TencentCloud Managed Service for Prometheus

Operation Guide

Product Documentation

Contents

Operation Guide
Instance
Creating Instance

Last updated：2024-01-29 16:01:55

This document uses custom configuration as an example to describe how to create a TMP instance.

Preparations

Before creating a TMP instance, you must complete the following operations:
Sign up for a Tencent Cloud account and complete identity verification.
Create a VPC in the target region and create a subnet in the target AZ in the VPC.

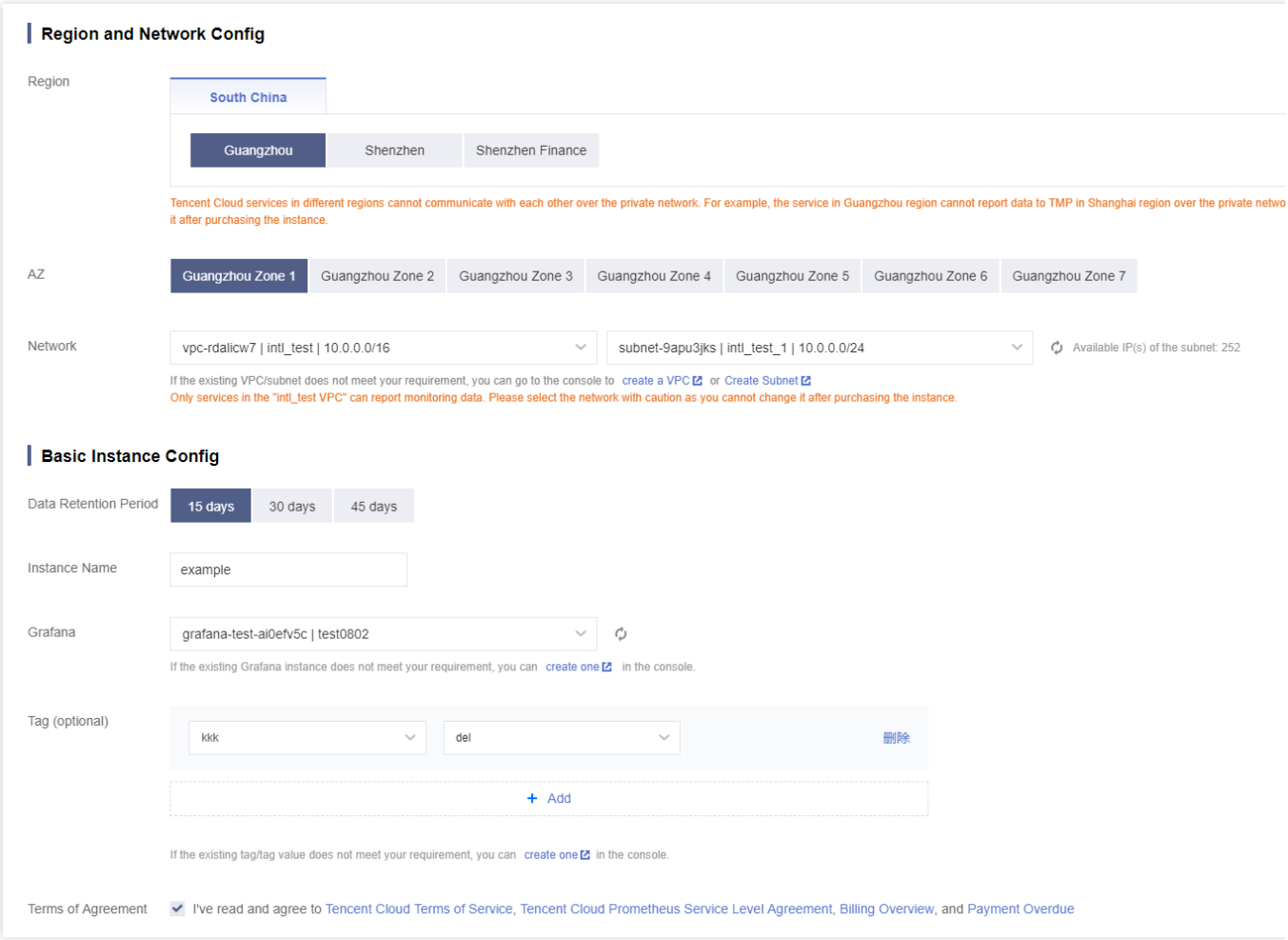
Directions

1. Log in to the TMP console.
2. Click **Create** and configure the following information as prompted:

Type	Required	Configuration Description
Region	Yes	Select a region based on the region of your Tencent Cloud service. The price may vary by region, and the real-time price displayed on the purchase page shall prevail. Tencent Cloud services in different regions cannot communicate with each other over the private network; for example, a service in the Guangzhou region cannot report data to a TMP instance in the Shanghai region over the private network. Once an instance is purchased, the region cannot be changed. Therefore, please select the region with caution.
AZ	Yes	Select as needed.
Network	Yes	It indicates a logically isolated network space in Tencent Cloud. A VPC consists of at least one subnet. The system will provide a default VPC and subnet for you in each region. If the existing VPCs/subnets don't meet your requirements, you can create new ones as instructed in Creating VPCs or Creating Subnets.
Product Specs	Yes	The price varies by product specs. For more information, please see Pricing.
Data Retention	Yes	The price varies by data retention period. For more information, please see

Period		Pricing.
Instance Name	Yes	Enter a custom name of the TMP instance.
Grafana/Grafana Password	Yes	Grafana is enabled by default. After the instance is successfully created, the system will generate a domain name accessible over the public network. The default account of the Grafana service is `admin`, and the password is user-defined.
Validity Period	Yes	Multiple validity periods are available for your choice.

3. After completing the configuration, click **Buy Now**.



Searching for Instance

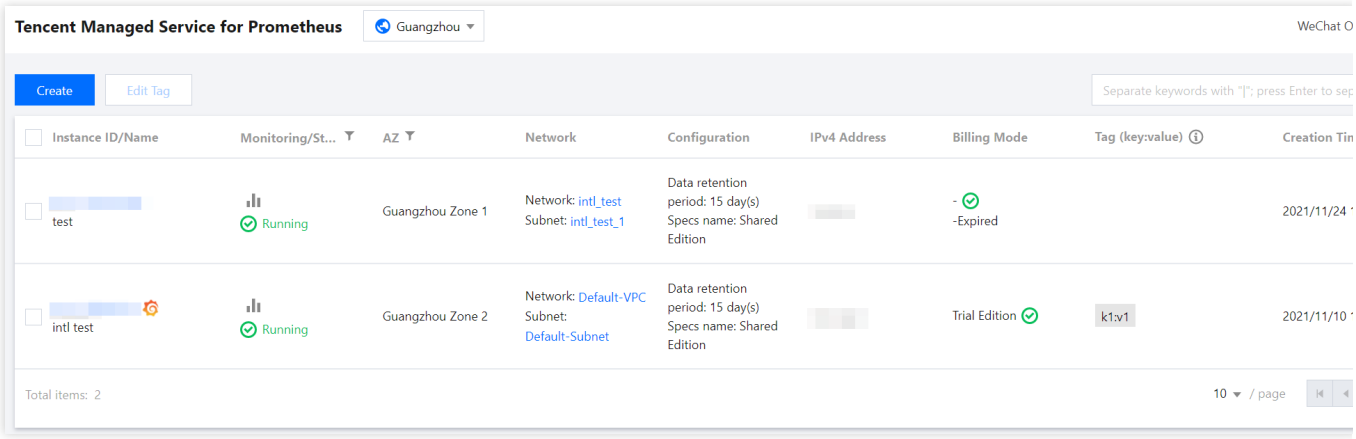
Last updated：2024-01-29 16:01:55

By default, TMP instances in the current region are displayed in the TMP console. To help you quickly find the instances in the current region, Tencent Cloud provides the search feature and allows you to filter instances by resource attributes such as instance ID, name, status, AZ, IPv4 address, and tag.

Directions

1. Log in to the TMP console.
2. Enter the conditions in the search box as needed and click

.



3. You can filter instances by different conditions. Currently, the following dimensions are supported:

Filter Condition	Description
Instance ID	You can directly enter multiple instance IDs for quick filtering. Each instance ID supports only exact match-based filtering.
Instance name	You can enter only one instance names for filtering. Fuzzy match-based filtering is supported.
Instance status	You can enter multiple status values for filtering. You can also configure the list header for quick filtering.
AZ	You can enter multiple AZs for filtering. After the region is switched, the corresponding AZs in the region will be displayed. You can also configure the list header for quick filtering.

IPv4 address	You can enter multiple IPv4 addresses for filtering. Each IPv4 address supports only exact match-based filtering.
Tag	You can enter multiple tags to filter instances. You can also directly click a tag value in the instance list for filtering.

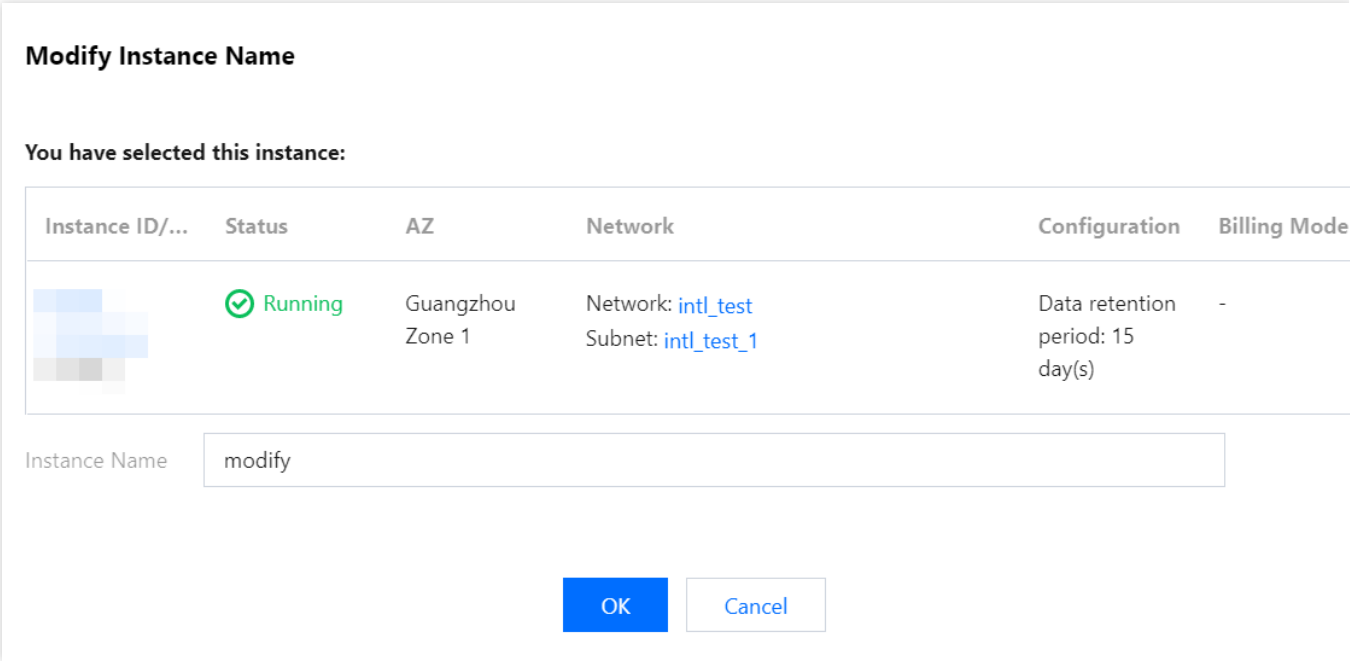
Renaming Instance

Last updated：2024-01-29 16:01:55

To help you quickly identify TMP instance names for easier instance management in the TMP console, Tencent Cloud allows you to name all instances and rename them at any time with immediate effect.

Directions

1. Log in to the TMP console.
2. In the instance list, select the TMP instance to be renamed and click **More** > **Instance Configuration** > **Rename** on the right.
3. In the **Rename** window that pops up, enter the new instance name and click **OK**.



Terminating Instance

Last updated：2024-01-29 16:01:56

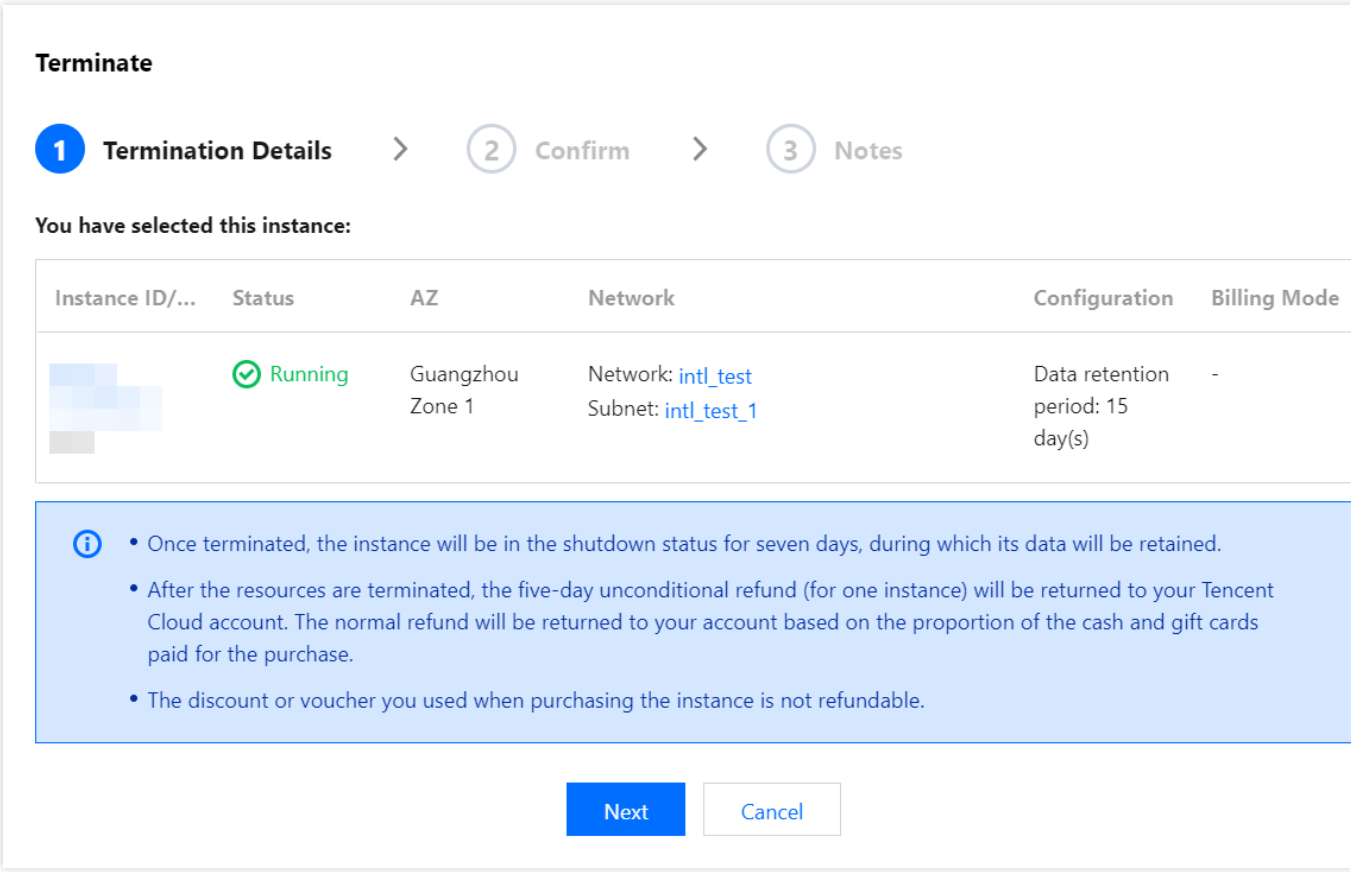
If you no longer use an instance, you can terminate it, and it will be suspended once terminated. You can reboot suspended instances according to different scenarios and needs.

Relevant Impact

Once an instance is suspended, its data will be affected as follows:
IP: the corresponding IP address is retained but does not provide normal services.
Grafana: Grafana cannot be accessed at the corresponding domain name.
Data: data on the corresponding instance will be retained for a certain number of days. The specific number displayed in the confirmation information during instance termination in the console shall prevail.

Directions

1. Log in to the TMP console.
2. In the instance list, select the TMP instance to be terminated and click **More** > **Instance Status** > **Terminate** on the right.
3. As termination is a high-risk operation, in the **Terminate** window that pops up, complete the termination steps as prompted and click **OK**.



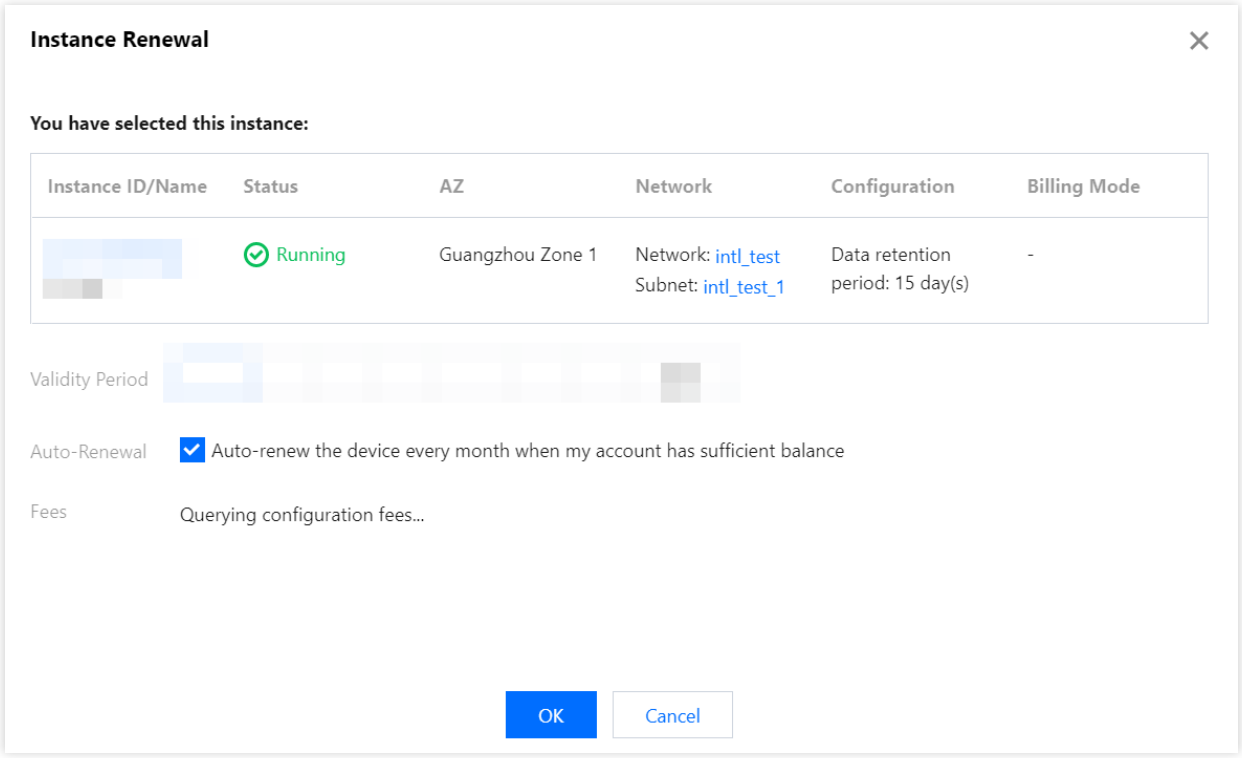
Rebooting Instance

Last updated：2024-01-29 16:01:55

You can reboot a suspended or abnormal instance in the console.

Directions

1. Log in to the TMP console.
2. In the instance list, select the TMP instance to be rebooted and click **More** > **Instance Status** > **Reboot** on the right.
3. In the **Reboot Notes** window that pops up, click **OK**.



Note:

If you reboot a running instance, the service will be interrupted during reboot. Please confirm the operation risks first.

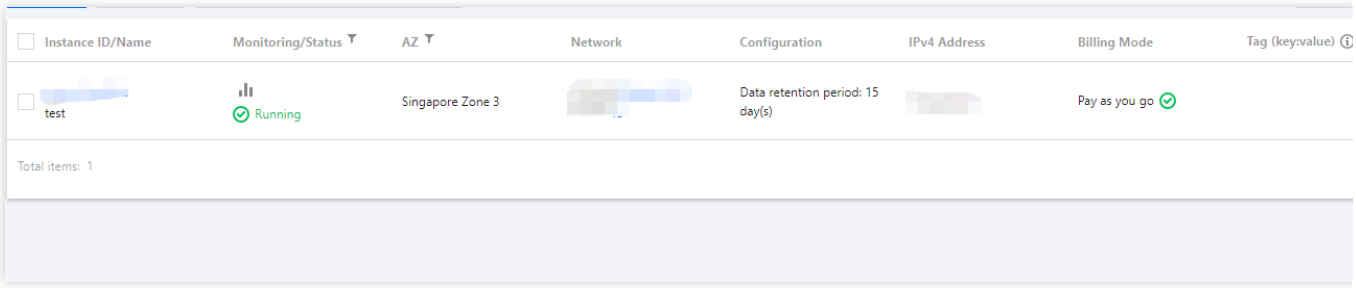
Modifying Storage Period

Last updated：2024-01-29 16:01:55

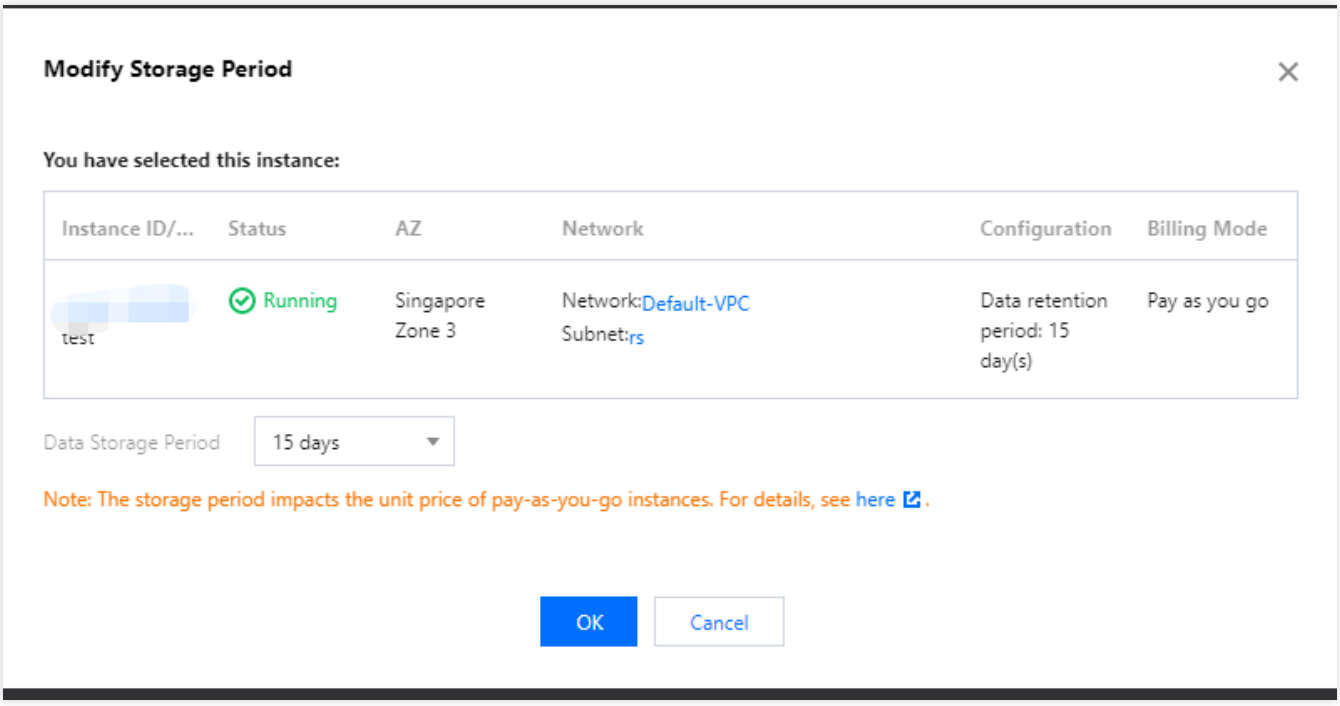
TMP allows you to modify the data storage period after creating an instance. The longer the storage period, the higher the unit price of the instance. For billing details, see Pay-as-You-Go.

Directions

1. Log in to the TMP console.
2. In the instance list, find the target TMP instance and click **More** > **Instance Configuration** > **Modify Storage Period** on the right.



3. In the pop-up window, select the target storage period and click **OK**.



Note:

After the modification, newly collected data will be billed based on the new storage period and unit price from 0:00 AM the next day.

Historical data is still stored based on the storage period configured before the modification.

