓存概述

使用镜像缓存可以在创建实例时加速拉取镜像，减少实例的启动耗时。该能力适用于 TKE Serverless 集群 Pod 和超级节点。本文主要介绍镜像缓存的工作原理、计费说明、创建和使用方式等。

工作原理

镜像缓存加速启动实例时会事先启动一个容器实例进行镜像拉取，该镜像存储在一个可自定义大小的数据盘中，同时将该数据盘作为云盘快照缓存，即镜像数据已被保存在快照中。在创建容器实例或 Pod 时选择自动匹配或手动匹配镜像缓存，会基于镜像快照创建相应数量的硬盘（数据盘），并直接挂载到实例上，避免镜像层下载，从而提升容器实例及 Pod 的创建速度。
相比 镜像复用 能力，镜像缓存的优势如下：
无时效限制，与镜像缓存（快照）的生命周期有关。
只需提前冷启动一个 Pod，创建快照后该 Pod 即被销毁。
无可用区限制，基于快照创建云盘时自动匹配可用区。
无工作负载的限制，相同地域中即可匹配。

计费说明

在创建镜像缓存时，会涉及到以下资源，相应的计费情况如下：

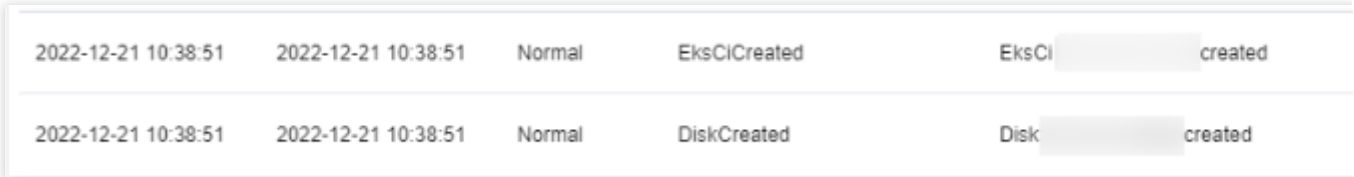
计费项	计费说明	计费文档
镜像缓存	创建镜像缓存时，需要运行一个2核4GiB容器实例以拉取镜像。在镜像缓存创建完成后，该容器实例会自动释放并停止计费。	容器实例计费
CBS 数据盘	创建镜像缓存时，需要绑定一个高性能数据盘存储镜像，该数据盘的大小支持自定义，默认为20G。在镜像缓存创建完成后，该数据盘会自动释放并停止计费。	云硬盘计费
快照	基于上述数据盘会创建一个快照，该快照的生命周期与镜像缓存的生命周期一致。快照按照使用时长及容量收费。	快照计费

使用镜像缓存时，会基于匹配的镜像缓存快照创建一个相同容量的高性能数据盘，并绑定到 Pod 上，因此除去创建 Pod 本身的费用，会额外收取**数据盘的费用**。
数据盘费用 = 容量 * *单价* *实例运行时长

操作步骤

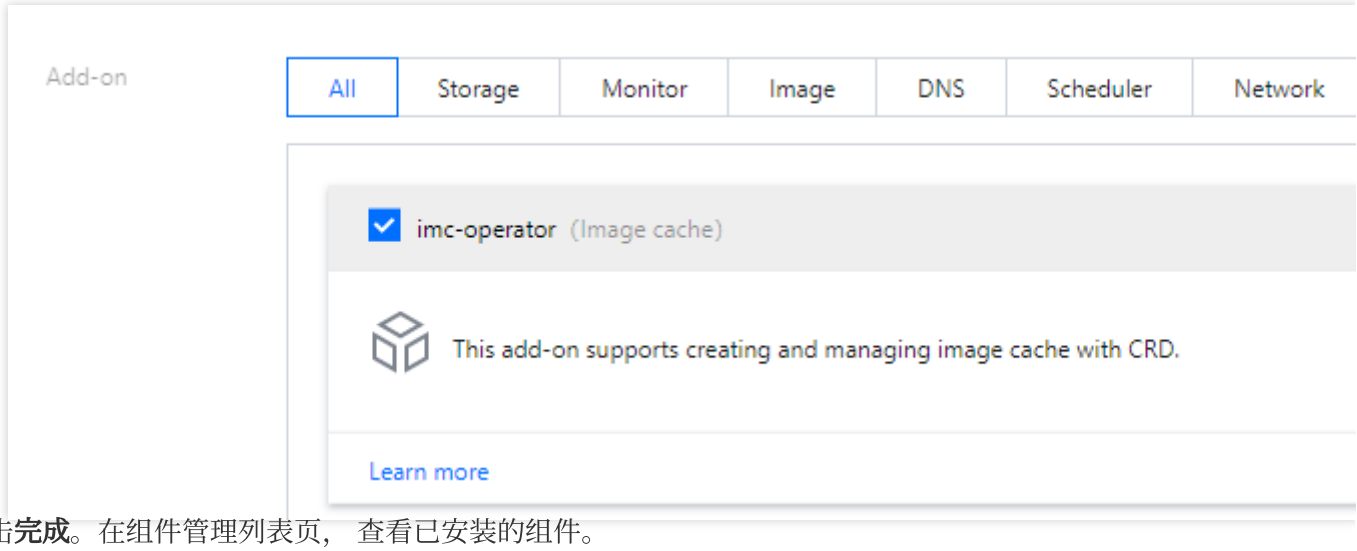
使用控制台创建镜像缓存

1. 登录 容器服务控制台。
2. 选择左侧导航栏的**应用中心 > 镜像缓存**，在镜像缓存页面单击**新建实例**。
3. 在**创建镜像缓存**页面，配置相关参数。
实例名称：自定义。
所在地域：按需选择。
容器网络：为容器实例分配在容器网络地址范围内的 IP 地址。
安全组：安全组具有防火墙的功能，可限制容器实例的网络通信，默认为 default。
OS类型：支持选择 Windows 和 Linux。
镜像：按需选择需要进行缓存的镜像及其版本。
镜像凭证：当选择 Dockerhub 及其他第三方镜像仓库的私有镜像时，必须填写镜像凭证，即仓库的访问地址、用户名及密码。
高级配置：
缓存大小：该大小决定快照及之后创建实例时绑定的数据盘的大小。
注意：
镜像仓库中显示的镜像大小为压缩后的数据，制作镜像缓存有拉取镜像并展开的过程，若镜像过大或镜像压缩比过大将导致默认的 20GB 数据盘不够用，建议设置更大的数据盘空间。
过期策略：选择镜像缓存的保留时间，默认为永久保留。
4. 单击**创建实例**。创建完成后可在镜像缓存的列表，单击事件名称查看创建进程。如下图所示：



使用 CRD 创建镜像缓存

若要使用 CRD 创建镜像缓存服务，需在集群内安装镜像缓存组件，安装后支持以 CRD+Controller 的模式，使用腾讯云镜像缓存服务，而不需要调用云 API 接口。操作步骤如下：
1. 登录 容器服务控制台。
2. 在集群列表中单击 Serverless 集群 ID，进入集群详情页。
3. 在左侧导航栏中选择**组件管理**，单击**新建。**
4. 在**新建组件管理**页面选择 imc-operator（镜像缓存） 组件。如下图所示：



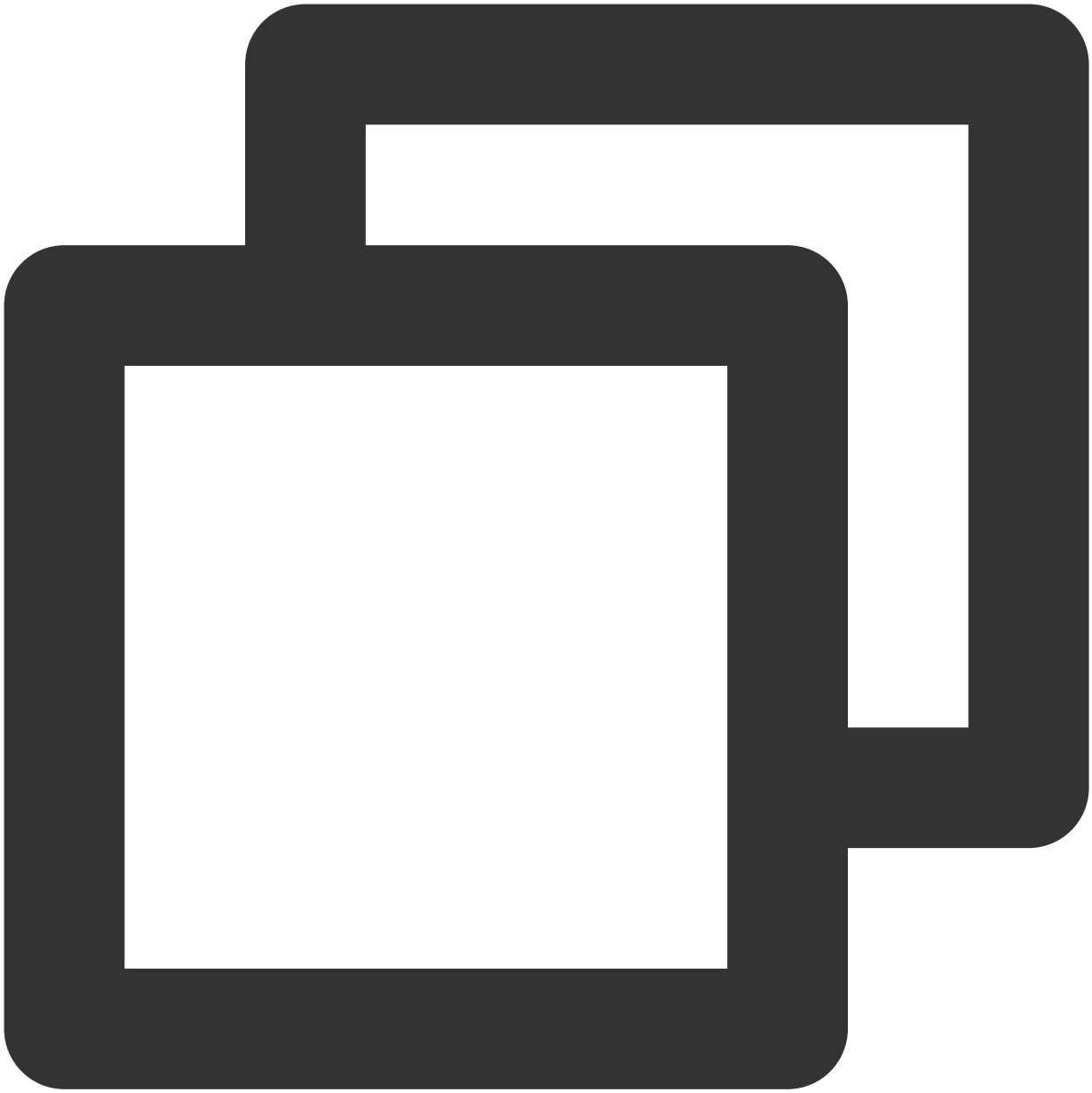
5. 单击**完成**。在组件管理列表页， 查看已安装的组件。

6. 编辑 YAML。 创建 ImageCache：

创建 ImageCache

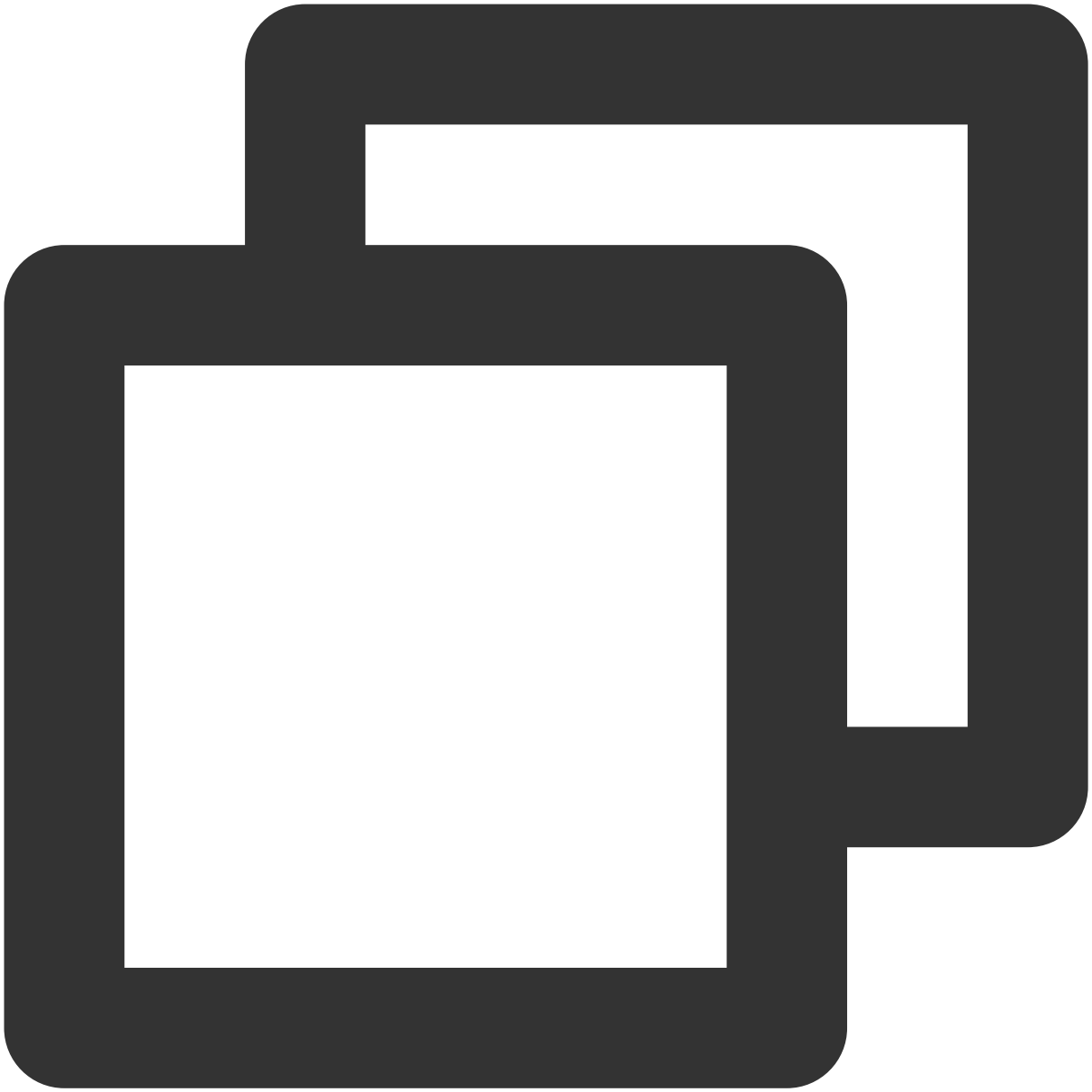
查看 ImageCache

示例：



```
apiVersion: eks.cloud.tencent.com/v1
kind: ImageCache
metadata:
  name: imagecache-sample
spec:
  images:
    - nginx
#  imageCacheSize: 30
  # TODO(user): Add fields here
```

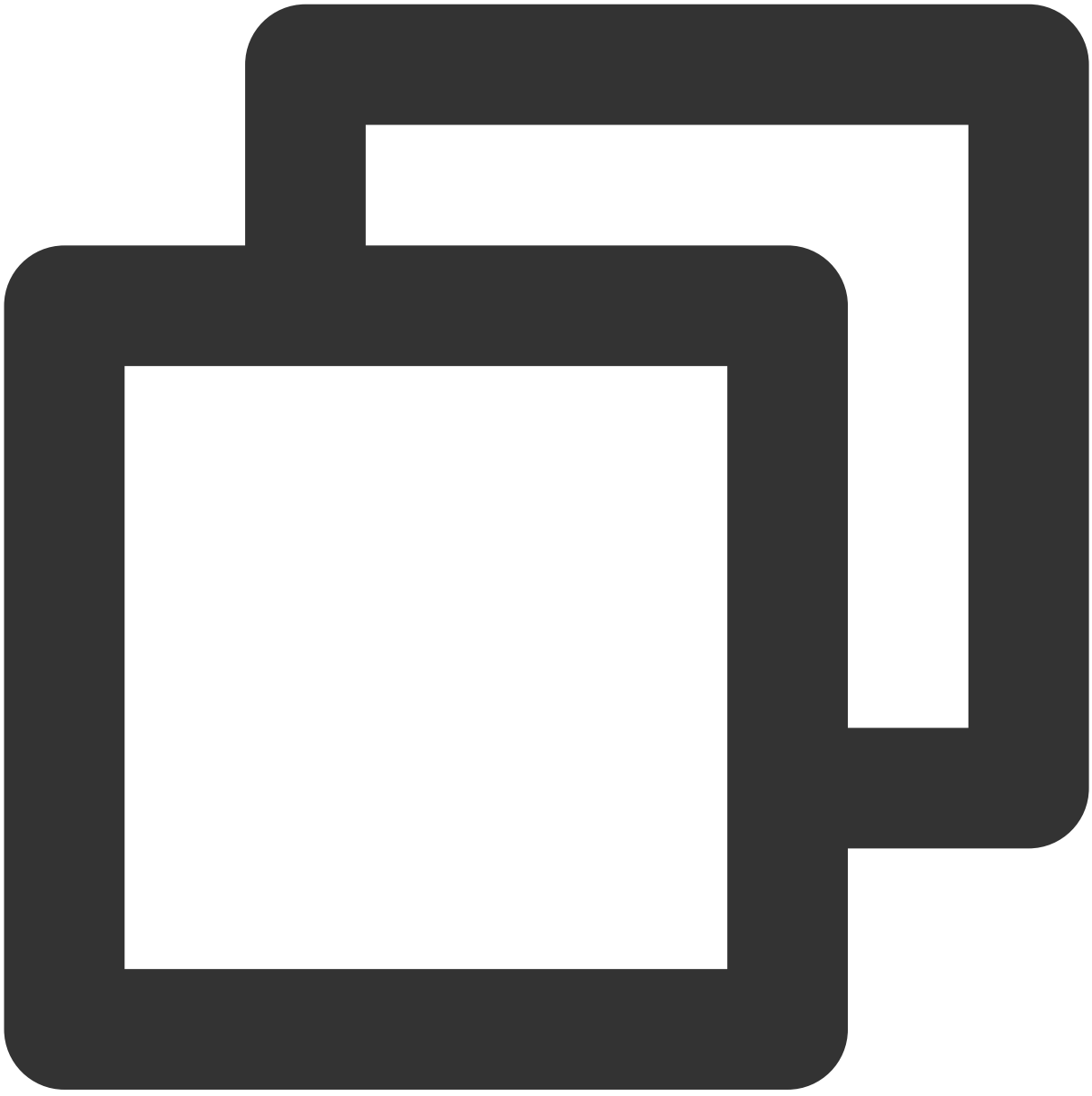
带有更多参数的示例：



```
apiVersion: eks.cloud.tencent.com/v1
kind: ImageCache
metadata:
  annotations:
    "eks.tke.cloud.tencent.com/eip-attributes": '{"InternetMaxBandwidthOut":2}' # 自
  name: imagecache-sample-more-para
spec:
  images:
    - nginx
    - mysql
  imageCacheSize: 30
```

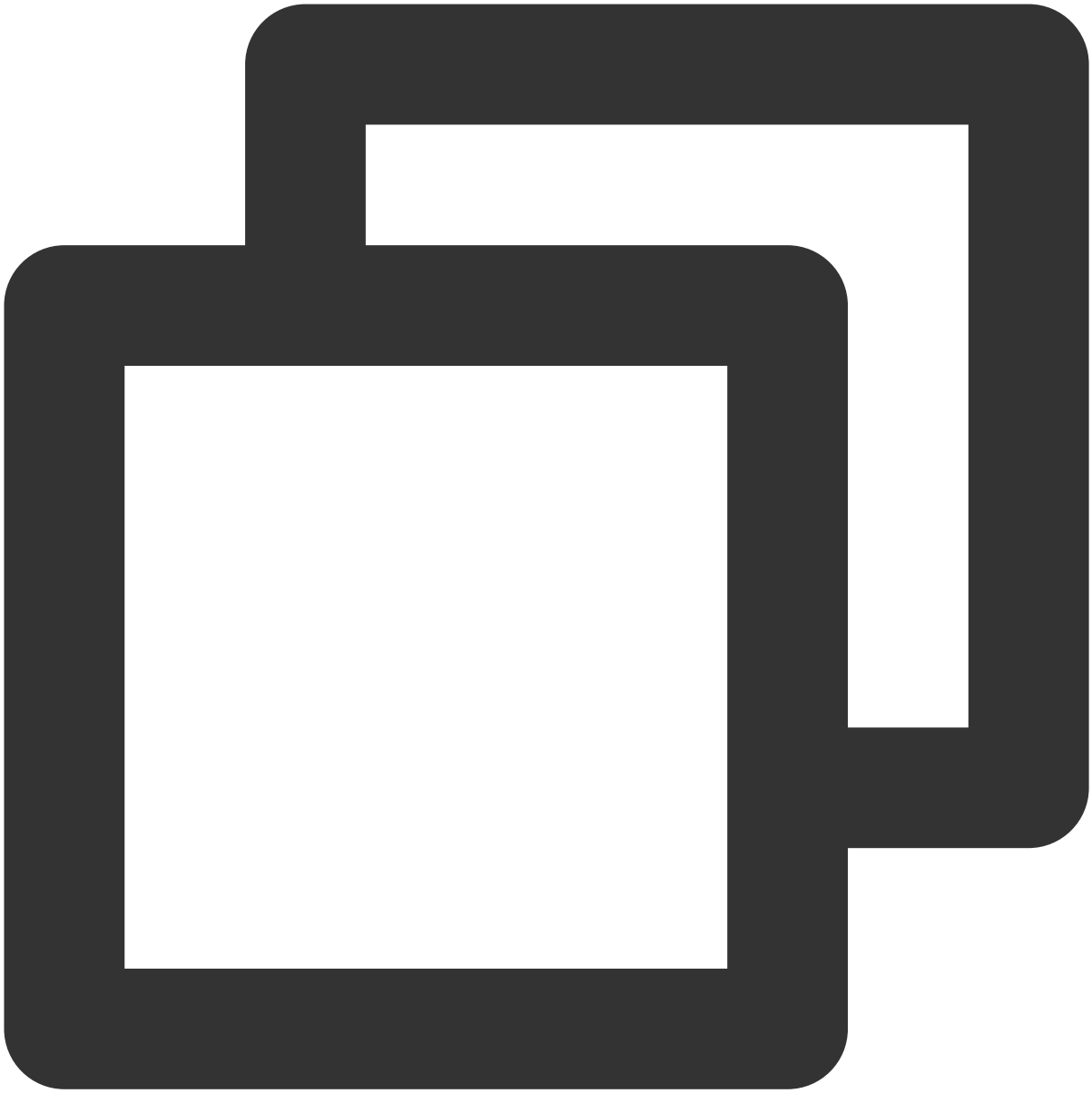
```
  retentionDays: 7
  imagePullSecrets:
    - imc-operator-system/qcloudregistrykey
```

示例：



```
kubectl get imc
```

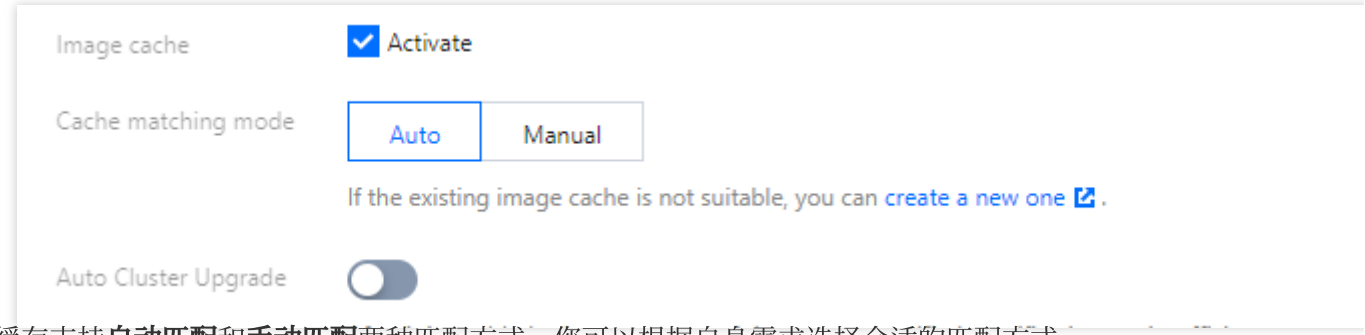
如有异常可查看 events：



```
kubectl describe imc xxx
```

使用已创建的镜像缓存

Serverless 集群内创建 Pod 时，可在**新建工作负载**页面，单击**显示高级设置**，勾选开启镜像缓存功能。如下图所示：



镜像缓存支持**自动匹配**和**手动匹配**两种匹配方式，您可以根据自身需求选择合适的匹配方式。

自动匹配

手动匹配

在缓存匹配模式中选择**自动匹配**时，根据以下匹配策略，将自动匹配最优的镜像缓存。

镜像名称及版本完全相同，则可匹配。

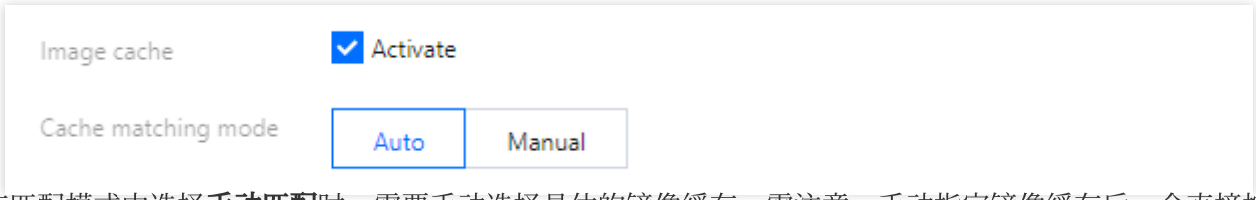
镜像缓存大小，小容量优先匹配。

创建时间，创建时间晚的优先匹配。

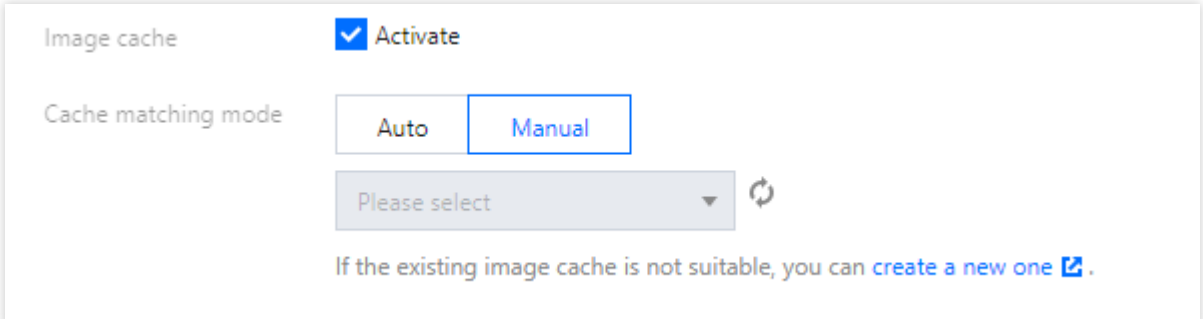
注意

若双方都为 nginx: latest，依旧可匹配，但是由于创建时间不同，可能存在版本不一致的情况，因此，建议创建时镜像缓存及实例时，明确注明版本。

若没有匹配到对应的镜像缓存，则会自动正常拉取镜像。

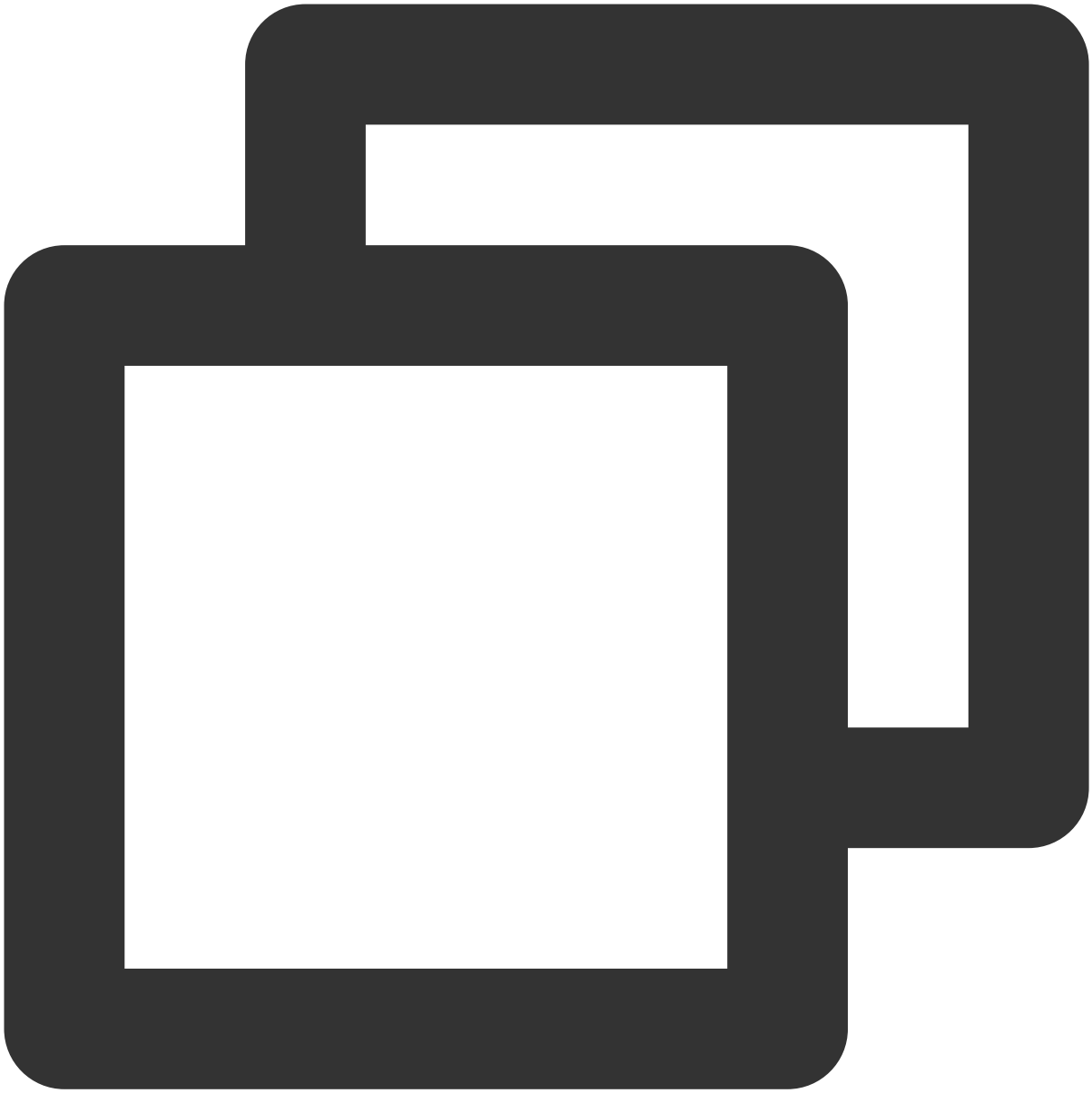


在缓存匹配模式中选择**手动匹配**时，需要手动选择具体的镜像缓存。需注意，手动指定镜像缓存后，会直接按照该镜像缓存快照创建数据盘并绑定到实例上，但是若数据盘中并没有创建时填写的镜像（即手动指定了错误的镜像缓存），会重新在新创建的数据盘中拉取镜像。



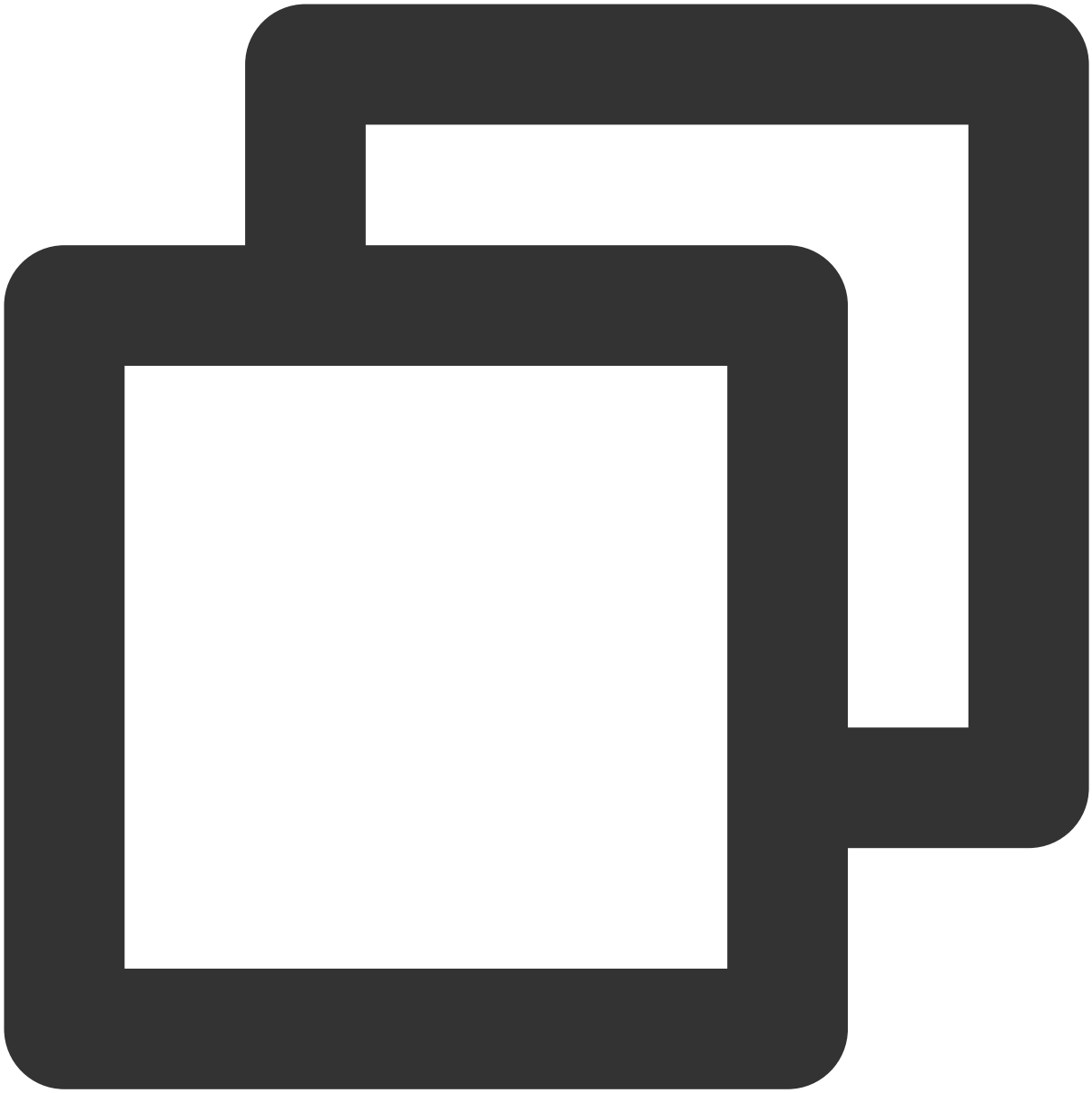
TKE 集群内的超级节点上的 Pod 可通过指定 Pod 的 Annotation 来使用镜像缓存，具体可参考超级节点 Annotation 说明。