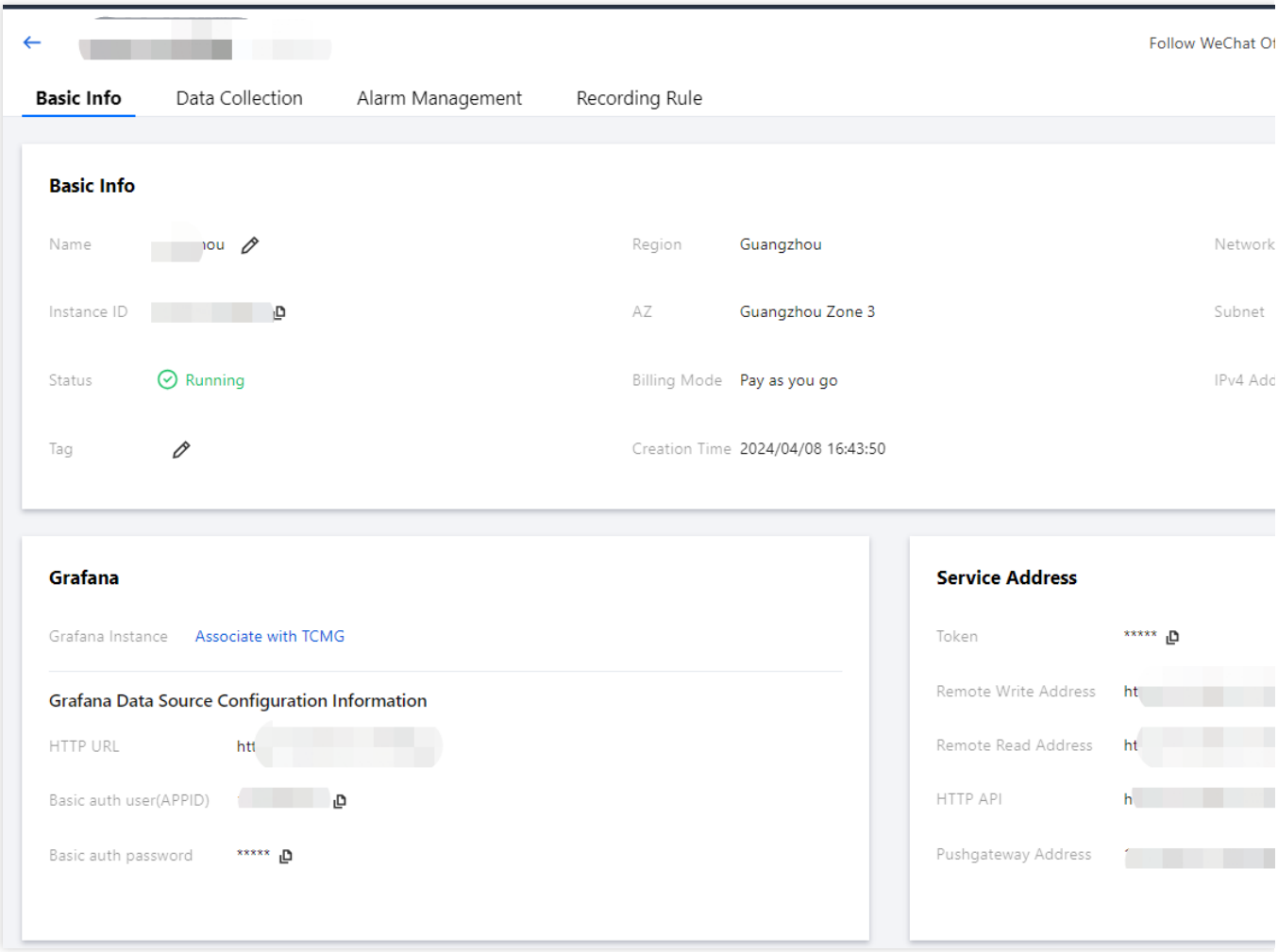
Viewing Instance's Basic Information

Last updated：2024-07-24 18:10:07

To help you view the basic information of TMP instances, TMP allows you to select a desired instance on the instance list page to enter its management page and view its basic information.

Directions

1. Log in to the TMP console.
2. In the instance list, select the TMP instance to be viewed and click its **instance ID** or **Manage** on the right.



3. You can perform the following operations on the basic information page:

Rename the instance.

Edit instance tags.

Change the Grafana Admin password.

Upgrade preset Grafana dashboards.

TKE

Integration with TKE

Last updated：2024-01-29 16:01:55

You can monitor the TKE service in business scenarios after integrating it. This document describes how to integrate the TKE service.

Tencent Kubernetes Engine (TKE) provides container-centric solutions based on native Kubernetes. It can solve environment-related issues in the process of development, testing, and Ops, helping you reduce costs and improve efficiency. Kubernetes is an open-source container orchestration tool developed by Google and has been used by Google for more than 15 years. As the de facto standard in the container field, Kubernetes can greatly simplify the management and deployment complexity of applications. By integrating with the TKE service, you can monitor the status of Kubernetes and the services running on it much more easily through Prometheus.

Note

In order to ensure normal operation, existing instances will automatically update the component version when you edit collection configuration and associate new clusters. During the update process, data breakpoints may occur in the associated clusters.

Directions

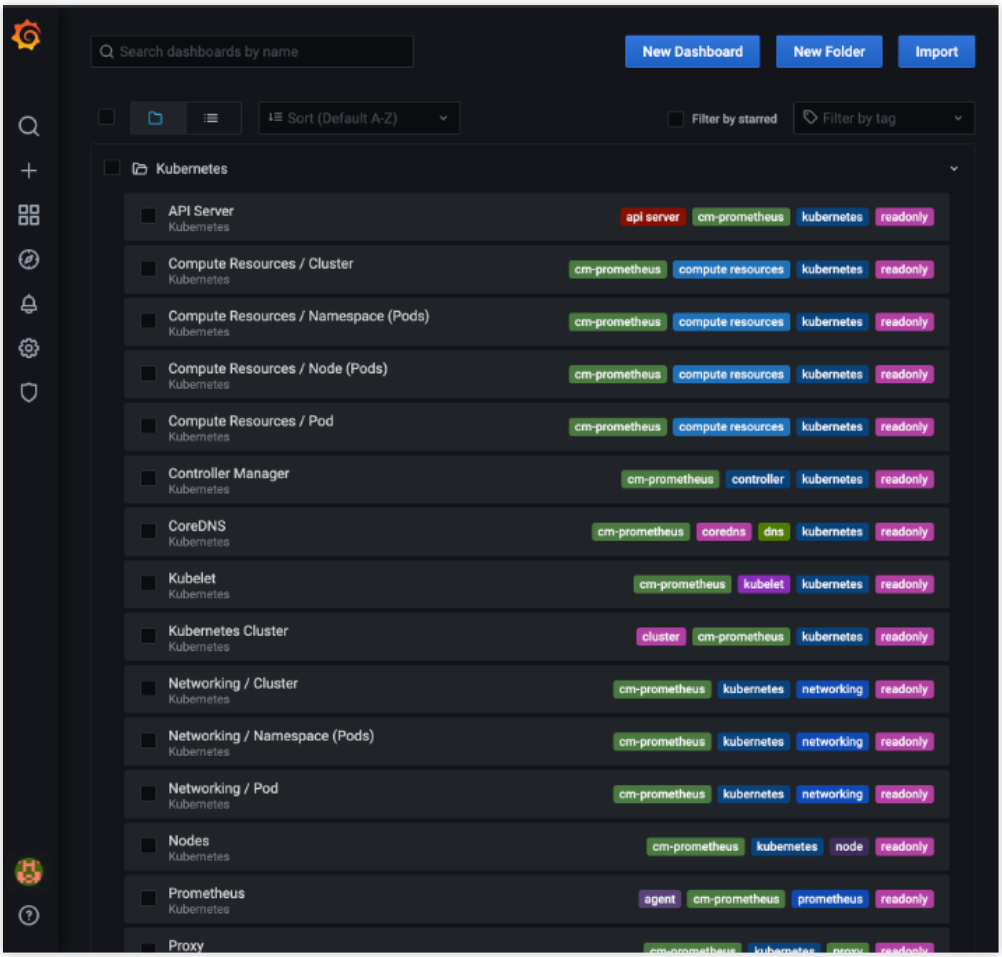
1. Log in to the TMP console.
2. In the TMP instance list, click the **ID/Name** of the newly created instance.
3. Go to the TMP management center and click **Integrate with TKE** on the left sidebar.
4. Perform the following operations on the cluster monitoring page:

Associate a cluster: Associate a cluster with a TMP instance as instructed in Associating with Cluster.

Configure data collection: Create a data collection rule to monitor your business data by adding the configuration in the console or via a YAML file. For more information, see Data Collection Configurations.

Streamline basic monitoring metrics: Select only the required metrics to avoid unnecessary fees as instructed Streamlining Monitoring Metrics.

5. After completing the above operations, you can view the monitoring data of your TKE service in Grafana.



Integration Center

Last updated：2024-01-29 16:01:55

TMP integrates commonly used programming languages, middleware, big data, and infrastructure databases. It supports quick installation and custom installation. You only need to follow the instructions to monitor the corresponding components. It also provides out-of-the-box Grafana monitoring dashboards. The integration center covers three major monitoring scenarios of basic service monitoring, application layer monitoring, and TKE cluster monitoring, making it easier for you to integrate and use.

List of Supported Services

Service Type	Service	Monitoring Metric	Quick Installation	Integration Guide
Big data	Elasticsearch	Cluster/Index/Node monitoring	Supported	Elasticsearch Exporter Integration
	Flink	Cluster/Job/Task monitoring	Not supported	Flink Integration
Development	CVM	The extended `cvm_sd_config` can be used to configure a CVM scrape task and collect Node Exporter or custom business metrics.	Supported	CVM Node Exporter
	Go	GC/Heap/Thread/Goroutine monitoring	Not supported	Go Application Integration
	JVM	Heap/Thread/GC/CPU/File monitoring	Not supported	JVM Integration
	Spring MVC	HTTP API/Exception/JVM monitoring	Not supported	Spring Boot Integration
Middleware	Kafka	Broker/Topic/Consumer group monitoring	Supported	Kafka Exporter Integration
	Consul	Consul monitoring	Supported	Consul Exporter Integration

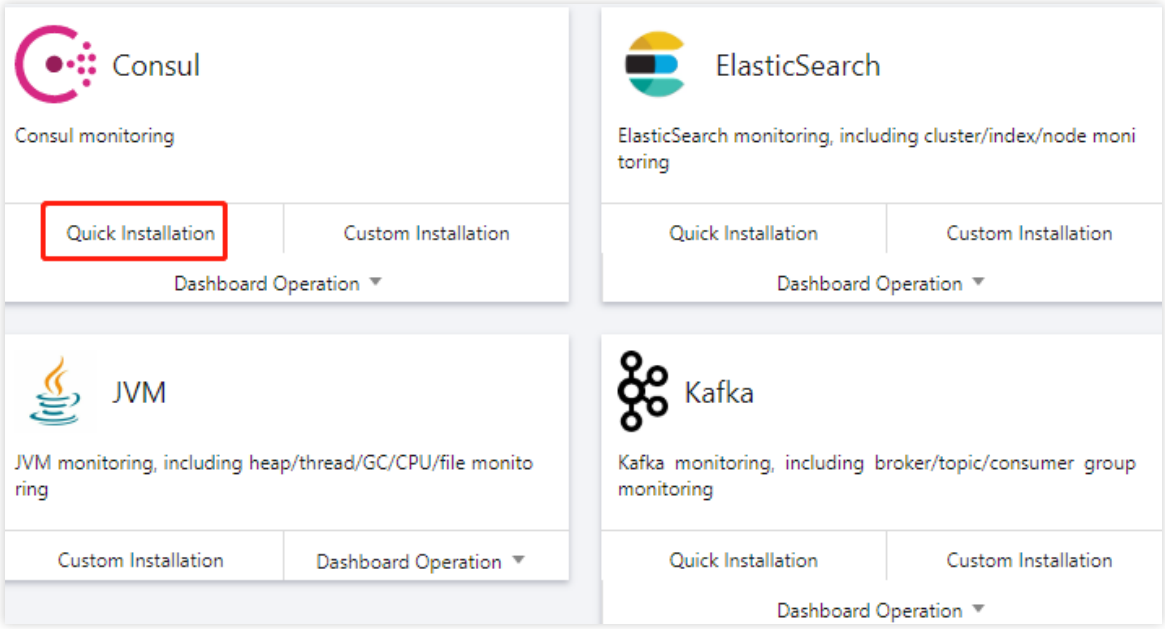
	Etcd	Etcd monitoring	Not supported	-
	Istio	Istio monitoring	Not supported	-
Infrastructure	Kubernetes	API server/DNS/Workload/Network monitoring	Supported	Agent Management
Database	TencentDB for MongoDB	File count/Read and write performance/Network traffic monitoring	Supported	MongoDB Exporter Integration
	TencentDB for MySQL	Network/Connection count/Slow query monitoring	Supported	MySQL Exporter Integration
	TencentDB for PostgreSQL	CPU/Memory/Transaction/Lock/Read/Write monitoring	Supported	PostgreSQL Exporter Integration
	TencentDB for Redis	Memory utilization/Connection count/Command execution status monitoring	Supported	Redis Exporter Integration
	TencentDB for Memcached	Memcached monitoring	Supported	Memcached Exporter Integration
Inspection	Health check	Blackbox can be used to regularly test the connectivity of the target service, helping you stay up to date with the service health and discover exceptions in time.	Supported	Health Check
CM	CM	Tencent Cloud service monitoring	Supported	-
Custom	Scrape task	The native `static_config` can be used to configure a scrape task.	Supported	Scrape Configuration Description
	CVM scrape task	The extended `cvm_sd_config` can be used to configure a CVM scrape task.	Supported	Scrape Configuration Description

Directions

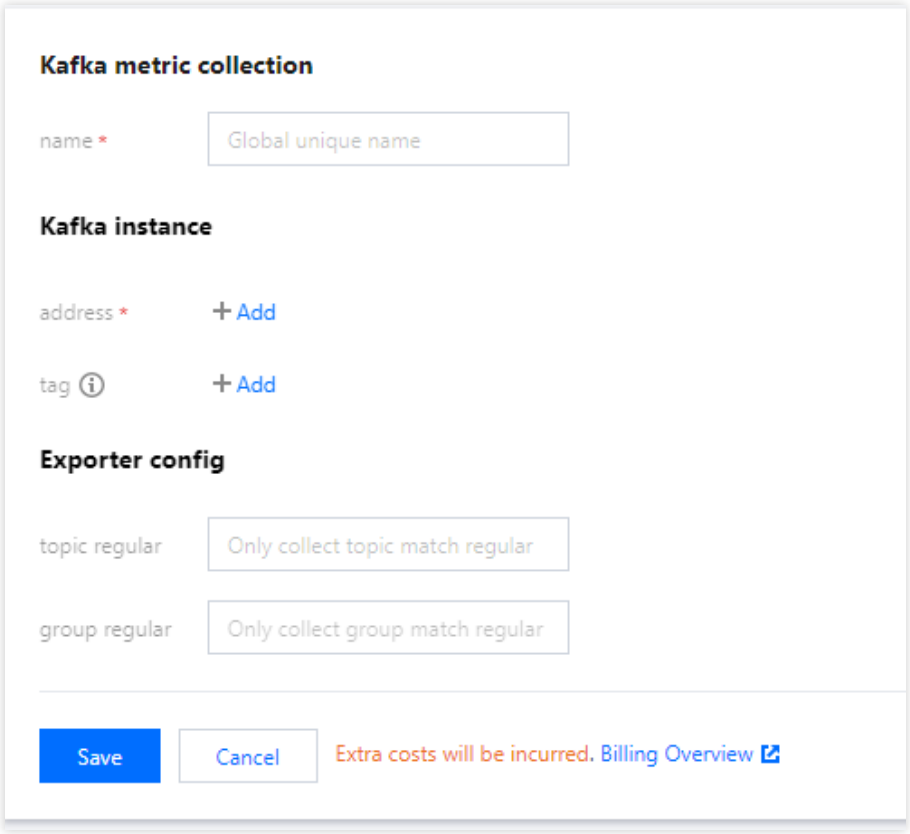
Quick installation

Some services support quick agent installation. For more information, see Integration Center > List of Supported Services.

1. Log in to the TMP console.
2. In the instance list, select the corresponding TMP instance.
3. Enter the instance details page and click **Integration Center**.
4. In the **Integration Center**, select the service that supports quick installation and click **Install** in the bottom-left corner.



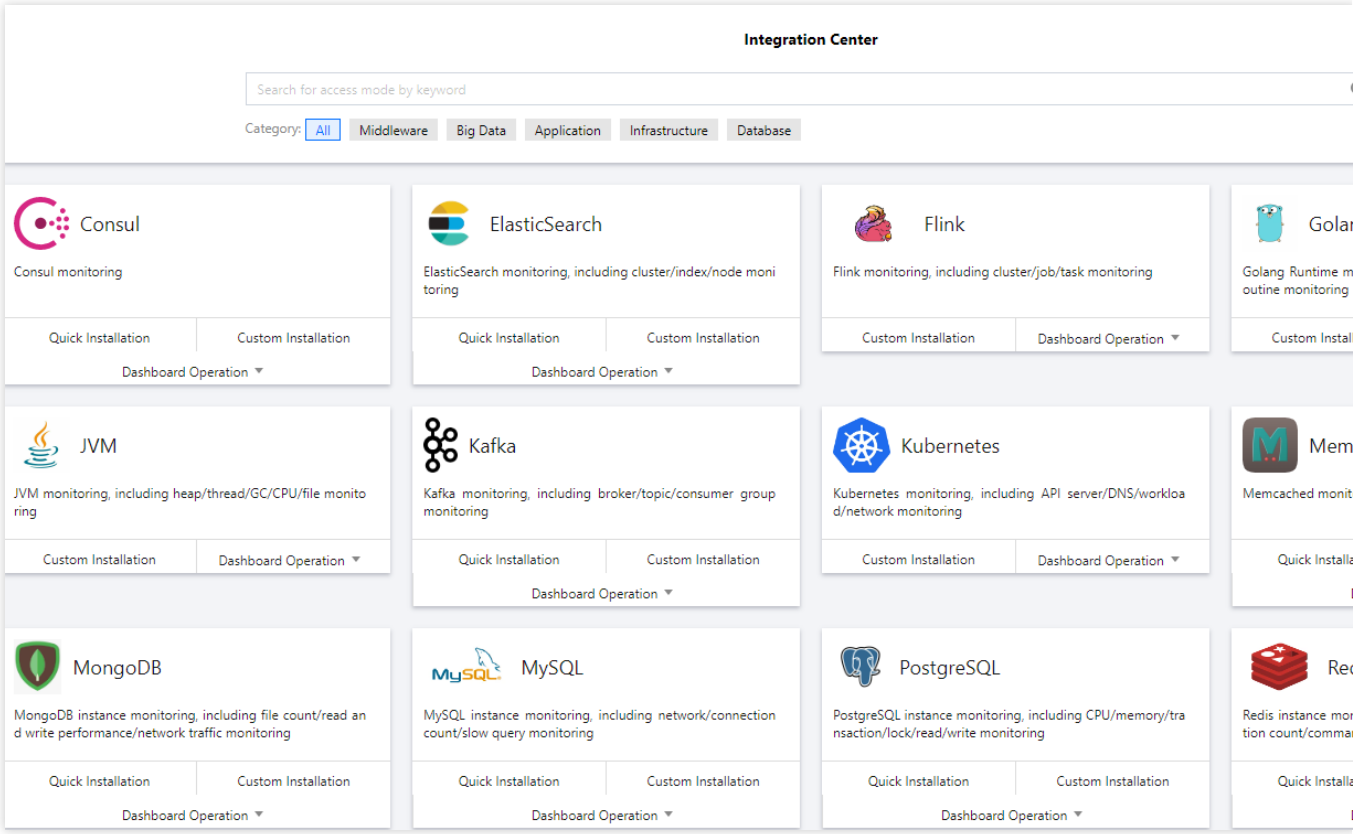
5. On the **Integration List** page, enter the metric collection name and address and click **Save**. Below is a sample for Kafka:



Custom installation

1. Log in to the TMP console.
2. In the instance list, select the corresponding TMP instance.
3. Enter the instance details page and click **Integration Center**.
4. Select the target service in the integration center. You can click **Integration Guide** to view the integration guide.

After successful integration, you can monitor the corresponding service in real time. You can also click **Install/Upgrade** in **Dashboard Operation** to install or upgrade the Grafana dashboard for the service.



Data Multi-Write

Last updated：2024-08-15 17:08:56

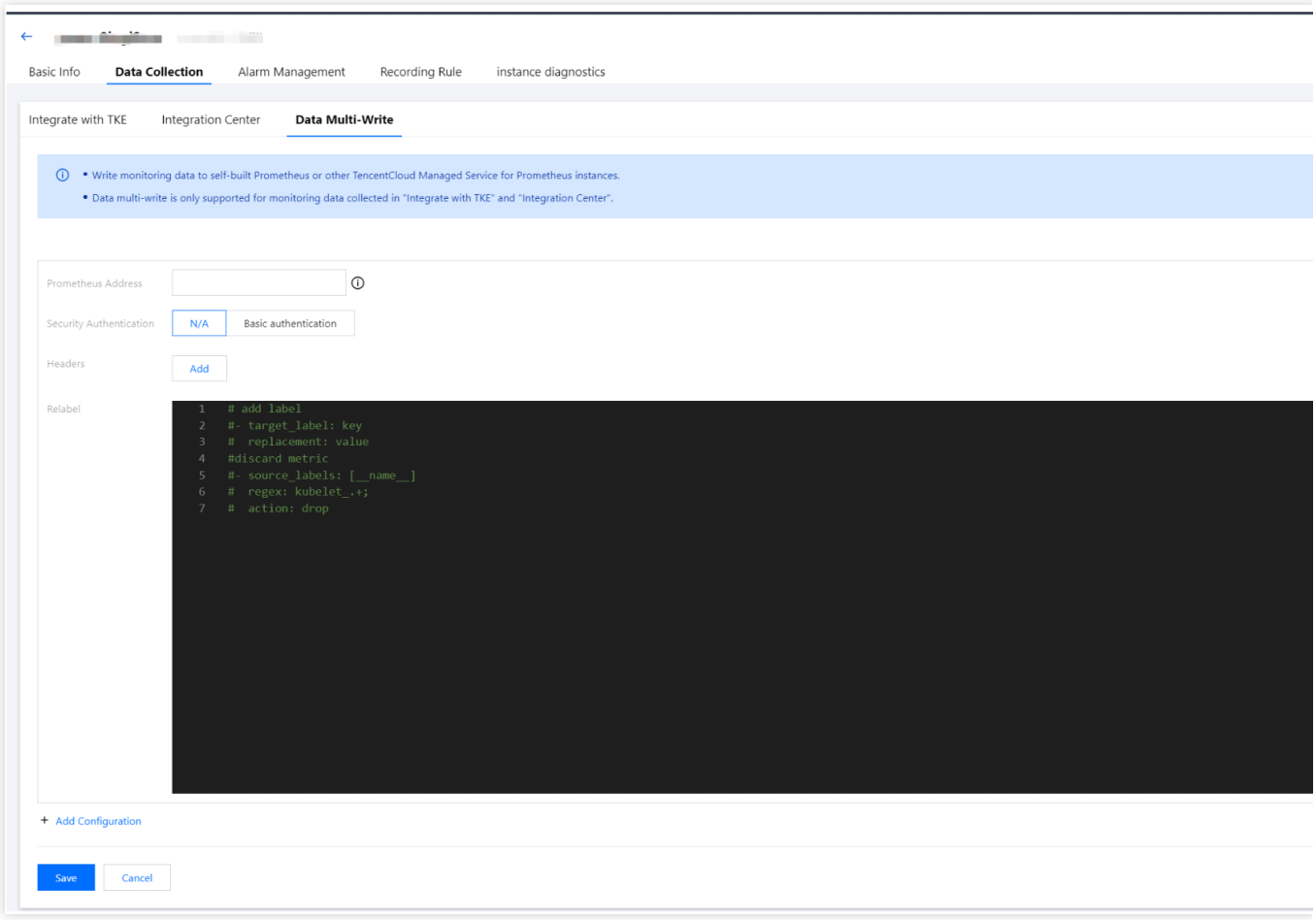
The data multi-write feature supports writing monitoring data to self-built Prometheus or other Prometheus monitoring instances.

Note:

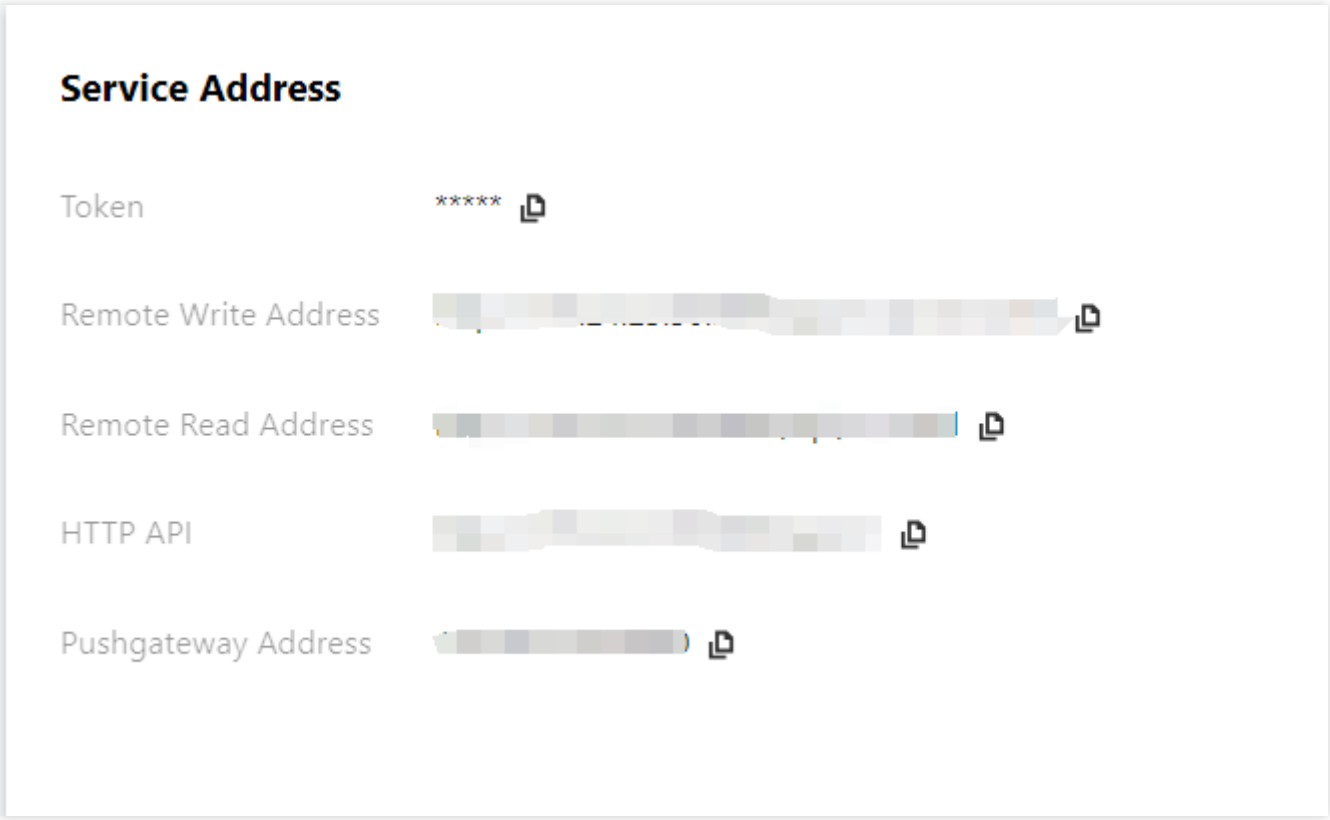
Currently, only the multi-write of monitoring data collected by Integrated Container Service and Integrated Center is supported, and others are not supported yet.