自动匹配：



```
eks.tke.cloud.tencent.com/use-image-cache: auto
```

手动指定：



```
eks.tke.cloud.tencent.com/use-image-cache: imc-xxx
```

匹配结果

可从创建实例的事件中，查看是否匹配成功。

若匹配成功，则显示以下事件：

Started	Started container container
Created	Created container container
Pulled	Container image "nginx" already present on machine
ImageCacheUsed	Image cache imc-7fwlzosl used. Disk disk-dz7qm9i8 attache

若未显示该事件，则表示没有匹配到合适的镜像缓存。

需要注意的是，若选择手动匹配，但是匹配的镜像缓存中并没有该镜像，会重新在新创建的数据盘中拉取镜像，并显示以下事件：

Started	Started container container
Created	Created container container
Pulled	Successfully pulled image "nginx" in 5.530971711s
Pulling	Pulling image "nginx"
ImageCacheUsed	Image cache imc-kcbee4nj used. Disk disk-d81vy7vw attached

运维中心
监控和告警

最近更新时间：2022-09-14 10:56:31

本文档介绍 TKE Serverless 集群提供的集群、工作负载、Pod、Container 4个层面的监控数据收集和展示功能。

前提条件

已创建状态为“运行中”的 Serverless 集群，详情请参见 创建集群。

操作步骤

1. 登录容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，选择 Serverless 集群 ID。
3. 在集群资源管理页面，参考以下文档查看监控及设置告警：

- 查看监控数据
- 设置告警

监控及告警指标列表

监控

目前 TKE Serverless 集群提供了以下维度的监控指标，所有指标均为统计周期内的**平均值**。

注意：

- 工作负载使用的 PV 的详细监控指标请参考 云硬盘监控、文件存储监控。
- Service 关联的 CLB 详细的网络监控指标请参考 负载均衡监控。
- 云监控创建告警策略指引，详情请参见 创建告警策略。

集群监控指标

指标	单位	说明

指标	单位	说明
CPU 使用量	核	集群当前所有运行中 Pod 使用的 CPU 规模
内存使用量	B	集群当前所有运行中 Pod 使用的内存规模

工作负载监控指标

指标	单位	说明
Pod 重启次数	次	工作负载内所有 Pod 的重启次数之和
CPU 使用量	核	工作负载内所有 Pod 的 CPU 使用量
CPU 利用率（占 Pod 规格）	%	工作负载内所有 Pod 的 CPU 使用量占分配资源总量之比
内存使用量	B	工作负载内所有 Pod 的内存使用量
内存利用率（占 Pod 规格）	%	工作负载内所有 Pod 的内存使用量占分配资源总量之比

Pod 监控指标

指标	单位	说明
异常状态	-	Pod 的状态，正常或异常
CPU 使用量	核	Pod 的 CPU 使用量
CPU 利用率（占 Request）	%	Pod 的 CPU 使用量和设置的 Request 值之比
CPU 利用率（占 Limit）	%	Pod 的 CPU 使用量和设置的 Limit 值之比
CPU 利用率（占 Pod 规格）	%	Pod 的 CPU 使用量占 Pod 分配总量之比
内存使用量	B	Pod 的内存使用量，含缓存
内存利用率（占 Request）	%	Pod 的内存使用量和设置的 Request 值之比
内存利用率（占 Limit）	%	Pod 的内存使用量和设置的 Limit 值之比
内存利用率（占 Pod 规格）	%	Pod 的内存使用量占 Pod 分配总量之比

Container 监控指标

指标	单位	说明

指标	单位	说明
CPU 使用量	核	Container 的 CPU 使用量
CPU 利用率（占 Request）	%	Container 的 CPU 使用量和设置的 Request 值之比
CPU 利用率（占 Limit）	%	Container 的 CPU 使用量和设置的 Limit 值之比
内存使用量	B	Container 的内存使用量，含缓存
内存利用率（占 Request）	%	Container 的内存使用量和设置的 Request 值之比
内存利用率（占 Limit）	%	Container 的内存使用量和设置的 Limit 值之比

告警

目前 TKE Serverless 集群提供了以下维度的告警指标，所有指标均为统计周期内的**平均值**。

Pod 告警指标

指标	单位	说明
CPU 利用率（占 Pod 规格）	%	Pod 的 CPU 使用量占 Pod 分配总量之比
内存利用率（占 Pod 规格）	%	Pod 的内存使用量占 Pod 分配总量之比
CPU 利用率（占 Request）	%	Pod 的 CPU 使用量和设置的 Request 值之比
内存利用率（占 Request）	%	Pod 的内存使用量和设置的 Request 值之比
CPU 利用率（占 Limit）	%	Pod 的 CPU 使用量和设置的 Limit 值之比
内存利用率（占 Limit）	%	Pod 的内存使用量和设置的 Limit 值之比
Pod 重启次数	次	Pod 的重启次数
Pod Ready	-	Pod 的状态，默认 False 时告警
CPU 使用量	核	Pod 的 CPU 使用量
内存使用量	MB	Pod 的内存使用量，含缓存

日志采集
使用环境变量配置日志采集

最近更新时间：2022-09-23 11:50:22

操作场景

在 TKE Serverless 集群中，用户可以通过环境变量配置日志采集，按行采集日志、不解析。也可以通过 自定义资源定义（CustomResourceDefinitions，CRD）的方式配置日志采集。

本文介绍如何通过环境变量实现 TKE Serverless 集群的日志采集功能。该功能支持将集群内服务的日志发送至 日志服务 CLS 或用户自建 Kafka，适用于需要对 TKE Serverless 集群内服务日志进行存储和分析的用户。

TKE Serverless 集群日志采集功能需要在创建工作负载时手动开启。您可根据以下操作开启日志采集功能：

- 通过控制台配置日志采集
- 通过 yaml 配置日志采集
- 更新日志采集

说明事项

TKE Serverless 集群日志采集功能开启后，日志采集 Agent 根据您配置的采集路径和消费端，将采集到的日志以 JSON 的形式发送到您指定的消费端。消费端及采集路径说明如下：

- **消费端**：日志采集服务支持 Kafka 或 CLS 作为日志的消费端。
- **采集路径**：需要采集的日志的路径。采集路径支持采集标准输出（stdout）和绝对路径，支持 * 通配，多个采集路径以“,”分隔。

前提条件

- 需确认 Kubernetes 集群能够访问日志消费端。
- 日志长度限制为单条2M，如果超过则会截断。

> 注意
> 若日志输出速率过快，为避免 OOM，需要调整此参数配置，详情请参见 如何调整日志采集配置。

操作步骤

通过控制台配置日志采集

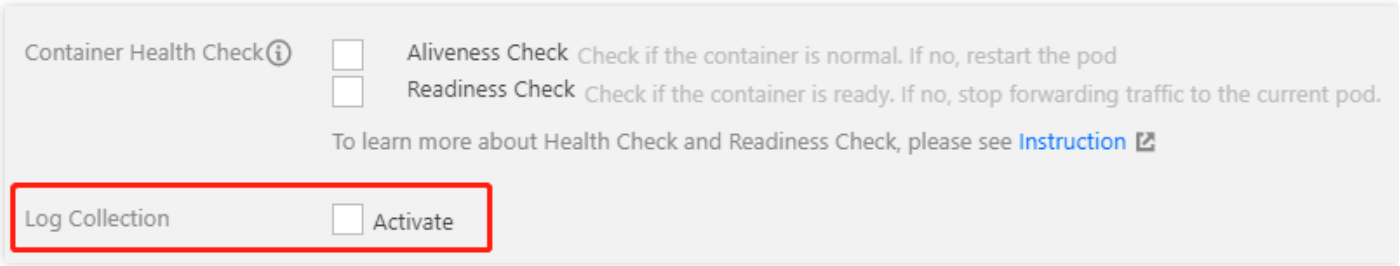
TKE Serverless 集群日志采集功能采集到的日志信息将会以 JSON 格式输出到您指定的消费端，并会附加相关的 Kubernetes metadata，包括容器所属 Pod 的 label 和 annotation 等信息。具体操作步骤如下：

1. 登录 容器服务控制台，选择左侧导航栏中的**集群**。

2. 在集群管理页面，单击 Serverless 集群 ID，进入集群详情页。

3. 在左侧“工作负载”中选择需要的工作负载类型，进入对应页面后选择**新建**。

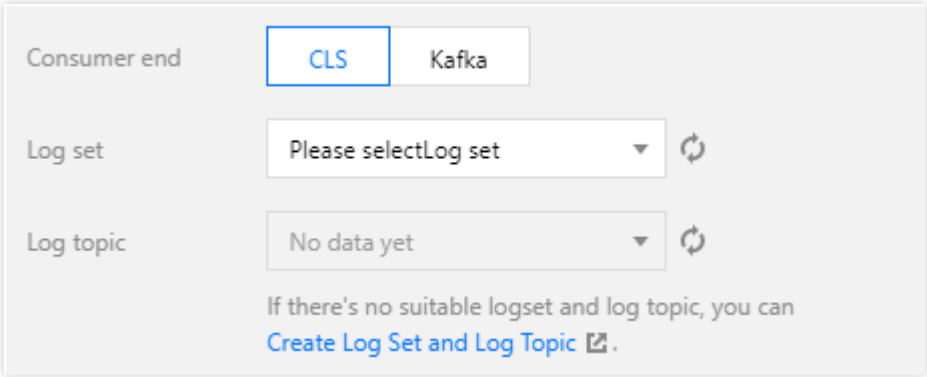
4. 在“实例内容器”中选择**显示高级设置**，并勾选“开启”日志采集。如下图所示：



5. 参考以下信息进行日志消费端配置，您可选择 CLS 或 Kafka 作为日志消费端。

   - 配置CLS作为日志消费端
   - 配置Kafka作为日志消费端

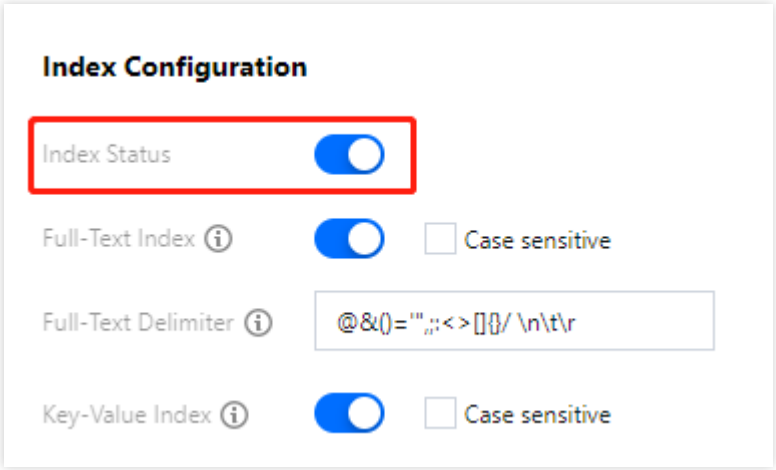
   i. 选择 CLS 作为日志消费端，并选择日志集和日志主题。如下图所示：



若无合适的日志集，请参考 新建日志集及日志主题。

日志服务 CLS 目前仅支持同地域的容器集群进行日志采集上报。

ii. 打开日志主题的**日志索引**。索引配置是使用日志服务 CLS 进行检索分析的必要条件。若未开启，则无法查看日志。配置索引的详细操作，请参见 日志服务配置索引。
您可在 **日志服务控制台**>**日志主题**中，选择日志主题名称，在“索引配置”页面开启索引。如下图所示：

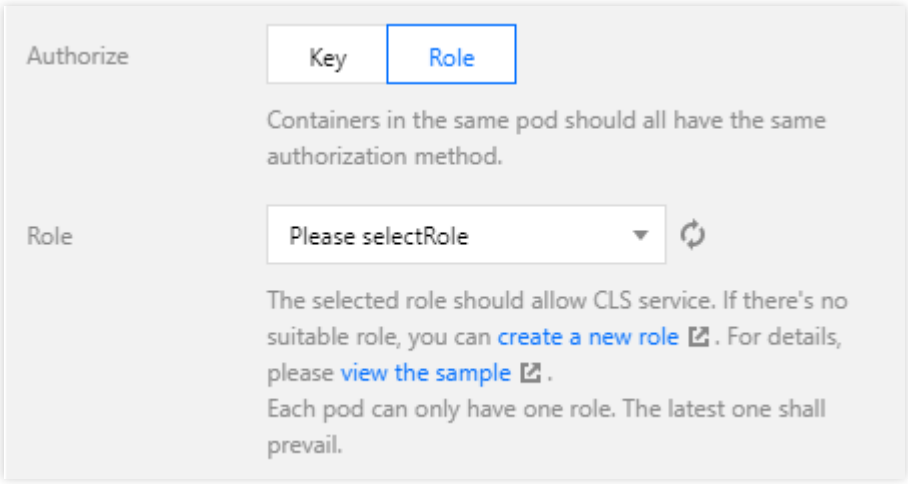


6. 选择角色或者密钥进行授权。

注意
- 同一 pod 下的容器只能选择同一种授权方式，以您最后修改的授权方式为准。例如第一个容器选择了密钥授权，第二个选择了角色授权，最终两个容器都是角色授权。
- 同一 pod 下的容器只能选择同一个角色授权。

- 角色授权
- 密钥授权

- 选择具有访问日志服务 CLS 权限的角色名称，如下图所示：



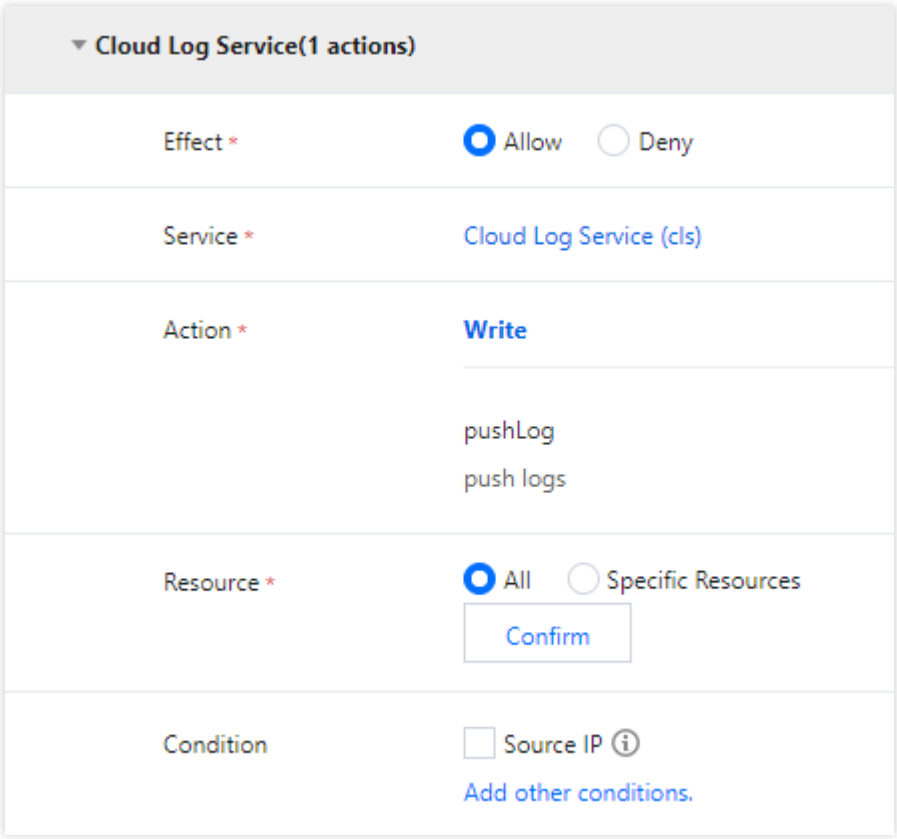
- 若无合适的角色，创建过程参考以下步骤：

  新建策略

  在新建角色之前，您需要创建一个策略，该策略决定了您的角色具备的权限。

  a. 登录访问管理控制台，在左侧导航栏选择 **策略**。
  b. 在“策略”页面，单击**新建自定义策略**。
  c. 在“选择创建策略方式” 弹窗中，选择**按策略生成器创建**。

d. 在“可视化策略生成器”中，“服务”选择 “日志服务(cls)”，“操作”选择“写操作:pushLog”，如下图所示：



e. 单击**下一步**，进入“关联用户/用户组”页面。

f. 确认策略名称，单击**完成**即可创建策略。

新建角色

创建策略完成后，需要将该策略绑定至一个角色，使得该角色具备策略相应的权限。

a. 登录访问管理控制台，在左侧导航栏选择 **角色**。

b. 在“角色”页面，单击**新建角色**。

c. 在“选择角色载体” 弹窗中，选择**腾讯云产品服务**，进入**新建自定义角色**页面。

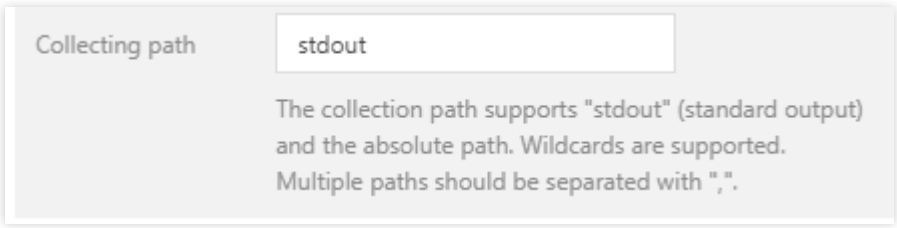
d. 在“输入角色载体信息”步骤中，选择**绑定**云服务器（cvm）**载体**，单击**下一步**。

> 注意
> 必须选择**云服务器（cvm）**作为角色载体，选择容器服务则无法完成授权。

e. 在“配置角色策略”步骤中，选择 已创建的策略，单击**下一步**。

f. 在“审阅”步骤中，输入您的角色名称，审阅您即将创建角色的相关信息，单击**完成**后即完成自定义角色创建。详情请参见 创建角色。

7. 配置采集路径。如下图所示：



至此已完成日志采集功能配置，您可按需进行该工作负载的其他配置。

通过 yaml 配置日志采集

本文提供采集日志到 Kafka、通过 secret 采集日志到 CLS 和通过 role 采集日志到 CLS 三种方式，请按需选择：

> 注意：
> 若 yaml 中同时配置了密钥和角色授权，pod 实际上采用的是角色授权。

- 采集日志到Kafka
- 通过secret采集日志到CLS
- 通过role采集日志到CLS

通过增加环境变量开启日志采集。

```
apiVersion: apps/v1beta2
kind: Deployment
metadata:
annotations:
deployment.kubernetes.io/revision: "1"
labels:
k8s-app: kafka
qcloud-app: kafka
name: kafka
namespace: default
spec:
replicas: 1
selector:
```

```
matchLabels:
k8s-app: kafka
qcloud-app: kafka
template:
metadata:
annotations:
eks.tke.cloud.tencent.com/cpu: "0.25"
eks.tke.cloud.tencent.com/mem: "0.5Gi"
labels:
k8s-app: kafka
qcloud-app: kafka
spec:
containers:
- env:
- name: EKS_LOGS_OUTPUT_TYPE
value: kafka
- name: EKS_LOGS_KAFKA_BROKERS
value: 10.0.16.42:9092
- name: EKS_LOGS_KAFKA_TOPIC
value: eks
- name: EKS_LOGS_KAFKA_MESSAGE_KEY #选填
valueFrom:
fieldRef:
fieldPath: metadata.name
- name: EKS_LOGS_METADATA_ON
value: "true"
- name: EKS_LOGS_LOG_PATHS
value: stdout,/tmp/busy*.log
image: busybox:latest
command: ["/bin/sh"]
args: ["-c", "while true; do echo hello world; date; echo hello >> /tmp/busy.log;
sleep 1; done"]
imagePullPolicy: Always
name: while
resources:
requests:
cpu: 250m
memory: 512Mi
```

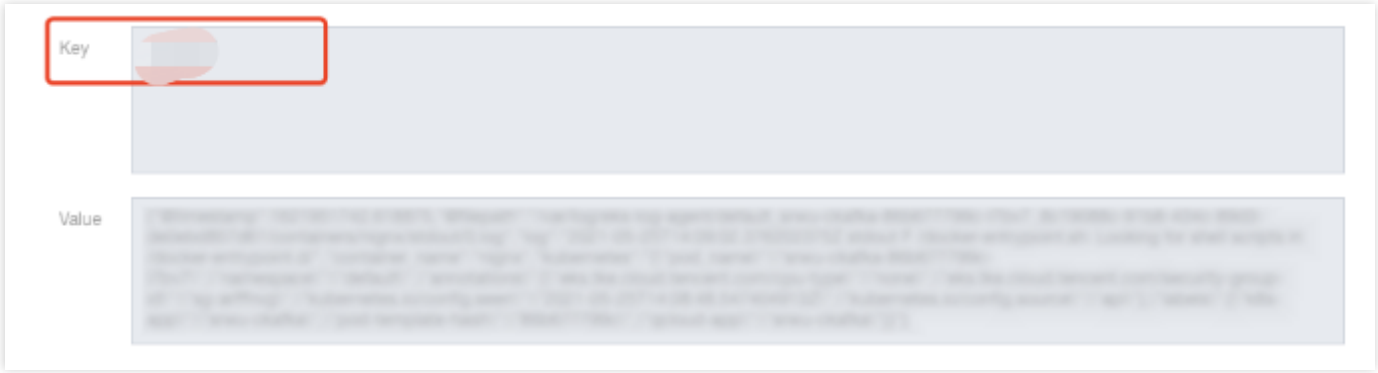
字段说明：

字段名	含义
EKS_LOGS_OUTPUT_TYPE	消费端支持 kafka 和 cls，根据该 key 判断是否启用日志收集。
EKS_LOGS_LOG_PATHS	日志路径，支持 stdout（表示采集标准输出）和绝对路径，支持 * 通

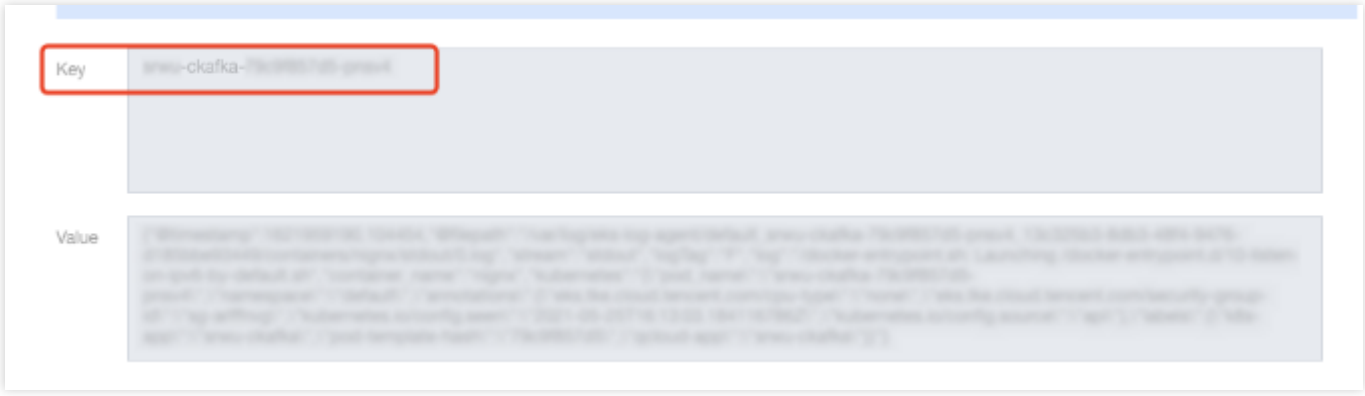
	配，多个路径用“,”分隔。
EKS_LOGS_METADATA_ON	支持 true 或 false。不填写则默认为 true。
EKS_LOGS_KAFKA_TOPIC	日志主题。
EKS_LOGS_KAFKA_BROKERS	kafka brokers，ip1:port1，ip1:port2，ip2:port2格式，多个用“,”分隔。对外用此环境变量，EKS_LOGS_KAFKA_HOST 以后不再对外可见。
EKS_LOGS_KAFKA_MESSAGE_KEY	非必填。支持指定一个 key，将日志投递到指定分区。 • 对于未开启按 key 投递，日志将随机投递到不同分区里。 • 开启按 key 投递，带有同样 key 的日志，将投递到相同的分区里。 **注意**：此处 key 从 Pod 的字段获取变量值，以上示例皆以 Pod name 为例，同时还支持 namespace、PodIP 等，详情可参见 kubernetes 社区文档。

对于开启按 key 投递到 kafka 指定分区，验证方式如下：

- 未开启时，查询消息不显示 key，如下图所示：



- 当开启后，查询消息显示 key，如下图所示：

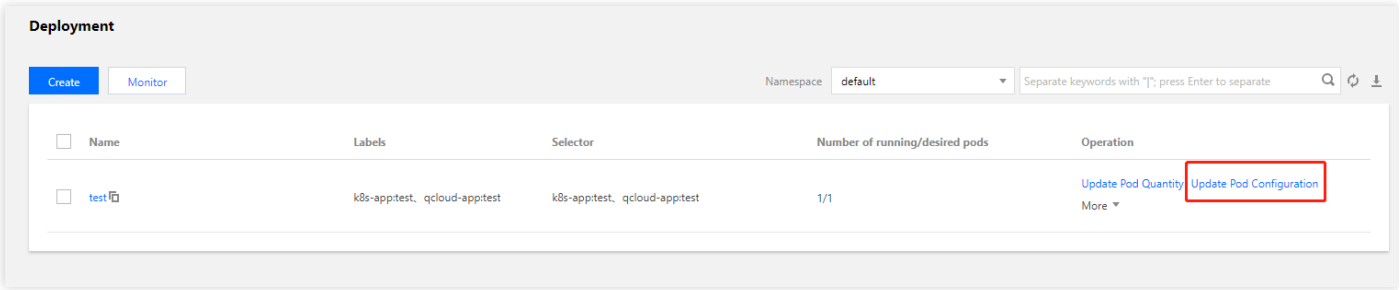


更新日志采集

您可通过控制台和 yaml 更新日志采集，请参考以下步骤：

- 通过控制台更新日志采集
- 通过yaml更新日志采集

1. 登录 容器服务控制台，选择左侧导航栏中的**集群**。
2. 在集群管理页面，单击 Serverless 集群 ID，进入集群详情页。
3. 选择左侧**工作负载**，单击需要更新日志采集的工作负载所在行右侧的**更新Pod配置** > **显示高级设置**，修改对应的配置。如下图所示：



4. 单击**完成**即可更新。

常见问题

如遇问题，您可先查询 Serverless 集群日志采集相关问题。如果您的问题仍未解决，请 联系我们。

使用 CRD 采集日志到 CLS 开启日志采集

最近更新时间：2022-09-14 18:10:51

操作场景

在 TKE Serverless 集群中，用户可以 通过环境变量配置日志采集。也可以通过自定义资源（CustomResourceDefinitions，CRD）的方式配置日志采集。

CRD 对 Pod 无侵入性，支持单行、多行、分隔符、完全正则、JSON 等多种日志解析方式，将标准输出、容器内文件日志发送至 腾讯云日志服务 CLS，提供检索分析、可视化应用、日志下载消费等服务。**推荐使用 CRD 配置日志采集。**

注意

- 使用 CRD 配置日志采集目前只对2021年5月25号后新建的 Pod 生效，若需为旧 Pod 配置日志采集，请销毁重建。
- 若采集的 Pod 同时配置环境变量及 CRD 采集日志，会造成重复采集、重复计费。故使用 CRD 配置日志采集时，请删除相关环境变量。

概念

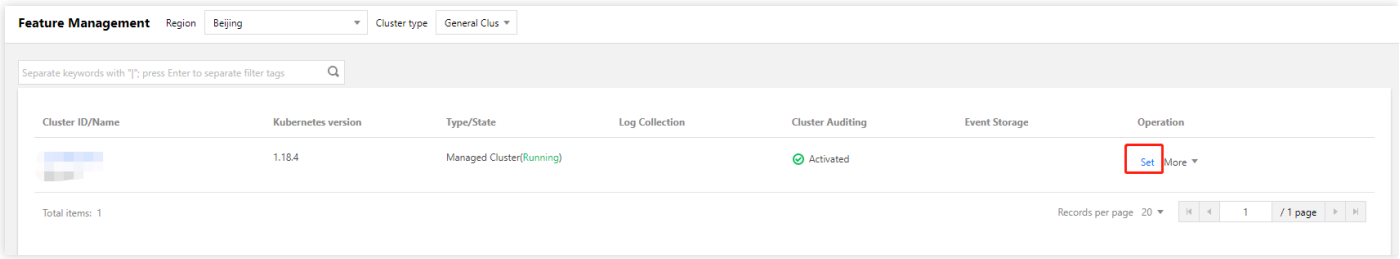
- **日志规则**：用户可以使用日志规则指定日志的采集源、消费端、日志解析方式、配置过滤器及上传解析失败等能力。日志采集器会监测日志采集规则的变化，变化的规则会在最多10s内生效。
- **日志源**：包含指定容器标准输出及容器内文件。
  - 在采集容器标准输出日志时，用户可选择所有容器、或指定工作负载和指定 Pod Labels 内的容器服务日志作为日志的采集源。
  - 在采集容器文件路径日志时，用户可指定工作负载或 Pod Labels 内容器的文件路径日志作为采集源。
- **消费端**：用户选择日志服务 CLS 的日志集和日志主题作为消费端。
- **提取模式**：日志采集 Agent 支持将采集到的日志以单行文本、JSON、分隔符、多行文本和完全正则的形式发送至用户指定的日志主题。
- **过滤器**：开启过滤器后可以根据用户指定的规则采集部分日志，key 支持完全匹配，过滤规则支持正则匹配，例如仅采集 ErrorCode = 404 的日志。
- **上传解析失败**：开启后，所有解析失败的日志，均以作为键名称（Key），原始日志内容作为值（Value）进行上传。关闭时会丢弃失败的日志。

操作步骤

首次授权

首次使用 TKE Serverless 集群的日志采集功能，需要您对 CLS 等相关权限进行授权，以保证将日志正常上传到 CLS。

1. 登录 容器服务控制台，选择左侧导航栏中的**运维功能管理**。
2. 在“功能管理”页面上方选择地域和 Serverless 集群，单击需要开启日志采集的集群右侧的**设置**。如下图所示：



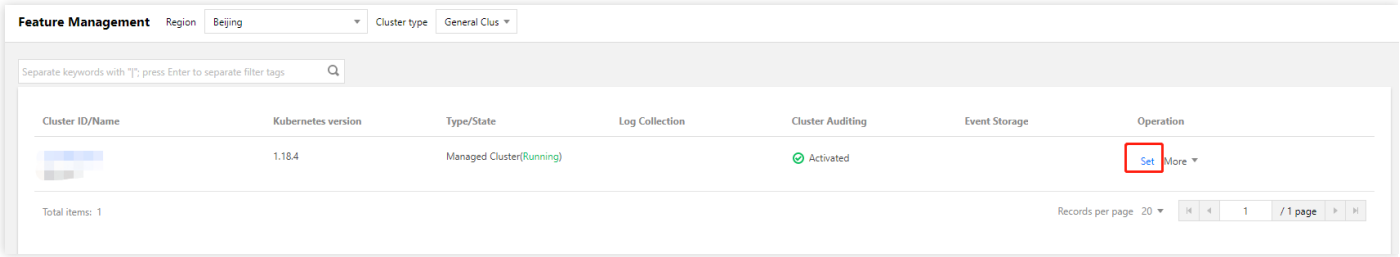
3. 在“设置功能”页面，单击**访问管理**，为服务授权。
   完成授权后，会默认为您的账号绑定角色 TKE_QCSLinkedRoleInEKSLog，该角色配置的预设策略为 QcloudAccessForTKELinkedRoleInEKSLog。

> 说明：
> 您只需在首次使用日志采集功能时进行授权，后续使用无需再操作。若您删除了以上角色，则会再次触发授权操作。

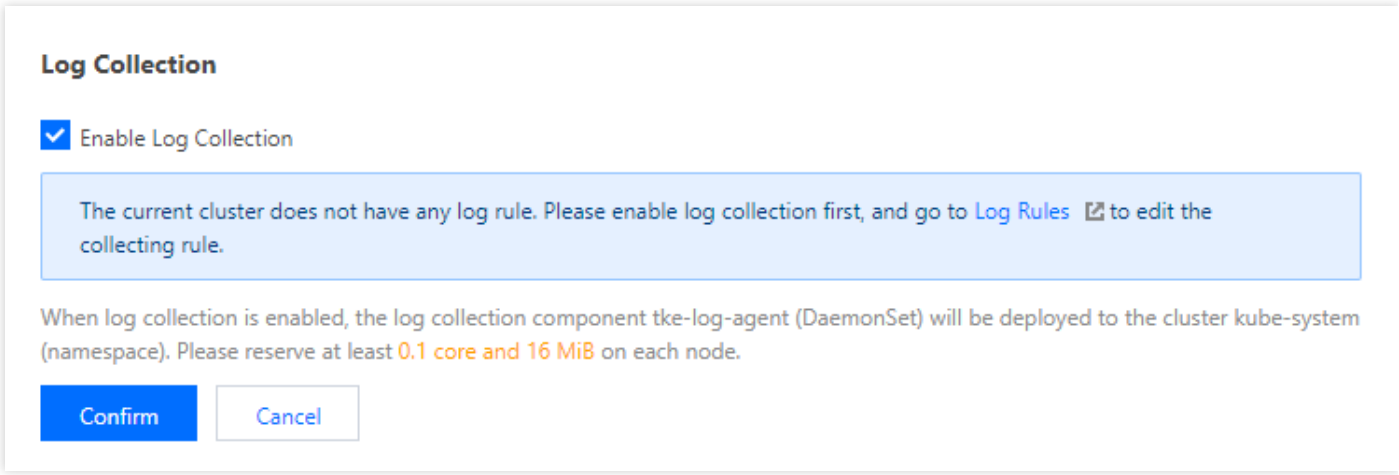
开启日志采集

完成授权后，即可开启日志采集。

1. 登录 容器服务控制台，选择左侧导航栏中的**运维功能管理**。
2. 在“功能管理”页面上方选择地域和 Serverless 集群，单击需要开启日志采集的集群右侧的**设置**。如下图所示：



3. 在“设置功能”页面，单击日志采集**编辑**，开启日志采集后确认。如下图所示：



下一步操作

开启日志采集后，您可根据以下操作使用 CRD 配置 TKE Serverless 集群的日志采集功能：

- 通过控制台配置日志采集
- 通过 YAML 配置日志采集

常见问题

如遇问题，请 提交工单 联系我们。

通过控制台配置日志采集

最近更新时间：2023-05-22 16:49:10

本文介绍使用 CRD 配置 TKEServerless 集群的日志采集功能。

前提条件

登录 容器服务控制台，并为 Serverless 集群开启日志采集功能。操作详情请参见 开启日志采集。

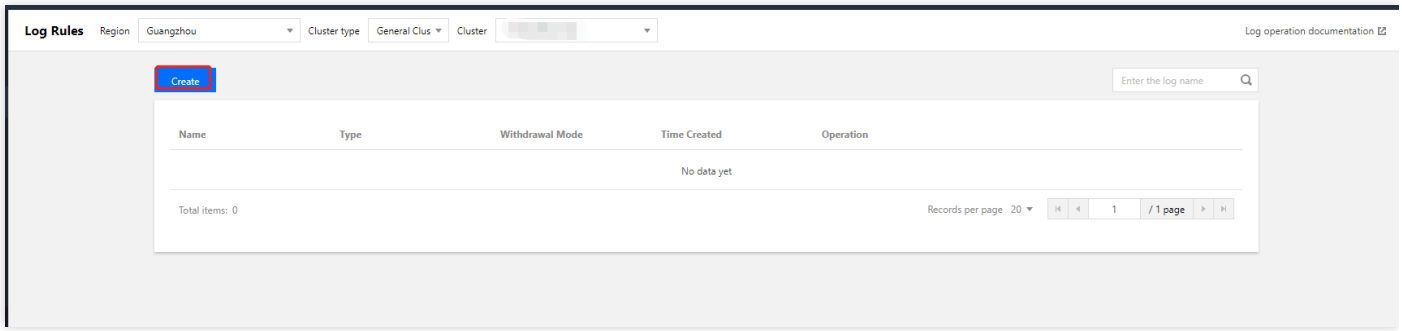
操作步骤

为集群开启日志采集功能后，您可根据以下操作进行配置：

配置日志规则

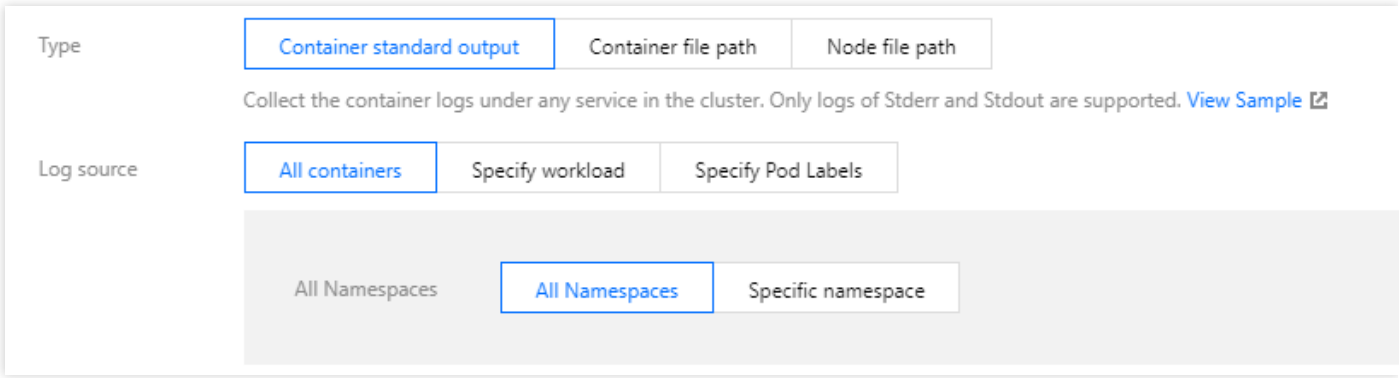
开启日志采集后，需要配置日志规则，确认日志源、消费端、日志解析方式等。

1. 登录 容器服务控制台，选择左侧导航栏中的**日志管理** > **日志规则**。
2. 在“日志采集”页面上方选择地域和需要配置日志采集规则的 TKE Serverless 集群，单击**新建**。如下图所示：



3. 在“新建日志采集规则”页面中，选择采集类型，并配置日志源、消费端、日志解析方式。目前采集类型支持 容器标准输出 和 容器文件路径。
   - 采集容器标准输出日志
   - 采集容器内文件日志

选择**容器标准输出**采集类型，并根据自身需求，配置日志源。如下图所示：



该类型日志源支持采集：

- **所有容器**：所有 Namespace 或某个 Namespace 下的所有容器。
- **指定工作负载**：某 Namespace 下，指定工作负载下的某些容器，并支持添加多个 Namespace。
- **指定 Pod Labels**：某 Namespace 下，指定多个 Pod Labels，采集符合该 Lables 的所有容器。

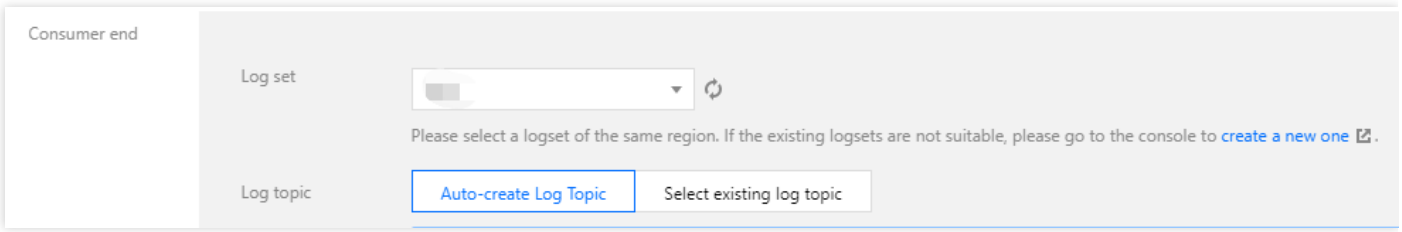
说明

对于容器的标准输出及容器内文件，除了原始的日志内容，还会带上容器或 kubernetes 相关的元数据（例如，产生日志的 Pod name）一起上报到 CLS，方便用户查看日志时追溯来源或根据容器标识、特征（例如，容器名、Labels）进行检索。

容器或 kubernetes 相关的元数据请参考下方表格：

字段名	含义
cluster_id	日志所属的集群 ID。
container_name	日志所属的容器名称。
image_name	日志所属容器的镜像名称 IP。
namespace	日志所属 pod 的 namespace。
pod_uid	日志所属 pod 的 UID。
pod_name	日志所属 pod 的名字。
pod_ip	日志所属 pod 的IP。
pod_lable_{label name}	日志所属 pod 的 label（例如一个 pod 带有两个 label：app=nginx，env=prod，则在上传的日志会附带两个 metedata：pod_label_app:nginx，pod_label_env:prod）。

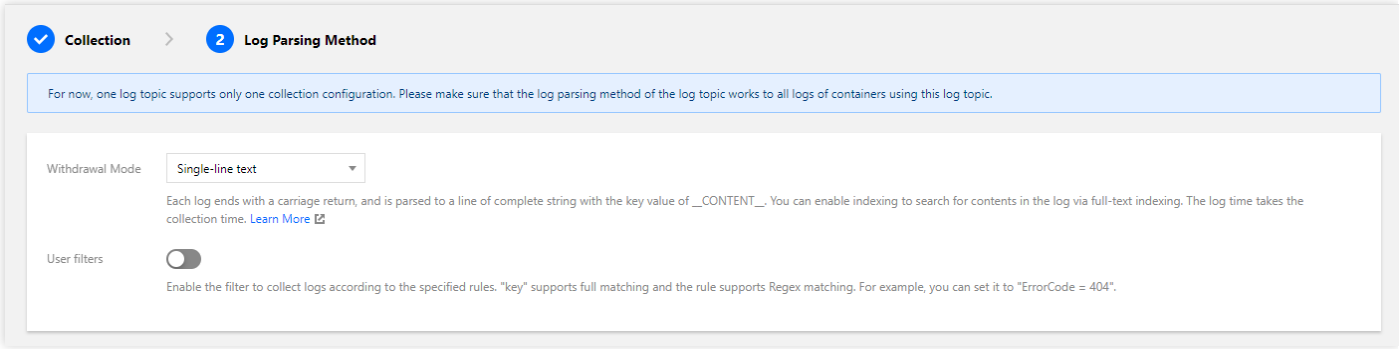
4. 配置日志服务 CLS 为消费端。选择日志集和相应的日志主题，建议选择自动新建日志主题。如下图所示：



注意

- 日志服务 CLS 目前只能支持同地域的容器集群进行日志采集上报。
- 日志集和日志主题在日志规则完成后不可更新。

5. 单击**下一步**，选择日志提取模式。如下图所示：



展开全部

多类提取模式说明

展开&收起

解析模式	说明	相关文档
单行全文	一条日志仅包含一行的内容，以换行符 \n 作为一条日志的结束标记，每条日志将被解析为键值为 **CONTENT** 的一行完全字符串，开启索引后可通过全文检索搜索日志内容。日志时间以采集时间为准。	单行全文格式
多行全文	指一条完整的日志跨占多行，采用首行正则的方式进行匹配，当某行日志匹配上预先设置的正则表达式，就认为是一条日志的开头，而下一个行首出现作为该条日志的结束标识符，也会设置一个默认的键值 **CONTENT**，日志时间以采集时间为准。支持 自动生成正则表达式。	多行全文格式

解析模式	说明	相关文档
单行 - 完全正则	指将一条完整日志按正则方式提取多个 key-value 的日志解析模式，您需先输入日志样例，其次输入自定义正则表达式，系统将根据正则表达式里的捕获组提取对应的 key-value。支持 自动生成正则表达式。	单行 - 完全正则格式
多行 - 完全正则	适用于日志文本中一条完整的日志数据跨占多行（例如 Java 程序日志），可按正则表达式提取为多个 key-value 键值的日志解析模式，您需先输入日志样例，其次输入自定义正则表达式，系统将根据正则表达式里的捕获组提取对应的 key-value。支持 自动生成正则表达式。	多行-完全正则格式
JSON	JSON 格式日志会自动提取首层的 key 作为对应字段名，首层的 value 作为对应的字段值，以该方式将整条日志进行结构化处理，每条完整的日志以换行符 \n 为结束标识符。	JSON 格式
分隔符	指一条日志数据可以根据指定的分隔符将整条日志进行结构化处理，每条完整的日志以换行符 \n 为结束标识符。日志服务在进行分隔符格式日志处理时，您需要为每个分开的字段定义唯一的 key，无效字段即无需采集的字段可填空，不支持所有字段均为空。	分隔符格式

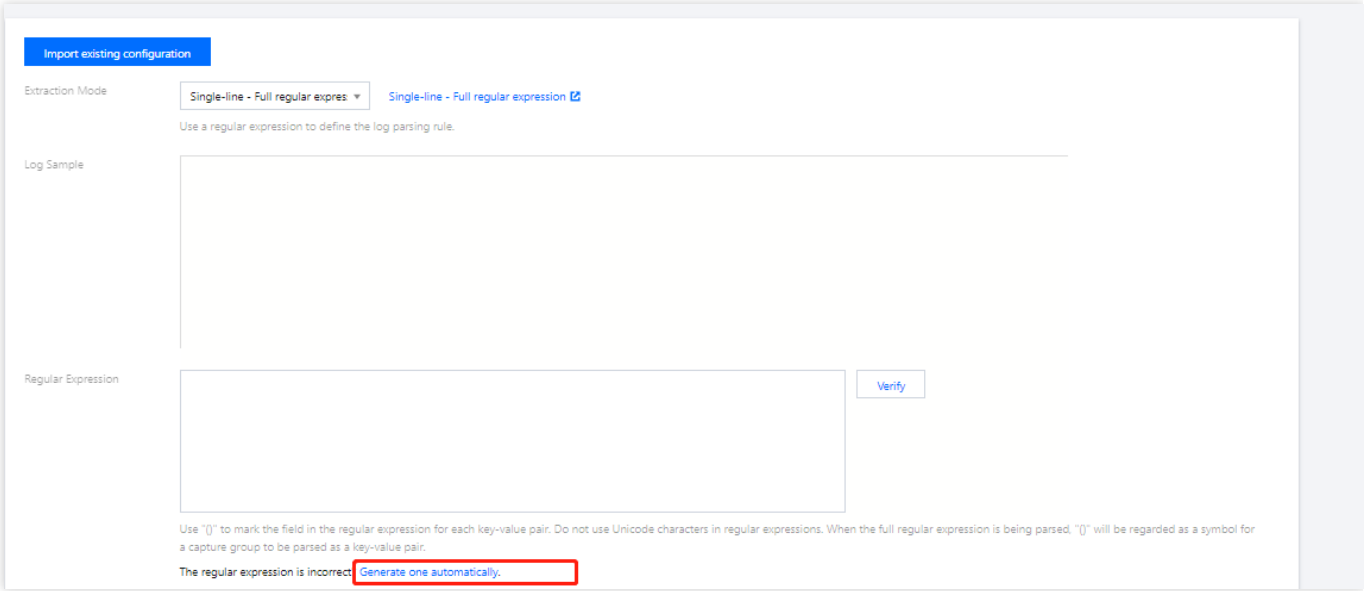
正则表达式自动生成说明

展开&收起

为方便您的使用，在选择**多行 - 完全正则**、**单行 - 完全正则**、**多行全文**的提取模式时，支持**根据日志样例自动生成正则表达式**。

以**单行 - 完全正则**为例，使用步骤如下：

i. 单击**正则表达式自动生成**。如下图所示：



ii. 在“正则表达式自动生成”弹窗中，选中日志样例中需要提取的字段，填写 key 值。

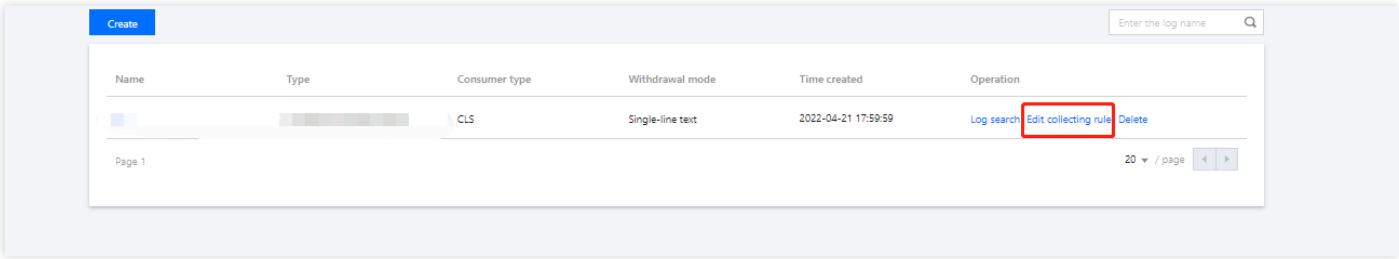
iii. 单击**确认提取**，即可生成该字段对应的正则表达式，并自动填写提取结果。重复此操作，直至日志完全被提取完成。

> 注意
> 一个日志主题目前仅支持一个采集配置，请保证选用该日志主题的所有容器的日志都可以接受采用所选的日志解析方式。若在同一日志主题下新建了不同的采集配置，旧的采集配置会被覆盖。

6. 根据自身需求开启其他功能。
   - 开启过滤器，并配置规则。
     开启后，仅采集符合过滤器规则的日志，Key 支持完全匹配，过滤规则支持正则匹配，如仅采集 ErrorCode = 404 的日志；
   - 开启上传解析失败日志。
     开启后，所有解析失败的日志，均以作为键名称（Key），原始日志内容作为值（Value）进行上传。关闭时会丢弃失败的日志。
7. 单击**完成**，完成创建。

更新日志规则

1. 登录 容器服务控制台，选择左侧导航栏中的**日志管理** > **日志规则**。
2. 在“日志规则”页面中，选择需要更新的日志规则，单击右侧的**编辑收集规则**。如下图所示：



3. 根据需求更新相应配置，单击**完成**，完成更新。

通过 YAML 配置日志采集

最近更新时间：2023-03-14 18:19:11

本文介绍通过 YAML 方式使用 CRD 配置 TKE Serverless 集群的日志采集功能。

前提条件

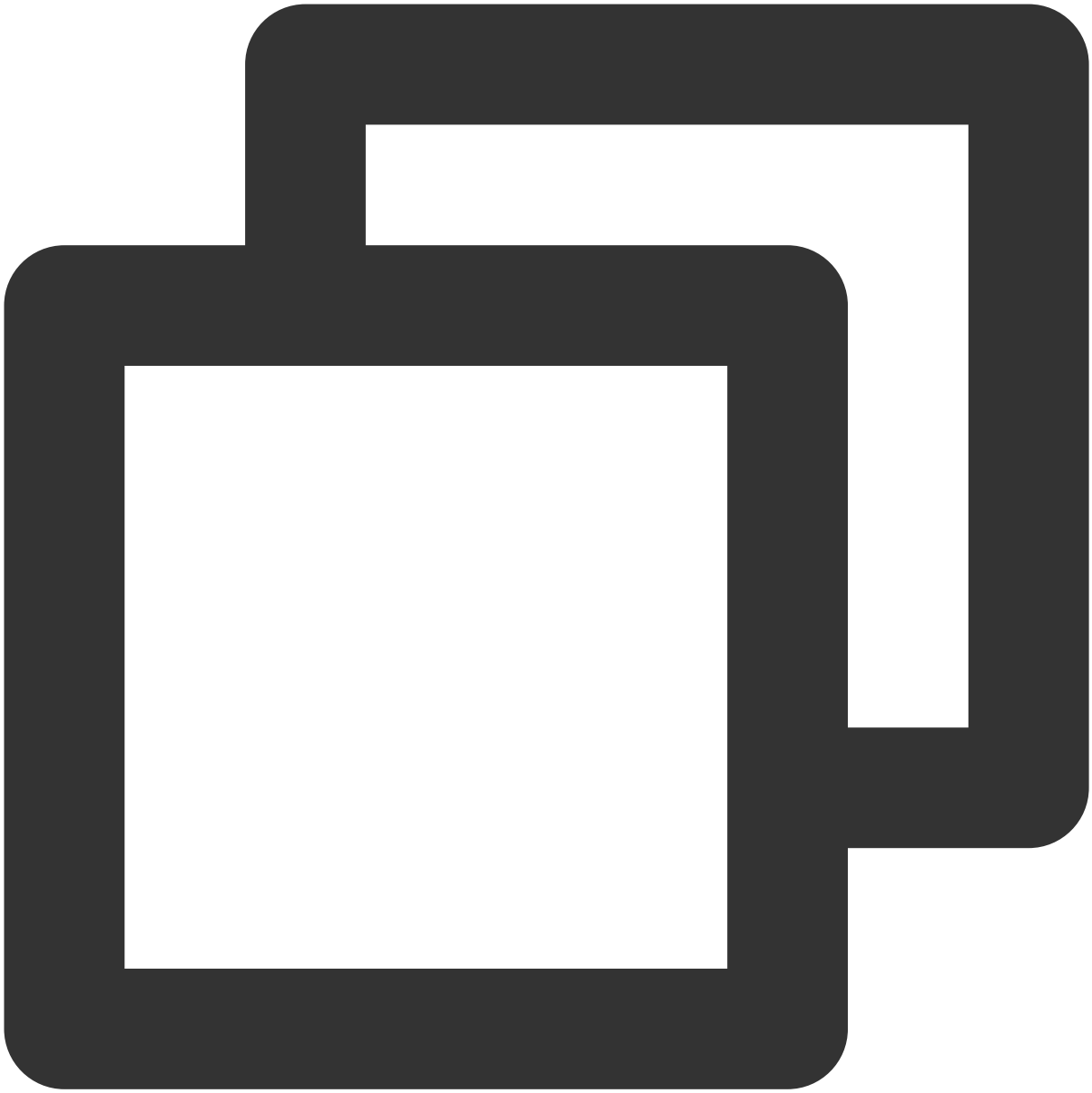
登录 容器服务控制台，并为 Serverless 集群开启日志采集功能。操作详情请参见 开启日志采集。

创建 CRD

您只需要定义 LogConfig CRD 即可创建采集配置，采集组件根据 LogConfig CRD 的变化修改相应的日志服务 CLS 日志主题，并设置绑定的机器组。CRD 的格式如下:

clsDetail 字段说明

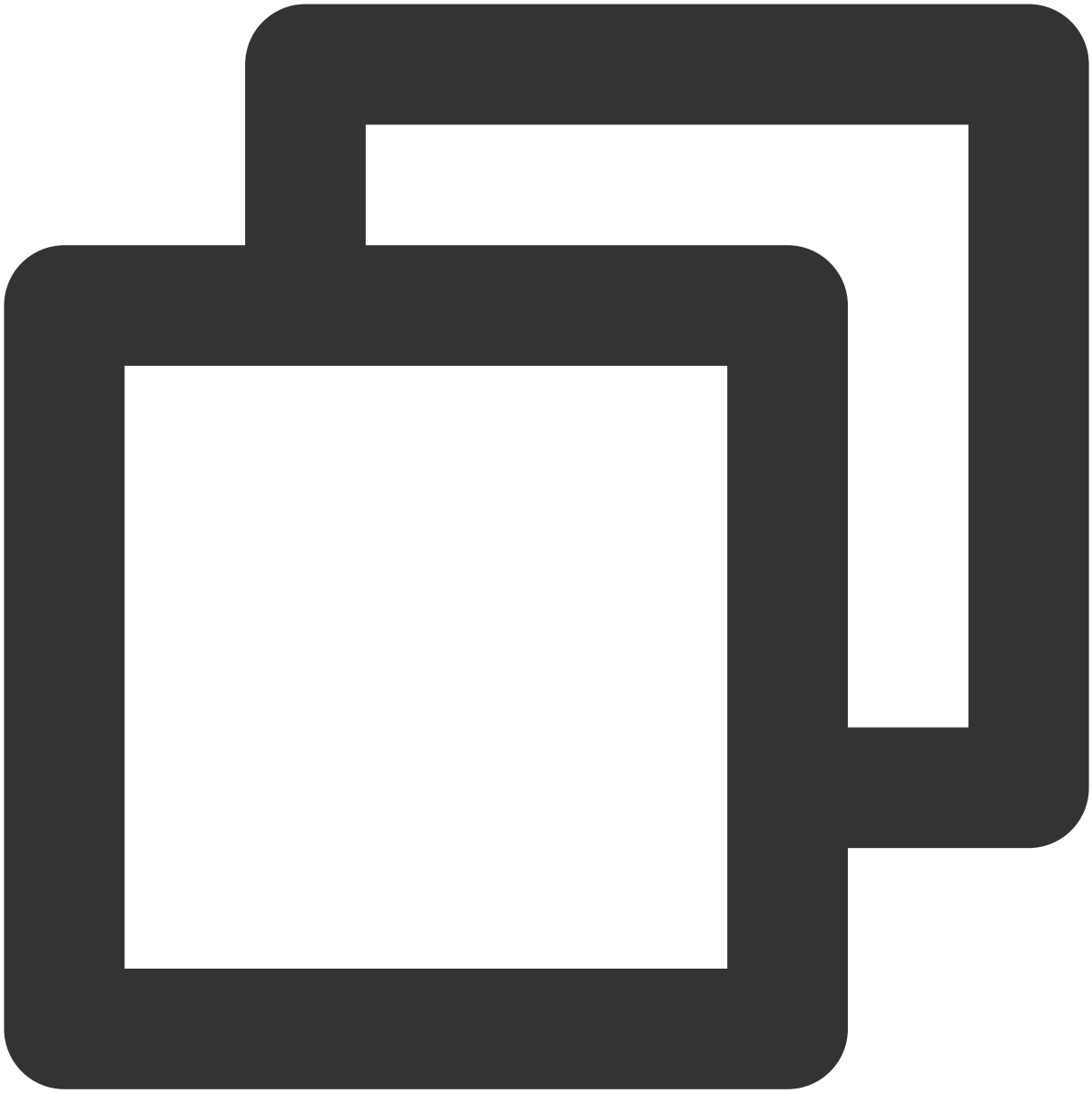
注意：
topic 指定后不允许修改。
如果选择采集类型为“容器文件路径”时，对应的“容器文件路径”不能为软链接，否则会导致软链接的实际路径在采集器的容器内不存在，采集日志失败。



```
clsDetail:
  ## 自动创建日志主题，需要同时指定日志集和主题的name
  logsetName: test                        ## CLS日志集的name，若无该name的日志集，会自动创建，
  topicName: test                         ## CLS日志主题的name，若无该name的日志主题，会自动创

  # 选择已有日志集日志主题，  如果指定了日志集未指定日志主题，则会自动创建一个日志主题
  logsetId: xxxxxx-xx-xx-xx-xxxxxxxx   ## CLS日志集的ID，日志集需要在CLS中提前创建
  topicId: xxxxxx-xx-xx-xx-xxxxxxxx    ## CLS日志主题的ID，日志主题需要在CLS中提前创建，且注

  logType: json_log  ## 日志采集格式，json_log代表 json 格式，delimiter_log代表分隔符格式，
  logFormat: xxx                          ## 日志格式化方式
```

```
period: 30                              ## 生命周期，单位天，可取值范围1~3600。取值为3640时
partitionCount:                         ## Integer 类型，日志主题分区个数。默认创建1个，最大
tags:                                  ## 标签描述列表，通过指定该参数可以同时绑定标签到相应的
 - key: xxx                             ## 标签key
 value: xxx                            ## 标签value
autoSplit: false                        ## boolean 类型，是否开启自动分裂，默认值为true
maxSplitPartitions:
storageType: hot                       ## 日志主题的存储类型，可选值 hot（标准存储），cold（低
excludePaths:                           ## 采集黑名单路径列表
  - type: File                            ## 类型，选填File或Path
      value: /xx/xx/xx/xx.log                ## type 对应的值
indexs:                                ## 创建 topic 时可自定义索引方式和字段
  - indexName:     ## 需要配置键值或者元字段索引的字段，元字段Key无需额外添加__TAG__.前缀，
      indexType:     ## 字段类型，目前支持的类型有：long、text、double
      tokenizer:     ## 字段的分词符，其中的每个字符代表一个分词符；仅支持英文符号及\\n\\t\\
      sqlFlag:     ## boolean 字段是否开启分析功能
      containZH:   ## boolean 是否包含中文
region: ap-xxx                          ## topic 所在地域，用于跨地域投递
userDefineRule: xxxxxx                  ## 用户自定义采集规则，Json格式序列化的字符串
extractRule: {}                         ## 提取、过滤规则。 如果设置了ExtractRule，则必须设
```

inputDetail 字段说明



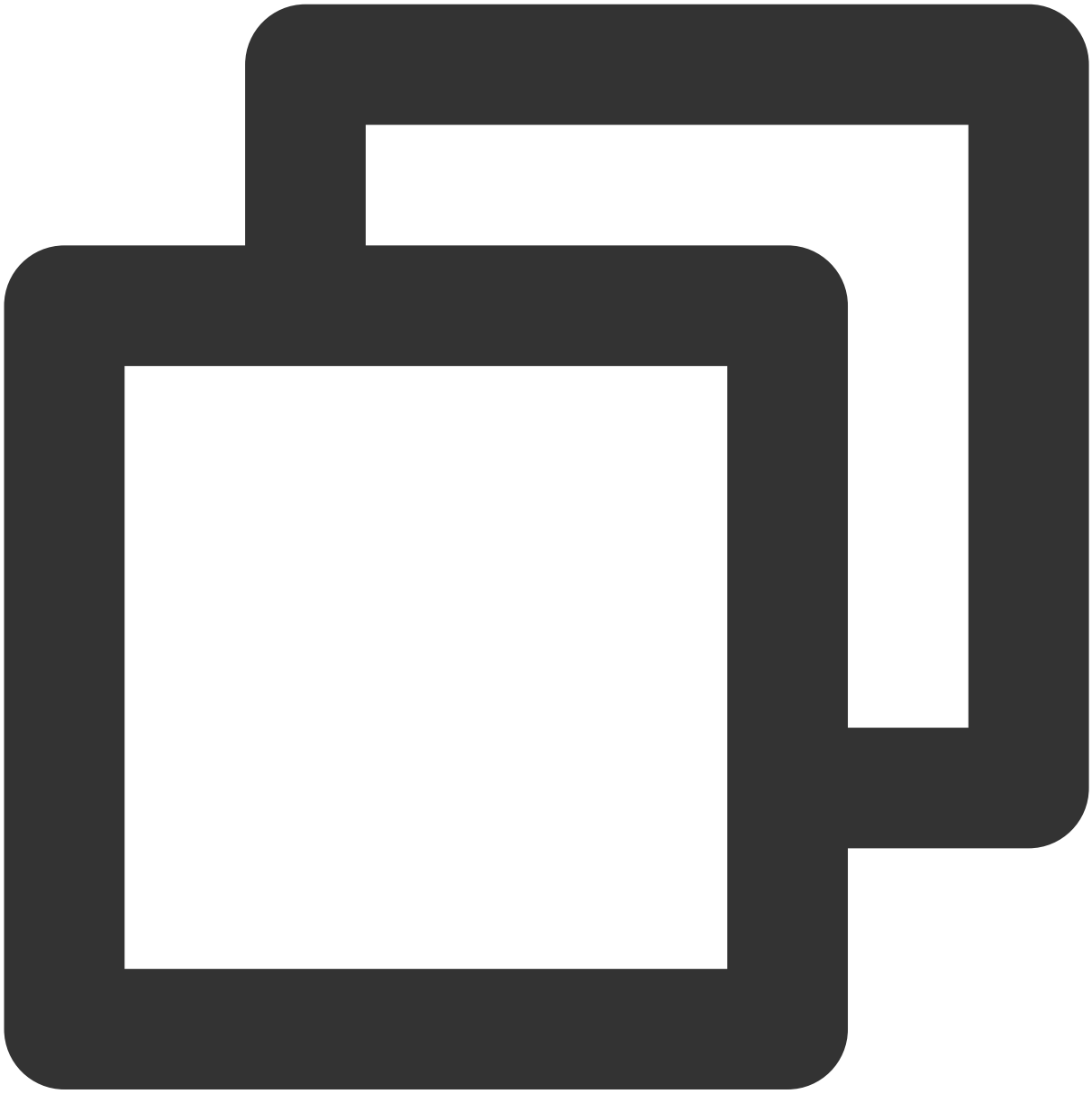
```
inputDetail:
  type: container_stdout  ## 采集日志的类型，包括container_stdout（容器标准输出）、contai

  containerStdout:        ## 容器标准输出
    namespace: default    ## 采集容器的kubernetes命名空间。支持多个命名空间，如果有多个命名空
    excludeNamespace: nm1,nm2  ## 排除采集容器的kubernetes命名空间。支持多个命名空间，如果有
    nsLabelSelector: environment in (production),tier in (frontend) ## 根据命名空间la
    allContainers: false       ## 是否采集指定命名空间中的所有容器的标准输出。注意:allConta
    container: xxx             ## 采集日志的容器名，为空时，代表采集所有符合容器的日志名。 注
    excludeLabels:  ## 采集不包含包含指定label的Pod，与workload，namespace 和 excludeNam
      key2: value2  ## 支持匹配同一个key下多个value值的pod，例填写enviroment = productio
```

```
    includeLabels:  ## 采集包含指定label的Pod，与workload，namespace 和 excludeNamespac
      key: value1  ## 收集规则收集的日志会带上metadata，并上报到消费端。支持匹配同一个key下多

    metadataLabels:           ## 指定具体哪些pod label被当做元数据采集，如果不指定，则采集所
    - label1
    customLabels:             ## 用户自定义metadata
      label: l1

    workloads:
    - container: xxx  ## 要采集的容器名，如果不指定，代表workload Pod中的所有容器
      kind: deployment  ## workload类型，支持deployment、daemonset、statefulset、job、
      name: sample-app  ## workload的名字
      namespace: prod  ## workload的命名空间

  containerFile:  ## 容器内文件
    namespace: default      ## 采集容器的kubernetes命名空间，必须指定一个命名空间
    excludeNamespace: nm1,nm2   ## 排除采集容器的kubernetes命名空间。支持多个命名空间，如果
    nsLabelSelector: environment in (production),tier in (frontend) ## 根据命名空间la
    container: xxx          ## 采集日志的容器名，为 * 时，代表采集所有符合容器的日志名
    logPath: /var/logs      ## 日志文件夹，不支持通配符
    filePattern: app_*.log  ## 日志文件名，支持通配符 * 和 ? ，* 表示匹配多个任意字符，? 表
    customLabels:  ## 用户自定义metadata
      key: value
    excludeLabels:  ## 采集不包含包含指定label的Pod，与workload不能同时指定
      key2: value2  ## 支持匹配同一个key下多个value值的pod，例填写enviroment = productio

    includeLabels:  ## 采集包含指定label的Pod，与workload不能同时指定
      key: value1  ## 收集规则收集的日志会带上metadata，并上报到消费端。支持匹配同一个key下多
    metadataLabels:      ## 指定具体哪些pod label被当做元数据采集，如果不指定，则采集所有poc
    - label1             ## pod label
    workload:
      container: xxx    ## 要采集的容器名，如果不指定，代表workload Pod中的所有容器
      name: sample-app  ## workload的名字
```