Operation Guide

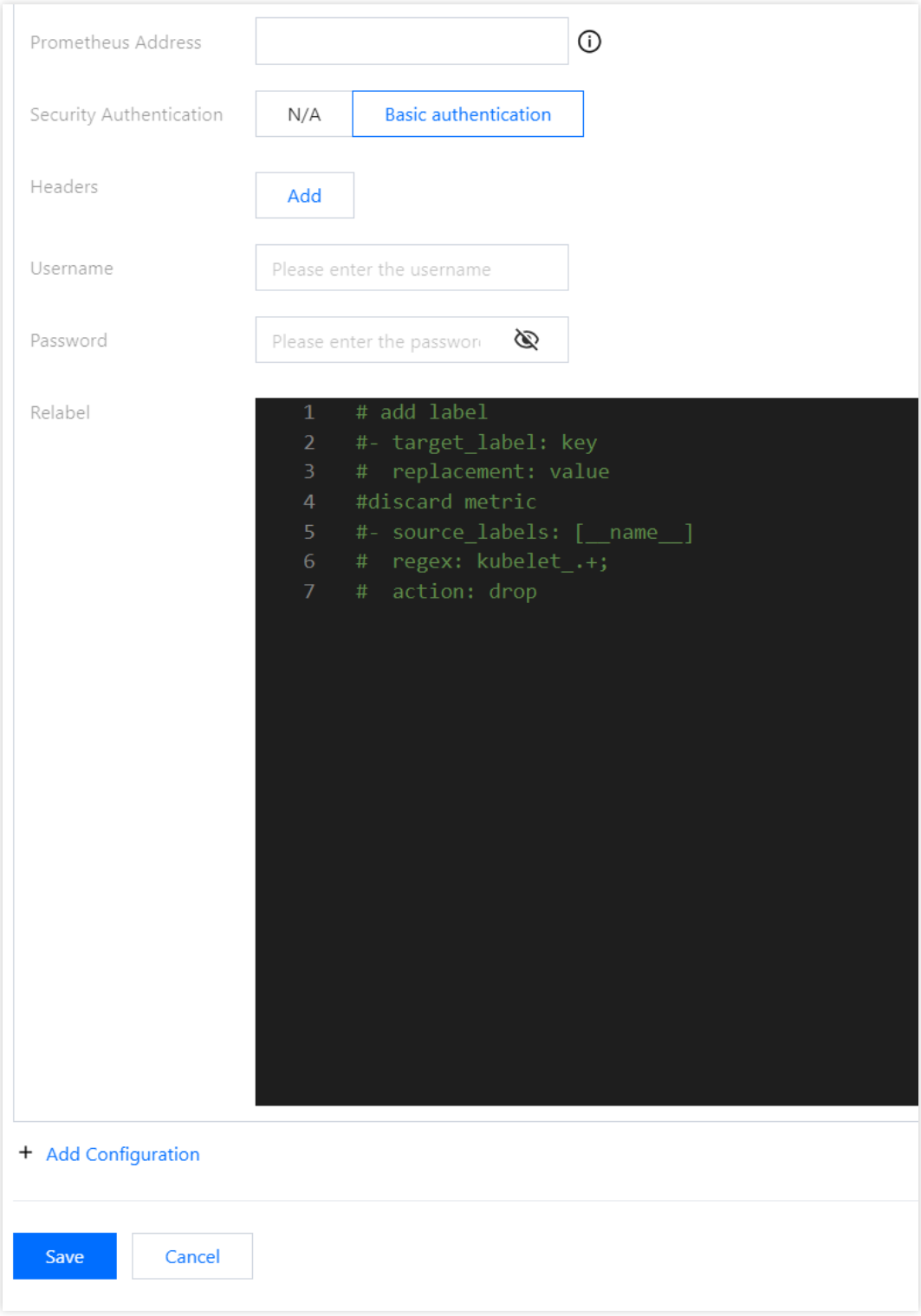
1. Log in to TMP Console.
2. In the left sidebar, click **TencentCloud Managed Service for Prometheus.**
3. Click the corresponding instance name to enter the instance details page, then click **Data Collection** > **Data Multi-Write** on the top navigation bar.
4. In the Data Multi-Write sub-window, click **Add** and see the following description to configure the data multi-write information.



Prometheus address: Obtain the HTTP API address from the basic information of instances.



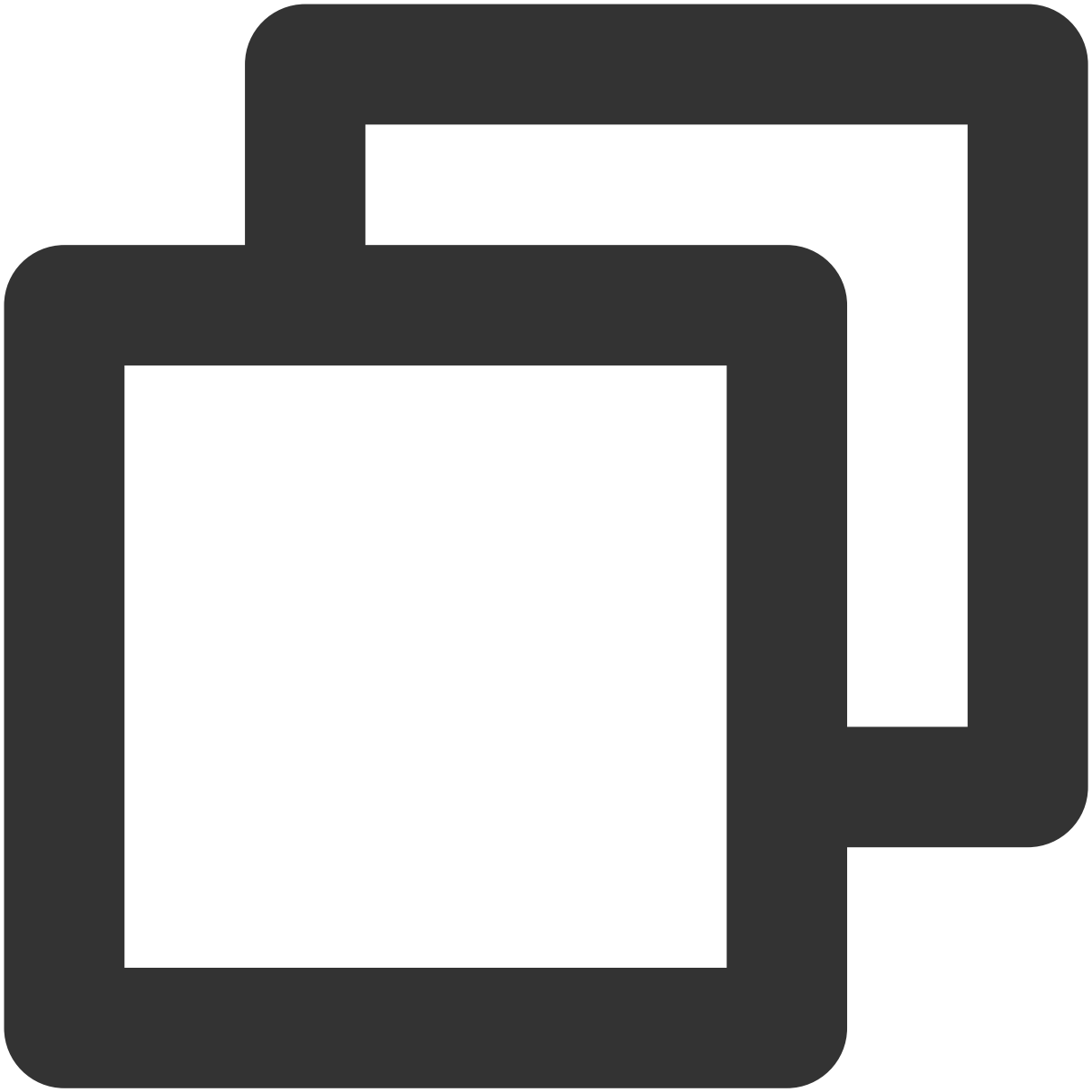
Security verification: Whether to enable security authentication. Once enabled, the username and password needs to be customized.



Recording Rule Overview

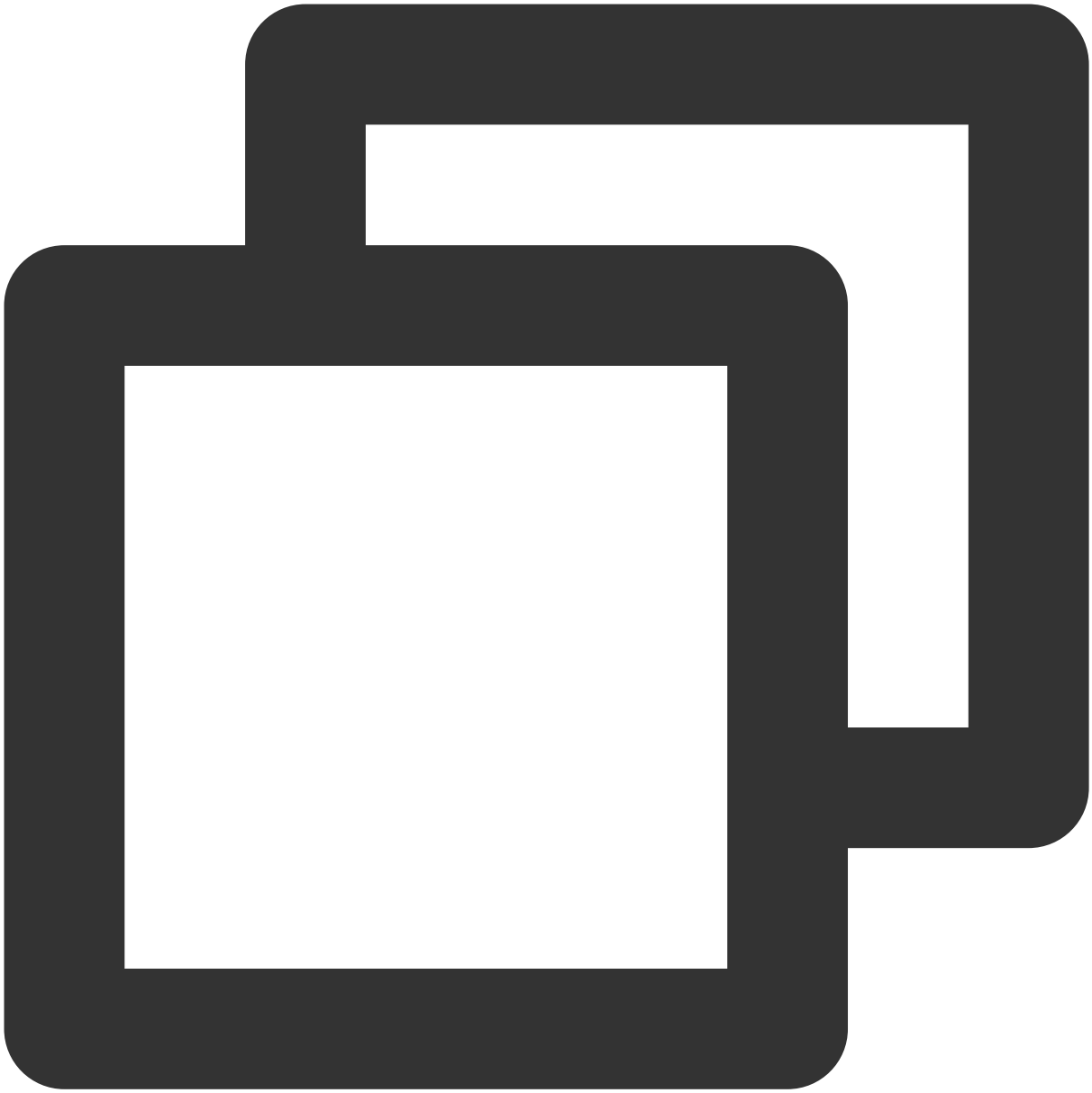
Last updated：2024-01-29 16:01:55

A recording rule allows you to calculate some commonly used or complex metrics in advance and then store the calculated data in new data metrics. In this way, querying the calculated data will be faster and easier than querying the original data. This is very suitable for dashboard scenarios and can solve the problems of complicated user configuration and slow query.
Recording rules exist in the form of rule group, and rules in the same group are executed sequentially at a certain interval. Rule names must conform to the corresponding Prometheus specification.
Generally, a rule file is as follows:



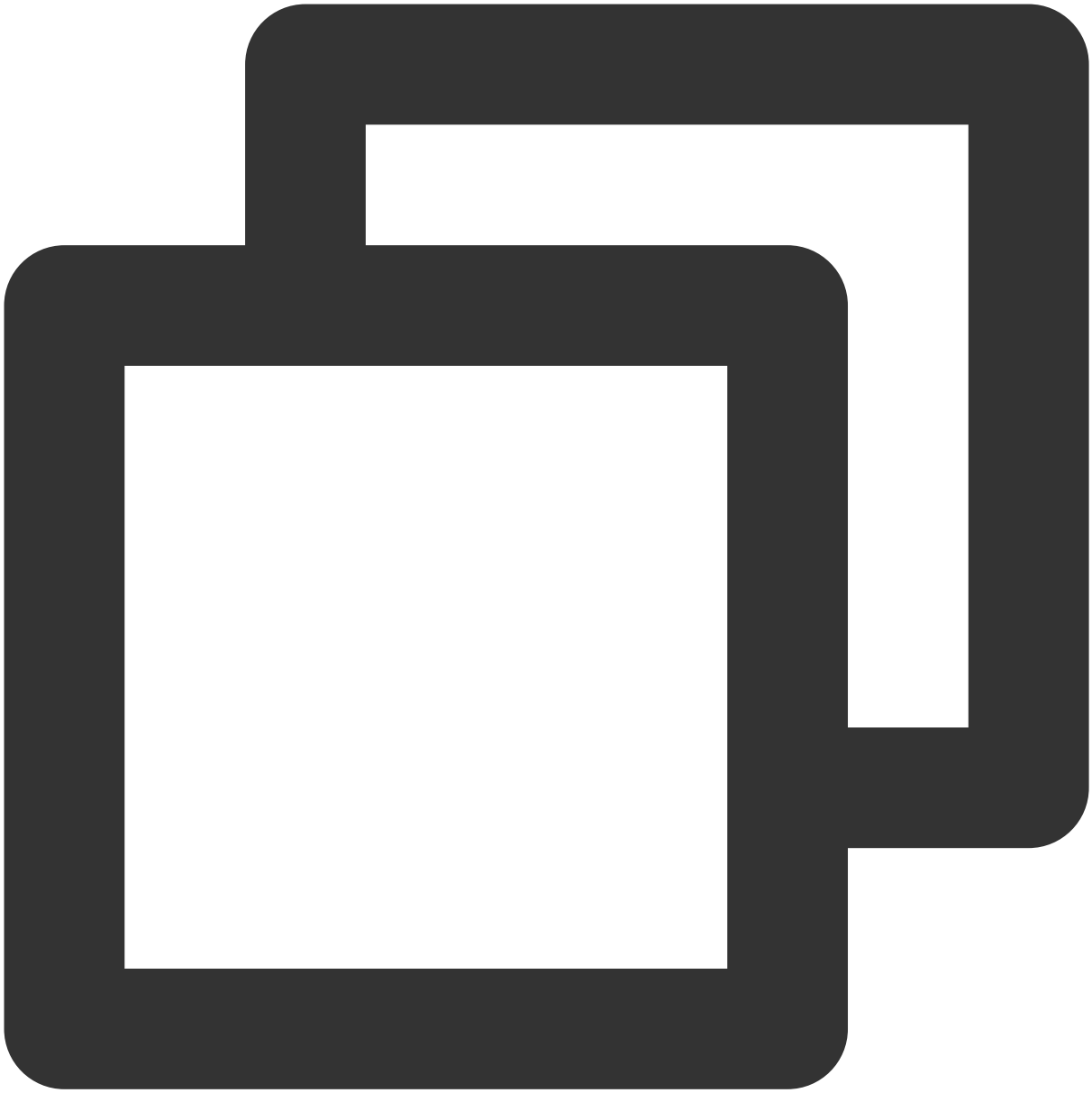
```
groups:
  [ - <rule_group> ]
```

Below is a simple example of recording rule:



```
groups:
  - name: example
    rules:
    - record: job:http_inprogress_requests:sum
      expr: sum by (job) (http_inprogress_requests)
```

Rule Group



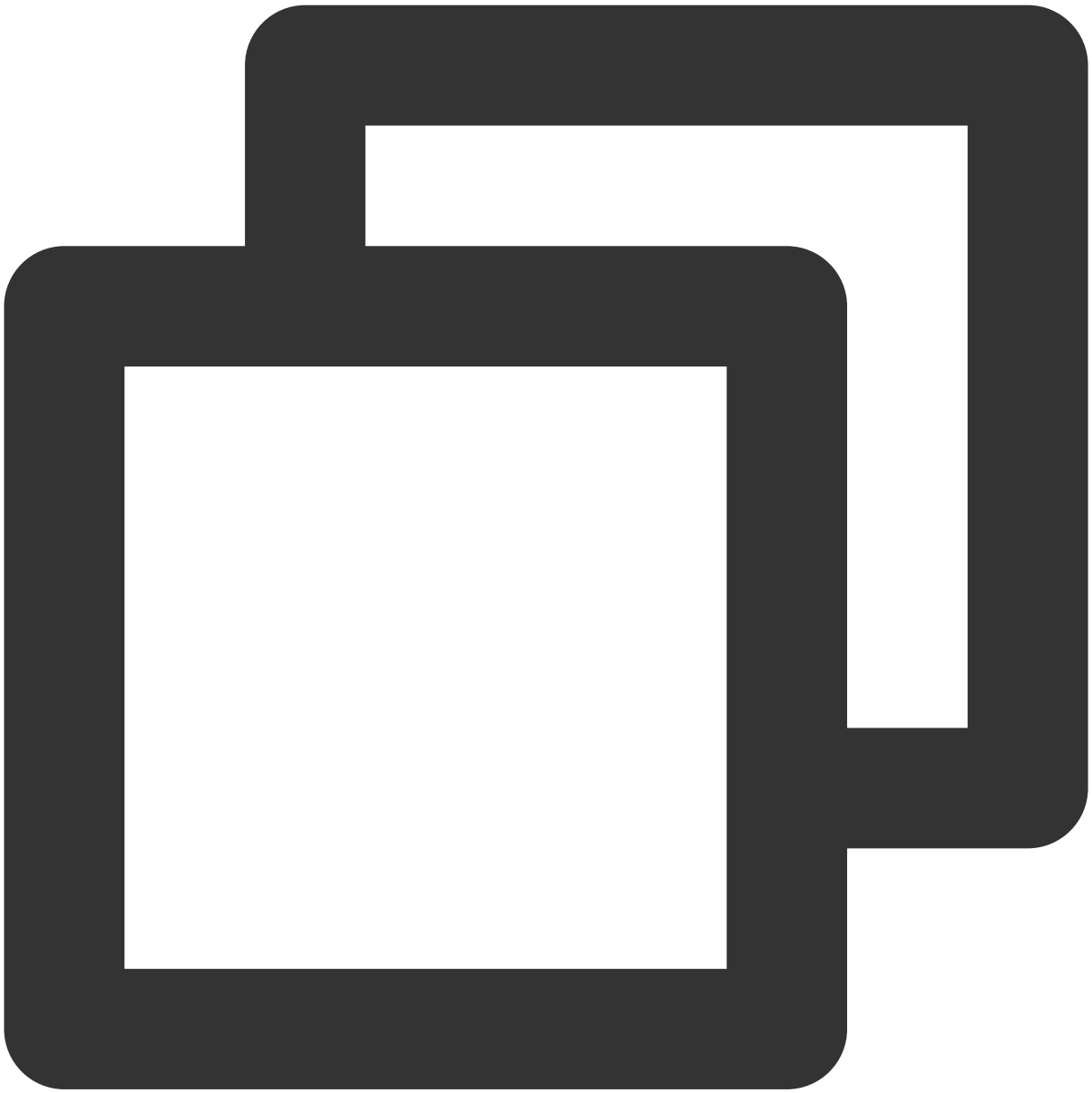
```
# A rule group name must be unique in the same file
name: <string>

# Rule detection interval
[ interval: <duration> | default = global.evaluation_interval ]

rules:
  [ - <rule> ... ]
```

Rule

The recording rule syntax is as follows:



```
# The generated new metric name, which must be valid
record: <string>

# PromQL expression. Each calculated data entry will be stored in the new metric na
expr: <string>

# The label to be added or overwritten in the data to be stored
```

```
labels:
  [ <labelname>: <labelvalue> ]
```

Recommended Name Format

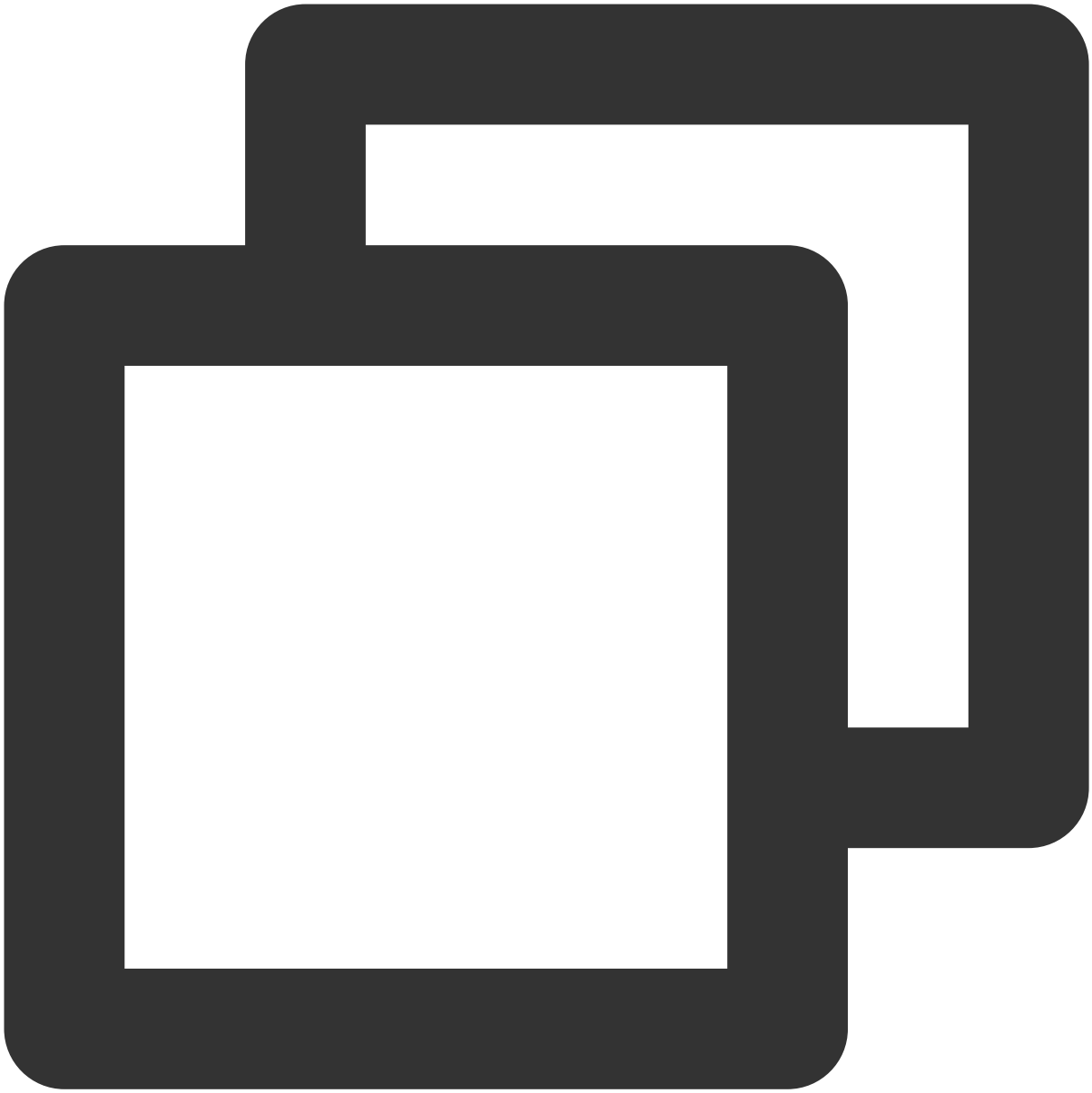
The recommended format for naming recording rules is `level:metric:operations` .

level: indicates the recording level and the output label of the rule.

metric: indicates the metric name.

operations: indicates the list of operations applied to the metric.