日志解析格式

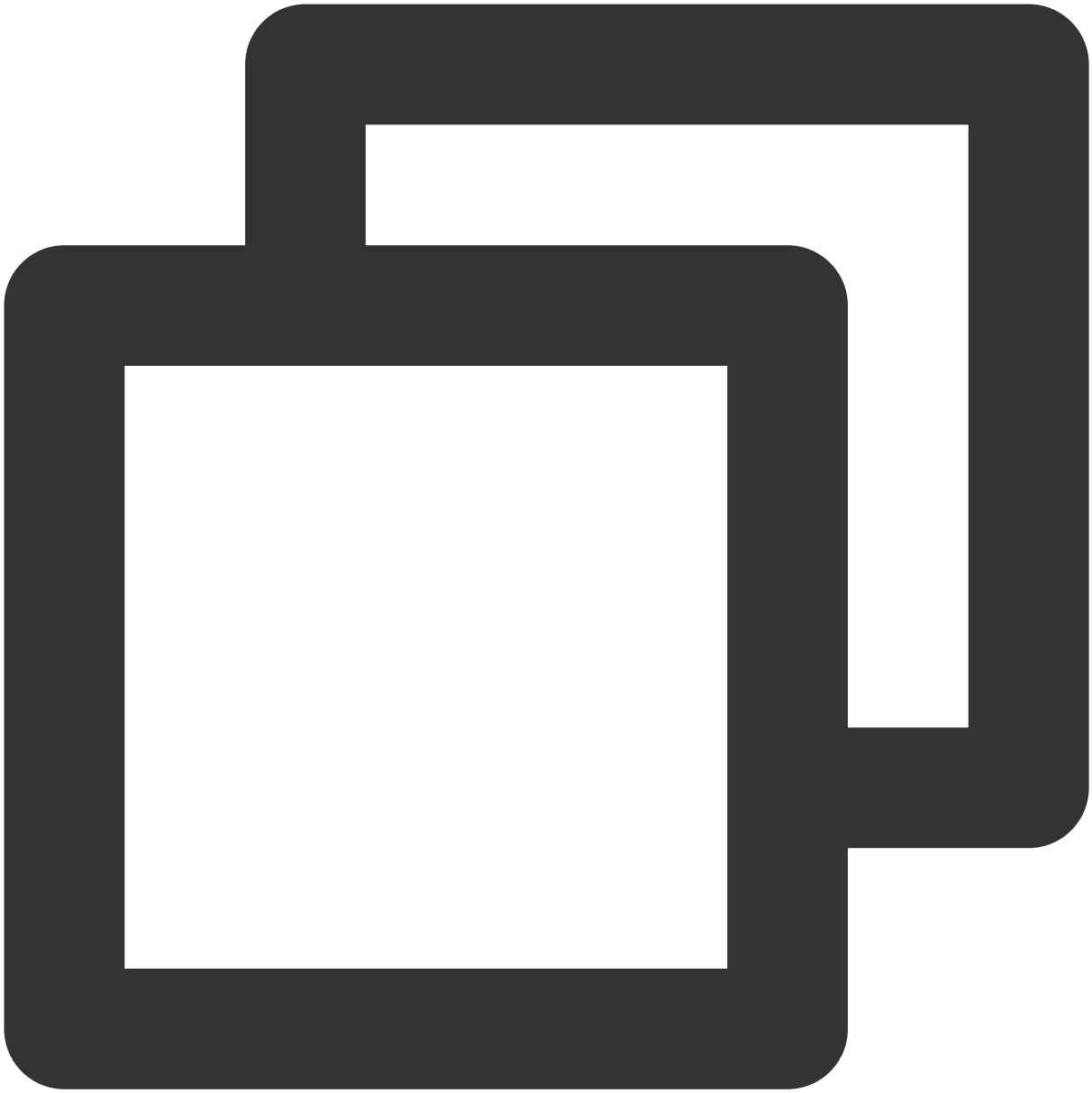
单行全文格式
多行全文格式
单行-完全正则格式
多行-完全正则格式
JSON 格式
分隔符格式

单行全文日志是指一行日志内容为一条完整的日志。日志服务在采集的时候，将使用换行符 `\\n` 来作为一条日志的结束符。为了统一结构化管理，每条日志都会存在一个默认的键值 `__CONTENT__` ，但日志数据本身不再进行日志结构化处理，也不会提取日志字段，日志属性的时间项由日志采集的时间决定。详情请参见 单行文本格式。
假设一条日志原始数据为：



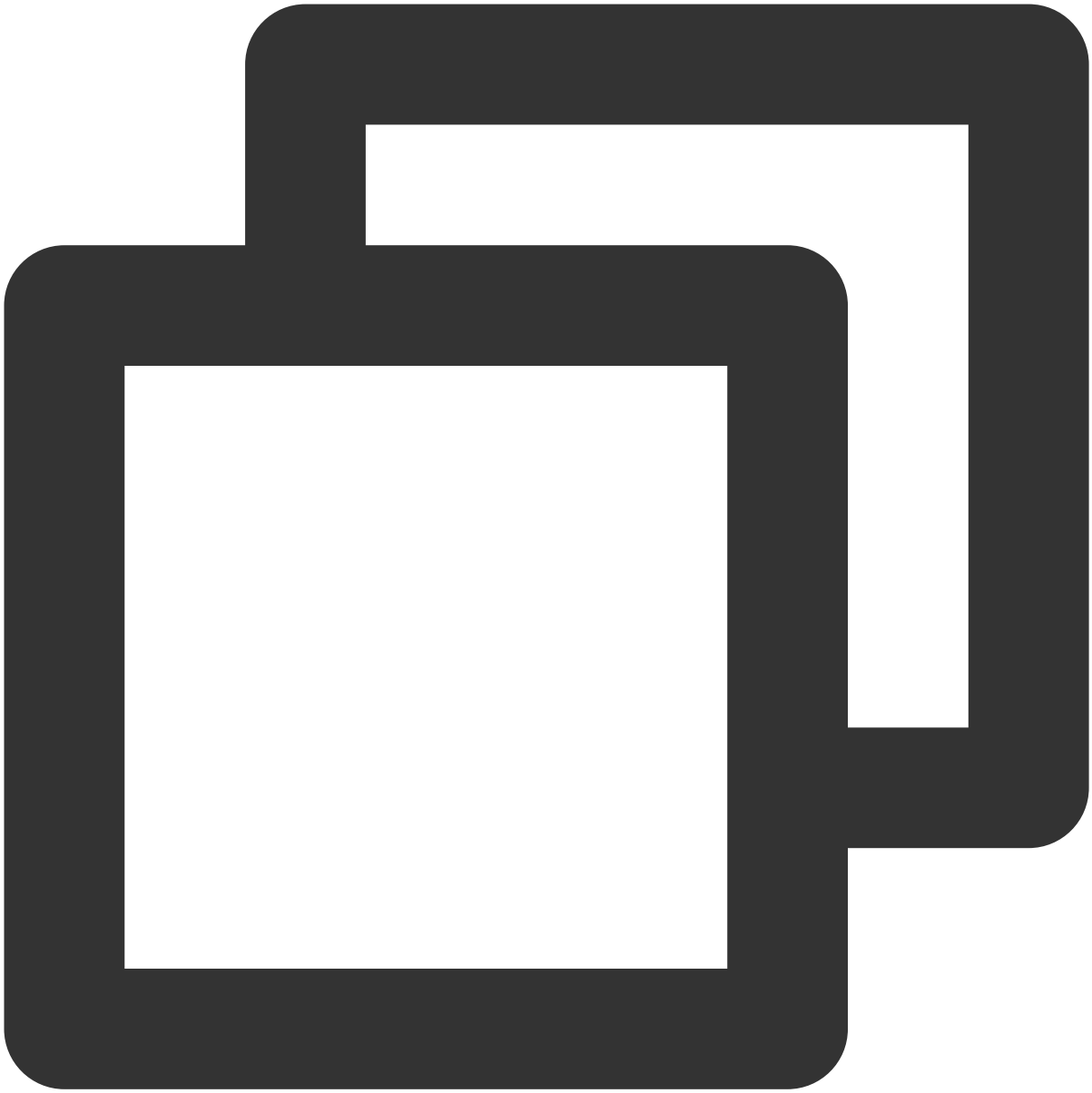
```
Tue Jan 22 12:08:15 CST 2019 Installed: libjpeg-turbo-static-1.2.90-6.el7.x86_64
```

LogConfig 配置参考示例如下：



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   clsDetail:
     topicId: xxxxxx-xx-xx-xx-xxxxxxxx
     # 单行日志
     logType: minimalist_log
```

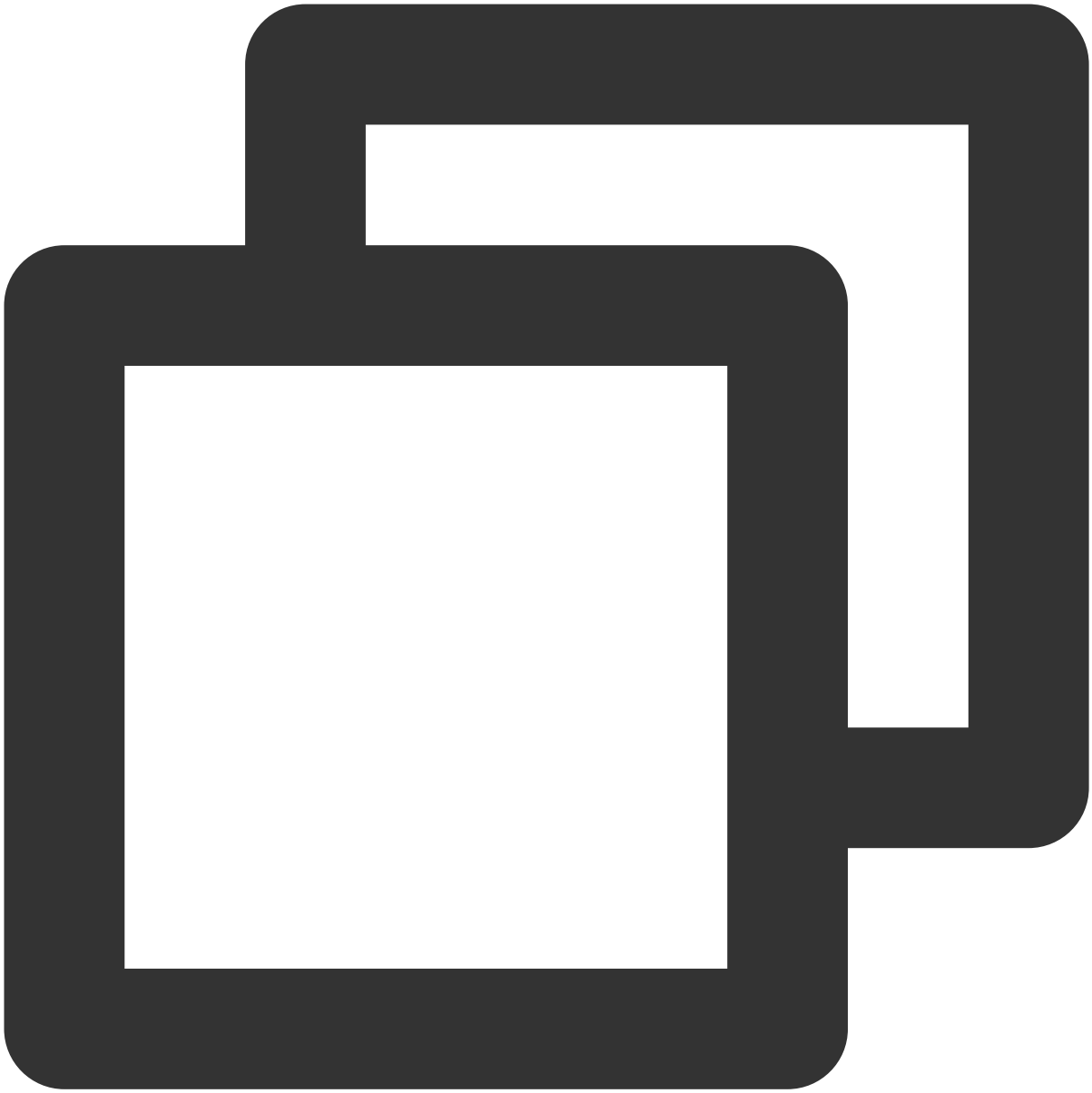
采集到日志服务的数据为：



```
__CONTENT__:Tue Jan 22 12:08:15 CST 2019 Installed: libjpeg-turbo-static-1.2.90-6.e
```

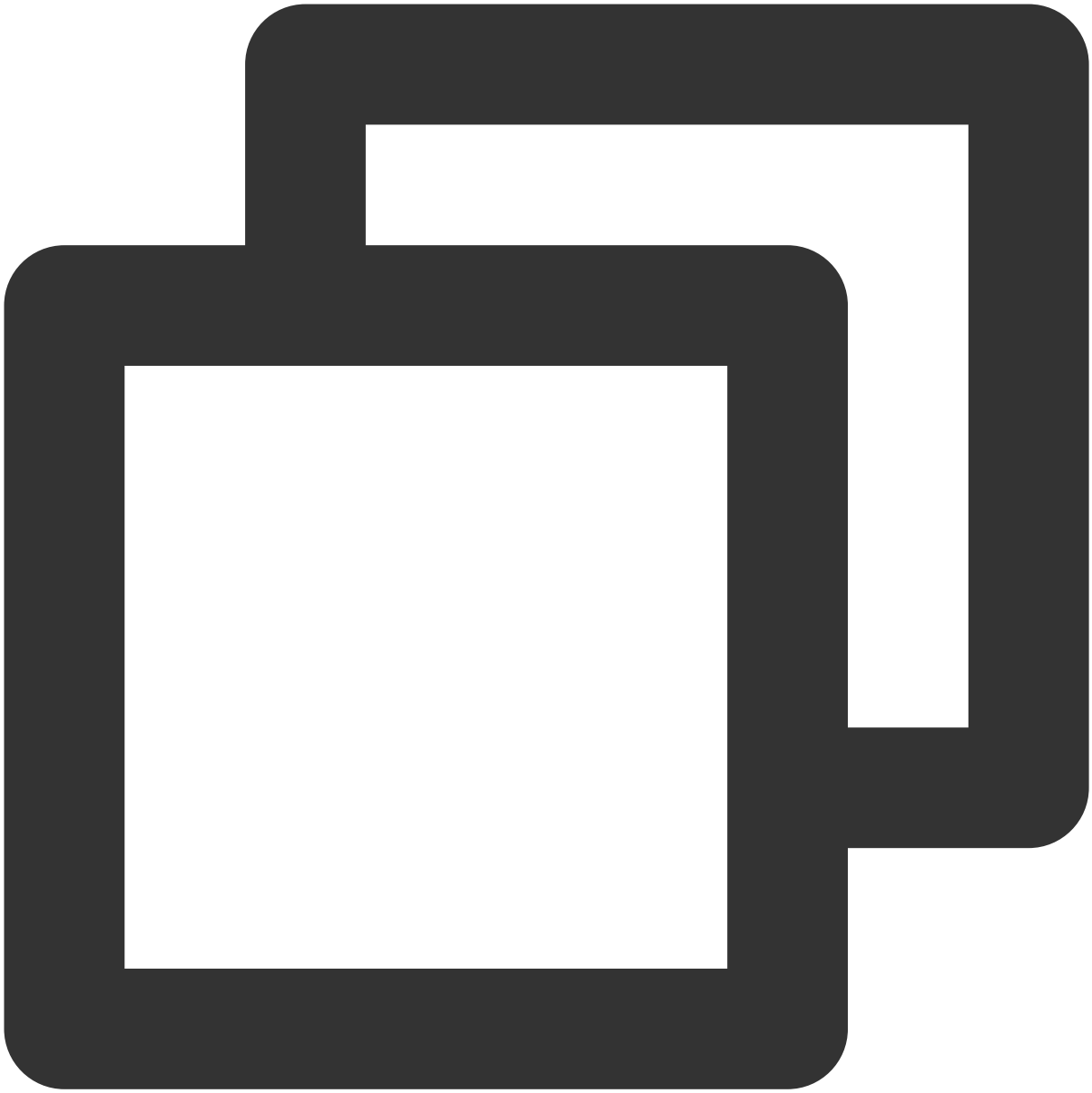
多行全文日志是指一条完整的日志数据可能跨占多行（例如 Java stacktrace）。该情况下无法使用换行符 `\\n` 作为日志的结束标识符，为了使日志系统明确区分每条日志，采用首行正则的方式进行匹配，当某行日志匹配预先设置的正则表达式，即为一条日志的开头，而下一行首出现则作为该条日志的结束标识符。多行全文也会设置一个默认的键值 `__CONTENT__` ，但日志数据本身不再进行日志结构化处理，也不会提取日志字段，日志属性的时间项由日志采集的时间决定。详情请参见 多行文本格式。

假设一条多行日志原始数据为：



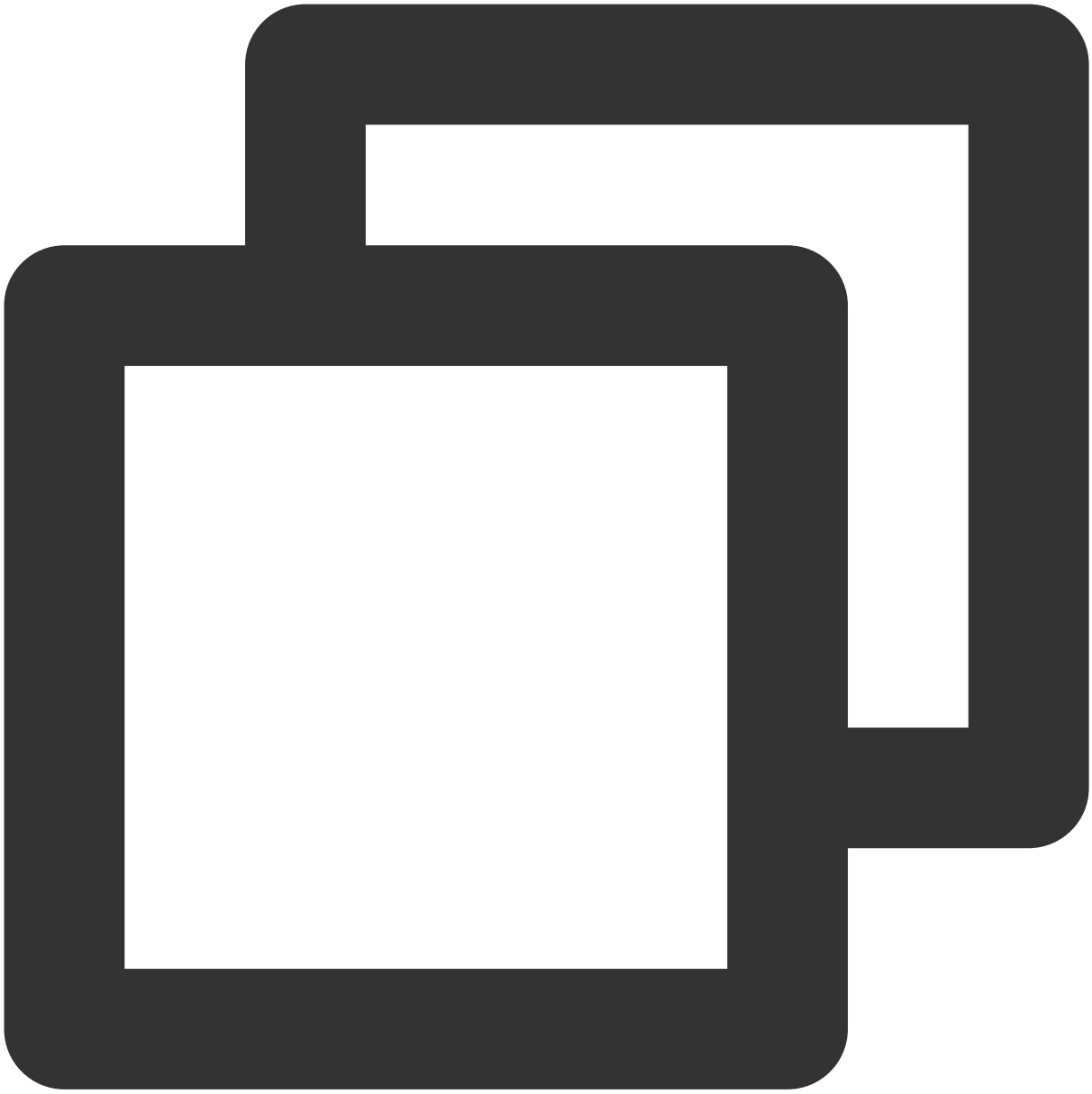
```
2019-12-15 17:13:06,043 [main] ERROR com.test.logging.FooFactory:
java.lang.NullPointerException
     at com.test.logging.FooFactory.createFoo(FooFactory.java:15)
     at com.test.logging.FooFactoryTest.test(FooFactoryTest.java:11)
```

LogConfig 配置的参考如下：



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   clsDetail:
     topicId: xxxxxx-xx-xx-xx-xxxxxxxx
     #多行日志
     logType: multiline_log
     extractRule:
       #只有以日期时间开头的行才被认为是新一条日志的开头，否则就添加换行符\\n并追加到当前日志的尾
       beginningRegex: \\d{4}-\\d{2}-\\d{2}\\s\\d{2}:\\d{2}:\\d{2},\\d{3}\\s.+
```

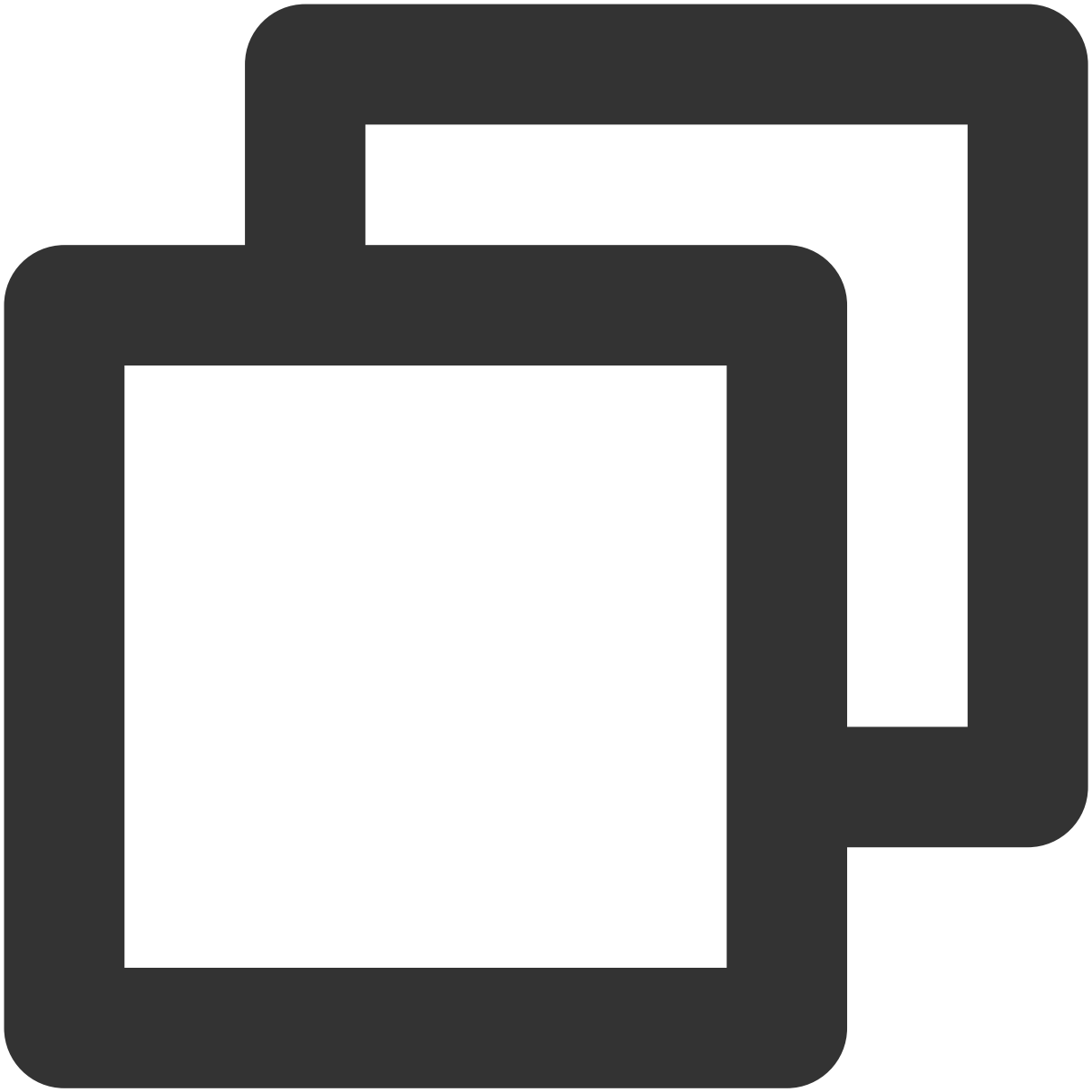
采集到日志服务的数据为：



```
\\_\\_CONTENT__:2019-12-15 17:13:06,043 [main] ERROR com.test.logging.FooFactory:\\
```

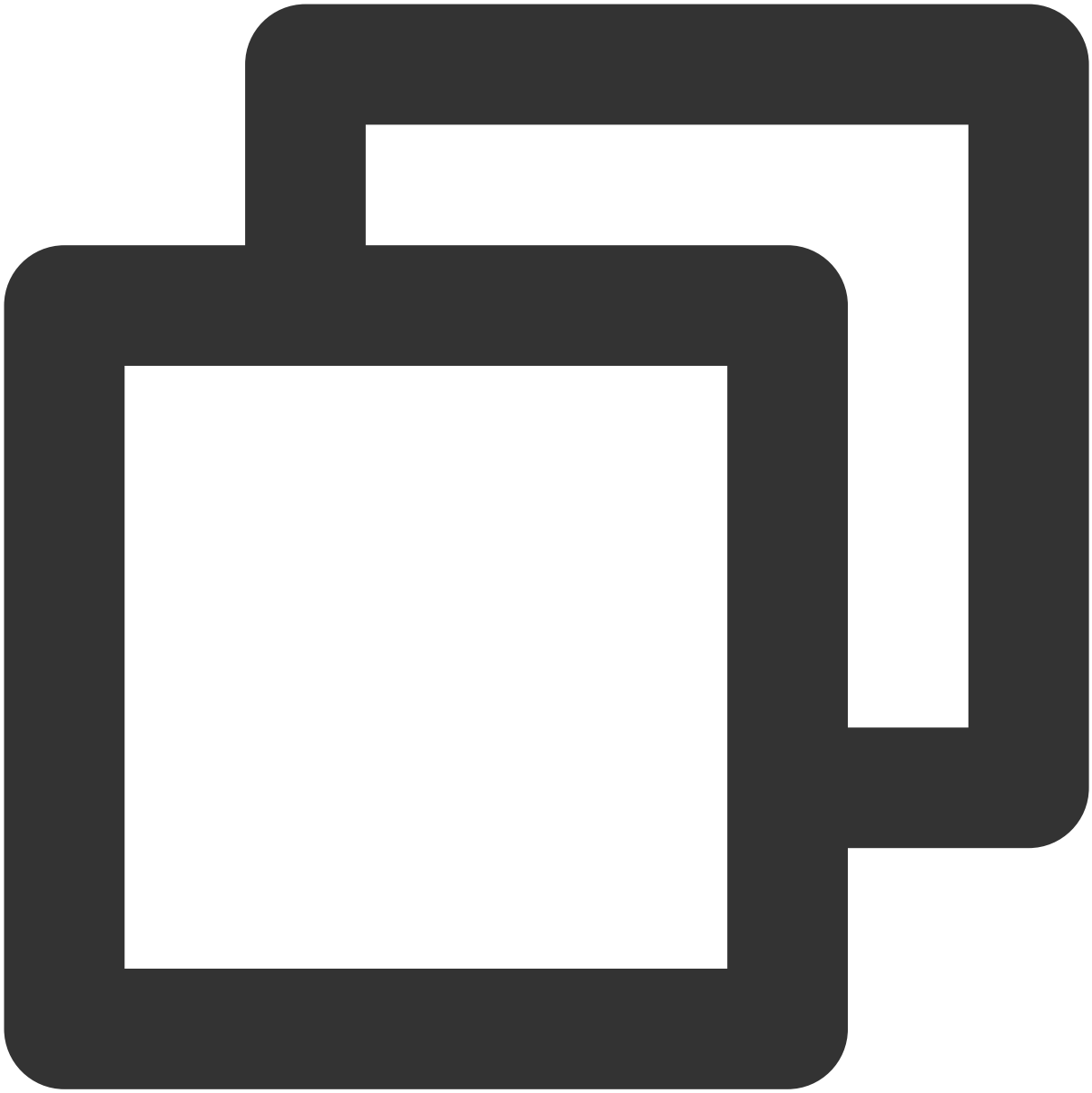
完全正则格式通常用来处理结构化的日志，指将一条完整日志按正则方式提取多个 key-value 的日志解析模式。详情请参见 完全正则格式。

假设一条日志原始数据为：



```
10.135.46.111 - - [22/Jan/2019:19:19:30 +0800] "GET /my/course/1 HTTP/1.1" 127.0.0.
```