Example:



```
- record: instance_path:requests:rate5m
  expr: rate(requests_total{job="myjob"}[5m])

- record: path:requests:rate5m
  expr: sum without (instance)(instance_path:requests:rate5m{job="myjob"})
```

Rule Management

Last updated：2024-01-29 16:01:55

Overview

You can manage the TMP recording rules in the TMP console to avoid the hassle of having to modify the configuration file in native Prometheus.

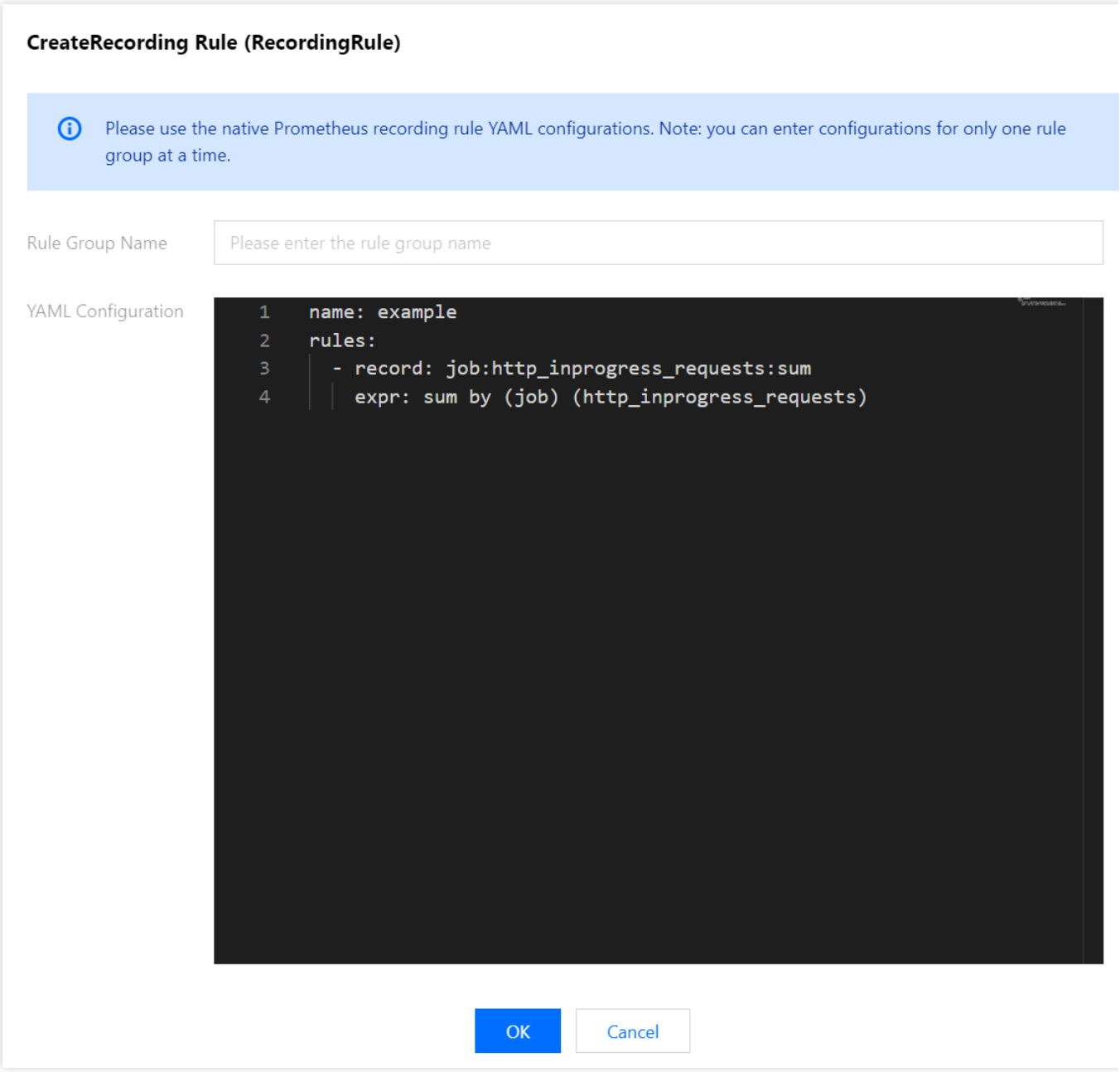
Preparations

1. Log in to the TMP console.
2. Create a TMP instance as instructed in Creating Instance.
3. Enter the TMP instance management page through the instance list.
4. Manage recording rules as instructed in Overview.

Directions

Creating rule

1. In the menu on the left of the instance management page, click **Recording Rule** > **Create** to enter the rule creation page, adjust the rule expression and the name of the new metric to be recorded according to your actual needs as shown below. For specific terms, please see Overview.



2. Click **OK**.

Managing rule

In the rule list, you can temporarily **disable** rules or enable rules that are **not enabled**. Once disabled, a rule will stop working, and the collection of related recording metrics will also stop.

Deleting rule

1. You can delete rules that are no longer used.
2. Select the rule to be deleted in the list and confirm in the pop-up window. Once deleted, a rule will stop working.

List of Default Recording Rules

Last updated：2024-01-29 16:01:56

Recording rules are created for associated clusters by default. For free metrics in pay-as-you-go mode, the following new metrics will be created after recording and will be billed normally. If you don't need them for data collection, you can disable the default recording rules when you associate the cluster for the first time, or later in the recording rule list on the recording page.

Metric	Preset Dashboard	A F T
:node_memory_MemAvailable_bytes:sum	Kubernetes / Compute Resources / Cluster	-
node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate	Kubernetes / Compute Resources / ClusterKubernetes / Compute Resources / WorkloadKubernetes / Compute Resources / PodKubernetes / Compute Resources / Node (Pods)Kubernetes / Compute Resources / Namespace (Workloads)Kubernetes / Compute Resources / Namespace (Pods)Kubernetes / Networking / WorkloadKubernetes / Networking / Namespace (Workload)	-
node_namespace_pod_container:container_memory_working_set_bytes		-
node_namespace_pod_container:container_memory_rss		-
node_namespace_pod_container:container_memory_cache		-
node_namespace_pod_container:container_memory_swap		-
namespace_workload_pod:kube_pod_owner:relabel		-
namespace:kube_pod_container_resource_requests_memory_bytes:sum	-	K re
namespace:kube_pod_container_resource_requests_cpu_cores:sum	-	K re

Alerting Rule Overview

Last updated：2024-01-29 16:01:56

TMP allows you to define alert conditions based on Prometheus expressions. Once a metric reaches an alert condition, you will receive notifications through email and SMS, so you can take corresponding measures promptly.
Note:
Alerting in TMP combines the alarming capabilities of Cloud Monitor and the alerting capabilities of the open-source Prometheus ecosystem, making alerting more accurate and reasonable.

Features

TMP supports Alertmanager features such as grouping and silencing to make alerts more reasonable.
You can group the metric data and regularly check and calculate the alerting rules to make alerting quicker and more convenient.
Alerting rules can be defined based on PromQL, making alerting more flexible.
Cloud Monitor's alarm notification templates can be used, which support multiple receiving channels such as email and SMS.

Components

Term	Description
Rule name	Custom alerting rule name.
Alerting rule	It contains alert trigger condition and duration and should be defined through PromQL.
Alert object	Custom alert title.
Alert message	Custom alert content.
Label	A set of specified labels to be added to the alert.
Annotation	Custom additional alert message.
Notification template	It contains the template name, notification type, recipient, receiving channel. You can define the receiving channel and grouping method.

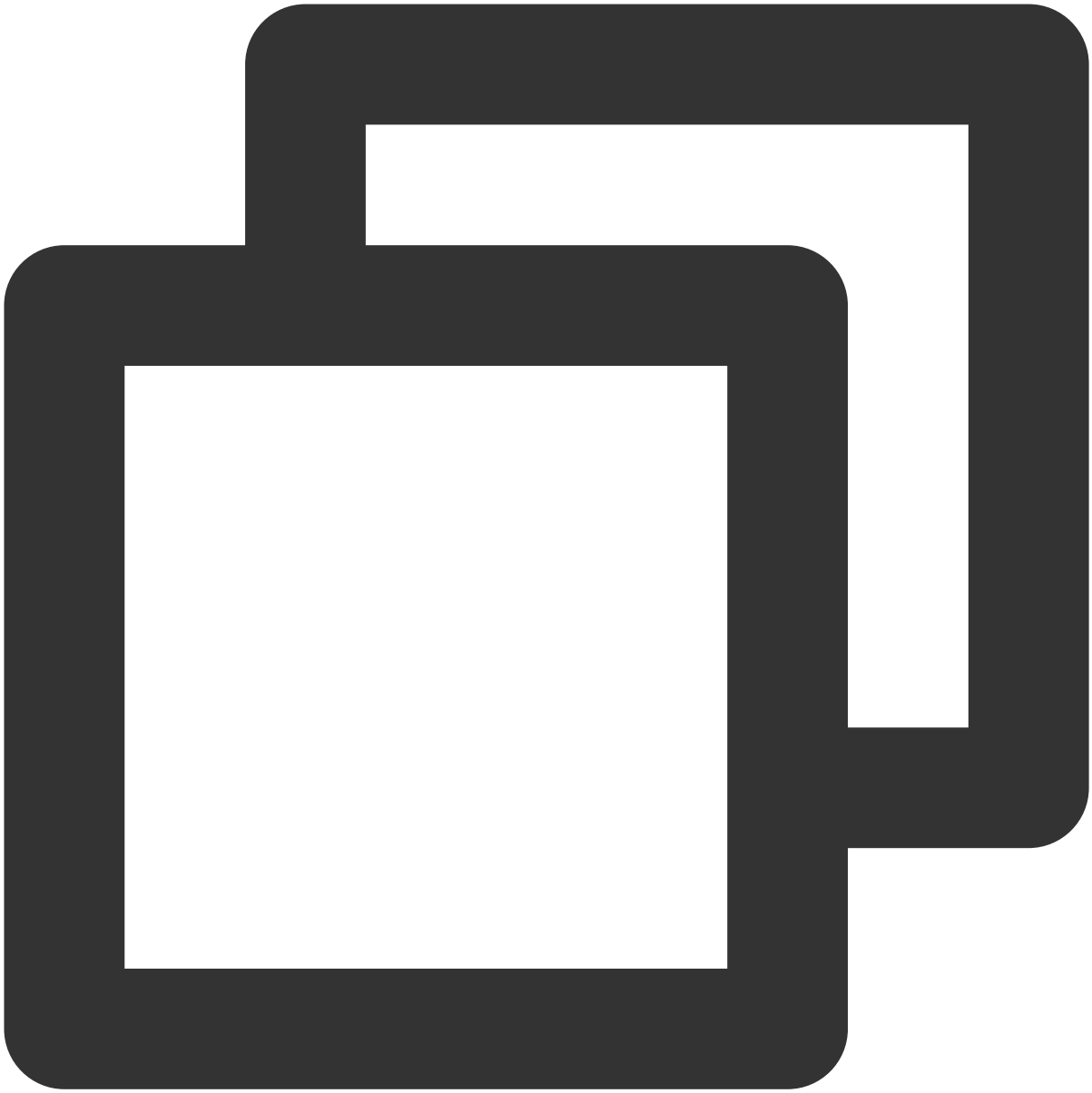
Alerting Rule Description

Last updated：2024-01-29 16:01:55

You can set alert conditions based on Prometheus expressions to monitor the service status in real time and receive prompt notifications when the service is exceptional.

Defining Alerting Rule

Defining an alerting rule in TMP is very similar to defining a recording rule. Below is a sample alerting rule:

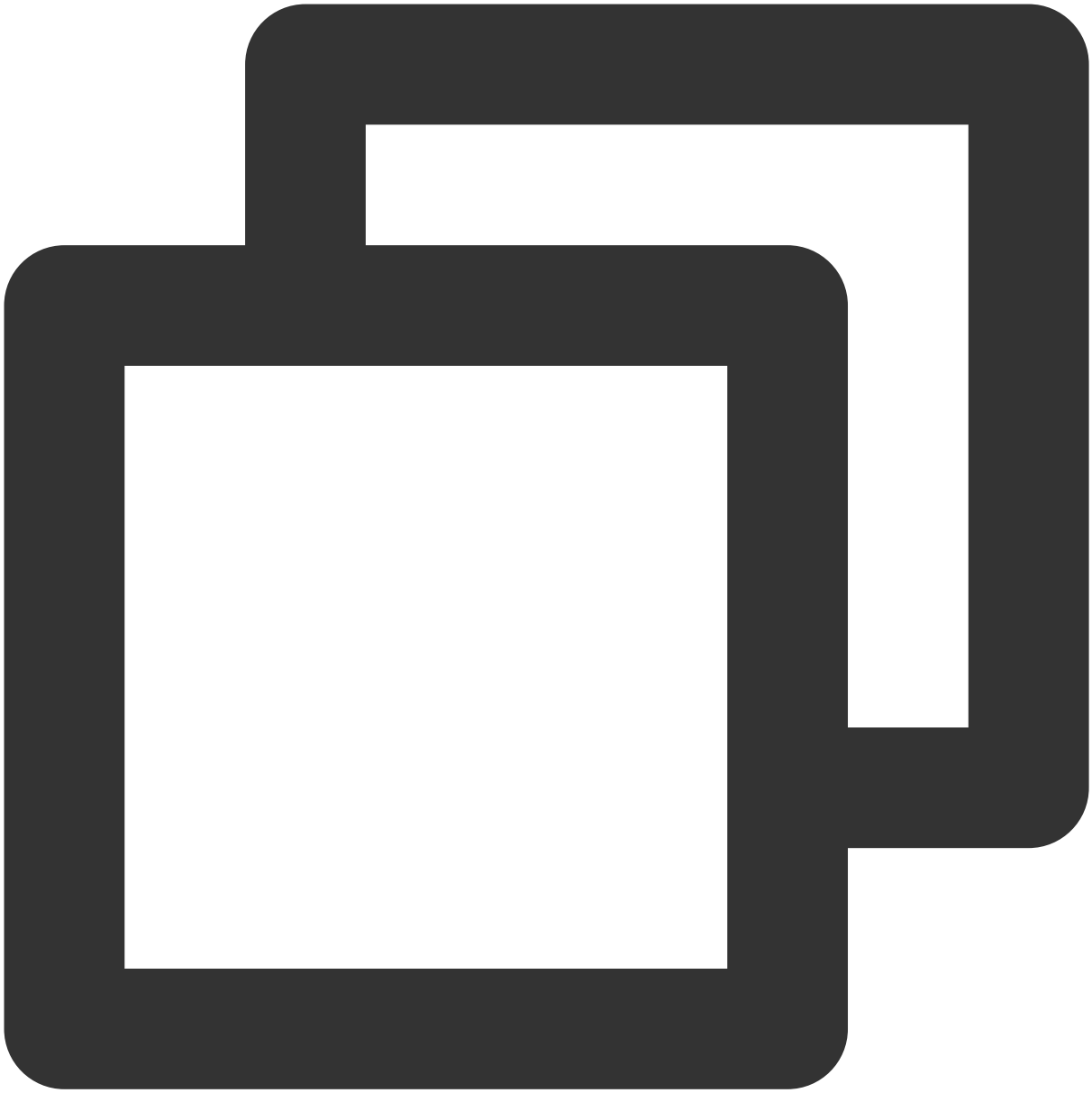


```
groups:
- name: example
  rules:
  - alert: HighRequestLatency
    expr: job:request_latency_seconds:mean5m{job="myjob"} > 0.5
    for: 10m
    labels:
      severity: page
    annotations:
      summary: High request latency
```

In an alerting rule file, you can define a set of relevant rules in the same group. In each group, you can define multiple alerting rules. A rule mainly consists of the following parts:
alert: alerting rule name.
expr: alert trigger condition based on a PromQL expression, which is used to calculate whether there is time series data meeting the condition.
for: assessment wait time, which is optional. It indicates how long a trigger condition can last before an alert is sent. New alerts generated during the wait time are in "Pending" status.
labels: custom labels, which are a set of specified labels to be added to alerts.
annotations: it is used to specify a set of additional information, such as text that describes alert details. It will be sent to Alertmanager as a parameter when an alert is generated.

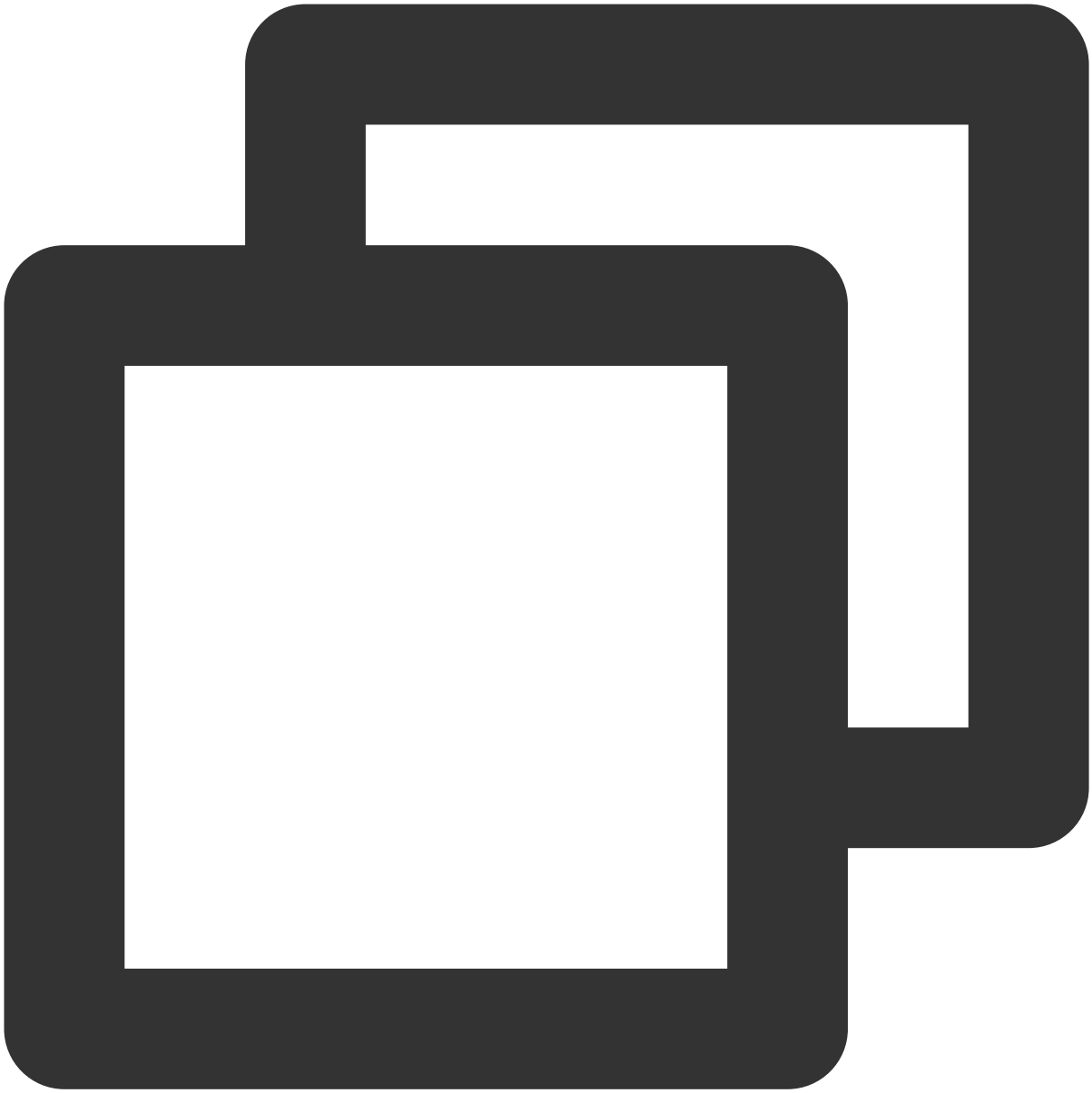
Template

Generally, `annotations` in an alerting rule file uses `summary` to describe the summary of alerts and `description` to describe alert details. In addition, Alertmanager UI will also display the alert information based on the two label values. To make the alert information more readable, TMP allows you to convert label values in `labels` and `annotations` into a template.
You can use the `$labels.<labelname>` variable to access the value of the specified label on the current alert instance and use `$value` to get the sample value calculated through the current PromQL expression.



```
# To insert a firing element's label values:
{{ $labels.<labelname> }}
# To insert the numeric expression value of the firing element:
{{ $value }}
```

For example, you can use a template to optimize the readability of the content of `summary` and `description`:



```
groups:
- name: example
  rules:

  # Alert for any instance that is unreachable for >5 minutes.
  - alert: InstanceDown
    expr: up == 0
    for: 5m
    labels:
      severity: page
    annotations:
```

```
      summary: "Instance {{ $labels.instance }} down"
      description: "{{ $labels.instance }} of job {{ $labels.job }} has been down f

  # Alert for any instance that has a median request latency >1s.
  - alert: APIHighRequestLatency
    expr: api_http_request_latencies_second{quantile="0.5"} > 1
    for: 10m
    annotations:
      summary: "High request latency on {{ $labels.instance }}"
      description: "{{ $labels.instance }} has a median request latency above 1s (c
```

Creating Alerting Rule

Last updated：2024-01-29 16:01:55

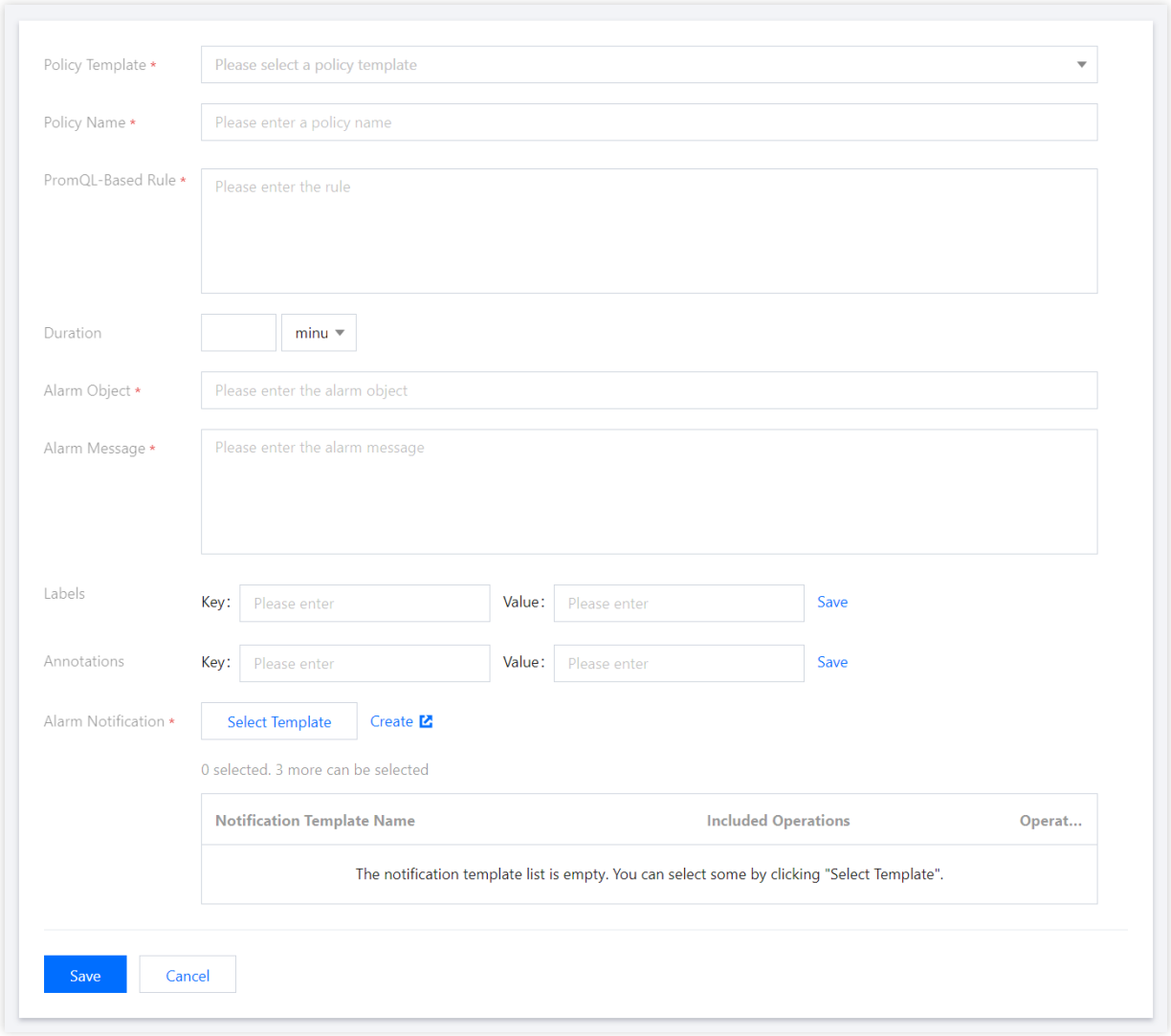
This document describes how to create an alerting rule in the TMP console. With such a rule, you will receive notifications when some metrics are exceptional, so that you can take corresponding measures promptly.

Prerequisites

You have created a managed TKE cluster.
You have created a TMP instance.
You have installed the Prometheus agent and other monitoring components.

Directions

For more information on alerting rule, please see Alerting Rule Description.
1. Log in to the TMP console.
2. In the instance list, select the corresponding TMP instance and click **Alerting Rule** on the left.
3. On the alerting rule management page, click **Create** and configure the alerting rule information in the pop-up window.
Rule Template Type: select a rule template type. For more information, please see Alerting Rule Type Description.
Rule Name: you can use the default rule name or customize it.
PromQL-Based Rule: you can use the default value or customize it. It indicates an alert trigger condition based on a PromQL expression, which is used to calculate whether there is time series data meeting the condition.
Duration: you can use the default value or customize it. It indicates how long a trigger condition can last before an alert is sent.
Alert Object: you can customize the alert title.
Alert Message: you can use the default value or customize it. It indicates the custom alert content.
Advanced Configuration: you can toggle it on for configuration, which contains configuration items for labels and annotations.
Labels: you can use the default value or customize it. It indicates a set of specified labels to be added to alerts. You can match the corresponding processing method based on the labels of a received alert.
Annotations: you can use the default value or customize it. It indicates the user-defined additional alert information.
Alert Notification: you can customize the alert notification template, which contains the template name, notification type, recipient, and receiving channel. For more information, please see Notification Template.



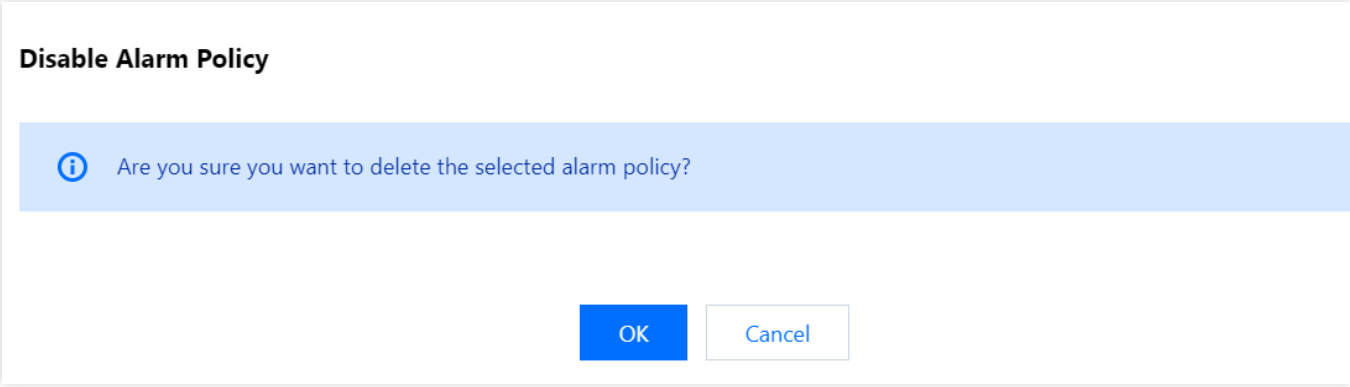
Disabling Alerting Rule

Last updated：2024-01-29 16:01:55

This document describes how to disable an alerting rule on the target instance in the TMP console.

Directions

1. Log in to the TMP console.
2. In the instance list, select the corresponding TMP instance and click **Alerting Rule** on the left.
3. Find the alerting rule to be disabled and click **Disable** in the **Operation** column.
4. In the pop-up window, click **OK**.



Rule Type Description(old)

Last updated：2024-01-29 16:01:55

TMP presets the master component, kubelet, resource use, workload, and node alert templates for TKE clusters.

Kubernetes master component

The following metrics are provided for non-managed clusters:

Rule Name	Rule Expression	Duration	Desc
Error with client access to APIServer	(sum(rate(rest_client_requests_total{code=~"5.."}[5m])) by (instance, job, cluster_id) / sum(rate(rest_client_requests_total[5m])) by (instance, job, cluster_id))> 0.01	15m	The rate acce the APIS abov
Imminent expiration of the client certificate for APIServer access	apiserver_client_certificate_expiration_seconds_count{job="apiserver"} > 0 and on(job) histogram_quantile(0.01, sum by (cluster_id, job, le) (rate(apiserver_client_certificate_expiration_seconds_bucket{job="apiserver"}[5m]))) < 86400	None	The certif for APIS acce expir hours
Recording API error	sum by(cluster_id, name, namespace) (increase(aggregator_unavailable_apiservice_count[5m])) > 2	None	The recor API repor error last 5 minu
Low recording API availability	(1 - max by(name, namespace, cluster_id) (avg_over_time(aggregator_unavailable_apiservice[5m]))) * 100 < 90	5m	The avail of the recor API s in the

			minu belov
APIServer fault	absent(sum(up{job="apiserver"}) by (cluster_id) > 0)	5m	APIS disap from colle targe
Scheduler fault	absent(sum(up{job="kube-scheduler"}) by (cluster_id) > 0)	15m	The sche disap from colle targe
Controller manager fault	absent(sum(up{job="kube-controller-manager"}) by (cluster_id) > 0)	15m	The contr mana disap from colle targe

Kubelet

Rule Name	Rule Expression	Duration	Des
Exceptional node status	kube_node_status_condition{job=~".*kube-state-metrics",condition="Ready",status="true"} == 0	15m	The stat exc for min
Unreachable node	kube_node_spec_taint{job=~".*kube-state-metrics",key="node.kubernetes.io/unreachable",effect="NoSchedule"} == 1	15m	The unre and wor be sch aga
Too many Pods	count by(cluster_id, node) ((kube_pod_status_phase{job=~".*kube-state-metrics",phase="Running"} == 1) * on(instance,pod,namespace,cluster_id)	15m	The of P

running on node	group_left(node) topk by(instance,pod,namespace,cluster_id) (1, kube_pod_info{job=~".*kube-state-metrics"}))/max by(cluster_id, node) (kube_node_status_capacity_pods{job=~".*kube-state-metrics"} != 1) > 0.95		runr the clos upp
Node status fluctuation	sum(changes(kube_node_status_condition{status="true",condition="Ready"} [15m])) by (cluster_id, node) > 2	15m	The stat fluc betv norr exc
Imminent expiration of the kubelet client certificate	kubelet_certificate_manager_client_ttl_seconds < 86400	None	The clie cert will 24 h
Imminent expiration of the kubelet server certificate	kubelet_certificate_manager_server_ttl_seconds < 86400	None	The serv cert will 24 h
Kubelet client certificate renewal error	increase(kubelet_certificate_manager_client_expiration_renew_errors[5m]) > 0	15m	An occ whil rene kub cert
Kubelet server certificate renewal error	increase(kubelet_server_expiration_renew_errors[5m]) > 0	15m	An occ whil rene kub serv cert
Time-Consuming PLEG	histogram_quantile(0.99, sum(rate(kubelet_pleg_relist_duration_seconds_bucket[5m])) by (cluster_id, instance, le) * on(instance, cluster_id) group_left(node) kubelet_node_name{job="kubelet"}) >= 10	5m	The perc PLE ope dura exc sec

Time-Consuming Pod start	histogram_quantile(0.99, sum(rate(kubelet_pod_worker_duration_seconds_bucket{job="kubelet"}[5m])) by (cluster_id, instance, le)) * on(cluster_id, instance) group_left(node) kubelet_node_name{job="kubelet"} > 60	15m	The perc Poc dura exc sec
Kubelet fault	absent(sum(up{job="kubelet"}) by (cluster_id) > 0)	15m	Kub disa fron coll targ

Kubernetes Resource Use

Rule Name	Rule Expression	Duration	Description
Cluster CPU resource overload	sum by (cluster_id) (max by (cluster_id, namespace, pod, container) (kube_pod_container_resource_requests_cpu_cores{job=~".*kube-state-metrics"}) * on(cluster_id, namespace, pod) group_left() max by (cluster_id, namespace, pod) (kube_pod_status_phase{phase=~"Pending|Running"} == 1))/sum by (cluster_id) (kube_node_status_allocatable_cpu_cores)>(count by (cluster_id) (kube_node_status_allocatable_cpu_cores)-1) / count by (cluster_id) (kube_node_status_allocatable_cpu_cores)	5m	Too many CPU cores are applied for by Pods in the cluster, and no more failed nodes can be tolerated
Cluster memory resource overload	sum by (cluster_id) (max by (cluster_id, namespace, pod, container) (kube_pod_container_resource_requests_memory_bytes{job=~".*kube-state-metrics"}) * on(cluster_id, namespace, pod) group_left() max by (cluster_id, namespace, pod) (kube_pod_status_phase{phase=~"Pending|Running"} == 1))/sum by (cluster_id) (kube_node_status_allocatable_memory_bytes) > (count by (cluster_id) (kube_node_status_allocatable_memory_bytes)-1) / count by (cluster_id) (kube_node_status_allocatable_memory_bytes)	5m	Too much memory is applied for by Pods in the cluster, and no more failed nodes can be tolerated
Cluster CPU	sum by (cluster_id) (kube_resourcequota{job=~".*kube-state-metrics", type="hard", resource="cpu"})/sum by (cluster_id) (kube_node_status_allocatable_cpu_cores) > 1.5	5m	The CPU quota in the cluster

quota overload			exceeds the total number of allocable CPU cores
Cluster memory quota overload	sum by (cluster_id) (kube_resourcequota{job=~".*kube-state-metrics", type="hard", resource="memory"}) / sum by (cluster_id) (kube_node_status_allocatable_memory_bytes) > 1.5	5m	The memory quota in the cluster exceeds the total amount of allocable memory
Imminent runout of quota resources	sum by (cluster_id, namespace, resource) kube_resourcequota{job=~".*kube-state-metrics", type="used"} / sum by (cluster_id, namespace, resource) (kube_resourcequota{job=~".*kube-state-metrics", type="hard"} > 0) >= 0.9	15m	The quota resource utilization exceeds 90%
High proportion of restricted CPU execution cycles	sum(increase(container_cpu_cfs_throttled_periods_total{container!="", }[5m])) by (cluster_id, container, pod, namespace) /sum(increase(container_cpu_cfs_periods_total{}[5m])) by (cluster_id, container, pod, namespace) > (25 / 100)	15m	The proportion of restricted CPU execution cycles is high
High Pod CPU utilization	sum(rate(container_cpu_usage_seconds_total{job="kubelet", metrics_path="/metrics/cadvisor", image!="", container!="POD"}[1m])) by (cluster_id, namespace, pod, container) / sum(kube_pod_container_resource_limits_cpu_cores) by (cluster_id, namespace, pod, container) > 0.75	15m	The Pod CPU utilization exceeds 75%
High Pod memory utilization	sum(rate(container_memory_working_set_bytes{job="kubelet", metrics_path="/metrics/cadvisor", image!="", container!="POD"}[1m])) by (cluster_id, namespace, pod, container) /sum(kube_pod_container_resource_limits_memory_bytes) by (cluster_id, namespace, pod, container) > 0.75	15m	The Pod memory utilization exceeds 75%

Kubernetes Workload

Rule Name	Rule Expression	Duration
Frequent Pod restarts	increase(kube_pod_container_status_restarts_total{job=~".*kube-state-metrics"}[5m]) > 0	15m
Exceptional Pod status	sum by (namespace, pod, cluster_id) (max by(namespace, pod, cluster_id) (kube_pod_status_phase{job=~".*kube-state-metrics", phase=~"Pending|Unknown"}) * on(namespace, pod, cluster_id) group_left(owner_kind) topk by(namespace, pod) (1, max by(namespace, pod, owner_kind, cluster_id) (kube_pod_owner{owner_kind!="Job"}))) > 0	15m
Exceptional container status	sum by (namespace, pod, container, cluster_id) (kube_pod_container_status_waiting_reason{job=~".*kube-state-metrics"}) > 0	1h
Deployment version mismatch	kube_deployment_status_observed_generation{job=~".*kube-state-metrics"} !=kube_deployment_metadata_generation{job=~".*kube-state-metrics"}	15m
Deployment replica quantity mismatch	(kube_deployment_spec_replicas{job=~".*kube-state-metrics"} != kube_deployment_status_replicas_available{job=~".*kube-state-metrics"}) and (changes(kube_deployment_status_replicas_updated{job=~".*kube-state-metrics"}[5m]) == 0)	15m
StatefulSet version mismatch	kube_statefulset_status_observed_generation{job=~".*kube-state-metrics"} != kube_statefulset_metadata_generation{job=~".*kube-state-metrics"}	15m
StatefulSet	(kube_statefulset_status_replicas_ready{job=~".*kube-state-metrics"} !=	15m

replica quantity mismatch	kube_statefulset_status_replicas{job=~".*kube-state-metrics"}) and (changes(kube_statefulset_status_replicas_updated{job=~".*kube-state-metrics"}[5m]) == 0)	
Ineffective StatefulSet update	(maxwithout(revision)(kube_statefulset_status_current_revision{job=~".*kube-state-metrics"}unless kube_statefulset_status_update_revision{job=~".*kube-state-metrics"})*(kube_statefulset_replicas{job=~".*kube-state-metrics"}!=kube_statefulset_status_replicas_updated{job=~".*kube-state-metrics"})) and (changes(kube_statefulset_status_replicas_updated{job=~".*kube-state-metrics"}[5m])==0)	15m
Frozen DaemonSet change	((kube_daemonset_status_current_number_scheduled{job=~".*kube-state-metrics"}!=kube_daemonset_status_desired_number_scheduled{job=~".*kube-state-metrics"}) or (kube_daemonset_status_number_misscheduled{job=~".*kube-state-metrics"}!=0) or (kube_daemonset_updated_number_scheduled{job=~".*kube-state-metrics"}!=kube_daemonset_status_desired_number_scheduled{job=~".*kube-state-metrics"}) or (kube_daemonset_status_number_available{job=~".*kube-state-metrics"}!=kube_daemonset_status_desired_number_scheduled{job=~".*kube-state-metrics"})) and (changes(kube_daemonset_updated_number_scheduled{job=~".*kube-state-metrics"}[5m])==0)	15m
DaemonSet not scheduled on some nodes	kube_daemonset_status_desired_number_scheduled{job=~".*kube-state-metrics"} - kube_daemonset_status_current_number_scheduled{job=~".*kube-state-metrics"} > 0	10m
Faulty scheduling of DaemonSet on some nodes	kube_daemonset_status_number_misscheduled{job=~".*kube-state-metrics"} > 0	15m
Excessive Job execution	kube_job_spec_completions{job=~".*kube-state-metrics"} - kube_job_status_succeeded{job=~".*kube-state-metrics"} > 0	12h
Job execution failure	kube_job_failed{job=~".*kube-state-metrics"} > 0	15m
Mismatch between replica	(kube_hpa_status_desired_replicas{job=~".*kube-state-metrics"} != kube_hpa_status_current_replicas{job=~".*kube-state-metrics"}) and	15m

quantity and HPA	changes(kube_hpa_status_current_replicas[15m]) == 0	
Number of replicas reaching maximum value in HPA	kube_hpa_status_current_replicas{job=~".*kube-state-metrics"} == kube_hpa_spec_max_replicas{job=~".*kube-state-metrics"}	15m
Exceptional PersistentVolume status	kube_persistentvolume_status_phase{phase=~"Failed|Pending",job=~".*kube-state-metrics"} > 0	15m

Kubernetes Node

Rule Name	Rule Expression	Duration	Description
Imminent runout of filesystem space	(node_filesystem_avail_bytes{job="node-exporter",fstype!=""}/node_filesystem_size_bytes{job="node-exporter",fstype!=""}*100<15 and predict_linear(node_filesystem_avail_bytes{job="node-exporter",fstype!=""}[6h],4*60*60)<0 and node_filesystem_readonly{job="node-exporter",fstype!=""}==0)	1h	It is estimated that the filesystem space will be used up in 4 hours
High filesystem space utilization	(node_filesystem_avail_bytes{job="node-exporter",fstype!=""}/node_filesystem_size_bytes{job="node-exporter",fstype!=""}*100<5 and node_filesystem_readonly{job="node-exporter",fstype!=""}==0)	1h	The available filesystem space is below 5%
Imminent runout of filesystem inodes	(node_filesystem_files_free{job="node-exporter",fstype!=""}/node_filesystem_files{job="node-exporter",fstype!=""}*100<20 and predict_linear(node_filesystem_files_free{job="node-exporter",fstype!=""}[6h],4*60*60)<0 and node_filesystem_readonly{job="node-exporter",fstype!=""}==0)	1h	It is estimated that the filesystem inodes will be used up in 4 hours
High	(node_filesystem_files_free{job="node-	1h	The proportion of

filesystem inode utilization	exporter",fstype!=""}/node_filesystem_files{job="node-exporter",fstype!=""}*100<3 and node_filesystem_readonly{job="node-exporter",fstype!=""}==0)		available inodes is below 3%
Unstable network interface status	changes(node_network_up{job="node-exporter",device!~"veth.+"}[2m])	2m	The network interface status is unstable and frequently changes between "up" and "down"
Network interface data reception error	increase(node_network_receive_errs_total[2m]) > 10	1h	An error occurred while the network interface received data
Network interface data sending error	increase(node_network_transmit_errs_total[2m]) > 10	1h	An error occurred while the network interface sent data
Unsynced server clock	min_over_time(node_timex_sync_status[5m]) == 0	10m	The server time has not been synced recently. Please check whether NTP is correctly configured
Server clock skew	(node_timex_offset_seconds>0.05 and deriv(node_timex_offset_seconds[5m])>=0) or (node_timex_offset_seconds<-0.05 and deriv(node_timex_offset_seconds[5m])<=0)	10m	The server clock skew exceeds 300 seconds. Please check whether NTP is correctly configured

Notification Template

Last updated：2024-01-29 16:01:55

This document describes how to create a notification template in the Cloud Monitor alarming module.

Use Cases

One template can be quickly reused for multiple policies, eliminating the need to repeatedly configure user notifications.
User notification methods can be configured in a more personalized way. For example, you can configure the alarm receiving channel as SMS/email by day and phone by night.

Prerequisites

View notification templates: the sub-account must have the read permission of Cloud Monitor.
Create and edit notification templates: the sub-account must have the write permission of Cloud Monitor.
Note:
For more information on how to grant sub-accounts permissions, please see Cloud Access Management (CAM).

Use Limits

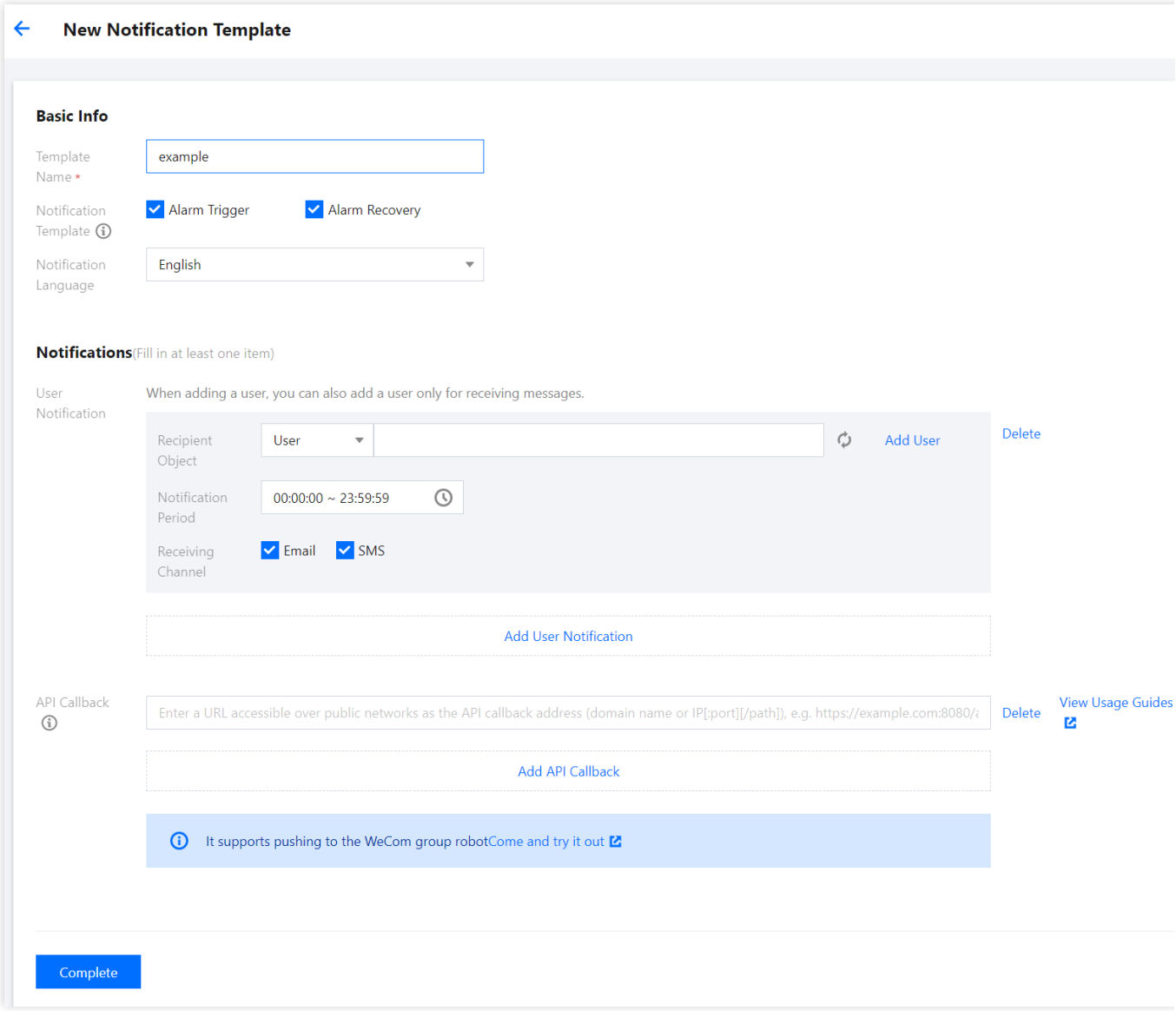
Feature	Limit
User notification	Up to five items can be added
API callback	Up to three URLs accessible over the public network can be entered

Directions

Creating notification template

1. Enter the Alarm Notification Template page in the Cloud Monitor console.
2. Click **Create** and enter relevant information in **Create Notification Template**.
Template Name: enter a custom template name.
Notification type:

Alarm triggered: a notification will be sent when an alarm is triggered.
Alarm cleared: a notification will be sent when an alarm is resolved.
User notification:
Recipient Object: you can choose a recipient group or recipient. If you need to create a group, please see Creating Alarm Recipient Group.
Notification Period: define the time period for receiving alarms.
Receiving Channel: four alarm channels are supported: email, SMS, WeChat, and phone. You can also set different channels and notification periods in different user dimensions.
Description of phone alarm settings:
Polling Times: the maximum number of dials for each polled recipient when there is no valid reach.
Polling Sequence: alarm calls will be dialed according to the order of the recipients. You can adjust the order of calling by dragging up and down recipients.
Polling Interval: time interval at which alarm calls will be dialed according to the order of the recipients.
Reach Notification: notifications will be to all recipients after successful reception of the call or calling all recipients. SMS messages are counted against the quota.
API Callback: you can enter a URL accessible over the public network as the callback API address, and Cloud Monitor will push alarm messages to it promptly. If the HTTP response returns code 200, the verification is successful. For more information on alarm callback fields, please see Alarm Callback.
Note:
After you save the callback URL, the system will automatically verify your URL once. The timeout threshold for this verification is 5 seconds. When an alarm policy created by the user is triggered or the alarm is resolved, the alarm messages will be pushed through the API callbacks. An alarm message can be pushed up to three times, and the timeout threshold for each request is 5 seconds.
When an alarm policy created by the user is triggered or the alarm is resolved, the alarm messages will be pushed through the API callbacks. API callbacks also support repeated alarms.
The outbound IP of the Cloud Monitor callback API is dynamically and randomly allocated, so no specific IP information can be provided to you, but the IP port is fixed at 80. We recommend you configure a weighted opening policy in the security group based on port 80.



Default notification template

The system automatically creates a default notification template for you as detailed below:

Feature	Default Configuration
Template name	Preset notification template
Notification type	Alarm trigger, alarm recovery
Alarm recipient	Root account admin
Notification period	00:00:00–23:59:59 (all day)
Receiving channel	Email, SMS

Deleting template

Note:

The default notification template cannot be deleted.

1. Find the name of the template to be deleted and click **Delete** in the **Operation** column.
2. In the pop-up window, click **OK**.

Replicating template

1. Find the name of the template to be replicated and click **Replicate** in the **Operation** column.
2. Modify the information in the redirected page or click **Complete** directly.

Tag
Examples

Last updated：2024-01-29 16:01:55

Overview

A tag is a key-value pair provided by Tencent Cloud to identify a resource in the cloud.
You can use tags to classify TMP resources based on various factors such as service, usage, and owner. With tags, you can quickly sift through the resource pool and find the corresponding resources. The values of tag keys do not mean anything to Tencent Cloud semantically and will be parsed and matched strictly according to the string. Tencent Cloud will not use your set tags, which are used only for resource management.
Below is a specific use case to show how a tag is used.

Use Case Background

A company owns 10 TMP instances in Tencent Cloud. Distributed in three departments (ecommerce, gaming, and entertainment), these instances are used to serve internal business lines such as marketing, game A, game B, and post-production. The OPS owners of the three departments are John, Jane, and Harry, respectively.

Setting Tag

To facilitate management, the company categorizes its TMP resources with tags and defines the following tag key-value pairs:

Tag Key	Tag Value
Department	Ecommerce, gaming, and entertainment
Business	Marketing, game A, game B, and post-production
OPS owner	John, Jane, and Harry

These tags are bound to TMP instances in the following way:

Instance ID	Department	Business	OPS Owner
prom-1jqwv1	Ecommerce	Marketing	Harry
prom-1jqwv12	Ecommerce	Marketing	Harry
prom-1jqwv13	Gaming	Game A	John
prom-1jqwv13	Gaming	Game B	John

prom-1jqwv14	Gaming	Game B	John
prom-1jqwv15	Gaming	Game B	Jane
prom-1jqwv16	Gaming	Game B	Jane
prom-1jqwv17	Gaming	Game B	Jane
prom-1jqwv18	Entertainment	Post-production	Harry
prom-1jqwv19	Entertainment	Post-production	Harry
prom-1jqwv110	Entertainment	Post-production	Harry

Using Tag

Filter out the TMP instances in the charge of Harry

Filter out the TMP instances where the OPS owner is "Harry". For detailed directions, please see Using Tag.