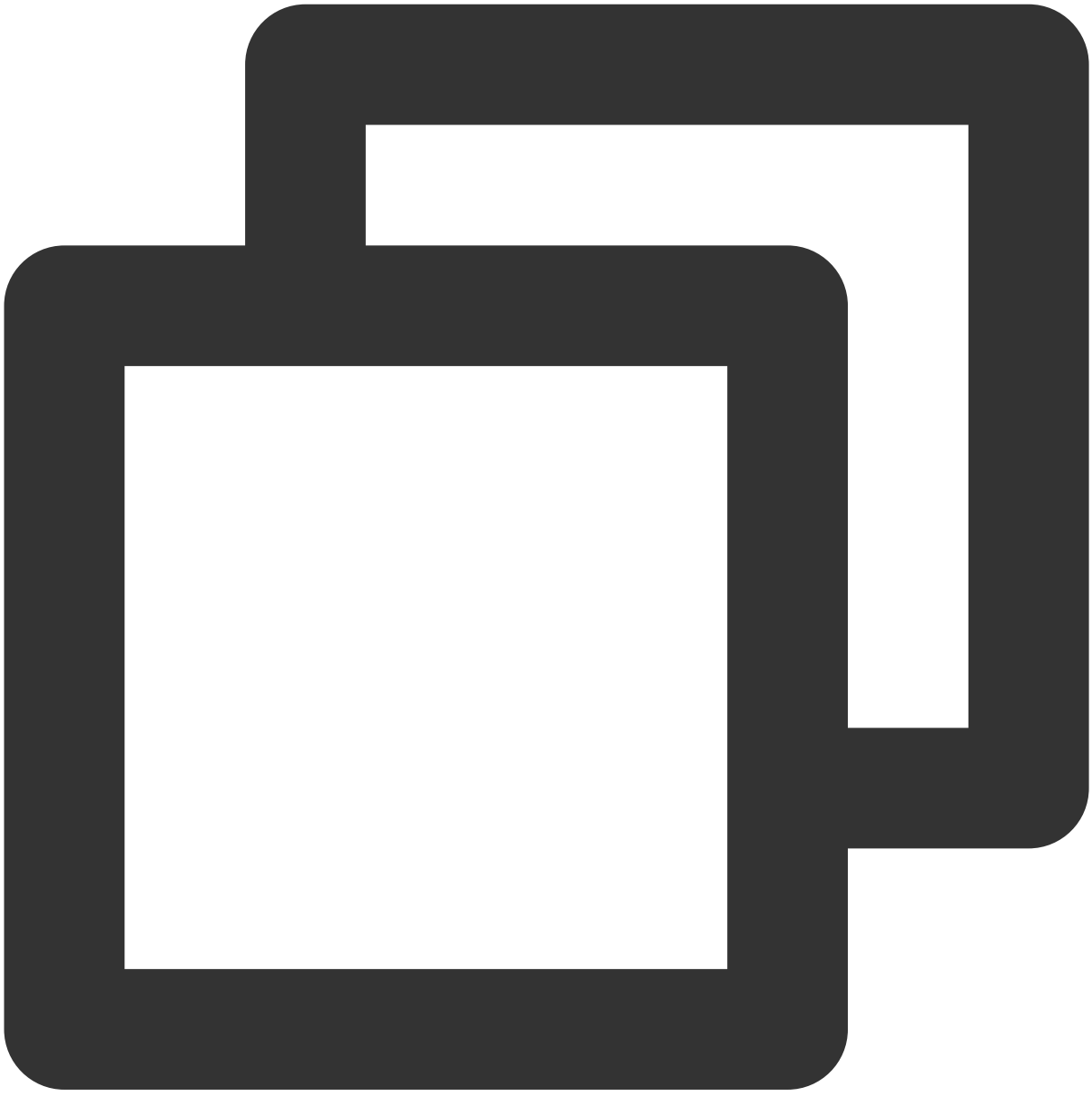
LogConfig 配置的参考如下：



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   clsDetail:
     topicId: xxxxxx-xx-xx-xx-xxxxxxxx
     # 完全正则格式
     logType: fullregex_log
     extractRule:
       # 正则表达式，会根据()捕获组提取对应的value
       logRegex: (\\S+)[^\\[]+(\\[[^:]+:\\d+:\\d+:\\d+\\s\\S+)\\s"(\\w+)\\s(\\S+)\\
       beginningRegex: (\\S+)[^\\[]+(\\[[^:]+:\\d+:\\d+:\\d+\\s\\S+)\\s"(\\w+)\\s(\
```

```
        # 提取的key列表，与提取的value的一一对应
        keys:   ['remote_addr','time_local','request_method','request_url','http_pro
```

采集到日志服务的数据为：

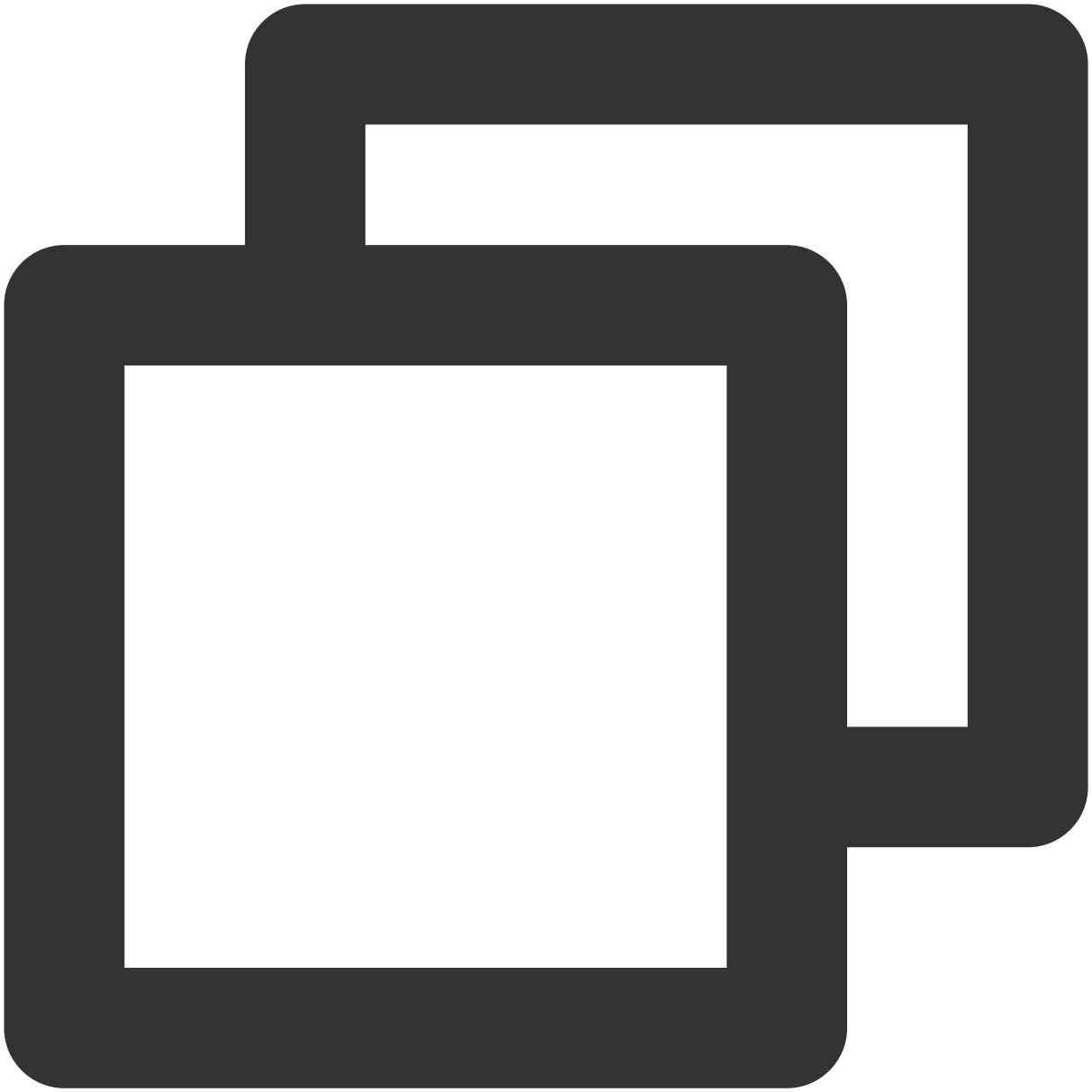


```
body_bytes_sent: 9703
http_host: 127.0.0.1
http_protocol: HTTP/1.1
http_referer: http://127.0.0.1/course/explore?filter%5Btype%5D=all&filter%5Bprice%5
http_user_agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:64.0) Gecko/20100101 Firef
remote_addr: 10.135.46.111
```

```
request_length: 782
request_method: GET
request_time: 0.354
request_url: /my/course/1
status: 200
time_local: [22/Jan/2019:19:19:30 +0800]
upstream_response_time: 0.354
```

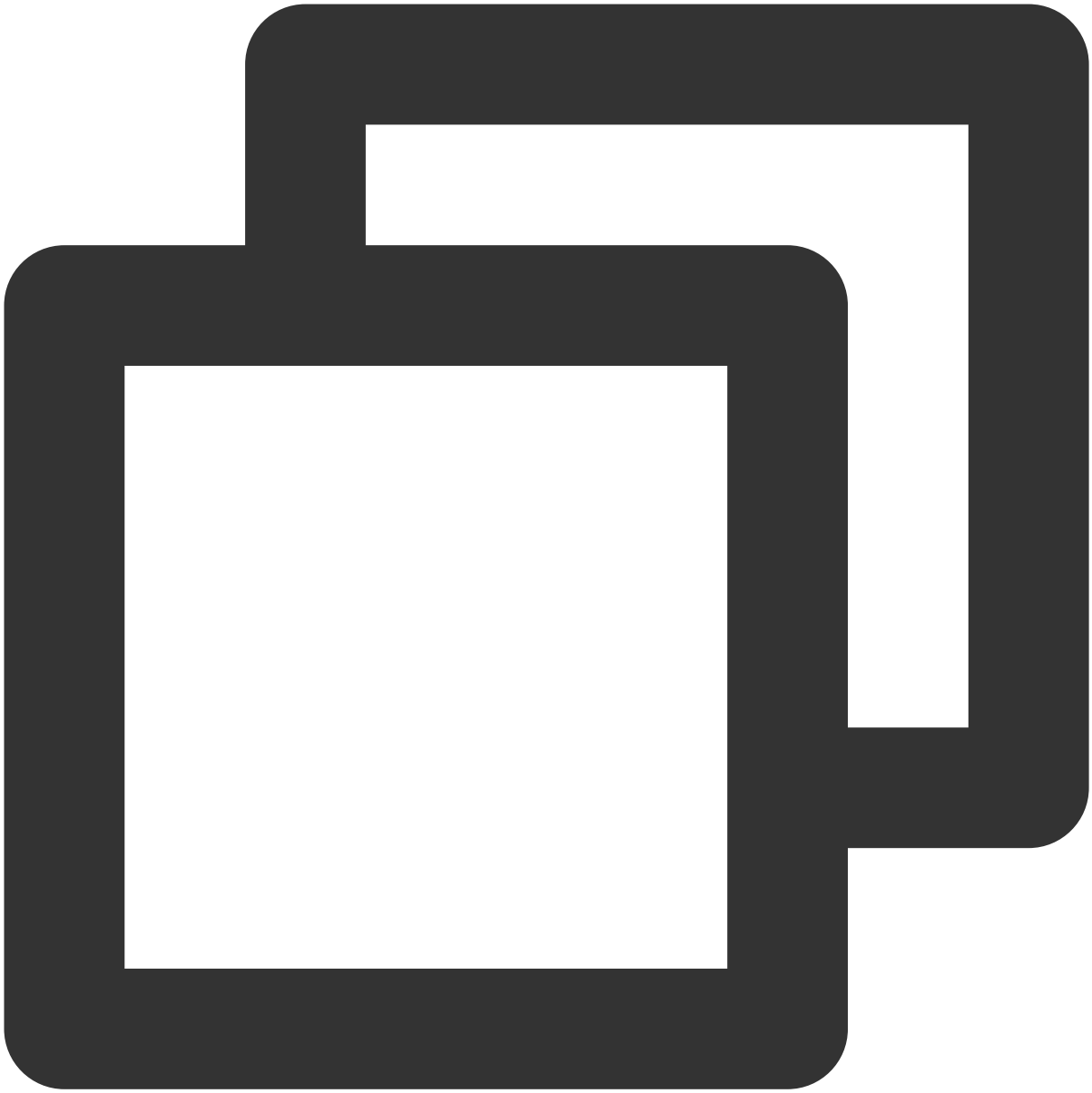
多行-完全正则模式适用于日志文本中一条完整的日志数据跨占多行（例如 Java 程序日志），可按正则表达式提取为多个 key-value 键值的日志解析模式。若不需要提取 key-value，请参阅多行全文格式进行配置。详情请参见 多行-完全正则格式。

假设一条日志原始数据为：



```
[2018-10-01T10:30:01,000] [INFO] java.lang.Exception: exception happened
   at TestPrintStackTrace.f(TestPrintStackTrace.java:3)
   at TestPrintStackTrace.g(TestPrintStackTrace.java:7)
   at TestPrintStackTrace.main(TestPrintStackTrace.java:16)
```

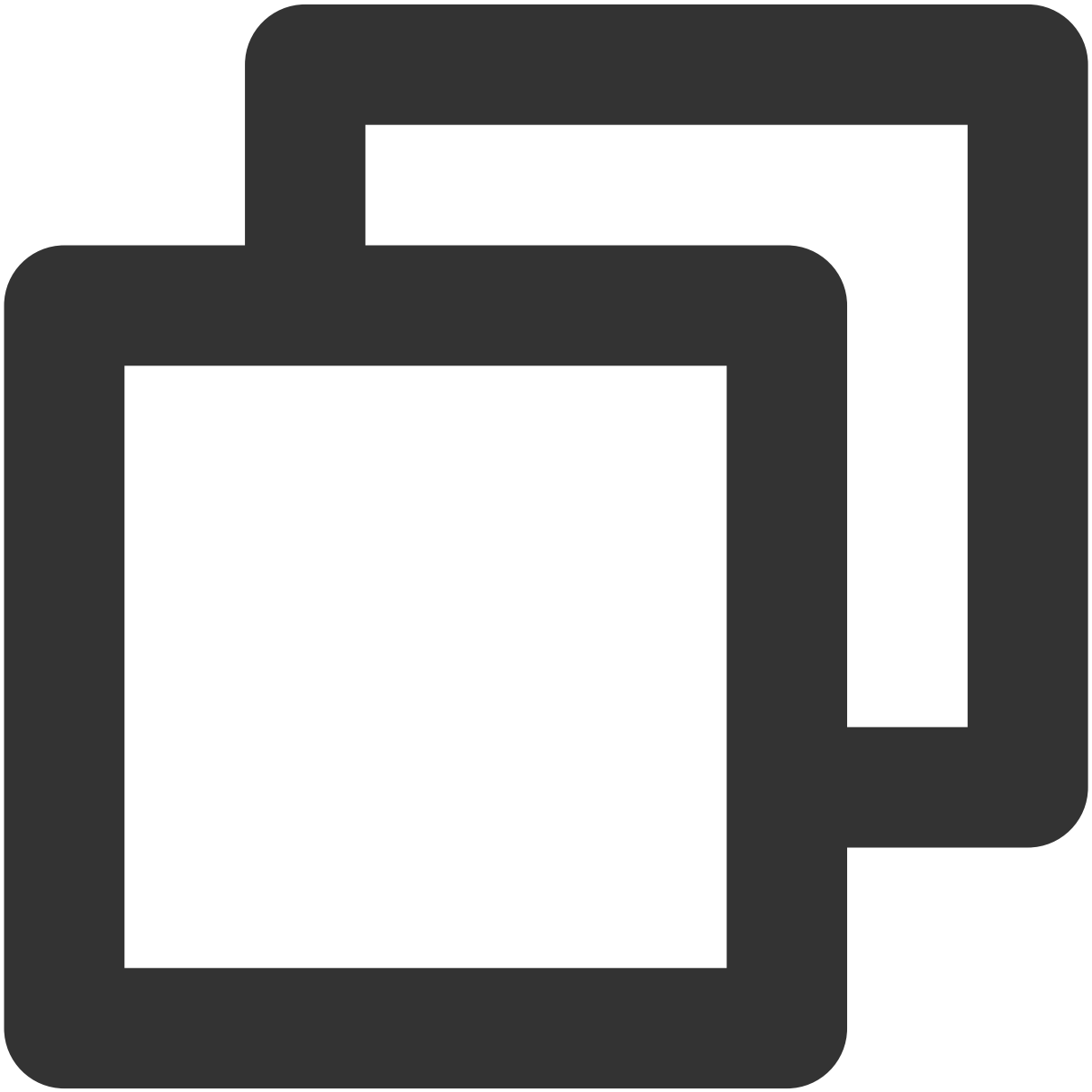
LogConfig 配置的参考如下：



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
  clsDetail:
    topicId: xxxxxx-xx-xx-xx-xxxxxxxx
        #多行-完全正则格式
        logType: multiline_fullregex_log
        extractRule:
      #行首完全正则表达式，只有以日期时间开头的行才被认为是新一条日志的开头，否则就添加换行符\\n并
           beginningRegex: \\[\\d+-\\d+-\\w+:\\d+:\\d+,\\d+\\]\\s\\[\\w+\\]\\s.*
      #正则表达式，会根据()捕获组提取对应的value
```

```
        logRegex: \\[(\\d+-\\d+-\\w+:\\d+:\\d+,\\d+)\\]\\s\\[(\\w+)\\]\\s(.*)
              # 提取的 key 列表，与提取的 value 的一一对应
              keys:
        - time
        - level
        - msg
```

根据提取的 key，采集到日志服务的数据为：

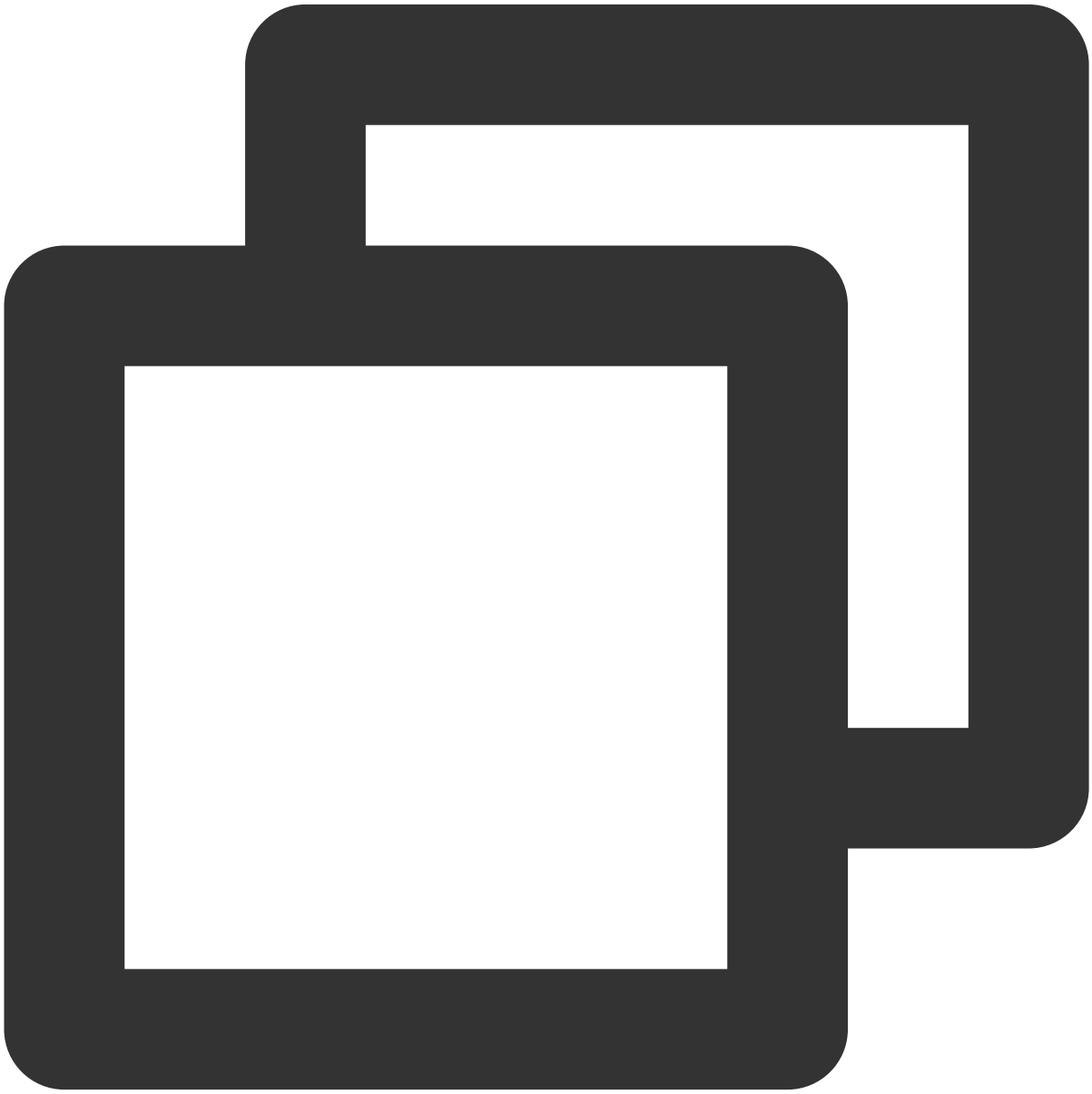


```
time： 2018-10-01T10:30:01,000`
level： INFO`
```

```
msg：java.lang.Exception: exception happened
    at TestPrintStackTrace.f(TestPrintStackTrace.java:3)
    at TestPrintStackTrace.g(TestPrintStackTrace.java:7)
    at TestPrintStackTrace.main(TestPrintStackTrace.java:16)
```

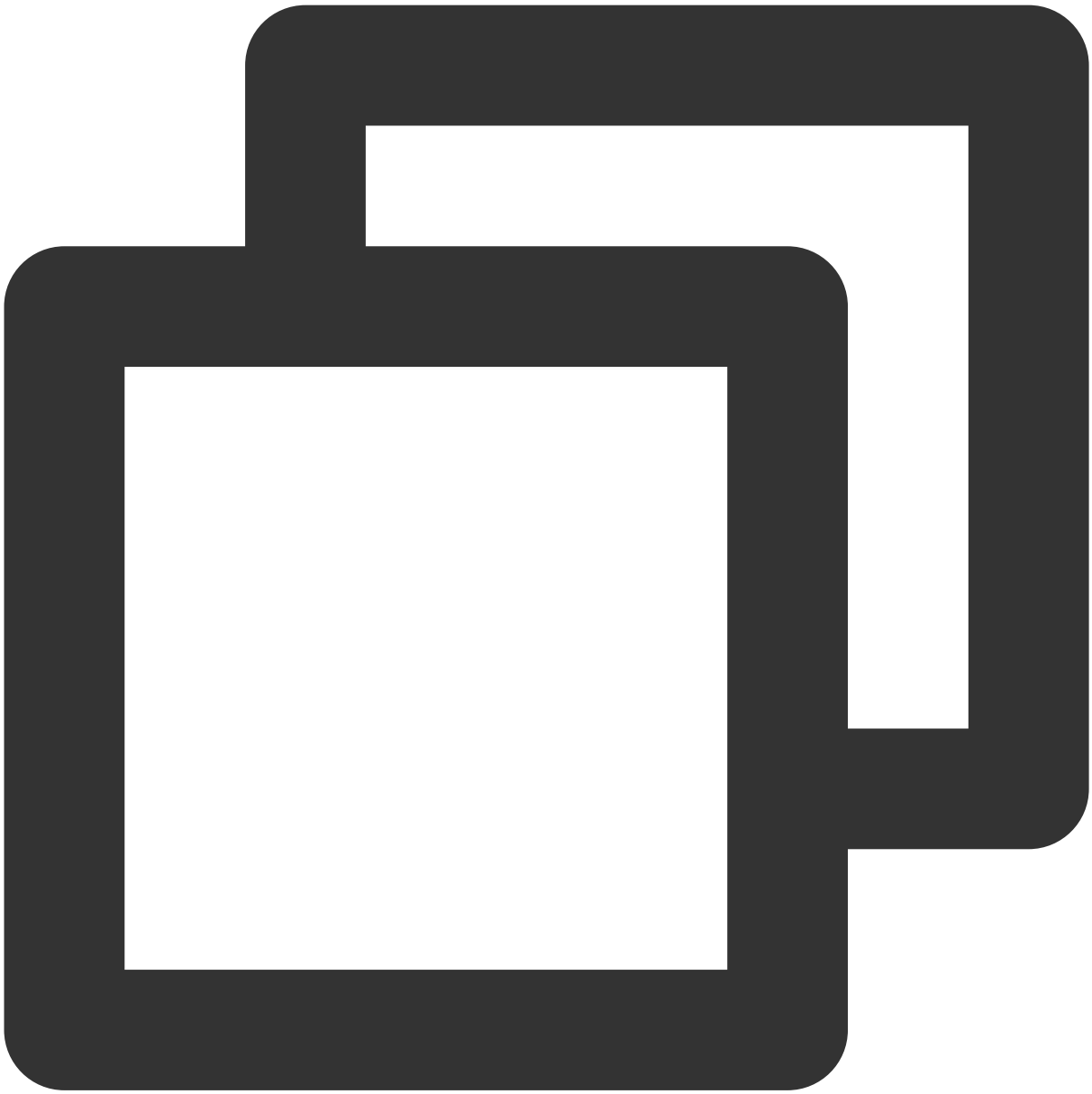
JSON 格式日志会自动提取首层的 key 作为对应字段名。首层的 value 作为对应的字段值，以该方式将整条日志进行结构化处理，每条完整的日志以换行符 `\\n` 为结束标识符。详情请参见 JSON 格式。

假设一条 JSON 日志原始数据为：



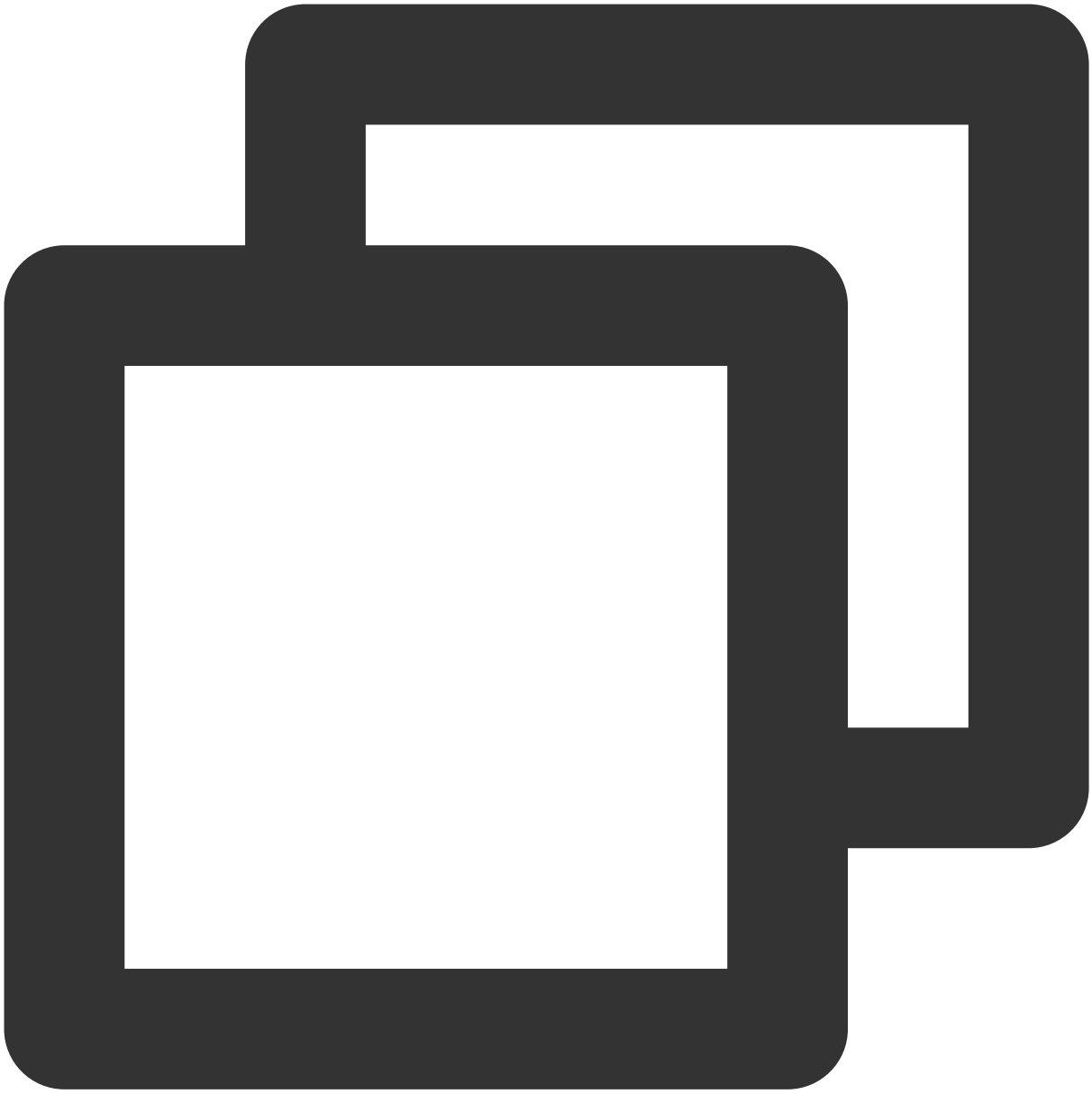
```
{"remote_ip":"10.135.46.111","time_local":"22/Jan/2019:19:19:34 +0800","body_sent":
```

LogConfig 配置的参考如下：



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   clsDetail:
     topicId: xxxxxx-xx-xx-xx-xxxxxxxx
     # JSON格式日志
     logType: json_log
```

采集到日志服务的数据为：

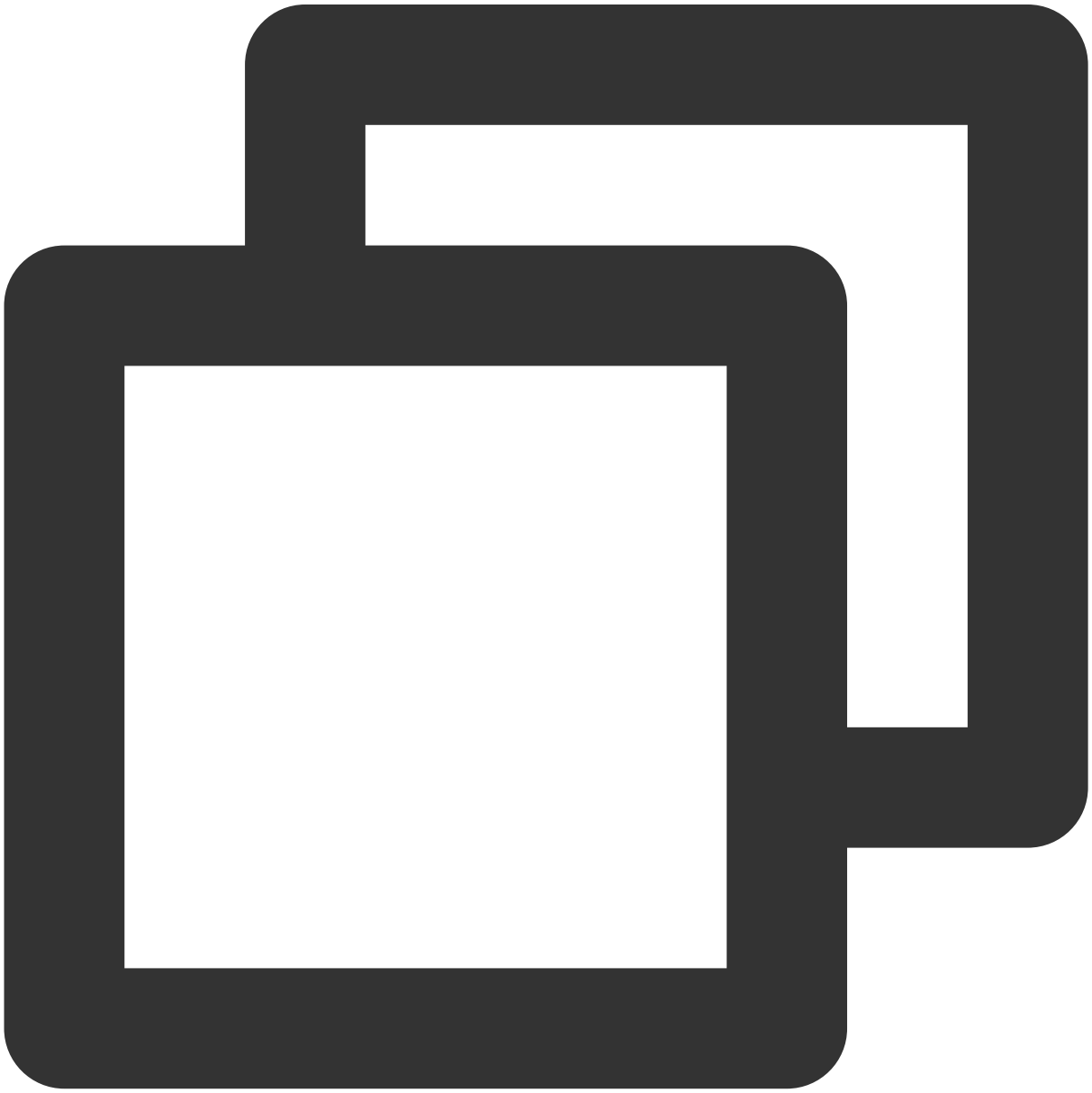


```
agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:64.0) Gecko/20100101 Firefox/64.0
body_sent: 23
http_host: 127.0.0.1
method: POST
referer: http://127.0.0.1/my/course/4
remote_ip: 10.135.46.111
request: POST /event/dispatch HTTP/1.1
response_code: 200
responsetime: 0.232
```