Filter out the TMP instances in the charge of Jane in the gaming department

Filter out the TMP instances where the department is "gaming" and OPS owner is "Jane". For detailed directions, please see Using Tag.

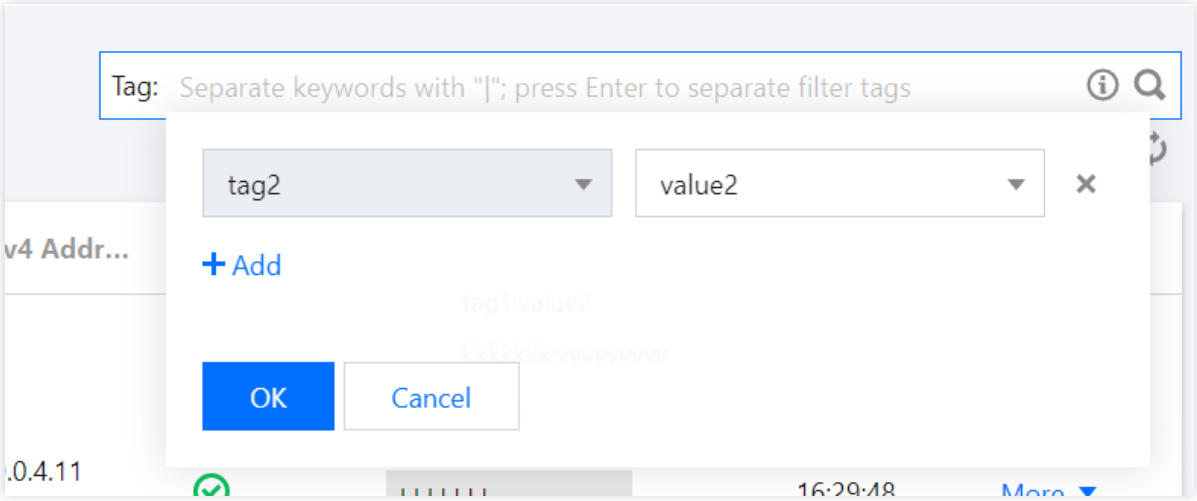
Using Tag

Last updated：2024-01-29 16:01:56

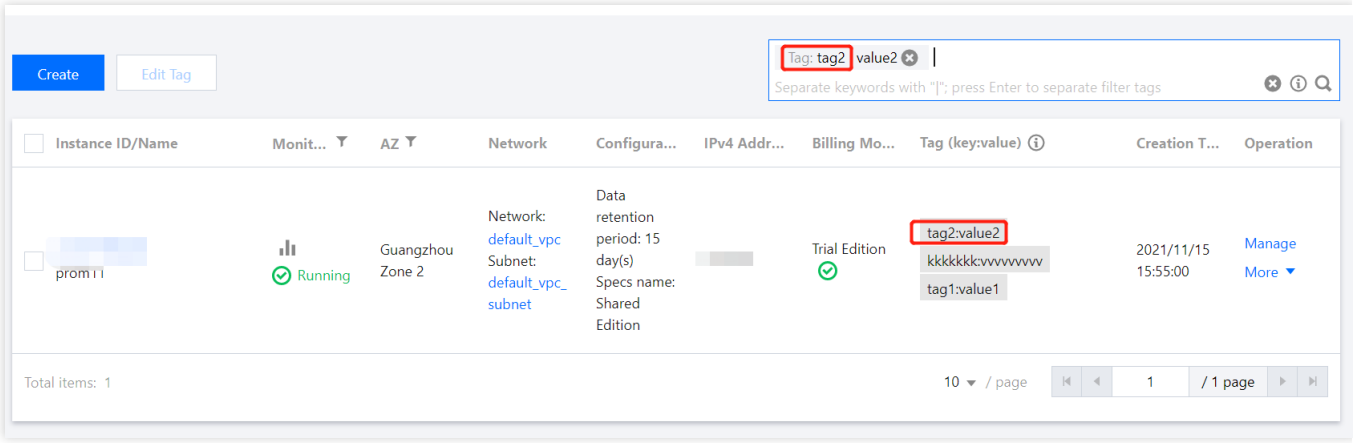
This document describes how to filter instance resources by tags in the TMP console.

Directions

1. Log in to the TMP console.
2. Select the region at the top of the instance list page.
3. In the search box in the top-right corner of the instance list, click the blank space and select **Tag**.



4. Select the corresponding conditions in the tag filter selection box and click **OK**.
5. If you need to adjust the tag conditions, click the tag content after **Tag:** in the search box to edit it.
6. You can also directly click the corresponding tag value in the instance list to filter instances as shown below:



Editing Tag

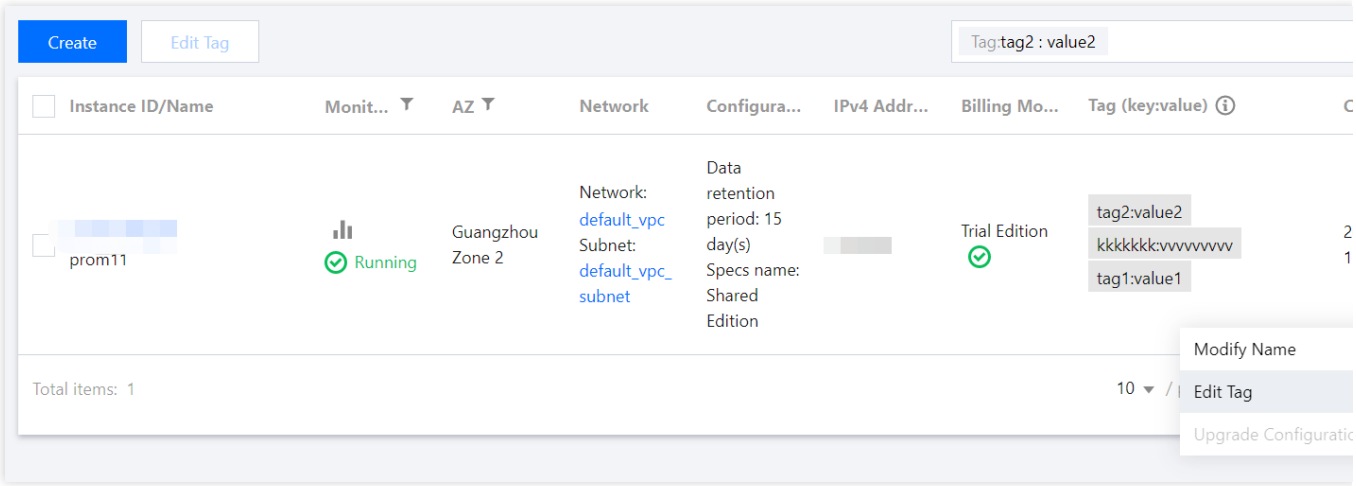
Last updated：2024-01-29 16:01:55

This document describes how to edit the tags of an instance in the TMP console.

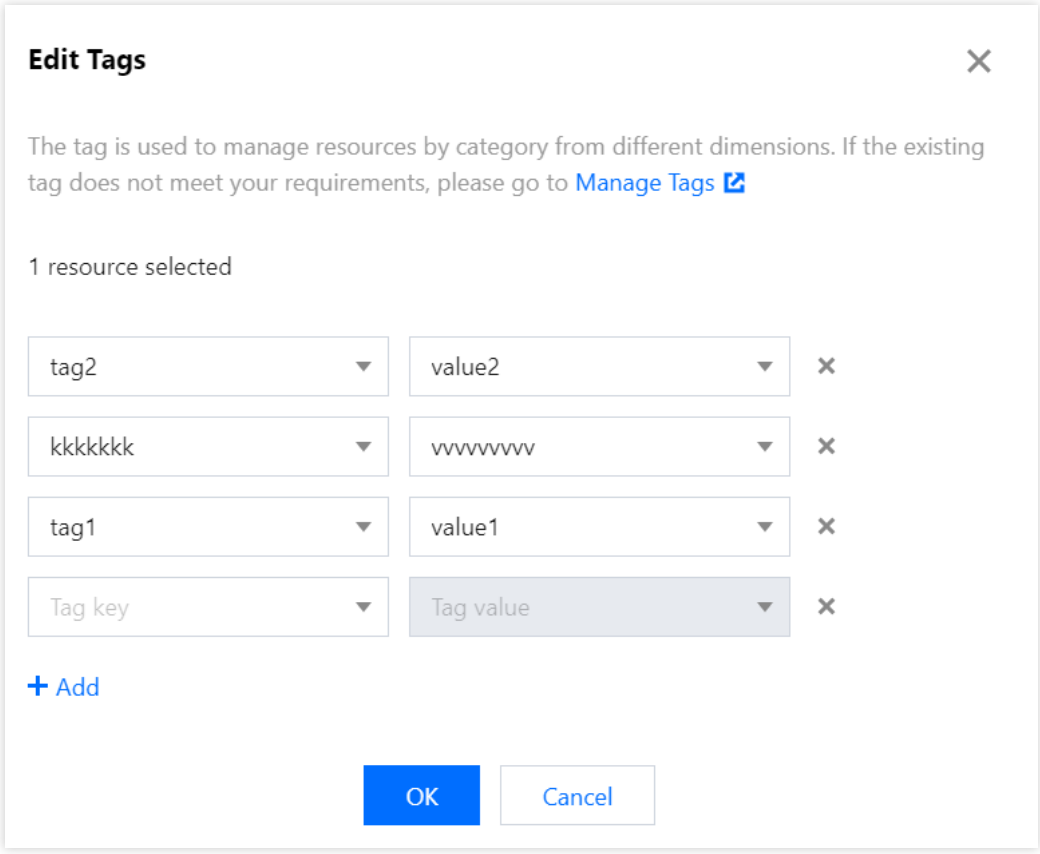
Directions

Edit tag for a single instance

1. Log in to the TMP console.
2. On the instance management page, select the instance for which to edit tags and click **More** > **Instance Configuration** > **Edit Tag** as shown below:



3. In the **Selected 1 resource** window that pops up, add, modify, or delete tags based on your actual needs.



Use Limits

Last updated：2024-01-29 16:01:55

A tag is a key-value pair. You can set tags for TMP instances in the TMP console to manage them in a categorized manner. Then, you can easily filter and find desired resources with tags.

Quantity Limit

One Tencent Cloud resource can have up to 50 tags.

Tag Key Limit

It cannot begin with "qcloud", "tencent", or "project" as they are reserved by the system.
It can contain only letters, digits, spaces, and certain special symbols (+, -, =, ., _, :, /, @).
It can contain up to 255 characters.

Tag Value Limit

It can contain only letters, digits, spaces, and certain special symbols (+, -, =, ., _, :, /, @).
It can contain up to 127 characters.

Access Control

Overview

Last updated：2024-01-29 16:01:55

If you have multiple users managing the TMP service, and they all share your Tencent Cloud account access key, you may face the following problems:
The risk of your key being compromised is high since multiple users are sharing it.
Your users might introduce security risks from maloperations due to the lack of user access control.
You can avoid the above problems by allowing different users to manage different services through sub-accounts. By default, sub-accounts have no permissions to use TMP. Therefore, you need to create a policy to grant different permissions to sub-accounts.

Overview

Cloud Access Management (CAM) is a web-based Tencent Cloud service that helps you securely manage and control access permissions of your Tencent Cloud resources. Using CAM, you can create, manage, and terminate users (groups), and control the Tencent Cloud resources that can be used by the specified user through identity and policy management.
When using CAM, you can associate a policy with a user or user group to allow or forbid them to use specified resources to complete specified tasks. For more information on CAM policies, please see Element Reference. For more information on how to use CAM policies, please see Policy.
You can skip this section if you don't need to manage permissions of TMP resources for sub-accounts. This won't affect your understanding and use of the other sections of the document.

Getting Started

A CAM policy must authorize or deny the use of one or more TMP operations. At the same time, it must specify the resources that can be used for the operations (which can be all resources or partial resources for certain operations). A policy can also include the conditions set for the manipulated resources.
Certain APIs of TMP don't support resource-level permissions, which means that for this type of API operations, you cannot specify a given resource for use when they are performed; instead, you must specify all resources for use.

Setting Policy

Last updated：2024-01-29 16:01:55

Overview

Access policies can be used to grant access to TMP instances. They use JSON-based access policy syntax. You can authorize specified principals to perform specified operations on specified TMP resources through the access policy syntax.
The access policy syntax describes the basic elements and usage of the policy. For the description of the policy syntax, please see Permission.

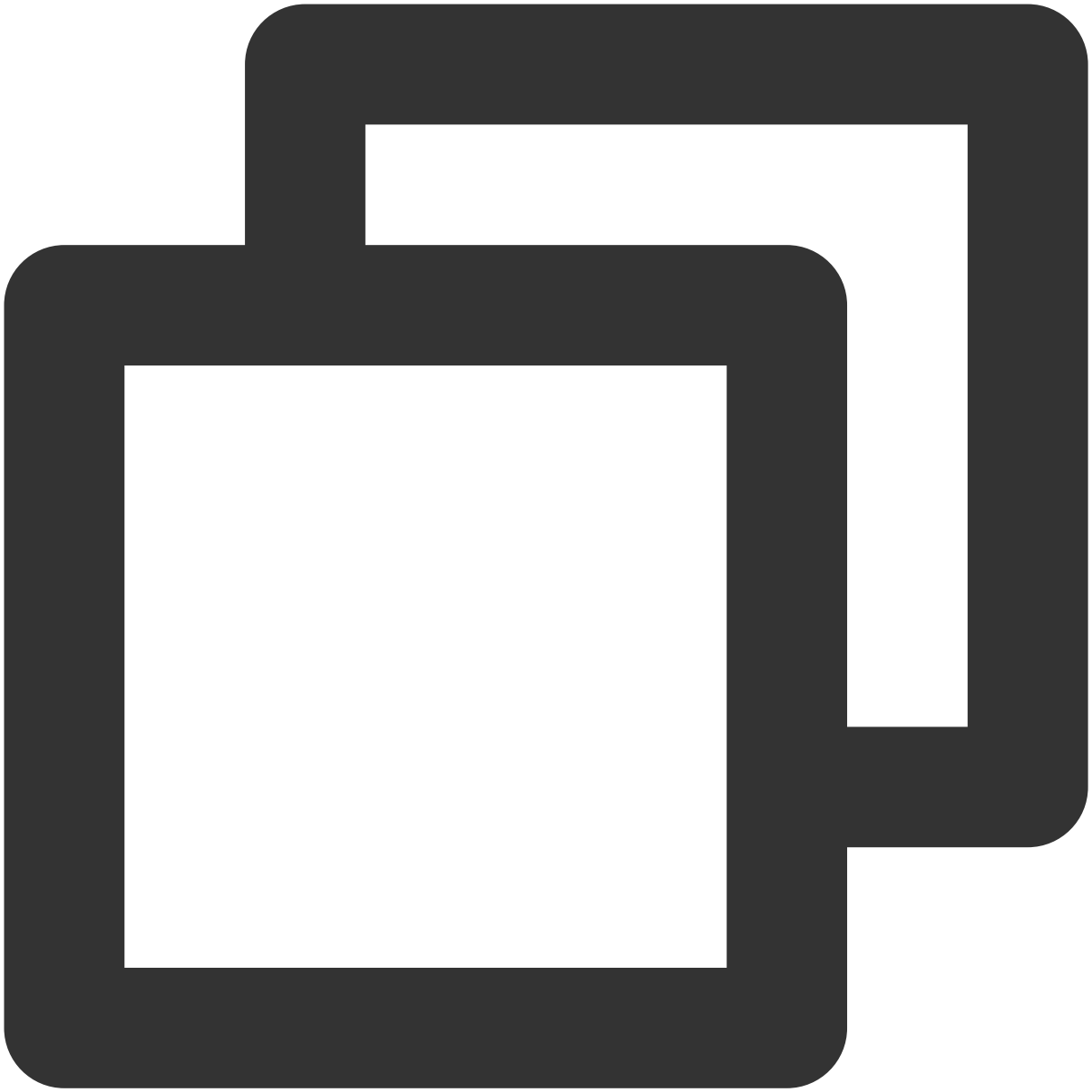
Elements in Access Policy

An access policy contains the following elements with basic meanings:
statement: it describes the details of one or more permissions. It contains a permission or permission set of multiple other elements such as `effect`, `action`, `resource`, and `condition`. One policy must and can have only one `statement`.
effect: it is required and describes the result of a statement. The result can be an "allow" or "explicit deny".
action: it is required and describes the allowed or denied action (operation). An operation can be an API (prefixed with "name") or a feature set (a set of specific APIs prefixed with "permid").
resource: it is required and describes the details of authorization. A resource is described in a six-segment format. Detailed resource definitions vary by product.
condition: it is optional and describes the condition for the policy to take effect. A condition consists of operator, action key, and action value. A condition value may contain information such as time and IP address. Some services allow you to specify additional values in a condition.

Element Usage

Specifying effect

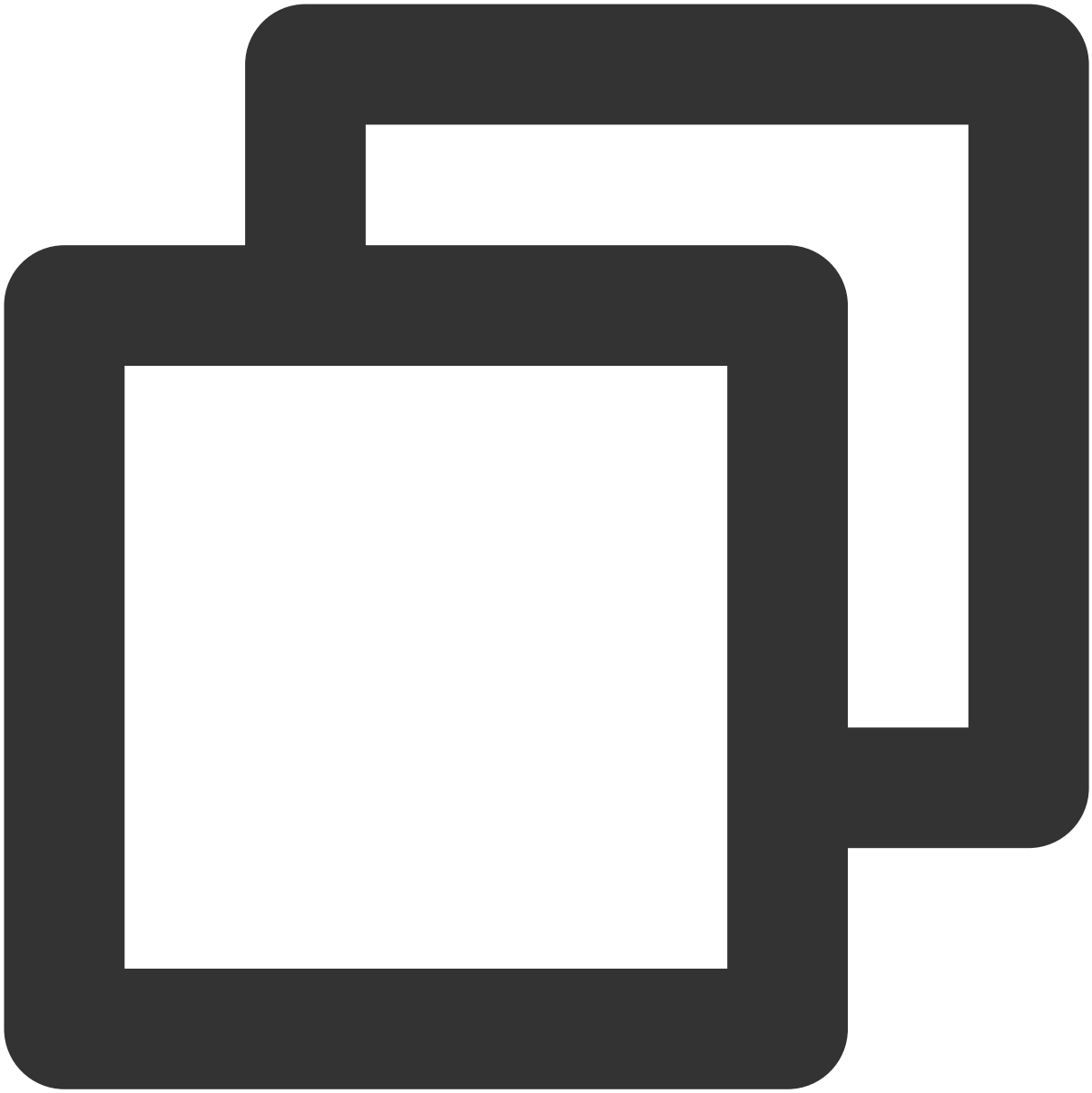
If access to a resource is not explicitly granted (allowed), then it is implicitly denied. It can also be explicitly denied, which ensures that users cannot access the resource even if they are granted the access permission by other policies. Below is an example of specifying the "allow" effect:



```
"effect" : "allow"
```

Specifying action

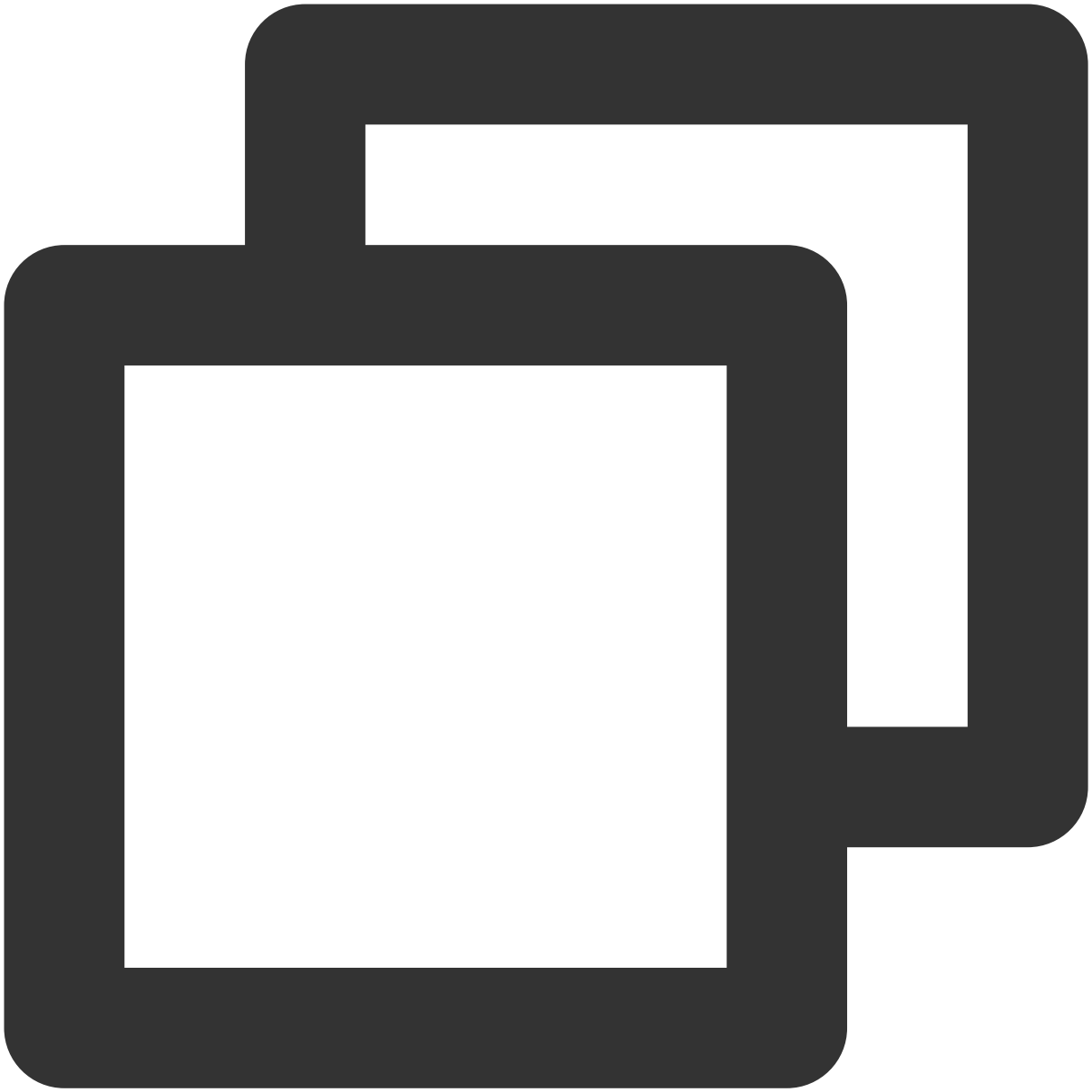
Cloud Monitor defines console operations that can be specified in a policy. The specified operations are divided into reading part of APIs (monitor:Describe*) and all APIs (monitor:*) according to the operation nature.
Below is an example of specifying the allowed operations:



```
"action": [
  "name/monitor:Describe*"
]
```

Specifying resource

The `resource` element describes one or more operation objects, such as TMP resources. All resources can use the following six-segment format:



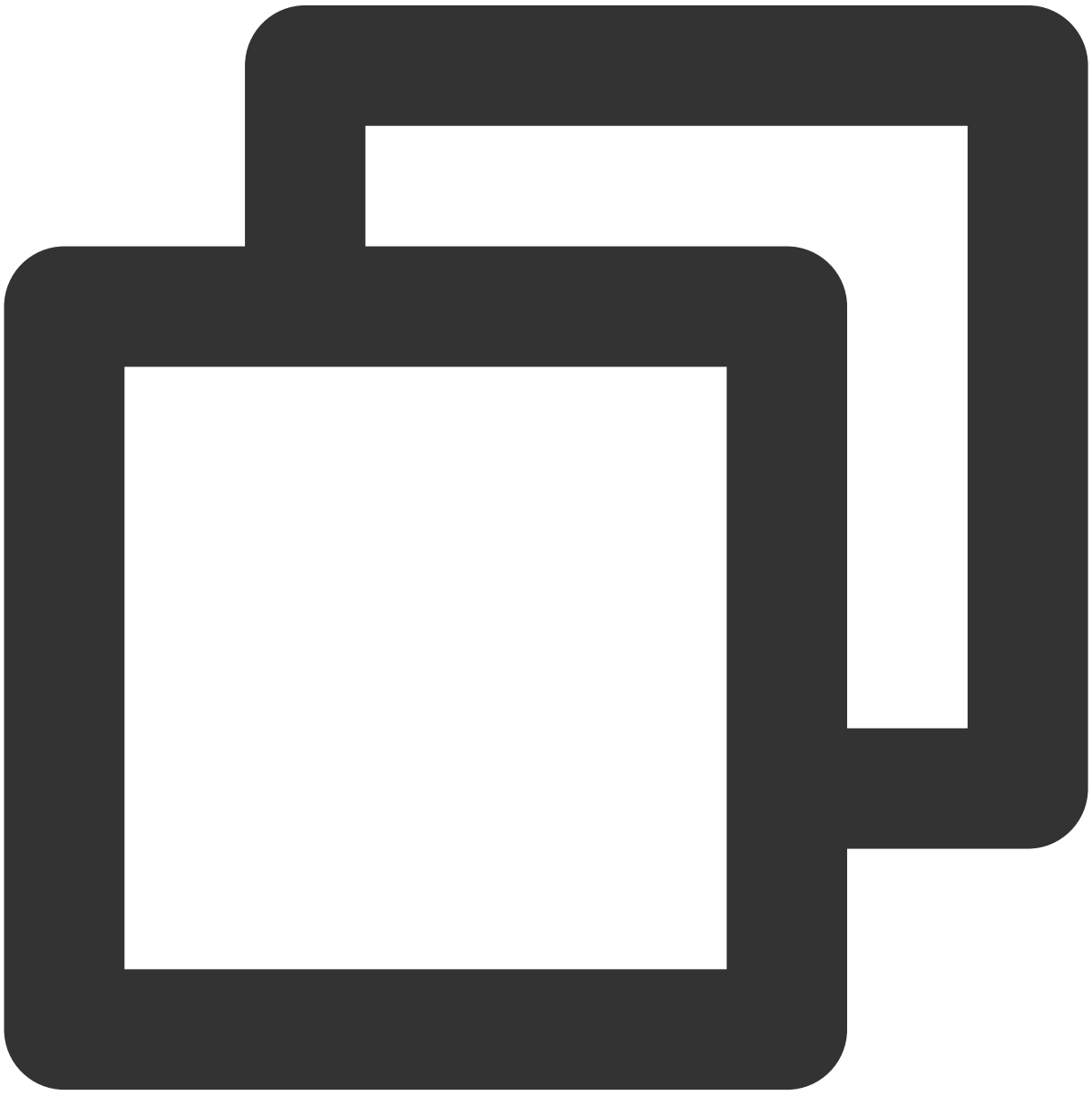
```
qcs:project_id:service_type:region:account:resource
```

The parameters are as detailed below:

Parameter	Description	Required
qcs	Tencent Cloud service abbreviation, which indicates a service of Tencent Cloud	Yes
project_id	Project information, which is only used to enable compatibility with legacy CAM logic and generally can be left empty	No

service_type	Product abbreviation, which is `monitor` here	Yes
region	Region information	Yes
account	Root account information of the resource owner, i.e., root account ID in the format of `uin/${OwnerUin}` , such as uin/100000000001	Yes
resource	Resource details prefixed with `instance`	Yes

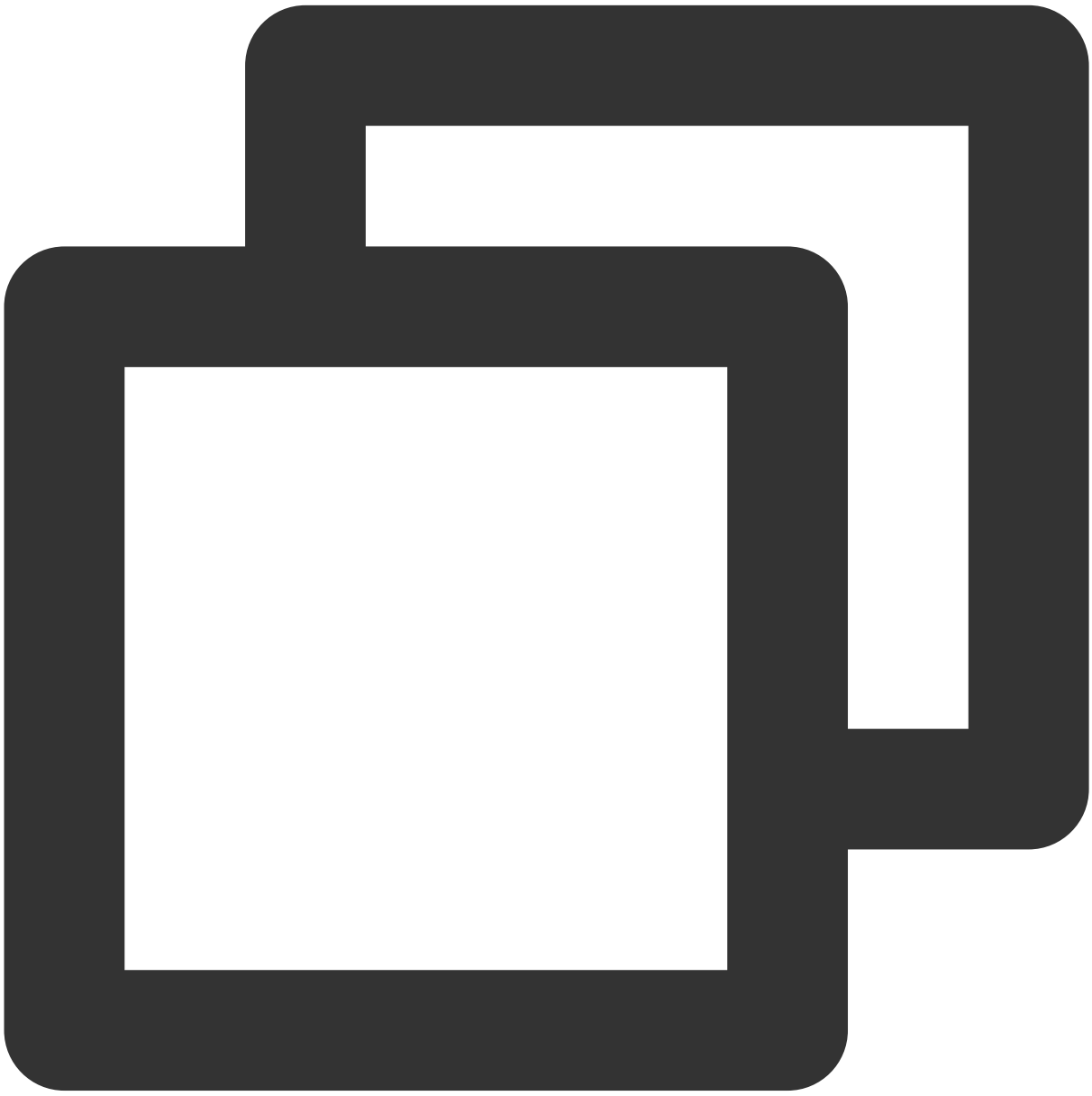
Below is a sample six-segment TMP resource description:



```
"resource":["qcs::monitor:ap-guangzhou:uin/100000000001:prom-instance/prom-73jingds
```

Specifying condition

The access policy syntax allows you to specify the condition when granting permissions, which is mainly used to set tag authentication. The tag condition takes effect only for clusters bound with the tag. Below is a sample tag policy:



```
"condition": {
    "for_any_value:string_equal": {
        "qcs:tag": [
```

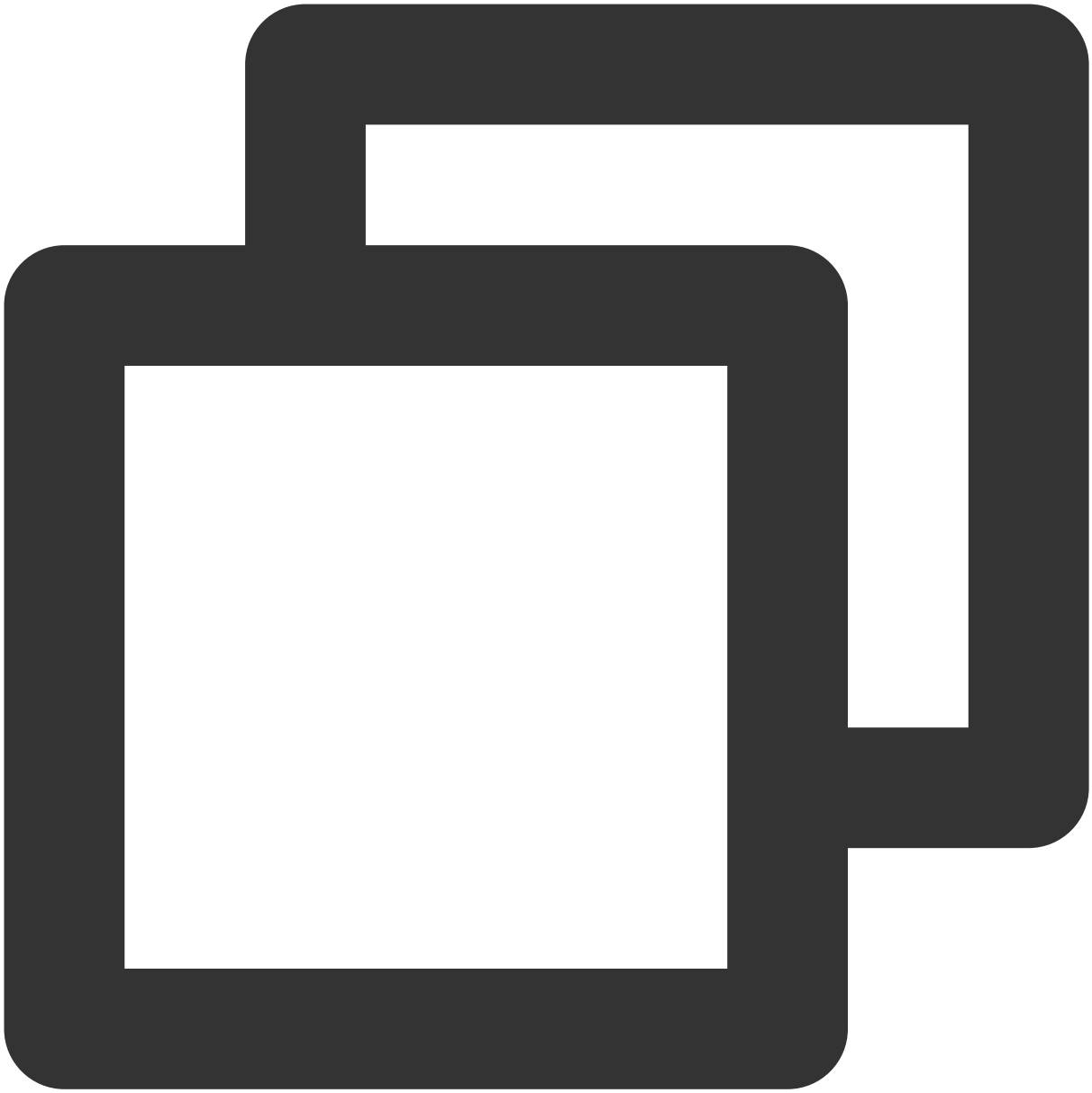
```
            "testkey&testvalue"
        ]
    }
}
```

This statement means that the policy contains resources whose tag key is `testkey` and tag value is `testvalue` .

Use Cases

Based on tag

In the following case, the policy is to allow access to the resource whose instance ID is `prom-73jingds` under UIN 1250000000 and the resources whose tag key is `testkey` and tag value is `testvalue` (if this instance doesn't have the `testkey& testvalue` tag, it cannot be accessed).



```
{
   "version": "2.0",
    "statement": [
        {
            "action": [
                "name/monitor:*"
            ],
            "condition": {
               "for_any_value:string_equal": {
                    "qcs:tag": [
                        "testkey&testvalue"
```

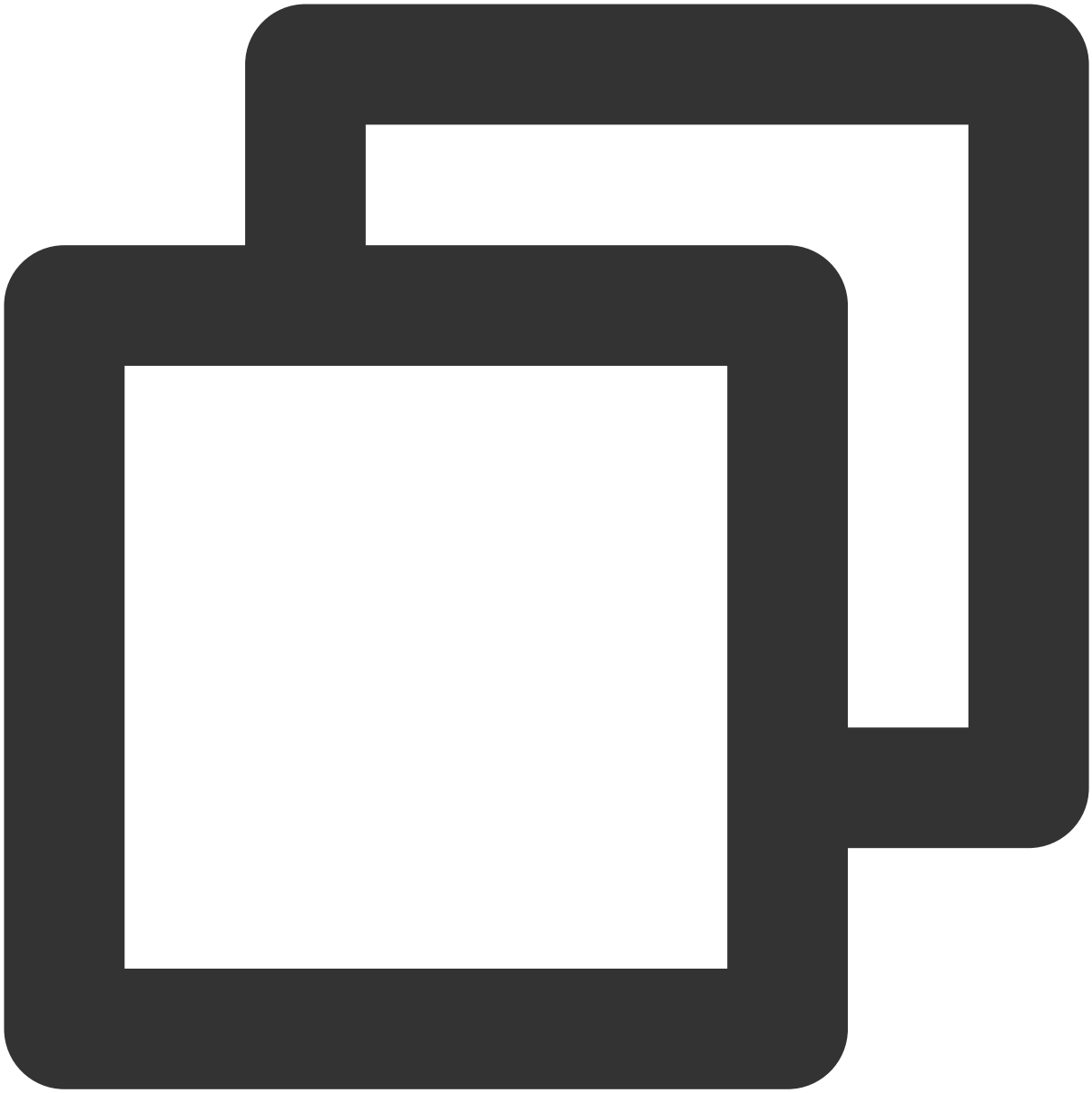
```
                    ]
                }
            },
            "effect": "allow",
            "resource": [
                "qcs::monitor:ap-guangzhou:uin/1250000000:prom-instance/prom-73jing
            ]
        }
    ]
}
```

Based on resource

Grant the read and write permissions of specified resources based on resource ID. The root account ID is 1250000000:

Configure read-only access to resources in the Guangzhou region.

Sample: granting the sub-user read-only access to the instance1(prom-73jingds) and instance2(prom-65jidfafk) resources.

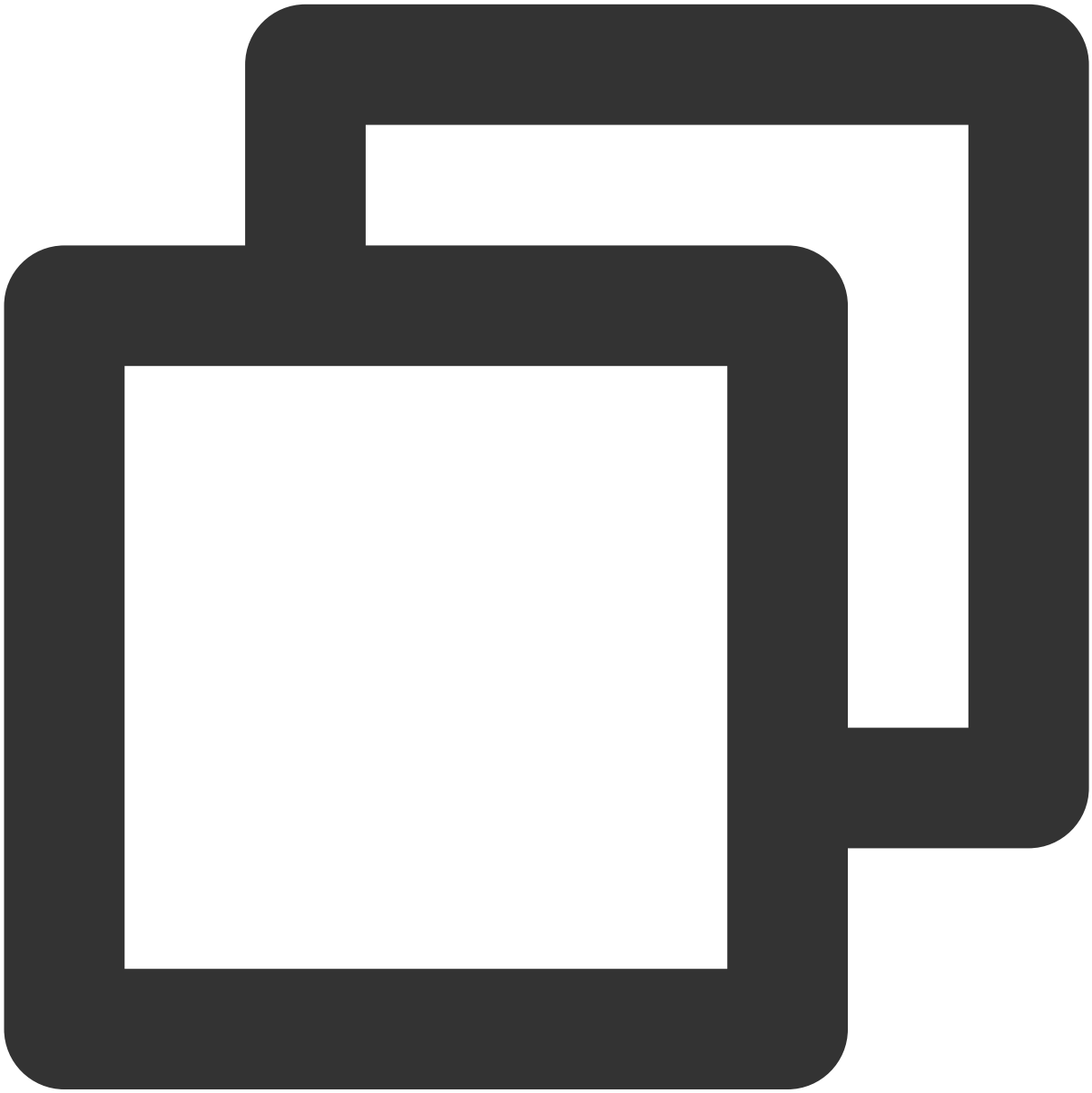


```
{
  "version": "2.0",
  "statement": [{
      "effect": "allow",
      "action": [
          "name/monitor:Describe*"
      ],
      "resource": [
          "qcs::monitor:ap-guangzhou:uin/1250000000:prom-instance/prom-73jingds",
          "qcs::monitor:ap-guangzhou:uin/1250000000:prom-instance/prom-65jidfafk"
      ],
```

```
        "condition": []
    }]
}
```

Configure partial read/write access to resources in the Guangzhou region.

Sample: granting the sub-user the permission to delete the instance1(prom-73jingds) resource



```
{
  "version": "2.0",
  "statement": [{
      "effect": "allow",
```

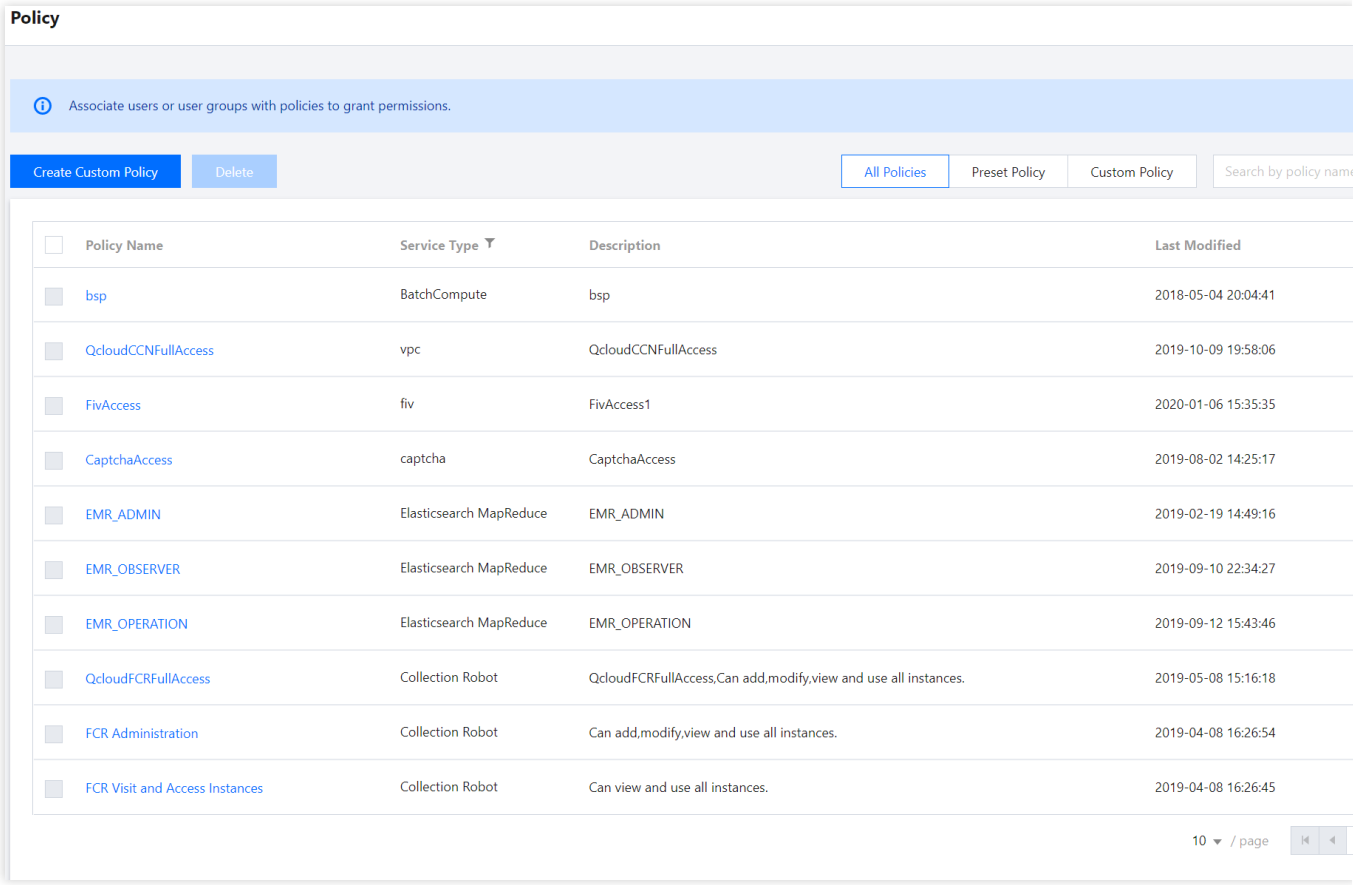
```
        "resource": [
            "qcs::monitor:ap-guangzhou:uin/1250000000:prom-instance/prom-73jingds"
        ],
        "action": [
            "name/monitor:TerminatePrometheusInstances"
        ]
    }]
}
```

Granting Policy

Last updated：2024-01-29 16:01:55

Custom Policy for TMP

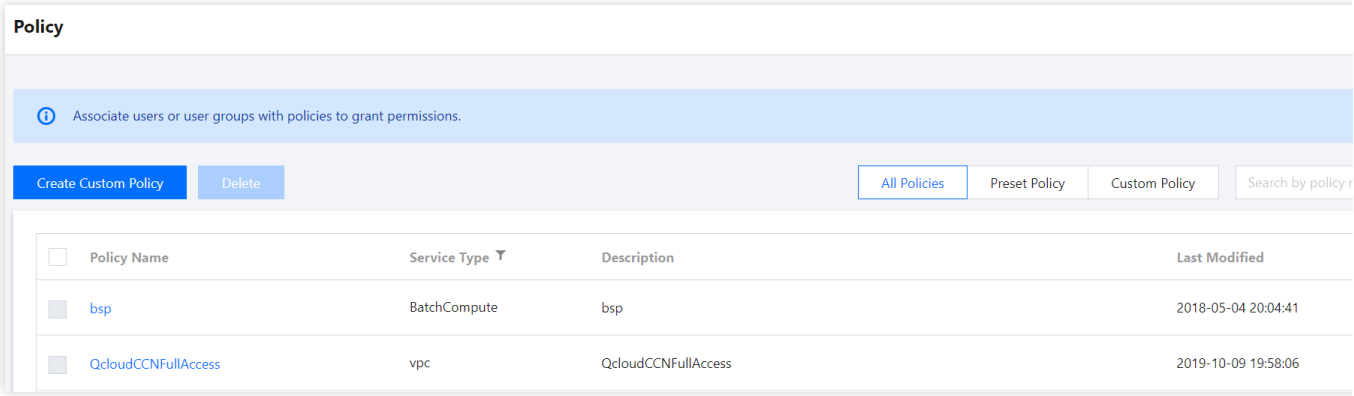
If preset policies cannot meet your needs, you can click **Create Custom Policy** to create custom policies.