```
time_local: 22/Jan/2019:19:19:34 +0800
upstreamhost: unix:/tmp/php-cgi.sock
upstreamtime: 0.232
url: /event/dispatch
xff: -
```

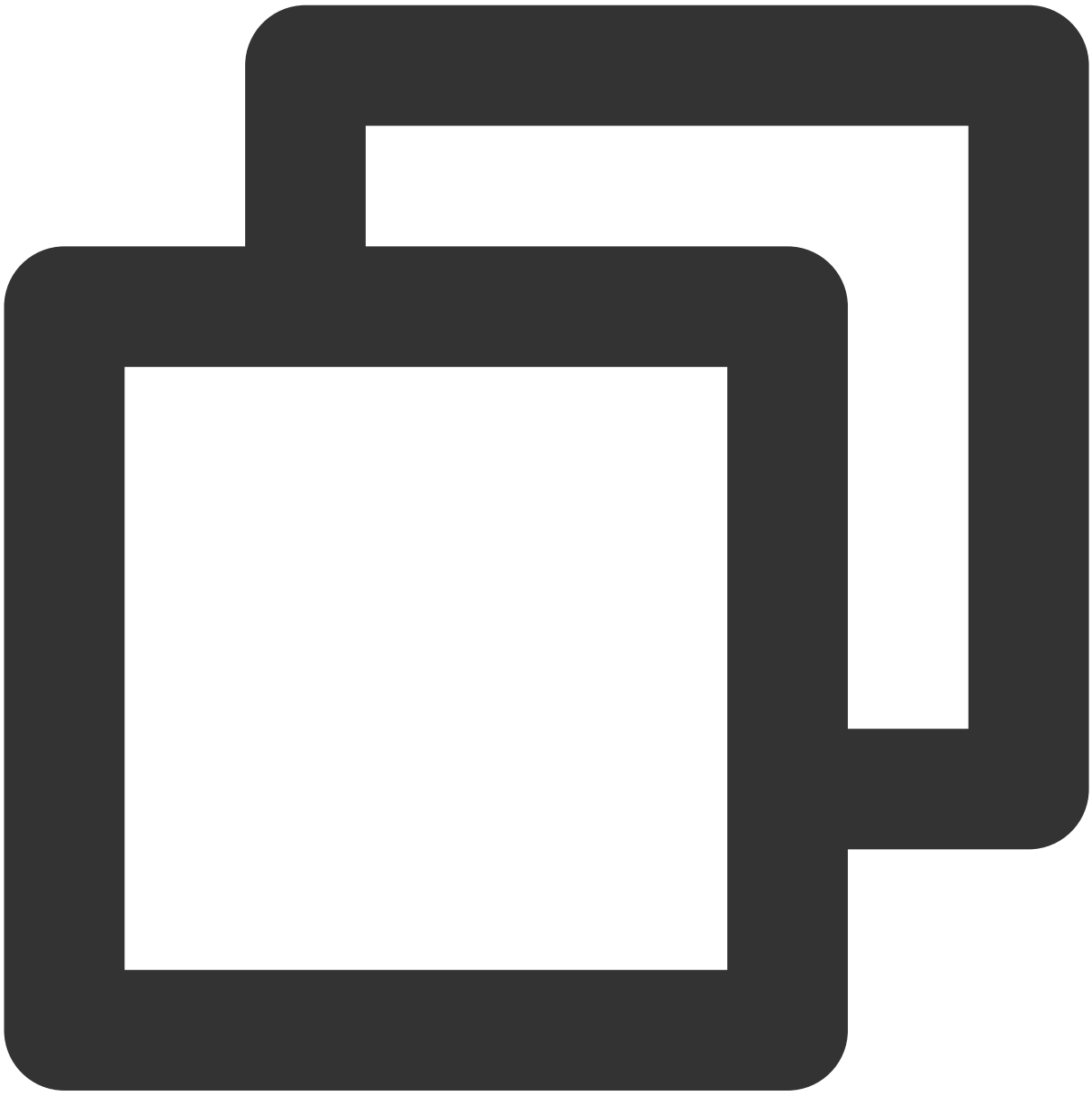
分隔符日志是指一条日志数据可以根据指定的分隔符将整条日志进行结构化处理，每条完整的日志以换行符 `\\n` 为结束标识符。日志服务在进行分隔符格式日志处理时，您需要为每个分开的字段定义唯一的 key。详情请参见 分隔符格式。

假设原始日志为：



```
10.20.20.10 ::: [Tue Jan 22 14:49:45 CST 2019 +0800] ::: GET /online/sample HTTP/1.
```

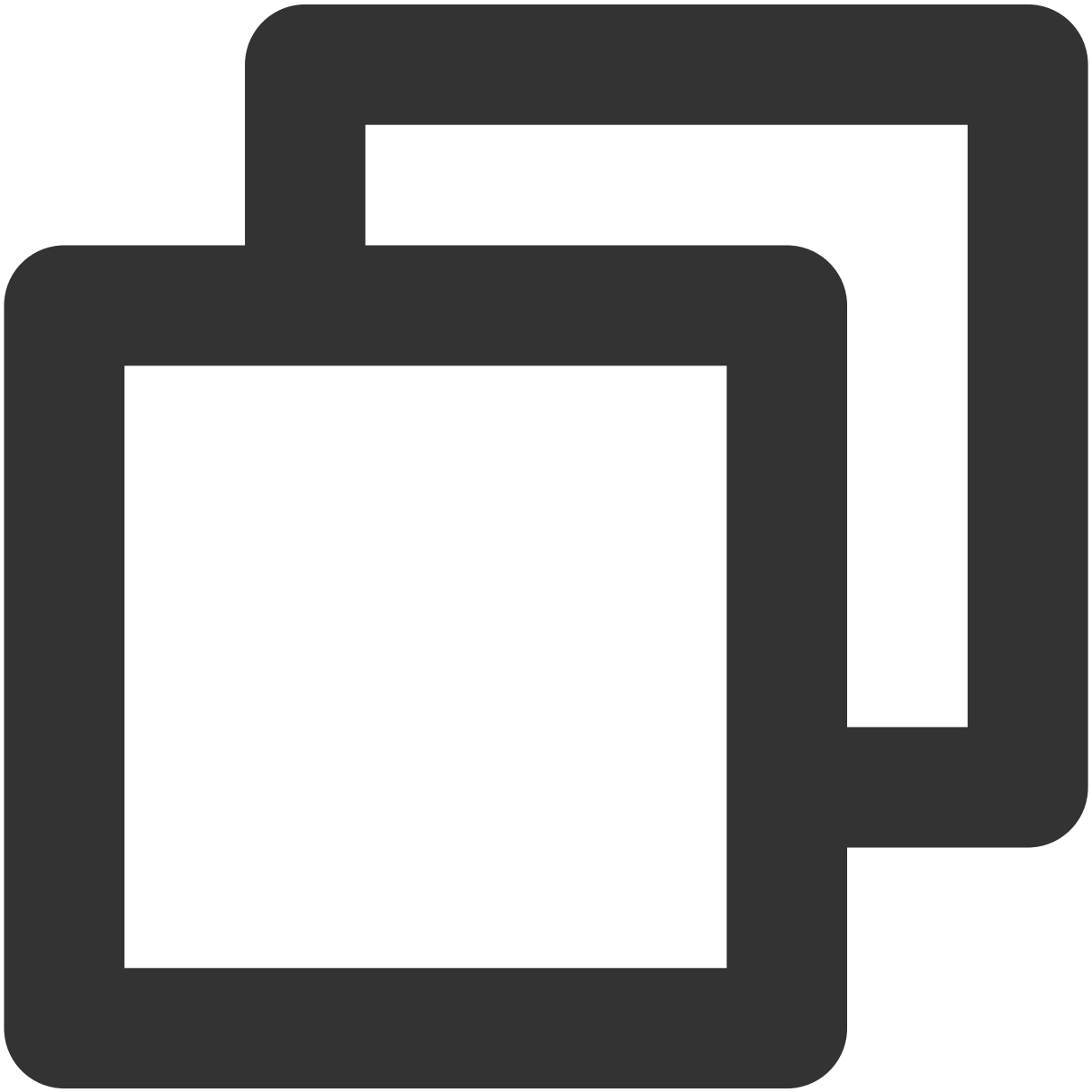
LogConfig 配置的参考如下：



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   clsDetail:
     topicId: xxxxxx-xx-xx-xx-xxxxxxxx
     #分隔符日志
```

```
      logType: delimiter_log
      extractRule:
        #分隔符
        delimiter: ':::'
        #提取的key列表，与被分割的字段一一对应
        keys: ['IP','time','request','host','status','length','bytes','referer']
```

采集到日志服务的数据为：



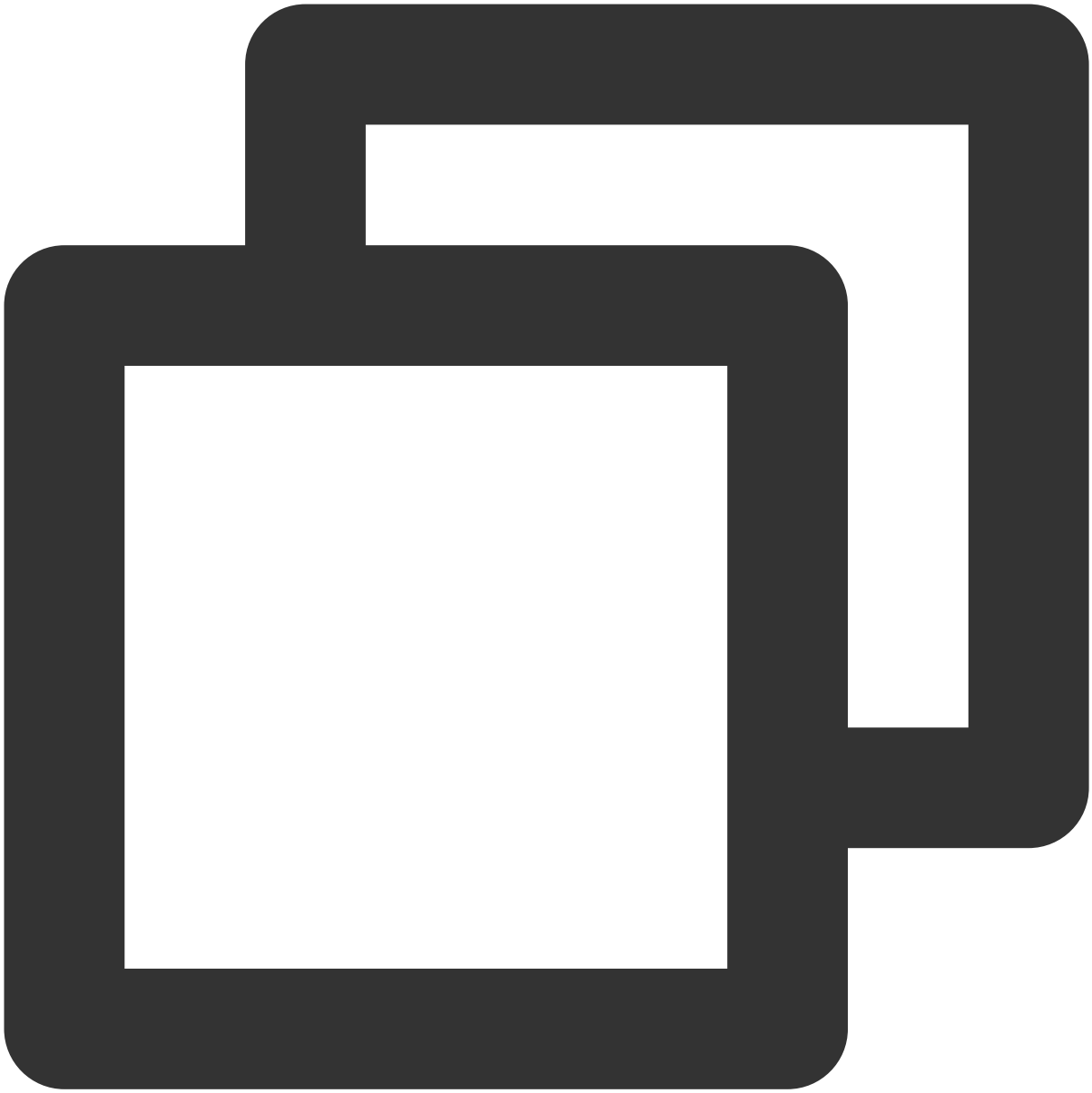
```
IP: 10.20.20.10
bytes: 35
```

```
host: 127.0.0.1
length: 647
referer: http://127.0.0.1/
request: GET /online/sample HTTP/1.1
status: 200
time: [Tue Jan 22 14:49:45 CST 2019 +0800]
```

采集日志的类型

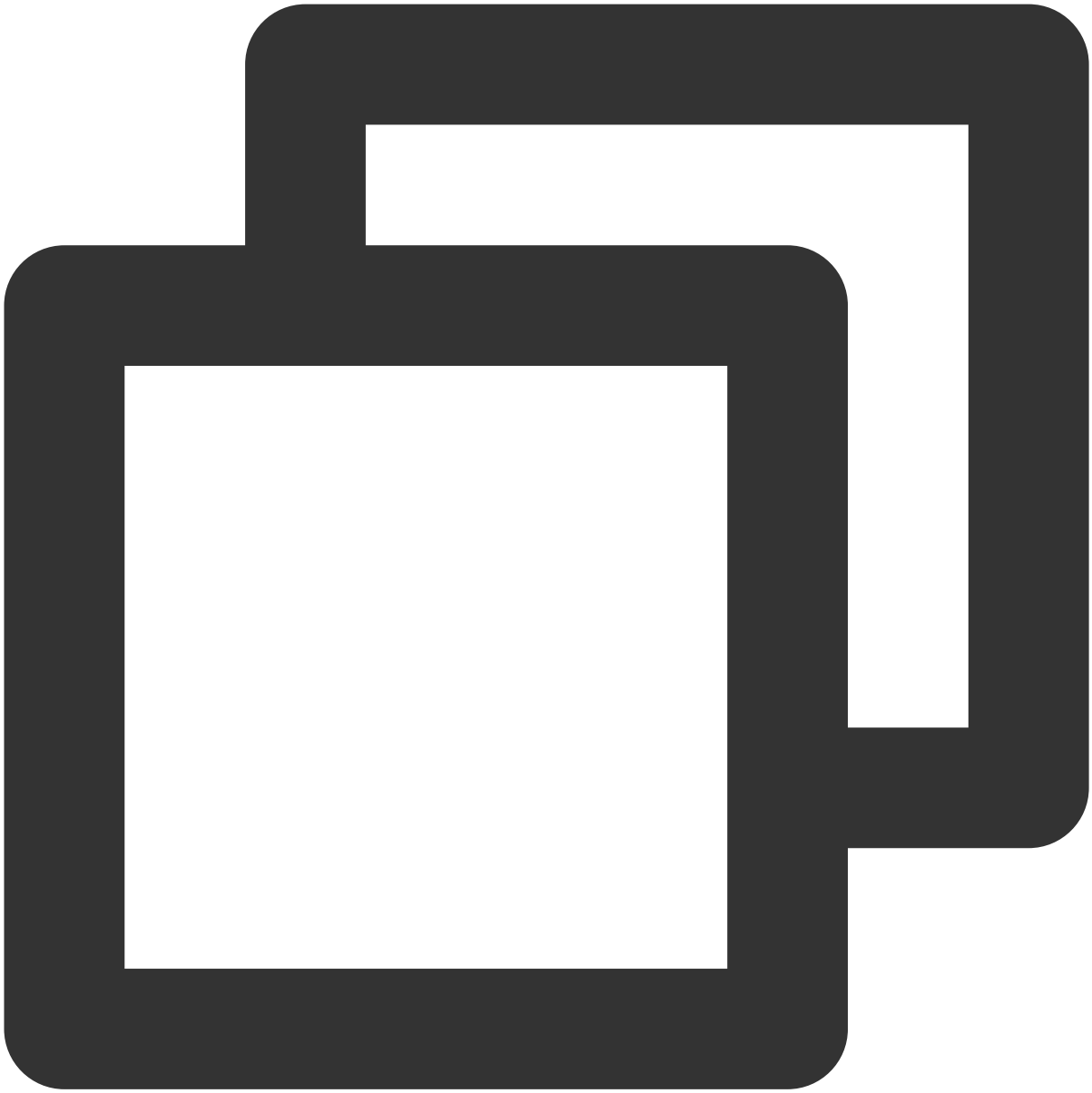
容器标准输出

示例1：采集 default 命名空间中的所有容器的标准输出



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   inputDetail:
     type: container_stdout
     containerStdout:
       namespace: default
       allContainers: true
 ...
```

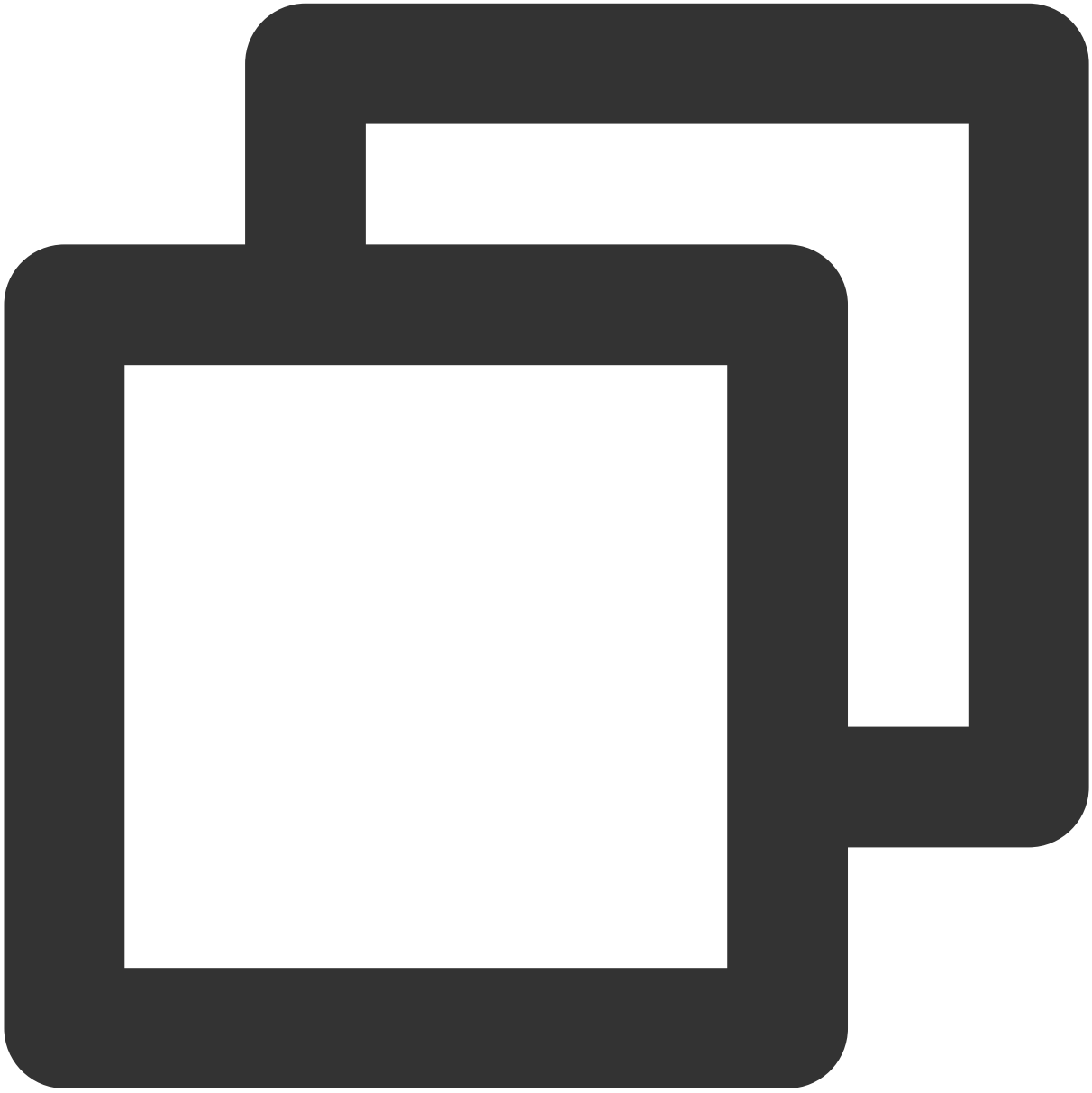
示例2：采集 production 命名空间中属于 ingress-gateway deployment 的 pod 中的容器的标准输出



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   inputDetail:
     type: container_stdout
     containerStdout:
       allContainers: false
       workloads:
       - namespace: production
```

```
        name: ingress-gateway
        kind: deployment
  ...
```

示例3：采集 production 命名空间下 pod 标签中包含 “k8s-app=nginx” 的 pod 中的容器的标准输出

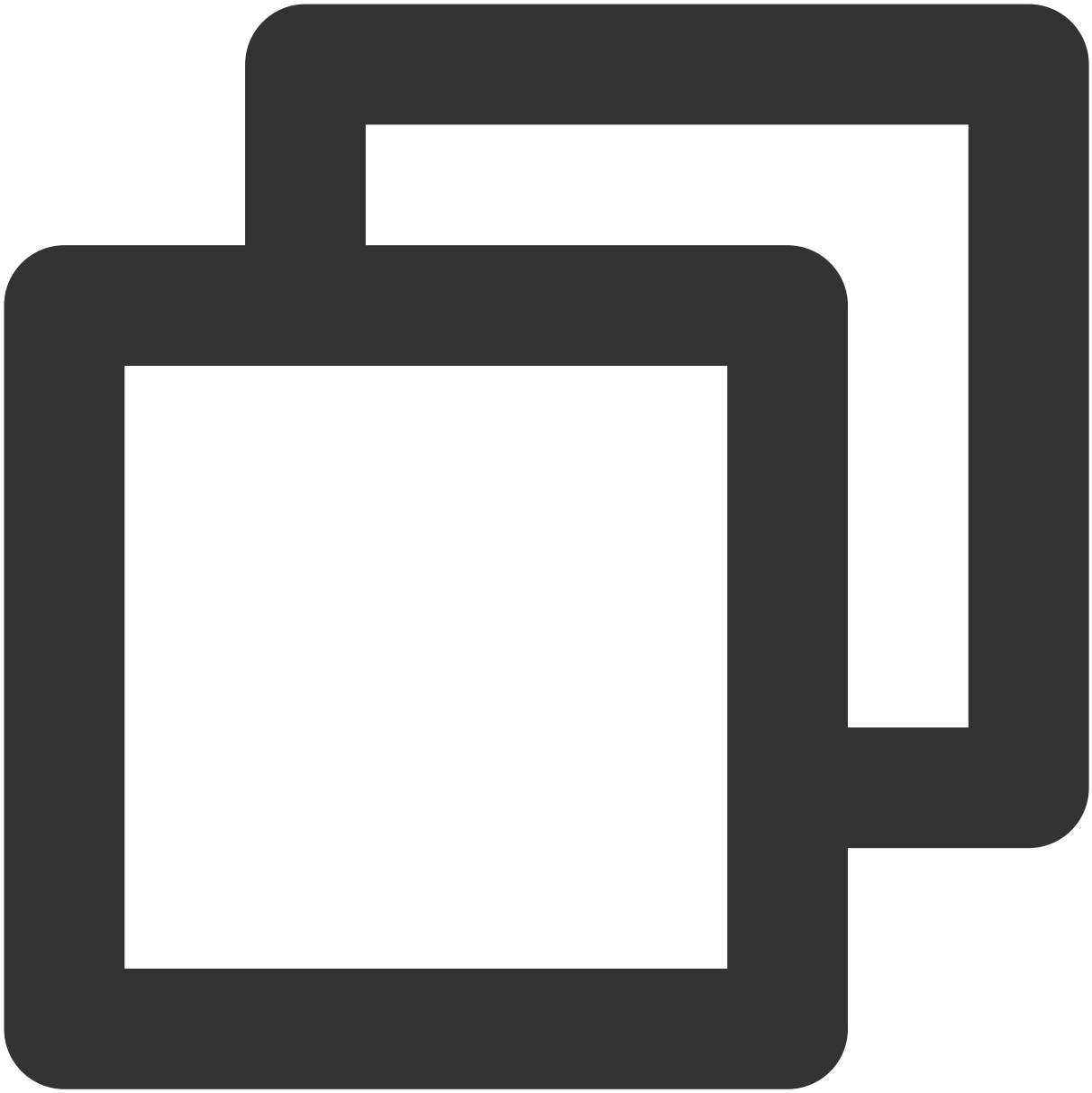


```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   inputDetail:
     type: container_stdout
```

```
      containerStdout:
        namespace: production
        allContainers: false
        includeLabels:
          k8s-app: nginx
  ...
```

容器文件

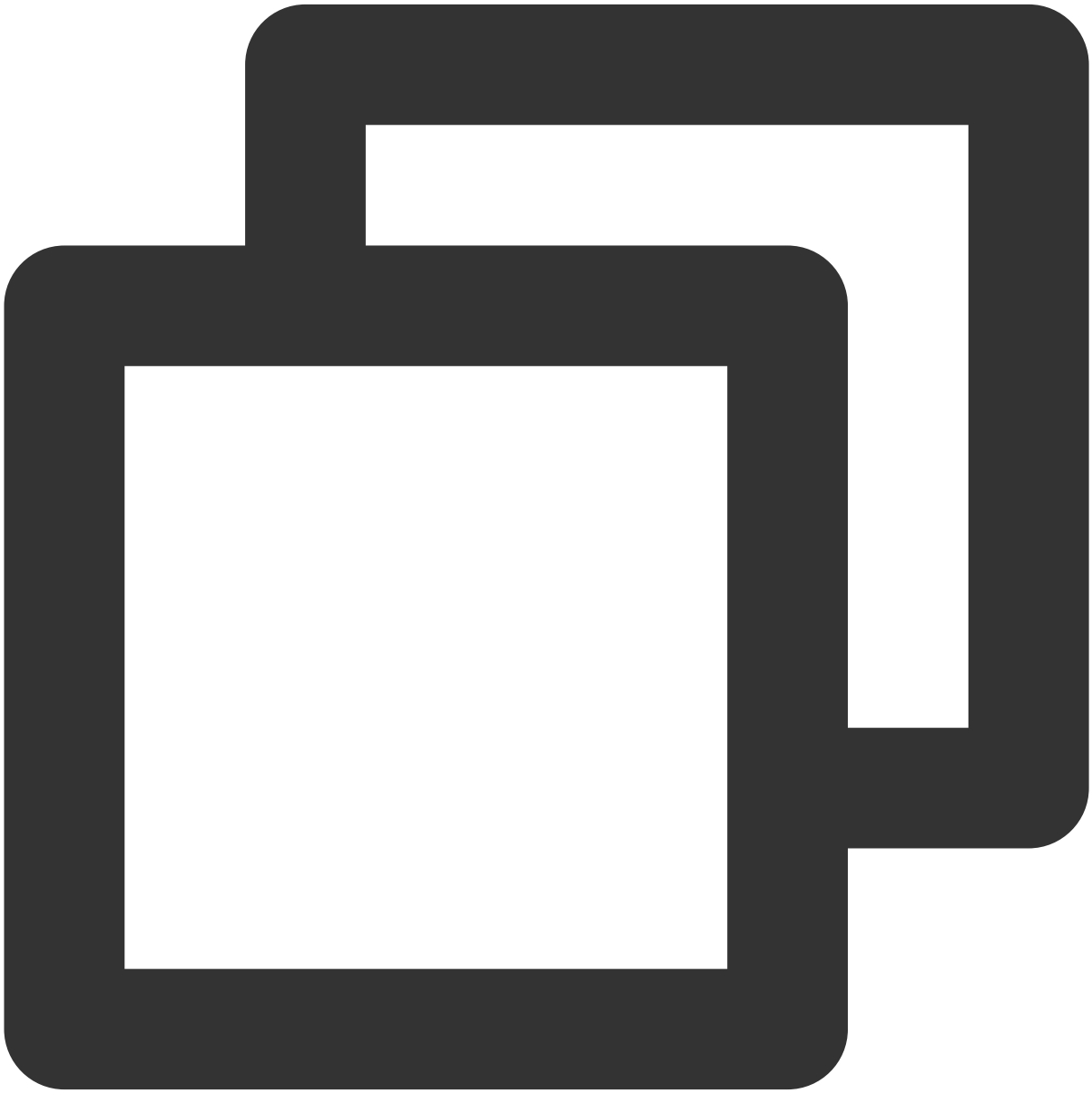
示例1：采集 production 命名空间下属于 ingress-gateway deployment 的 pod 中的 nginx 容器中 `/data/nginx/log/` 路径下名为 `access.log` 的文件



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
spec:
   topicId: xxxxxx-xx-xx-xx-xxxxxxxx
   inputDetail:
     type: container_file
     containerFile:
       namespace: production
       workload:
         name: ingress-gateway
         type: deployment
```

```
        container: nginx
        logPath: /data/nginx/log
        filePattern: access.log
    ...
```

示例2：采集 production 命名空间下 pod 标签包含 “k8s-app=ingress-gateway” 的 pod 中的 nginx 容器中 `/data/nginx/log/` 路径下名为 `access.log` 的文件



```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig
```

```
spec:
   inputDetail:
     type: container_file
     containerFile:
       namespace: production
       includeLabels:
         k8s-app: ingress-gateway
       container: nginx
       logPath: /data/nginx/log
       filePattern: access.log
   ...
```

元数据（Metadata）

容器标准输出（container_stdout）以及容器文件（container_file），除原始的日志内容外，还需携带容器场景的元数据（例如产生日志的容器 ID）一起上报至日志服务。方便用户查看日志时追溯来源或根据容器标识、特征（例如容器名及 labels）进行检索。

元数据如下表：

字段名	含义
cluster_id	日志所属的集群 ID。
container_name	日志所属的容器名称。
image_name	日志所属容器的镜像名称 IP。
namespace	日志所属 pod 的 namespace。
pod_uid	日志所属 pod 的 UID。
pod_name	日志所属 pod 的名字。
pod_ip	日志所属 pod 的IP。
pod_lable_{label name}	日志所属 pod 的 label（例如一个 pod 带有两个 label：app=nginx，env=prod，则在上传的日志会附带两个 metedata：pod_label_app:nginx，pod_label_env:prod）。

使用 CRD 采集日志到 Kafka

最近更新时间：2022-09-13 15:41:28

TKE Serverless集群不仅支持上传日志到 CLS，也支持采集日志到自建 Kafka 或者 CKafka。

创建 CRD

若需要采集日志到 Kafka，只需定义 CRD 即可。具体模板如下：

```
apiVersion: cls.cloud.tencent.com/v1
kind: LogConfig ## 默认值
metadata:
name: test ## CRD资源名，在集群内唯一
spec:
kafkaDetail:
brokers: xxxxxx # 必填，broker地址，一般是域名:端口，多个地址以“,”分隔
topic: xxxxxx # 必填，topicID
messageKey: # 选填，指定pod字段作为key上传到指定分区
valueFrom:
fieldRef:
fieldPath: metadata.name
timestampKey: #时间戳的key，默认是@timestamp
timestampFormat: #时间戳的格式，默认是double
inputDetail:
type: container_stdout ## 采集日志的类型，包括container_stdout（容器标准输出）、container_file（容器文件）
containerStdout: ## 容器标准输出
namespace: default ## 采集容器的kubernetes命名空间，如果不指定，代表所有命名空间
allContainers: false ## 是否采集指定命名空间中的所有容器的标准输出
container: xxx ## 采集日志的容器名，此处可填空
includeLabels: ## 采集包含指定label的Pod
k8s-app: xxx ## 只采pod标签中配置"k8s-app=xxx"的pod产生的日志，与workloads、allContainers=true不能同时指定
workloads: ## 要采集的容器的Pod所属的kubernetes workload
-namespace: prod ## workload的命名空间
name: sample-app ## workload的名字
kind: deployment ## workload类型，支持deployment、daemonset、statefulset、job、cronjob
container: xxx ## 要采集的容器名，如果填空，代表workload Pod中的所有容器
containerFile: ## 容器内文件
namespace: default ## 采集容器的kubernetes命名空间，必须指定一个命名空间
container: xxx ## 采集日志的容器名，此处可填*
includeLabels: ## 采集包含指定label的Pod
```

```
k8s-app: xxx ## 只采pod标签中配置"k8s-app=xxx"的pod产生的日志，与workload不能同时指定
workload: ## 要采集的容器的Pod所属的kubernetes workload
name: sample-app ## workload的名字
kind: deployment ## workload类型，支持deployment、daemonset、statefulset、job、cronjob
logPath: /opt/logs ## 日志文件夹，不支持通配符
filePattern: app_*.log ## 日志文件名，支持通配符 * 和 ? ，* 表示匹配多个任意字符，? 表示匹配单个任意字符
```

注意事项

若无法采集日志，请销毁重建 Pod 重试。

审计管理
集群审计

最近更新时间：2023-05-06 17:36:46

说明：
日志服务 CLS 为 TKE Serverless 集群产生的所有审计、事件数据提供**免费服务**至2021年12月31日。请选择自动创建日志集，或在已有日志集中选择自动创建日志主题。

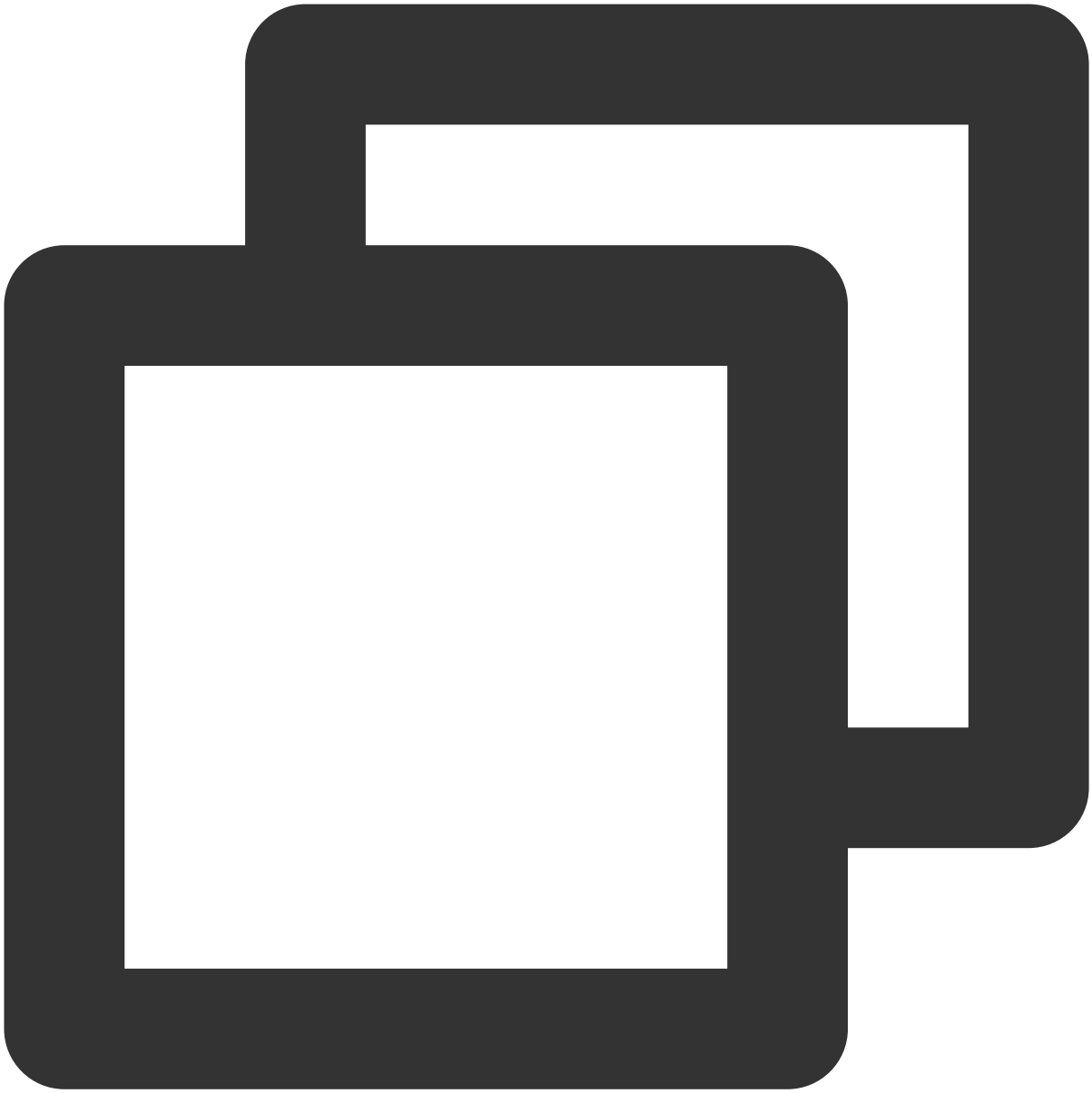
简介

集群审计是基于 Kubernetes Audit 对 kube-apiserver 产生的可配置策略的 JSON 结构日志的记录存储及检索功能。本功能记录了对 kube-apiserver 的访问事件，会按顺序记录每个用户、管理员或系统组件影响集群的活动。

功能优势

集群审计功能提供了区别于 metrics 的另一种集群观测维度。开启集群审计后，Kubernetes 可以记录每一次对集群操作的审计日志。每一条审计日志是一个 JSON 格式的结构化记录，包括元数据（metadata）、请求内容（requestObject）和响应内容（responseObject）三个部分。其中元数据（包含了请求的上下文信息，例如谁发起的请求、从哪里发起的、访问的 URI 等信息）一定存在，请求和响应内容是否存在取决于审计级别。通过日志可以了解到以下内容：
集群里发生的活动。
活动的发生时间及发生对象。
活动的触发时间、触发位置及观察点。
活动的结果以及后续处理行为。

阅读审计日志示例



```
{
  "kind":"Event",
  "apiVersion":"audit.k8s.io/v1",
  "level":"RequestResponse",
  "auditID":0a4376d5-307a-4e16-a049-24e017******,
  "stage":"ResponseComplete",
  // 发生了什么
  "requestURI":"/apis/apps/v1/namespaces/default/deployments",
  "verb":"create",
  // 谁发起的
  "user":{
```

```
      "username":"admin",
        "uid":"admin",
        "groups":[
          "system:masters",
          "system:authenticated"
        ]
    },
    // 从哪里发起
    "sourceIPs":[
      "10.0.6.68"
    ],
    "userAgent":"kubectl/v1.16.3 (linux/amd64) kubernetes/ald64d8",
    // 发生了什么
    "objectRef":{
      "resource":"deployments",
      "namespace":"default",
      "name":"nginx-deployment",
      "apiGroup":"apps",
      "apiVersion":"v1"
    },
    // 结果是什么
    "responseStatus":{
      "metadata":{
      },
      "code":201
    },
    // 请求及返回具体信息
    "requestObject":Object{...},
    "responseObject":Object{...},
    // 什么时候开始/结束
    "requestReceivedTimestamp":"2020-04-10T10:47:34.315746Z",
    "stageTimestamp":"2020-04-10T10:47:34.328942Z",
    // 请求被接收/拒绝的原因是什么
    "annotations":{
      "authorization.k8s.io/decision":"allow",
      "authorization.k8s.io/reason":""
    }
}
```

TKE Serverless 集群审计策略

审计级别（level）

和一般日志不同，Kubernetes 审计日志的级别更像是一种 verbose 配置，用于标示记录信息的详细程度。一共有4个级别，可参考以下表格内容：

参数	说明
None	不记录。
Metadata	记录请求的元数据（例如用户、时间、资源、操作等），不包括请求和响应的消息体。
Request	除了元数据外，还包括请求消息体，不包括响应消息体。
RequestResponse	记录所有信息，包括元数据以及请求、响应的消息体。

审计阶段（stage）

记录日志可以发生在不同的阶段，参考以下表格内容：

参数	说明
RequestReceived	一收到请求就记录。
ResponseStarted	返回消息头发送完毕后记录，只针对 watch 之类的长连接请求。
ResponseComplete	返回消息全部发送完毕后记录。
Panic	内部服务器出错，请求未完成。

审计策略

TKE Serverless 默认收到请求即会记录审计日志，且大部分的操作会记录 RequestResponse 级别的审计日志。但也会存在如下情况：

get、list 和 watch 会记录 Request 级别的日志。

针对 secrets 资源、configmaps 资源或 tokenreviews 资源的请求会在 Metadata 级别记录。

以下请求将不会进行记录日志：

`system:kube-proxy` 发出的监视 endpoints 资源、services 资源或 services/status 资源的请求。

`system:unsecured` 发出的针对 kube-system 命名空间中 configmaps 资源的 get 请求。

kubelet 发出的针对 nodes 资源或 nodes/status 资源的 get 请求。

`system:kube-controller-manager` 、 `system:kube-scheduler` 或 `system:serviceaccount:endpoint-controller` 发出的针对 kube-system 命名空间中 endpoints 资源的 get 和 update 请求。

`system:apiserver` 发出的针对 namespaces 资源、namespaces/status 资源或 namespaces/finalize 资源的 get 请求。

对与 `/healthz*` 、 `/version` 或 `/swagger*` 匹配的网址发出的请求。

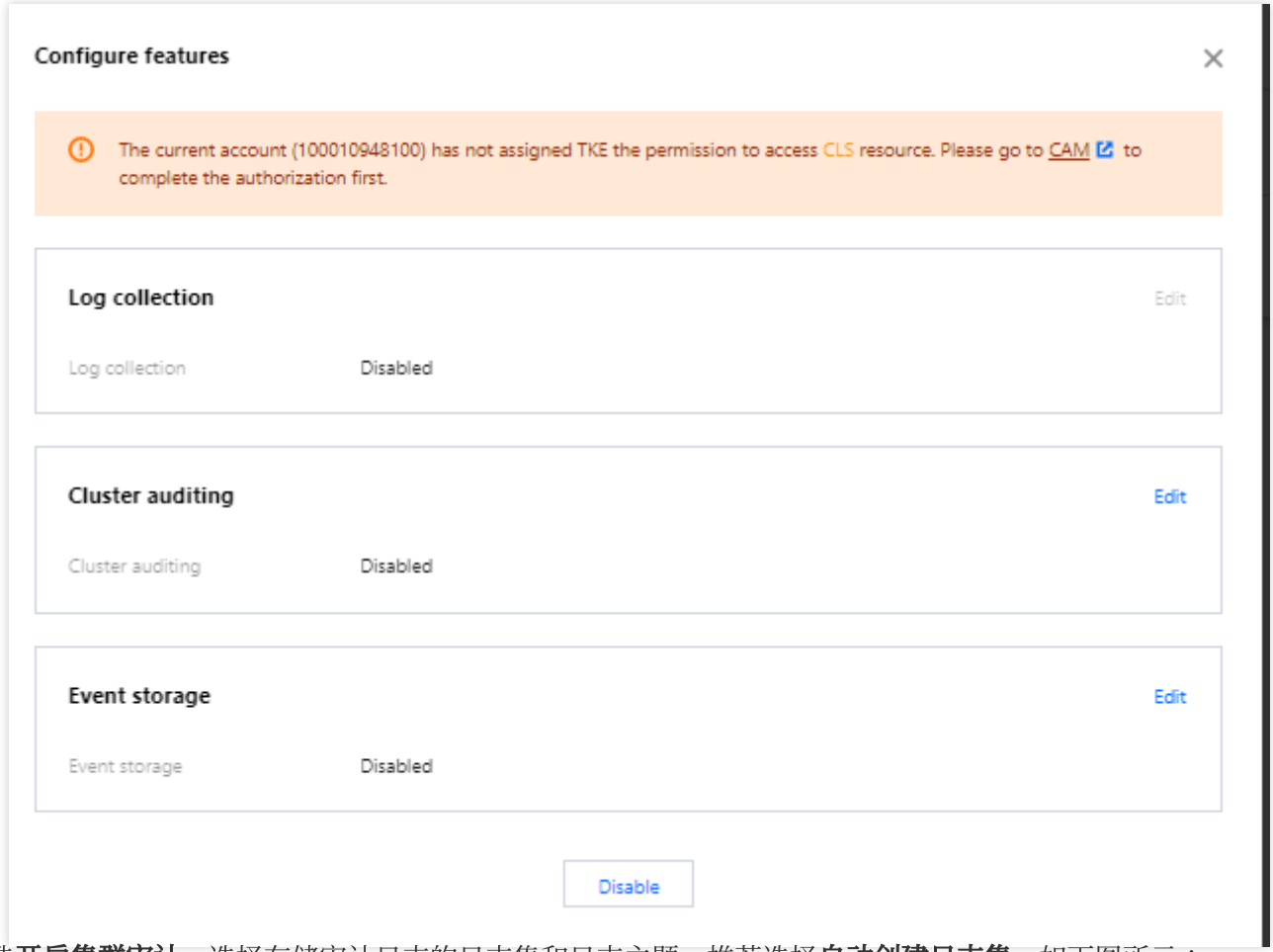
操作步骤

开启集群审计

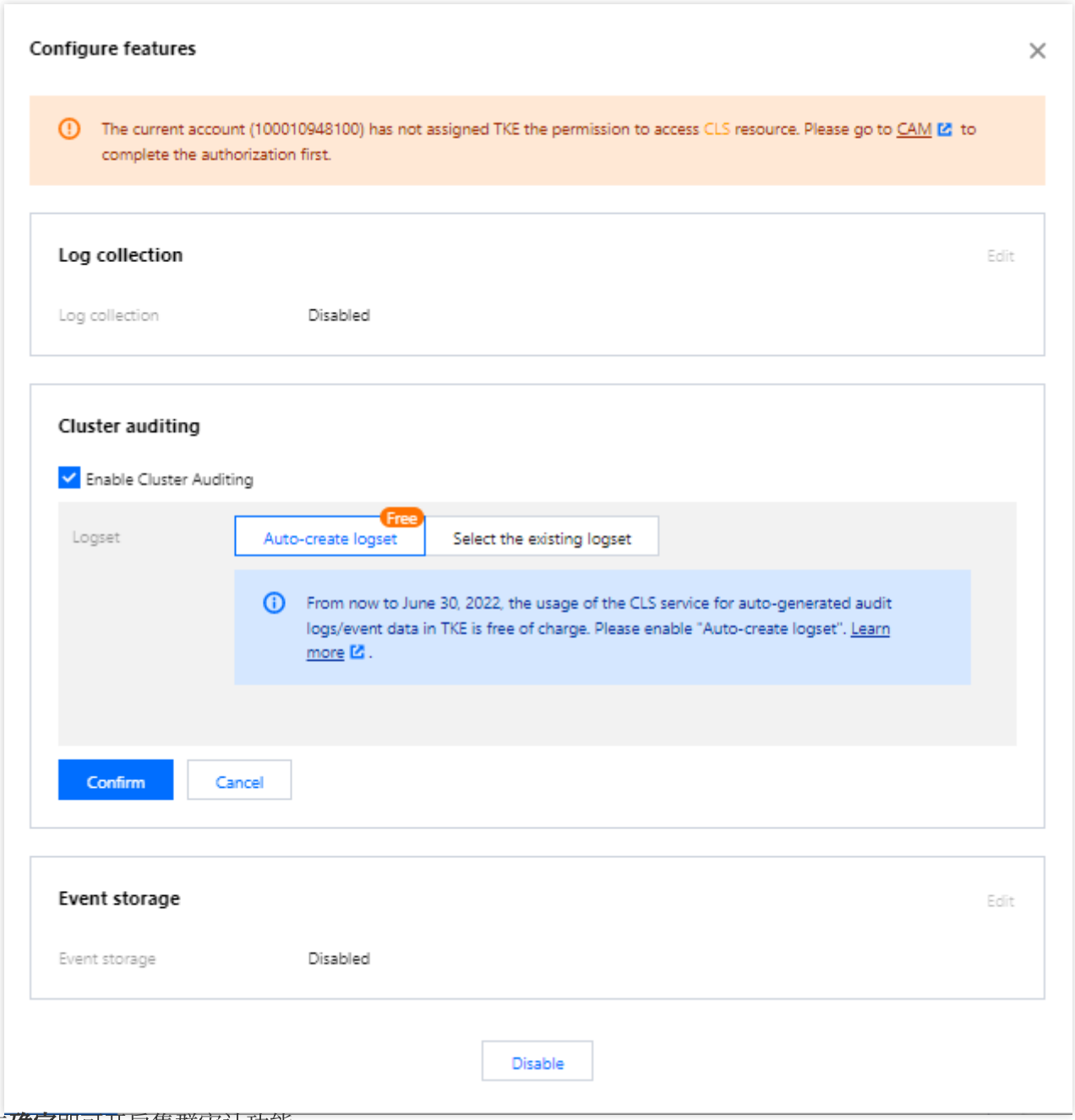
注意

开启集群审计功能需要重启 kube-apiserver ，建议不要频繁开关。

1. 登录 容器服务控制台。
2. 选择左侧导航栏中的**运维功能管理**，进入“功能管理”页面。
3. 在“功能管理”页面上方选择地域、选择 “Serverless 集群”类型。
4. 在下方集群列表找到您希望开启集群审计的集群，在其右侧操作栏中单击**设置**。
5. 在弹出的“设置功能”窗口，单击“集群审计”功能右侧的**编辑**。如下图所示：



6. 勾选**开启集群审计**，选择存储审计日志的日志集和日志主题，推荐选择**自动创建日志集**。如下图所示：



7. 单击**确定**即可开启集群审计功能。

审计仪表盘

最近更新时间：2022-09-14 15:35:30

操作场景

TKEServerless集群为用户提供了开箱即用的审计仪表盘。在集群开启集群审计功能后，TKEServerless集群将自动为该集群配置审计总览、K8S 对象操作概览、聚合检索仪表盘。还支持用户自定义配置过滤项，同时内置 CLS 的全局检索，方便用户观测和检索各类集群操作，以便于及时发现和定位问题。

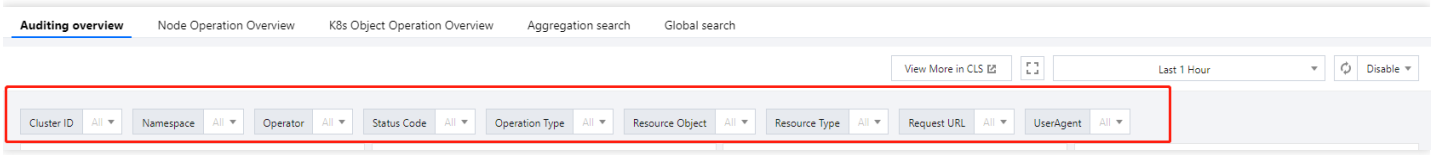
功能介绍

审计检索中配置了四个大盘，分别是“审计总览”、“K8S 对象操作概览”、“聚合检索”、“全局检索”。请按照以下步骤进入“审计检索”页面，开始使用对应功能：

1. 登录 容器服务控制台 。
2. 开启集群审计功能，详情请参见 集群审计。
3. 选择导航栏左侧**集群运维** > **审计日志**，进入“审计检索”页面。

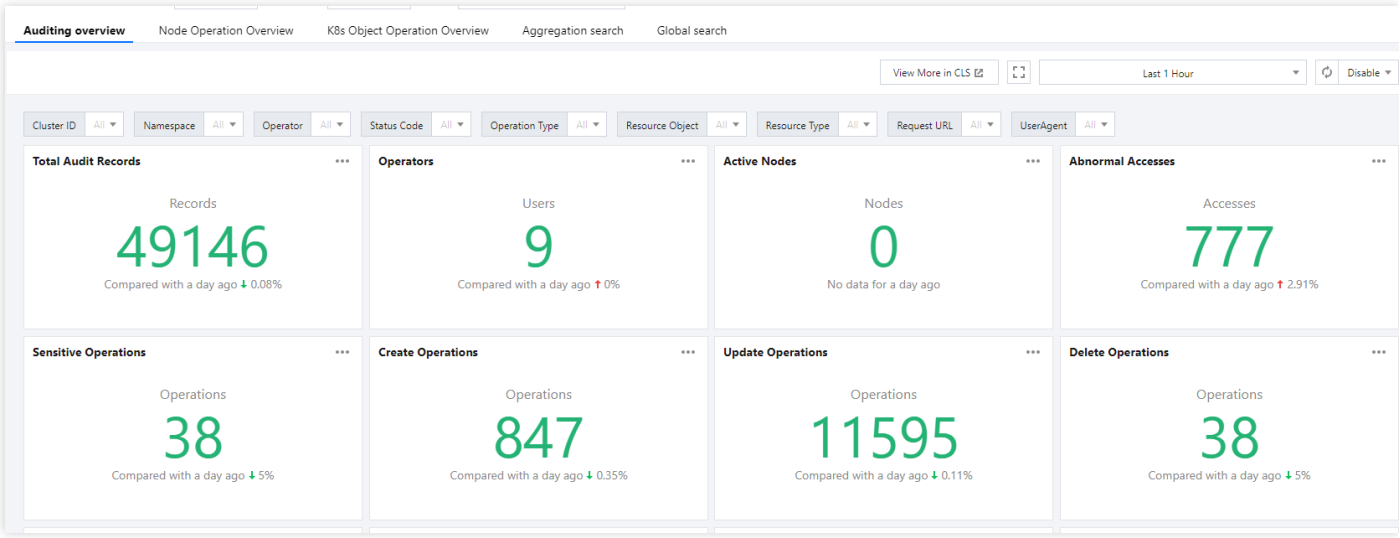
审计总览

当您需要观测整个集群 APIserver 操作时，可在“审计总览”页面设置过滤条件，查看核心审计日志的汇总统计信息，并展示一个周期内的数据对比。例如，核心审计日志的统计数、分布情况、重要操作趋势等。
在过滤项中，您可根据自己需求进行个性化配置，如下图所示：

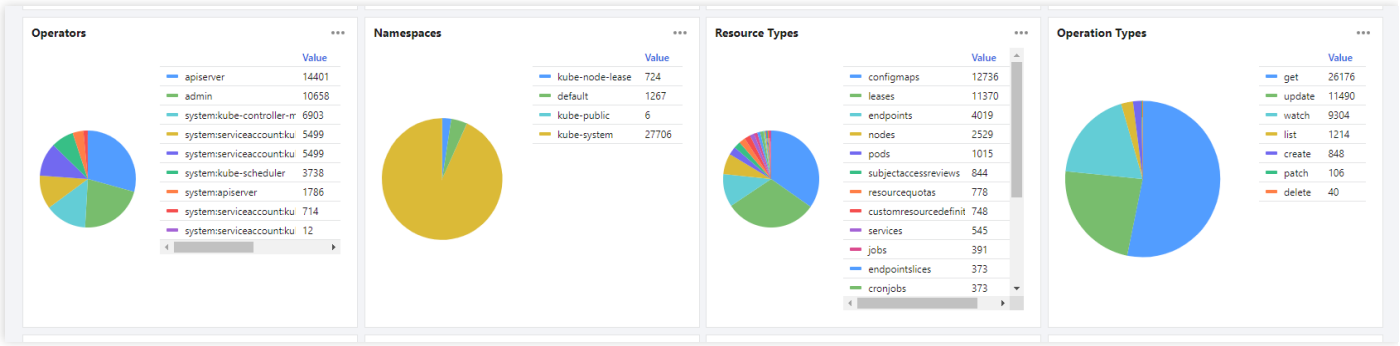


您还可在该页面中查看更多统计信息，如下所示：

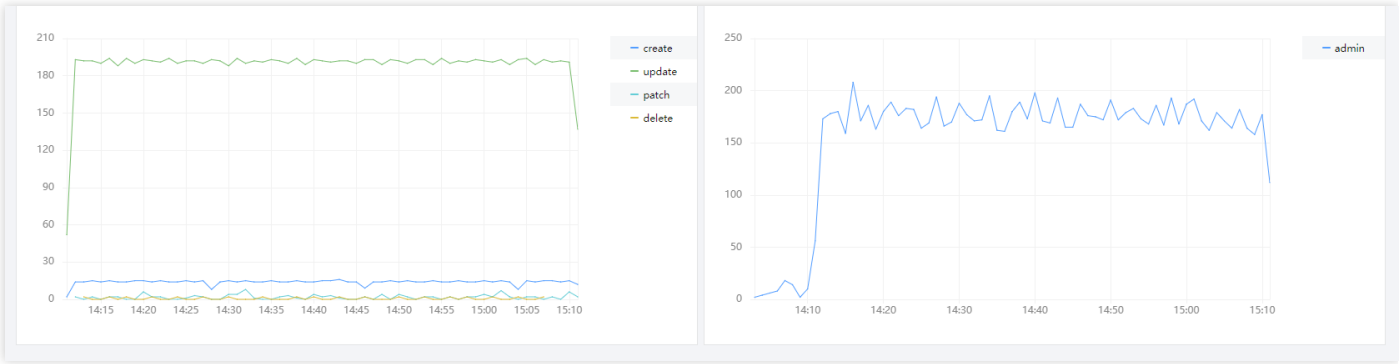
- 核心审计日志的统计数仪表盘：



- 分布情况仪表盘：



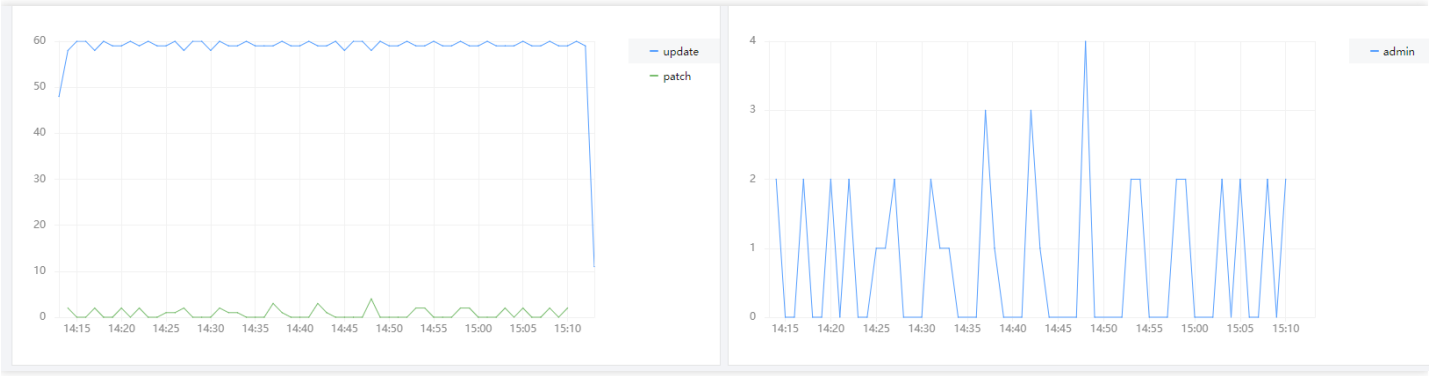
- 重要操作趋势仪表盘：



K8S 对象操作概览

当您需要排查 K8S 对象（例如某个工作负载）的相关问题时，可在切换至**K8S 对象操作概览**页签，在此页面设置过滤条件，查看各类 K8S 对象的操作概览、对应的操作用户、相应的审计日志列表，以查找更多的细节。

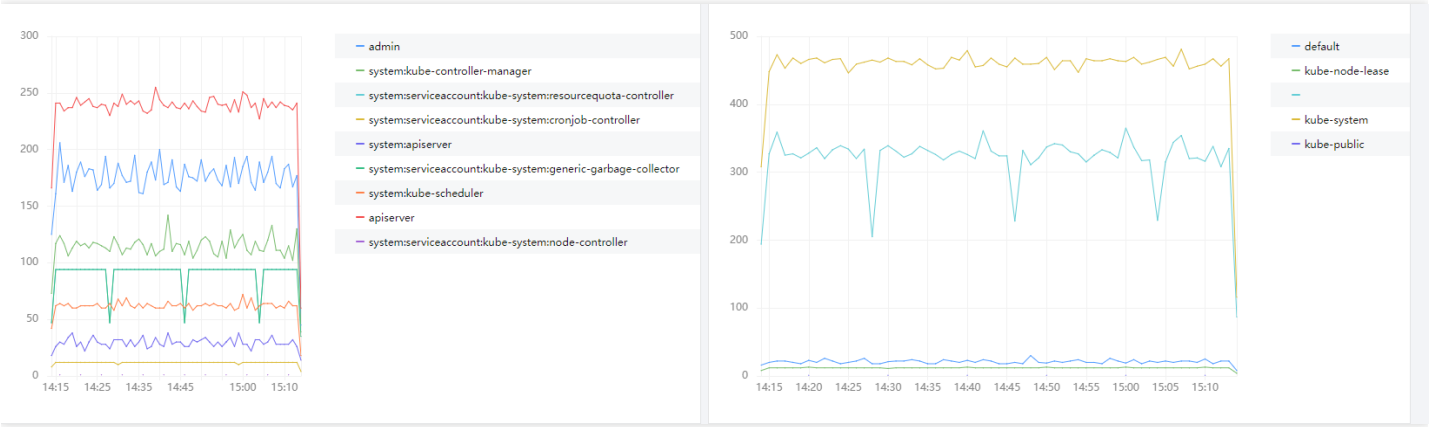
如下图所示：



聚合检索

当您需要观测某个维度下审计日志的分布趋势，可在“聚合检索”页面设置过滤条件，查看各类重要操作的时序图。纬度包括操作用户、命名空间、操作类型、状态码、资源类型以及相应的审计日志列表。

如下图所示：



全局检索

全局检索仪表盘中内嵌了 CLS 的检索分析页面，方便用户在容器服务控制台 也能快速检索全部审计日志。

如下图所示：

基于仪表盘配置告警

您可以通过以上预设的仪表盘配置告警，达到您所设置的条件则触发告警。具体操作如下：

1. 在需要配置告警的仪表盘右侧中单击 ，在下拉菜单中选择**快速添加告警**。
2. 配置告警策略详细步骤，请参见 配置告警策略。

事件管理
事件存储

最近更新时间：2022-09-14 17:01:10

> 说明：
> 日志服务 CLS 为 TKE Serverless 集群产生的所有审计、事件数据提供**免费服务**至2022年9月5日。请选择自动创建日志集，或在已有日志集中选择自动创建日志主题。活动详情请参见 TKE 容器服务审计与事件中心日志免费说明。

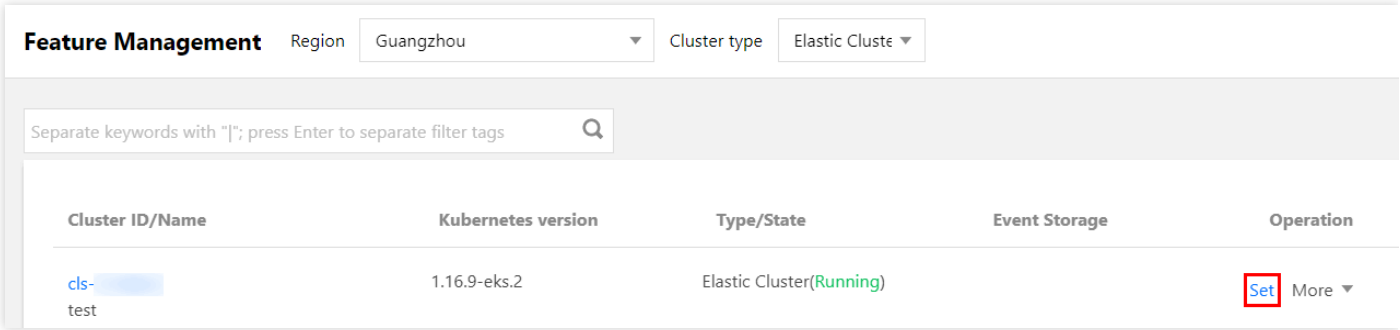
操作场景

Kubernetes Events 包括 Kuberntes 集群的运行和各类资源调度情况，有助于维护人员日常观察资源的变更以及定位问题。TKE Serverless 集群支持为您的所有集群配置事件持久化功能，还支持使用腾讯云提供的 PAAS 服务和开源软件对事件流水进行检索。开启集群事件持久化后，TKE Serverless集群会将您的集群事件实时导出至配置的存储端。本文档指导您如何开启集群事件持久化存储，您可参考本文开始使用事件存储功能。

操作步骤

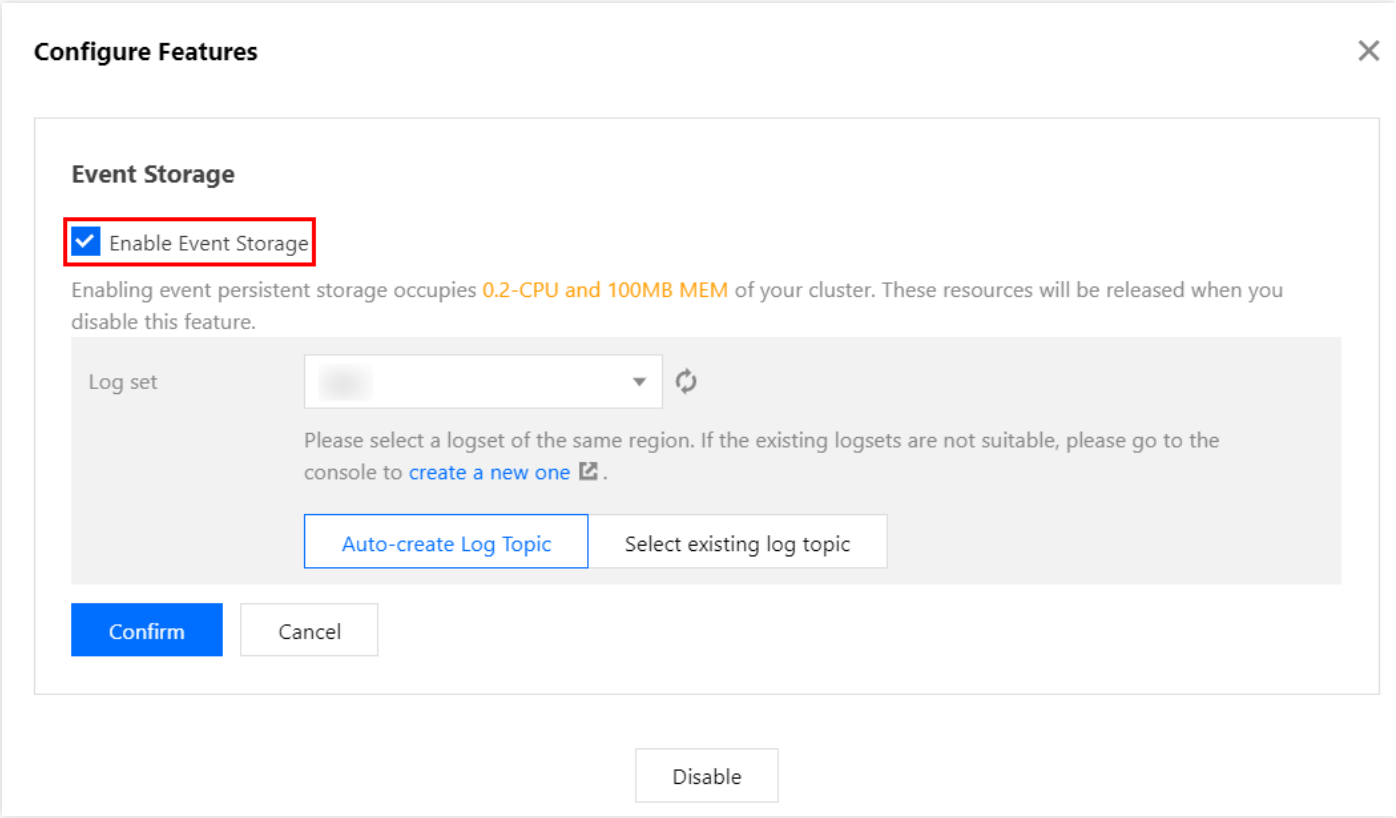
开启事件存储

1. 登录 容器服务控制台。
2. 在左侧导航栏中，选择**运维功能管理**，进入“功能管理”页面。
3. 在“功能管理”页面上方选择地域和集群类型“Serverless 集群”，单击需要开启事件存储的集群右侧的**设置**。如下图所示：



4. 在“设置功能”页面，单击事件存储**编辑**。

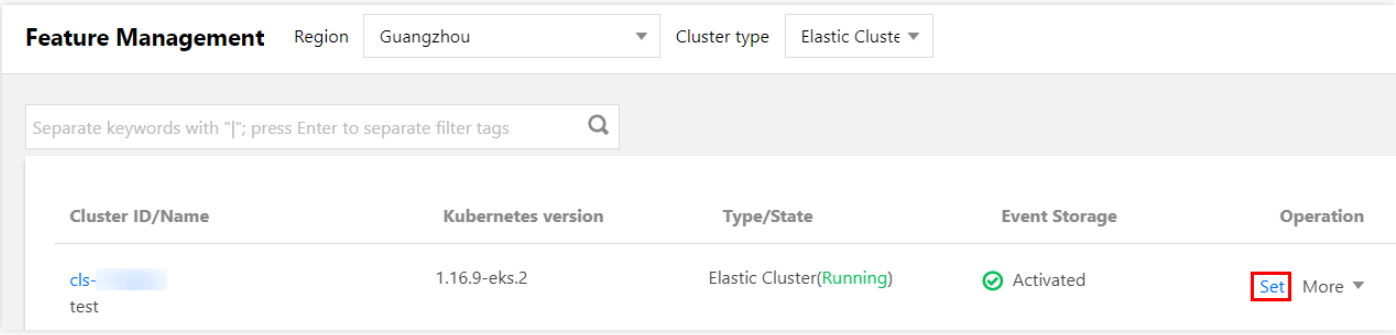
5. 在“事件存储”编辑页面，勾选**开启事件存储**，并配置日志集和日志主题。如下图所示：



6. 单击**确定**，即可开启事件存储。

更新日志集或日志主题

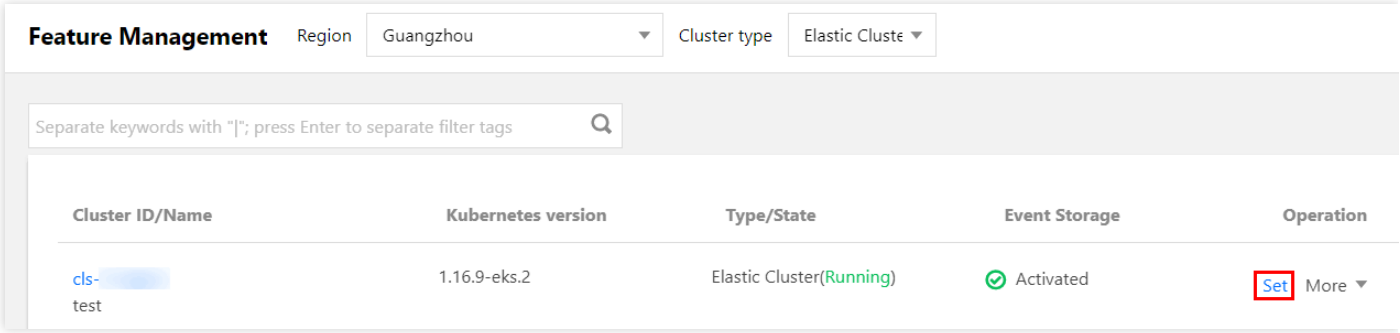
1. 登录 容器服务控制台。
2. 在左侧导航栏中，选择**运维功能管理**，进入“功能管理”页面。
3. 在“功能管理”页面上方选择地域和集群类型“Serverless 集群”，单击需要开启事件存储的集群右侧的**设置**。如下图所示



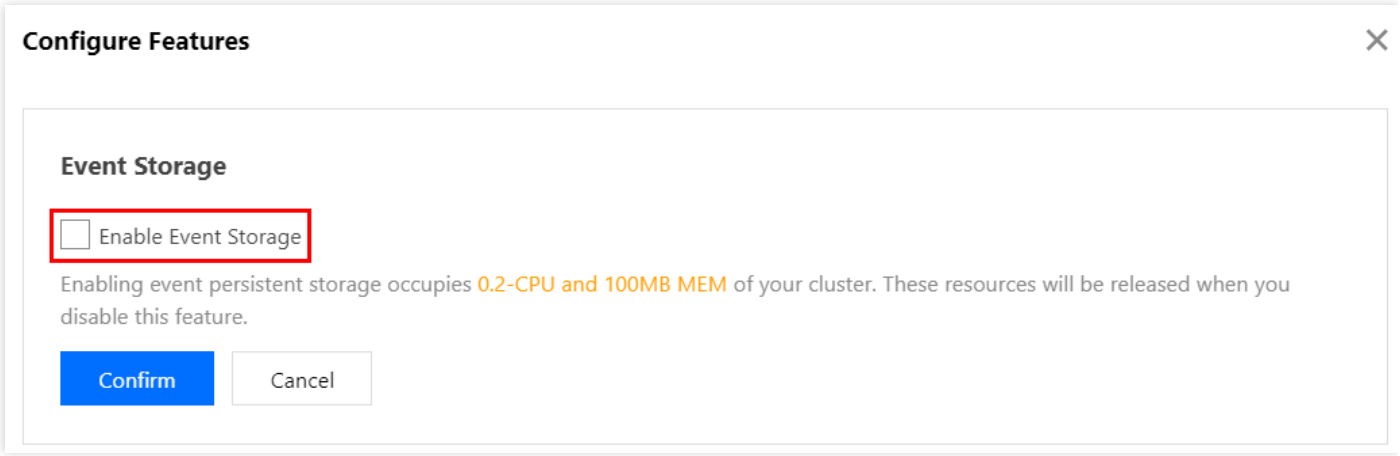
4. 在“设置功能”页面，单击事件存储**编辑**。
5. 在“事件存储”编辑页面，重新选择日志集和日志主题。单击**确定**即可更新日志集和日志主题。

关闭事件存储

1. 登录 容器服务控制台。
2. 在左侧导航栏中，选择**运维功能管理**，进入“功能管理”页面。
3. 在“功能管理”页面上方选择地域和集群类型“Serverless 集群”，单击需要开启事件存储的集群右侧的**设置**。如下图所示



4. 在“设置功能”页面，单击事件存储**编辑**。
5. 在“事件存储”编辑页面，取消勾选**开启事件存储**。如下图所示，

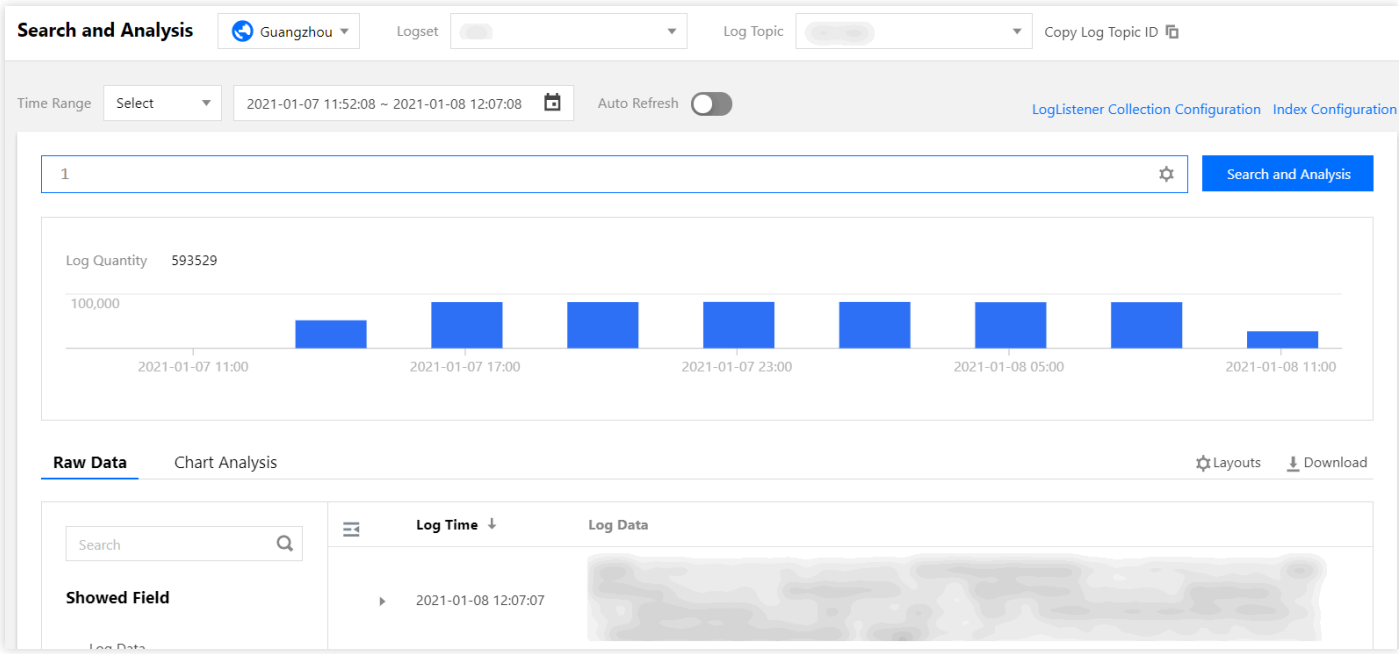


6. 单击**确定**，即可关闭事件存储。

在 CLS 控制台查看事件

1. 登录 日志服务控制台。

2. 在左侧导航栏中，选择**检索分析**，进入“检索分析”页面。如下图所示：



3. 选择在 事件存储 中配置的日志集和日志主题，按需自行配置显示字段并进行检索分析，详情请参见 日志检索分析。

> 注意：
> 当您开启事件存储时，将默认为您的 Topic 开启索引。

事件仪表盘

最近更新时间：2023-05-22 15:56:59

操作场景

TKE Serverless 集群为用户提供了开箱即用的事件仪表盘。在 Serverless 集群开启事件存储功能后，TKE Serverless 集群将自动为集群配置各类事件总览大盘和异常事件的聚合检索分析仪表盘。还支持用户自定义配置过滤项，同时内置 CLS 的事件全局检索，实现在容器服务控制台 全面观测、查找、分析、定位问题的能力。

功能介绍

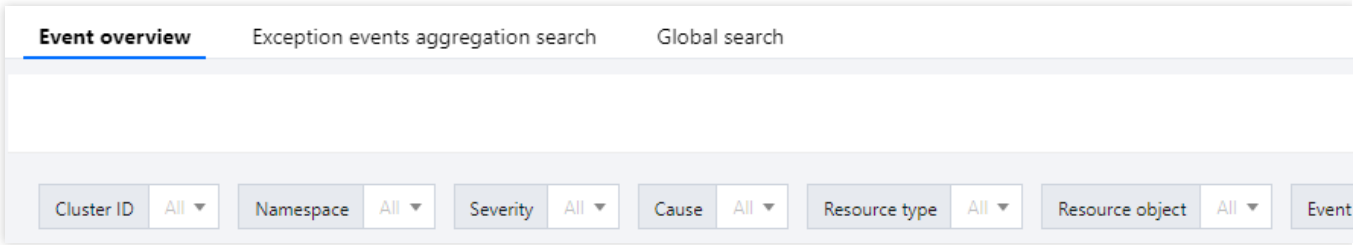
事件检索中配置了三个大盘，分别是“事件总览”、“异常事件聚合检索”、“全局检索”。请按照以下步骤进入“事件检索”页面，开始使用对应功能：

1. 登录 容器服务控制台。
2. 开启“事件存储功能”，详情请参见 事件存储。
3. 选择导航栏左侧**日志管理 > 事件日志**，进入“事件检索”页面。

事件总览

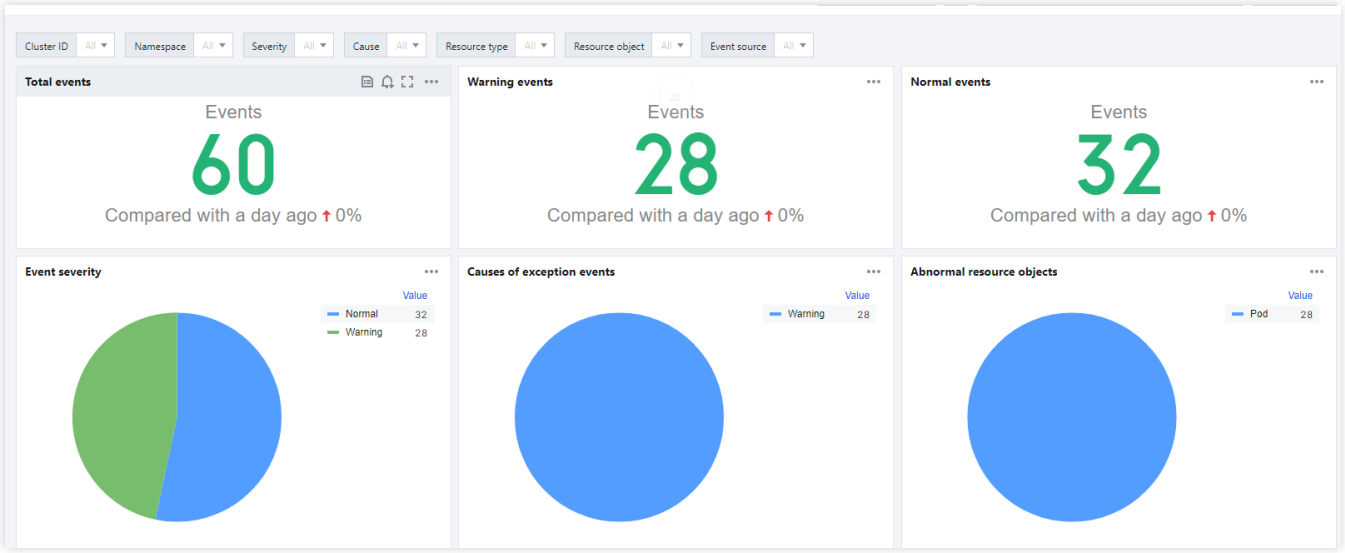
在“事件总览”页面，可根据集群ID、命名空间、级别、原因、资源类型、资源对象、事件源等维度过滤事件，查看核心事件的汇总统计信息，并展示一个周期内的数据对比。例如，事件总数及分布情况、节点异常、Pod OOM、重要事件趋势等仪表盘以及异常 TOP 事件列表。

在过滤项中，您可根据自己需求进行个性化配置。如下图所示：

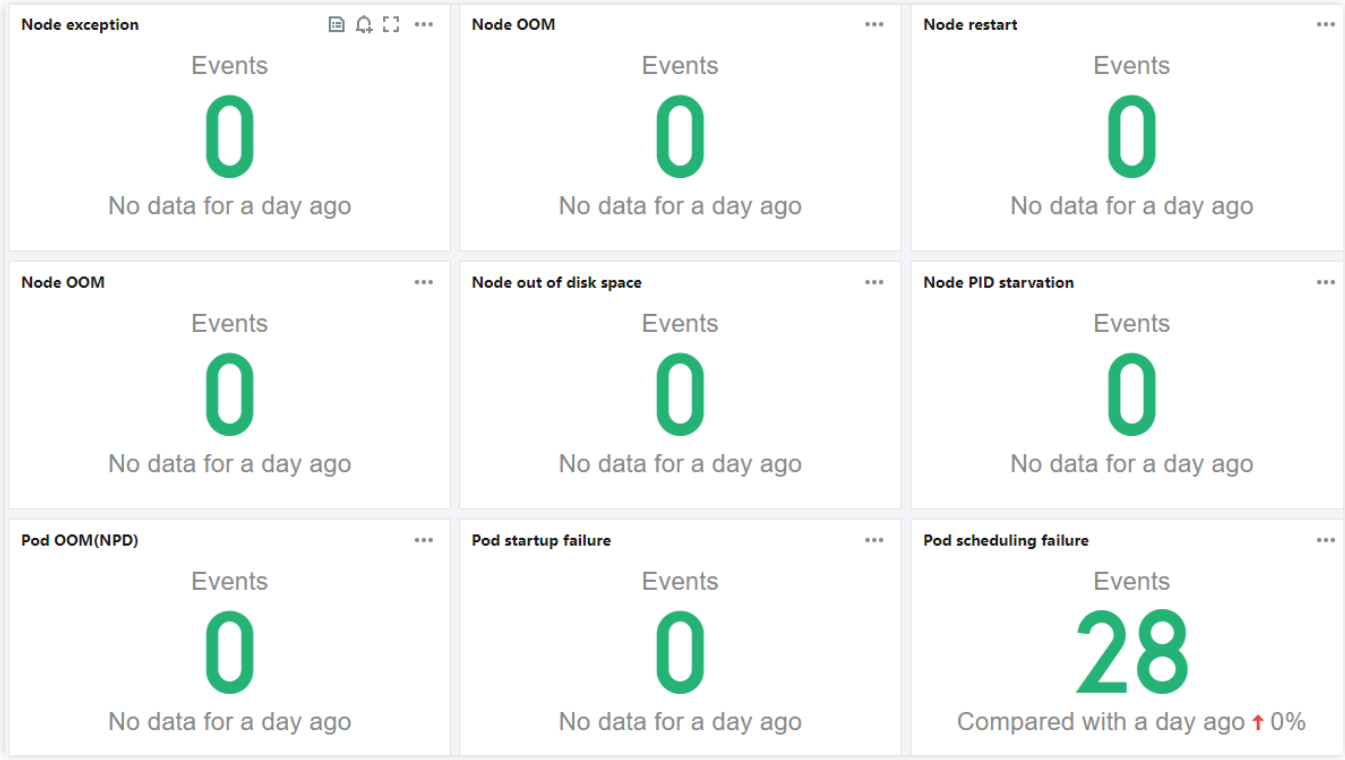


您还可在该页面中查看更多统计信息，如下所示：

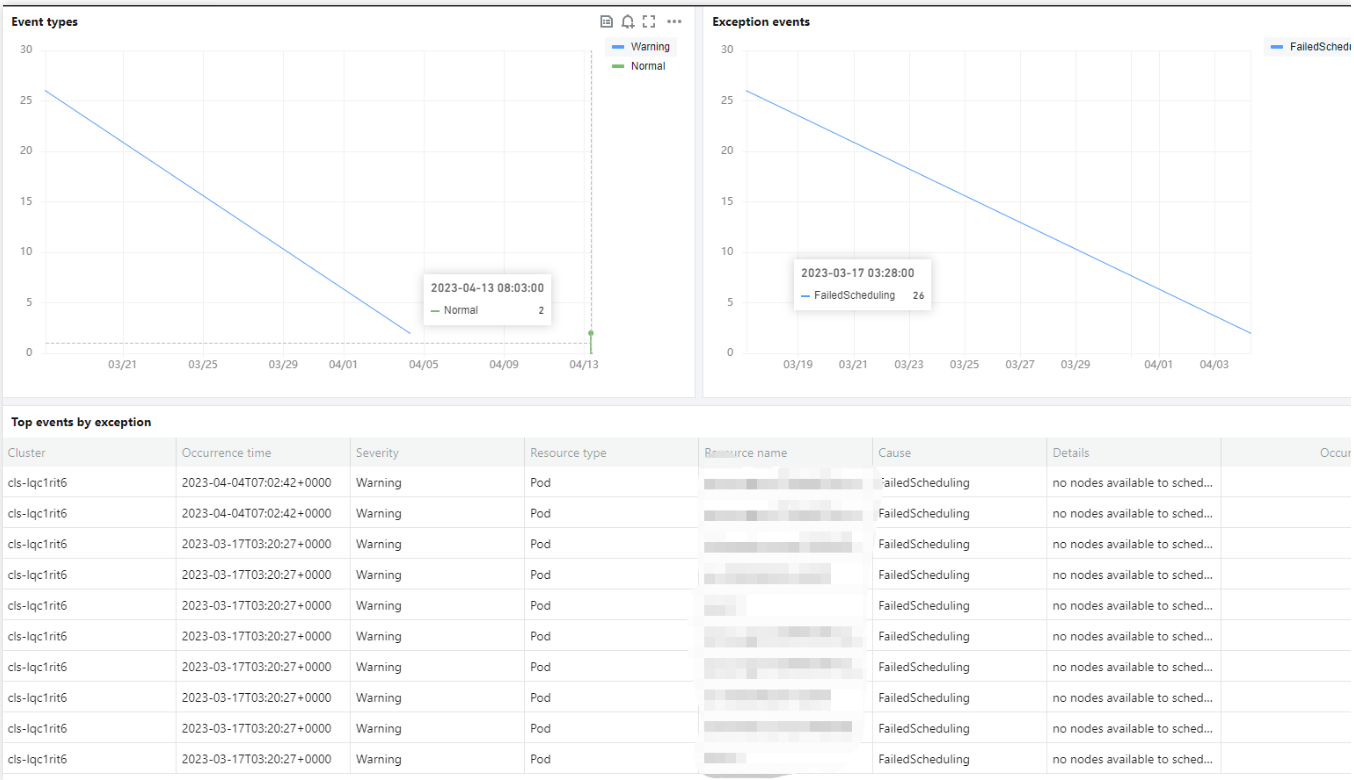
事件总数及级别分布情况，异常事件的原因、对象分布情况检索如下图所示：



各类常见事件的汇总信息检索如下图所示：

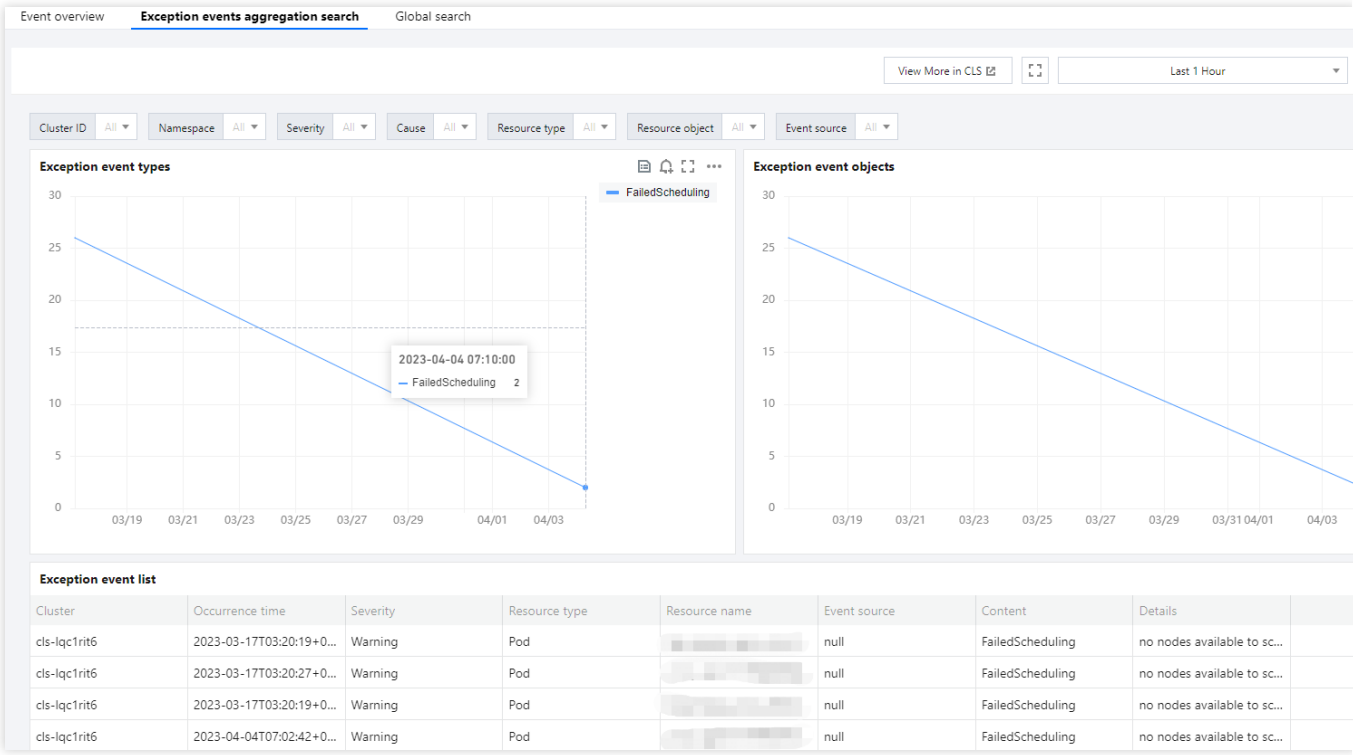


事件趋势及异常 TOP 事件列表检索如下图所示：



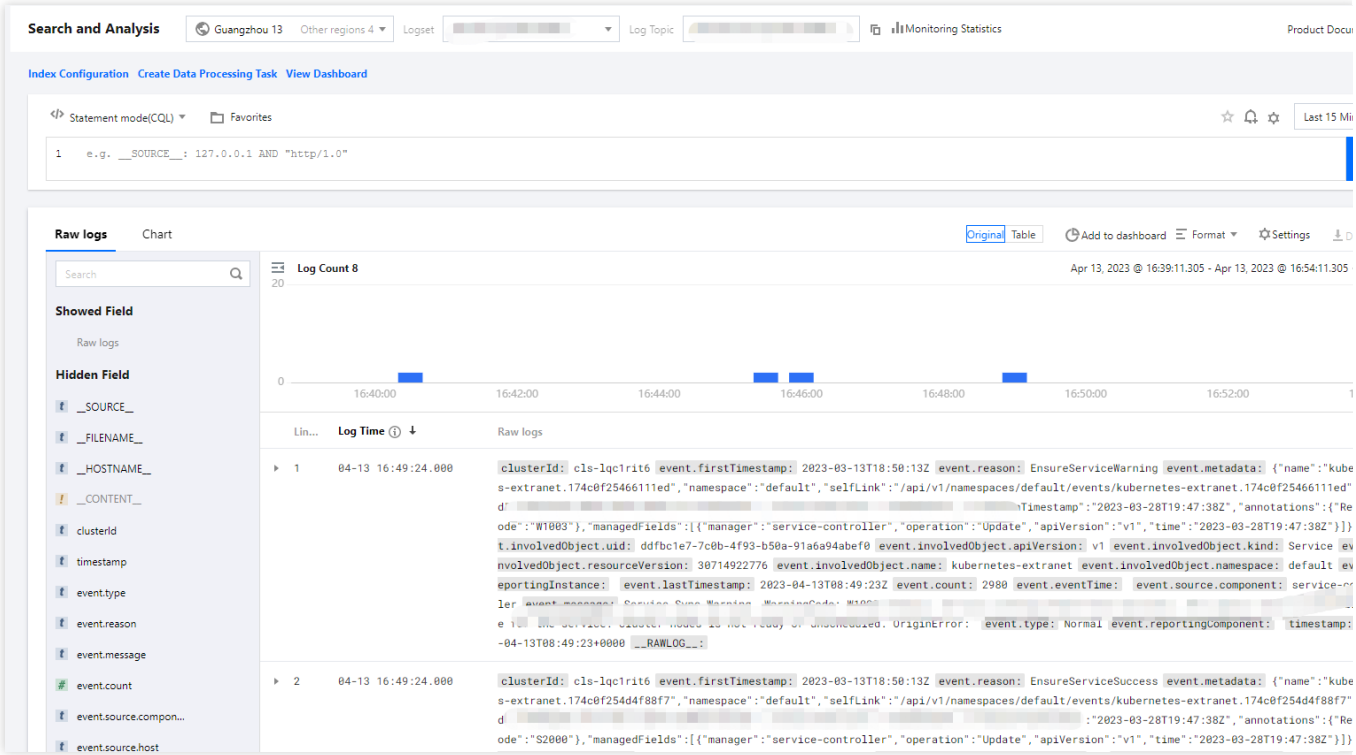
异常事件聚合检索

在“异常事件聚合检索”页面设置过滤条件，查看某个时间段内各类异常事件的 reason 和 object 分布趋势。在趋势下方展示了可供搜索的异常事件列表，帮助您快速定位到问题。如下图所示：



全局检索

全局检索仪表盘中内嵌了 CLS 的检索分析页面，方便用户在容器服务控制台 也能快速检索全部事件。如下图所示：

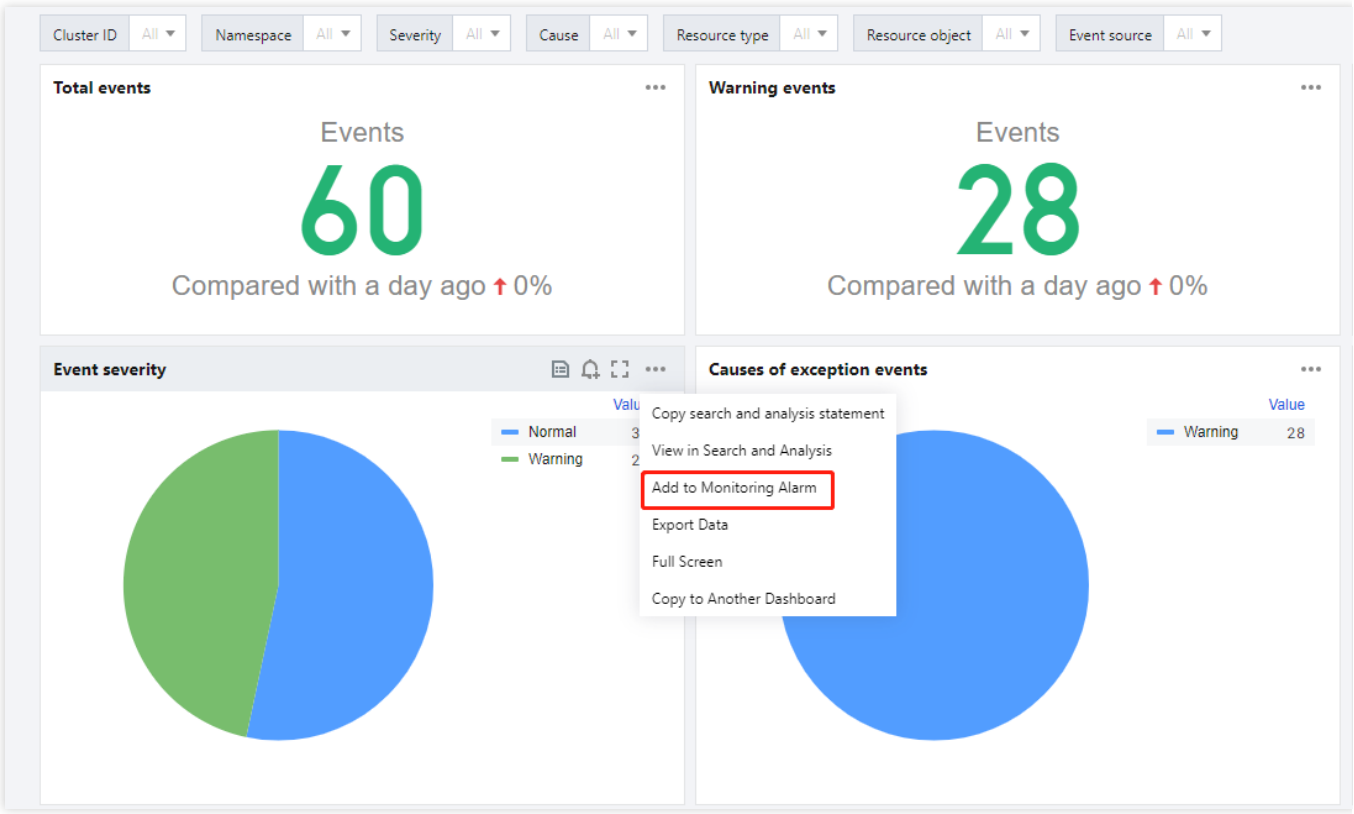


基于仪表盘配置告警

您可以通过以上预设的仪表盘配置告警，达到您所设置的条件则触发告警。具体操作如下：

1. 在需要配置告警的仪表盘右侧中单击

，在下拉菜单中选择**快速添加告警**，如下图所示：



2. 配置告警策略详细步骤，请参见 配置告警策略。

常见问题

最近更新时间：2023-05-22 17:02:01

TKE Serverless 集群相关问题

负载均衡相关问题

超级节点相关问题

如何禁止 Pod 调度到某个超级节点？
如何禁止 TKE 普通集群在资源不足时自动调度到超级节点？
如何手动调度 Pod 到超级节点？
如何强制调度 Pod 到超级节点，无论超级节点是否支持该 Pod？
如何自定义超级节点 DNS？

镜像仓库相关问题

TKE Serverless 集群如何使用容器镜像服务 TCR？
TKE Serverless 集群如何使用自建的自签名镜像仓库或 HTTP 协议镜像仓库？

日志采集相关问题

集群配置日志采集后，为什么在日志服务控制台查看不到日志？
配置好日志规则后，日志在哪查看？
Java 类应用如何实现多行日志合并？
如何调整日志采集配置，以适应不同的日志输出速率？
TKE Serverless 集群容器内应用输出日志的标准是什么？

联系我们

最近更新时间：2022-04-25 15:37:44

热线电话

当您在使用腾讯云遇到问题时，可以直接致电客服人员，寻求相应的帮助。

- 中国香港 +852 800-964-163 (免费热线)
- 美国：+1 888-652-2736 (免费热线)
- 其他：+86 4009100100

工单系统

当您遇到运维或技术类产品使用问题时，可以登录 腾讯云官网，根据界面指引提交工单，我们将尽快响应，期待收到您的宝贵意见。
工单相关入口如下：

- 工单提交：提交工单
- 状态查询：工单列表

工单状态说明如下：

- 未处理：新提交的工单，或者技术支持收到工单但评估还未完成。您可以对未处理工单进行补充和关闭操作。
- 处理中：技术支持收到工单并评估，正在实施过程中。您可以对处理中的工单进行补充和关闭操作。
- 待补充：技术支持收到工单并评估，未提交完整的信息，需要补充。您可以对待补充工单进行关闭操作。

> 说明：
> 当“待补充”的工单补充后重新提交，会再次进入“未处理”状态。

- 已结单：工单实施完成，或者您在技术支持操作前关闭工单。