For the method of custom policy creation, please see Setting Policy.

Policy Authorization

A configured policy can grant permissions by associating user groups or sub-users.



Resource Types Authorizable by Custom Policy

Resource-Level permission can be used to specify which resources a user can manipulate. TMP supports certain resource-level permissions. This means that for TMP operations that support resource-level permission, you can control the time when a user is allowed to perform operations or to use specified resources. The following table describes the types of resources that can be authorized in CAM.

Resource Type	Resource Description Method in Authorization Policy
TMP	`qcs::monitor:$region:$account:prom-instance/*` `qcs::monitor:$region:$account:prom-instance/$instanceId`

The following table describes the TMP API operations that currently support resource-level permissions. When setting a policy, you can enter the API operation name in the `action` field to control the individual API. You can also use `*` as a wildcard to set the `action` .

List of APIs supporting resource-level authorization

API Operation	API Description
DescribePrometheusInstances	Lists all TMP instances of the user
TerminatePrometheusInstances	Terminates TMP instance
RecreatePrometheusInstance	Reboots TMP instance
ModifyPrometheusInstanceAttributes	Modifies TMP instance attributes
ChangeGrafanaAdminPassword	Changes Grafana admin Password
UpgradeGrafanaDashboard	Upgrades Grafana dashboard

DescribePrometheusKubeClusters	Lists TKE clusters that can be integrated with TMP
InstallPrometheusAgent	Installs Prometheus agent
UninstallPrometheusAgent	Uninstalls Prometheus agent
DescribeServiceDiscovery	Lists TMP scrape configurations
CreateServiceDiscovery	Creates TMP scrape configuration
UpdateServiceDiscovery	Updates TMP scrape configuration
DeleteServiceDiscovery	Deletes TMP scrape configuration
DescribePrometheusKubeBasicMonitor	Queries basic monitoring status
EnablePrometheusKubeBasicMonitor	Enables basic monitoring
DisablePrometheusKubeBasicMonitor	Disables basic monitoring
DescribePrometheusAgentRuntime	Gets the runtime status of Prometheus agent
DescribePrometheusJobTargets	Lists the status information of TMP metric scrape tasks
DescribeRecordingRules	Queries recording rules
CreateRecordingRule	Creates recording rule
UpdateRecordingRule	Updates recording rule
DeleteRecordingRules	Deletes recording rule
DescribeAlertRules	Queries alarming rules
DeleteAlertRules	Deletes alarming rule
UpdateAlertRuleState	Updates alarming rule status
CreateAlertRule	Creates alarming rule
UpdateAlertRule	Updates alarming rule

List of APIs not supporting resource-level authorization

For TMP API operations that don't support resource-level authorization, you can still authorize a user to perform them, but you must specify `*` as the resource element in the policy statement.

API Operation	API Description

CreatePrometheusInstance	Creates TMP instance

Description of Role Permissions Related to Service Authorization

Last updated：2024-01-29 16:01:55

When you use TMP, in order to use related Tencent Cloud resources, you will encounter a variety of scenarios that require service authorization. The `CM_QCSRole` service role is mainly involved in the process of using TMP. This document describes the details, scenarios, and steps of each authorization policy by role.
The preset policies associated with the `CM_QCSRole` role by default include the following:
QcloudAccessForCMRoleInPromHostingService: TKE permission required by TMP.

Use Cases

After you successfully create a TMP instance, you need to monitor the services running on TKE. In order to integrate the TKE service more conveniently, you need to access TKE-related APIs. In this case, your authorization is required before TKE can be normally accessed to install basic monitoring components and get their running status information. This role doesn't need to actively look for configuration. If its permission hasn't been granted, after you successfully create a TMP instance, the authorization page will automatically pop up when you enter the **Integrate with TKE** page for instance management.

Authorization Steps

Authorizing by root account

1. After you successfully create a TMP instance, an authorization window will pop up when you access the **Integrate with TKE** page, and you need to authorize Cloud Monitor permissions as shown below:
2. Click **Authorize Now** in the window.
3. On the **CAM** > **Role Management** page, click **Grant**, and the system will prompt that the authorization is successful.

Note:
This authorization window will appear only once. If you have already authorized, it will not appear again.

Granting permissions to sub-account

After the root account completes the above authorization operations and successfully creates the `CM_QCSRole` role, the sub-account doesn't have permission to access it. The sub-account must be granted the `PassRole`

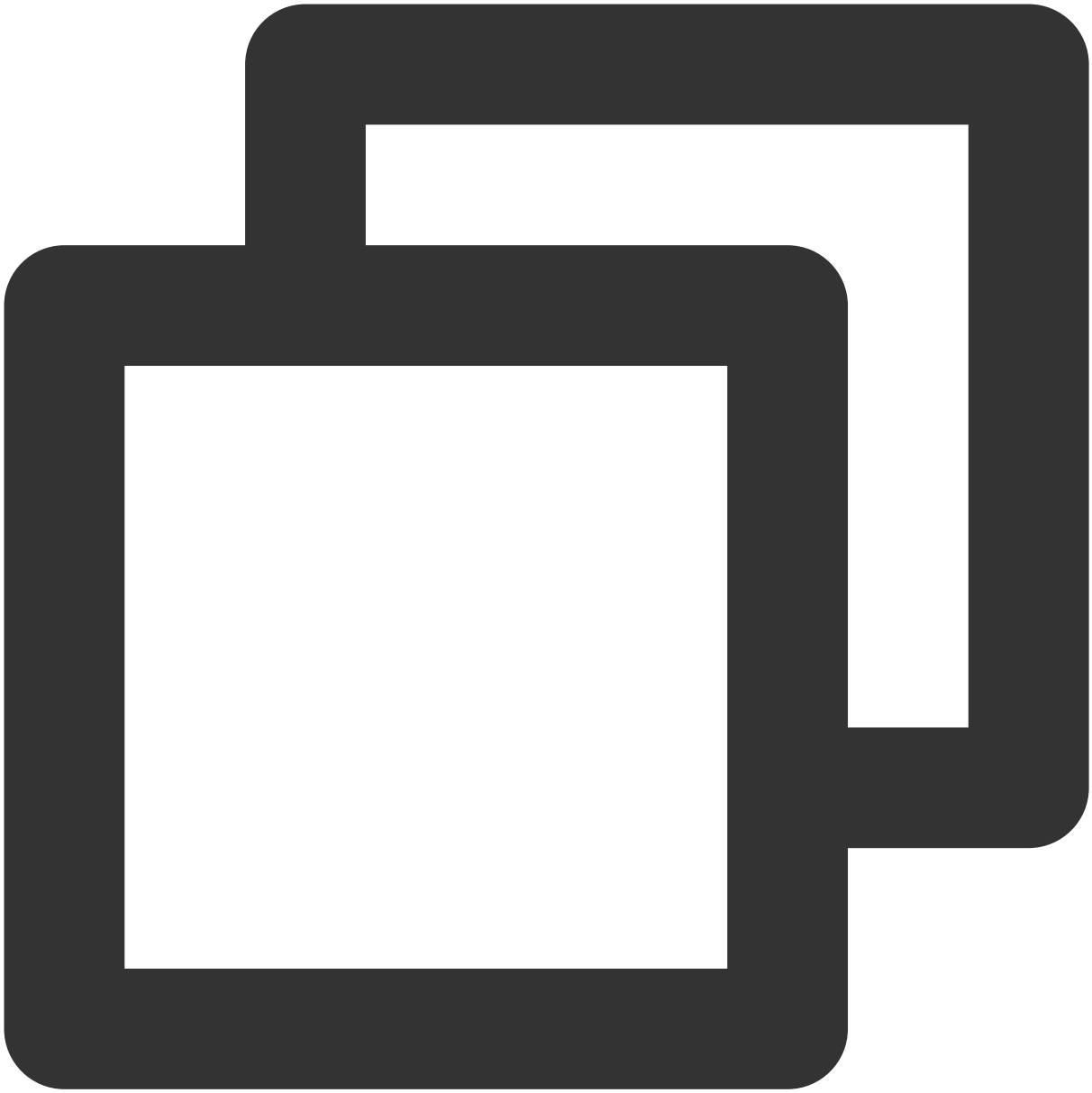
permission by the root account before it can normally access TKE in TMP; otherwise, an error will be displayed when it accesses the TKE cluster list.

When granting the `PassRole` permission to your sub-account, please make sure that your sub-account has the following permissions:

Permission Description	Granted Policy
The sub-account needs to be granted access to CAM before granting the `PassRole` permission to the sub-account by the root account can take effect	`QcloudCamReadOnlyAccess` or `QcloudCamFullAcces`
The Cloud Monitor policy depends on the Tencent Cloud service policy; therefore, before granting the `PassRole` permission to the sub-account, you need to make sure that the sub-account can normally access TKE resources	For more information, please see Permission Management

To ensure that the above permissions are granted successfully, please grant the `cam:PassRole` permission to the sub-account in the following steps.

1. Use the root account or a sub-account with administrative permissions to create the following custom policy:



```
{
 "version": "2.0",
 "statement": [
     {
         "effect": "allow",
         "action": "cam:PassRole",
         "resource": "qcs::cam::uin/${OwnerUin}:roleName/CM_QCSRole"
     }
 ]
}
```

2. After creation, associate the sub-account with the custom policy as instructed in CAM - Authorization Management. After granting the sub-account the `cam:PassRole` permission, access the **Integrate with TKE** page of the corresponding TMP instance, and an authorization window will pop up.

Grafana

Last updated：2024-01-29 16:01:55

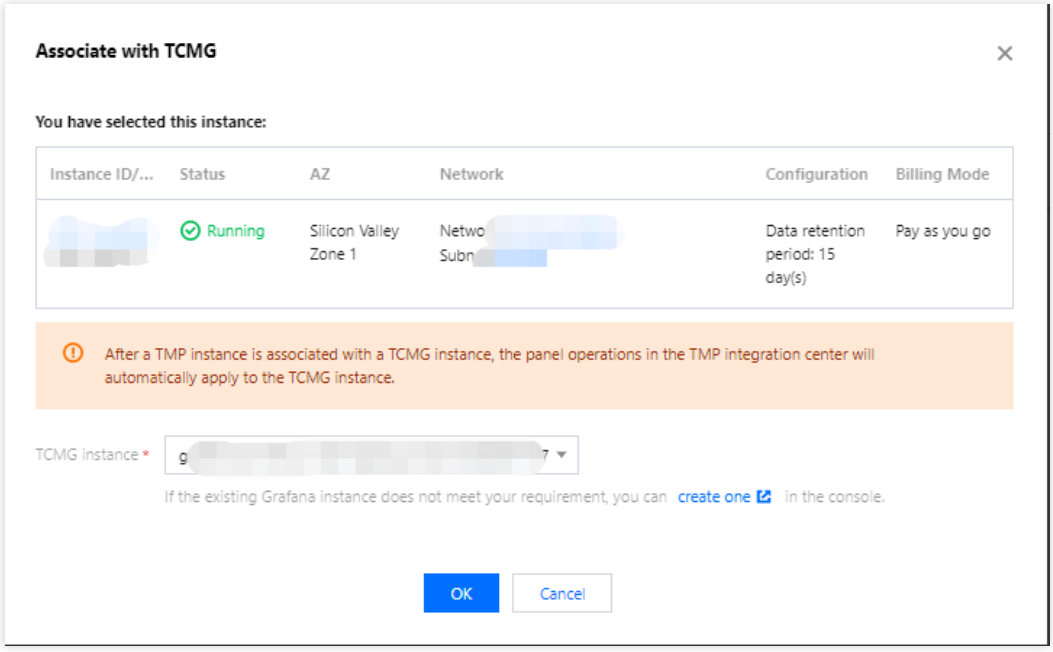
TMP is highly integrated with TencentCloud Managed Service for Grafana (TCMG). One TCMG instance can be bound by multiple TMP instances at the same time to visualized TMP data in a unified way.

Directions

TMP allows you to associate with a TCMG instance when creating a TMP instance. If you don't associate with the TCMG instance when purchasing the TMP service, you can follow the instructions below for binding.

Associating with a TCMG instance

1. Log in to the TMP console.
2. Find the corresponding TMP instance in the TMP instance list, and click **More**>**Grafana**>**Associate with TCMG** in the **Operation** column.
3. Select the TCMG instance in the pop-up window and click **OK**.

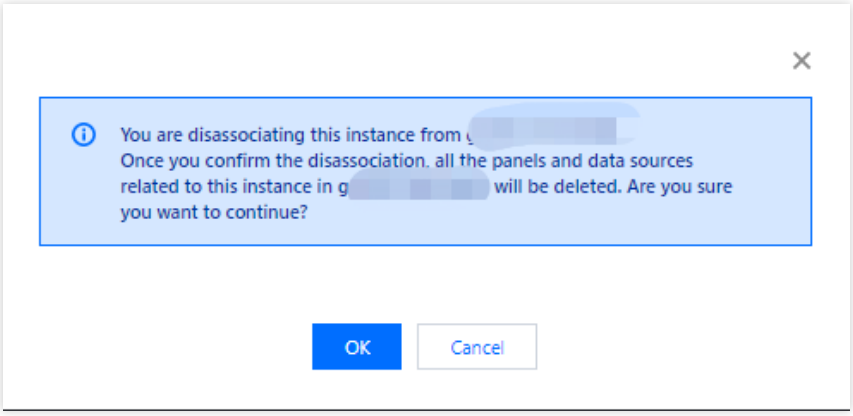


Note

You can only select the TCMG instance in the same VPC (private network) as the TMP instance. If there is no suitable TCMG instance, see Creating Instance to create one.

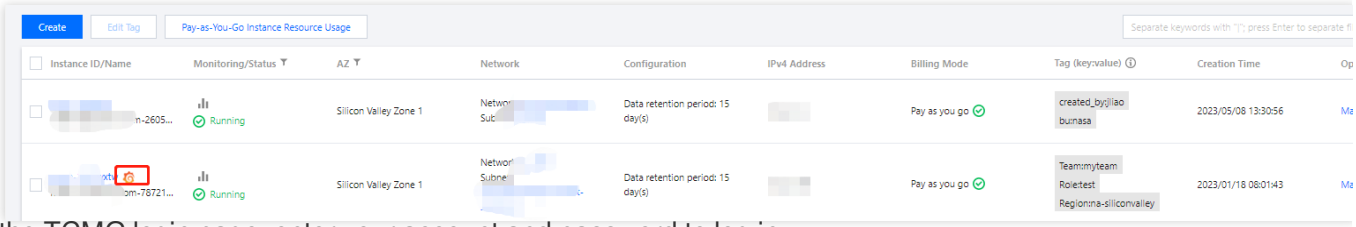
Disassociating from a TCMG instance

1. Log in to the TMP console.
2. Find the corresponding TMP instance in the TMP instance list, and click **More**>**Grafana**>**Disassociate from TCMG** in the **Operation** column.
3. In the pop-up window, click **OK**.



Logging in to the TCMG instance

1. Log in to the TMP console.
2. Find the corresponding TMP instance in the TMP instance list, and click the Grafana icon to the right of the instance ID/name.



3. On the TCMG login page, enter your account and password to log in.

Note

For more TCMG operations such as configuration and image rendering, see TencentCloud Managed Service for Grafana (TCMG).

API Guide
Overview

Last updated：2024-01-29 16:01:55

HTTP API

All stable HTTP APIs of Prometheus are under the path `/api/v1`. When you need to query monitoring data, you can request data through query APIs. To submit data, you can use the remote write protocol or Pushgateway.

Supported APIs

API	Description	Authentication Required	Method
/api/v1/query	Query	Yes	GET/POST
/api/v1/query_range	Range query	Yes	GET/POST
/api/v1/series	Series query	Yes	GET/POST
/api/v1/labels	Labels query	Yes	GET/POST
/api/v1/label/<label_name>/values	Label value query	Yes	GET
/api/v1/prom/write	Data submission through remote write	Yes	remote write
Pushgateway	Data submission through Pushgateway	Yes	SDK

Authentication Method

Authentication is enabled by default, so all APIs require authentication, and all authentication methods support bearer token and basic authentication.

Bearer token

A bearer token is generated as an instance is generated and can be queried in the console. For more information on bearer token, please see Bearer Authentication.

Basic auth

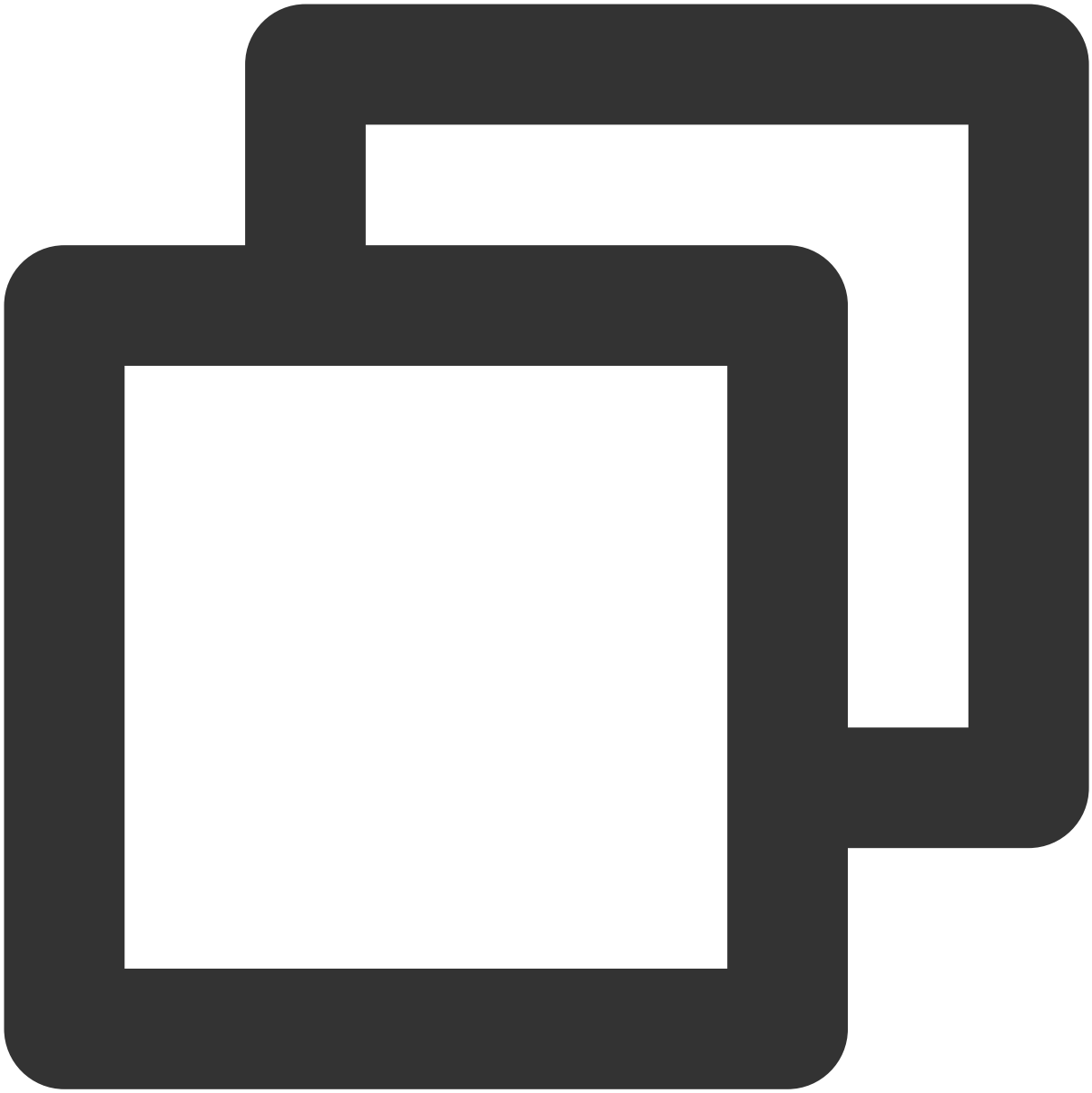
Basic auth is compatible with the native Prometheus query authentication method. The username is your `APPID`, and the password is the bearer token (generated when the instance is generated), which can be queried in the console. For more information on basic auth, please see Basic Authentication.

Data Return Format

The response data of all APIs is in JSON format. Every successful request will return a status code of `2xx`.
An invalid request will return a piece of JSON data containing an error object and a status code as described below:

Status Code	Description
401	Authentication failed
400	A parameter was missing or incorrect
422	An invalid expression couldn't be specified (RFC4918)
503	The query was unavailable or canceled

Below is a response template for invalid requests:



```
{
  "status": "success" | "error",
  "data": <data>,

  // When the `status` is `error`, the following data will be returned
  "errorType": "<string>",
  "error": "<string>",

  // When there is a warning message during request execution, this field will be f
  "warnings": ["<string>"]
}
```

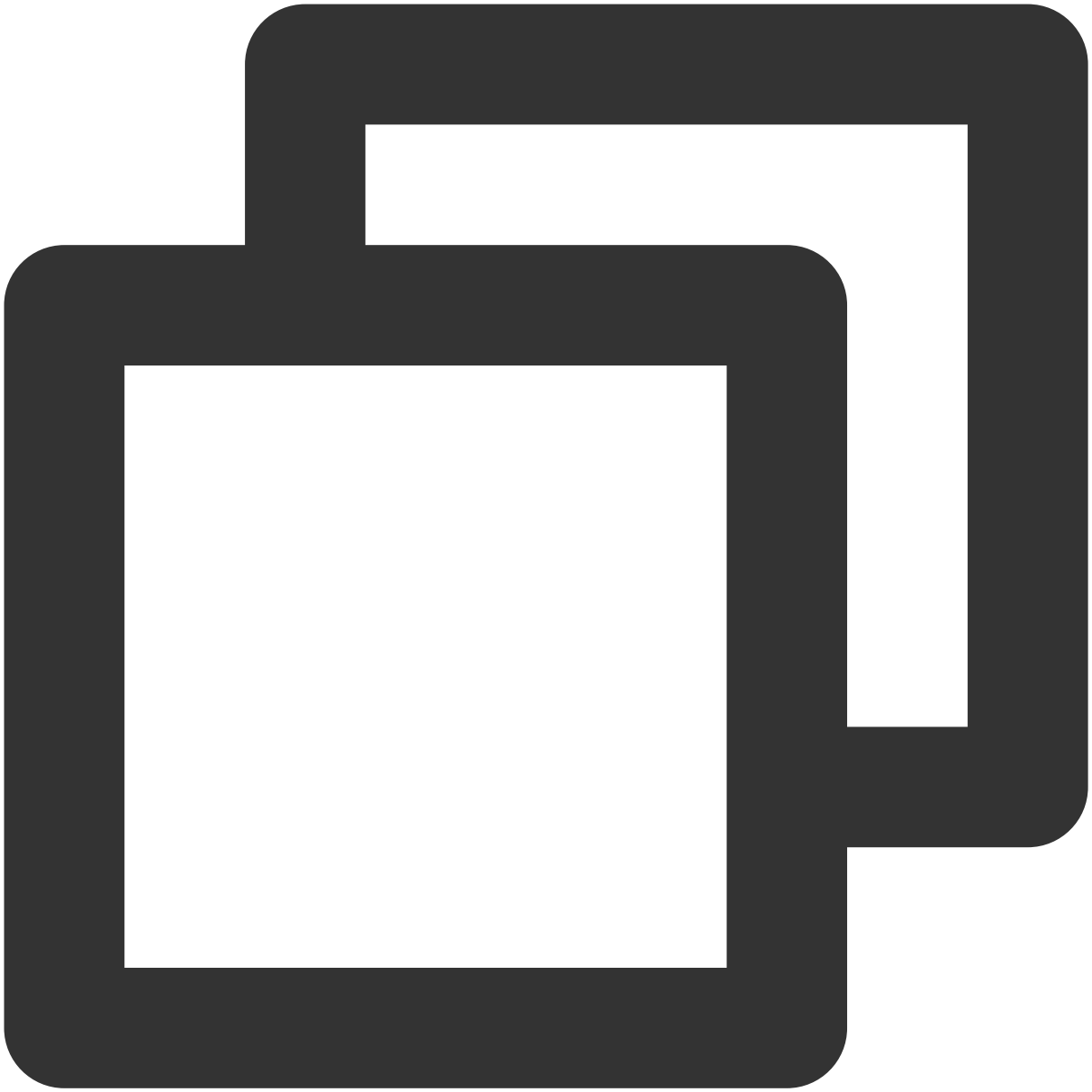
Writing Data

Last updated：2024-01-29 16:01:55

Overview

Similar to the scenario of data reporting by Flink jobs, you need to write data directly to Prometheus through APIs, as the lifecycle of these jobs may be very short, and it would be too late to wait for Prometheus to pull the data. To write data, you can directly use the remote write protocol or Pushgateway.

Remote Write



```
POST /api/v1/prom/write
```

Remote write is a standard protocol of Prometheus. For more information, please see REMOTE WRITE TUNING. With remote write, you can write other Prometheus data in the VPC to TMP, which helps improve the data stability and make migration easier.

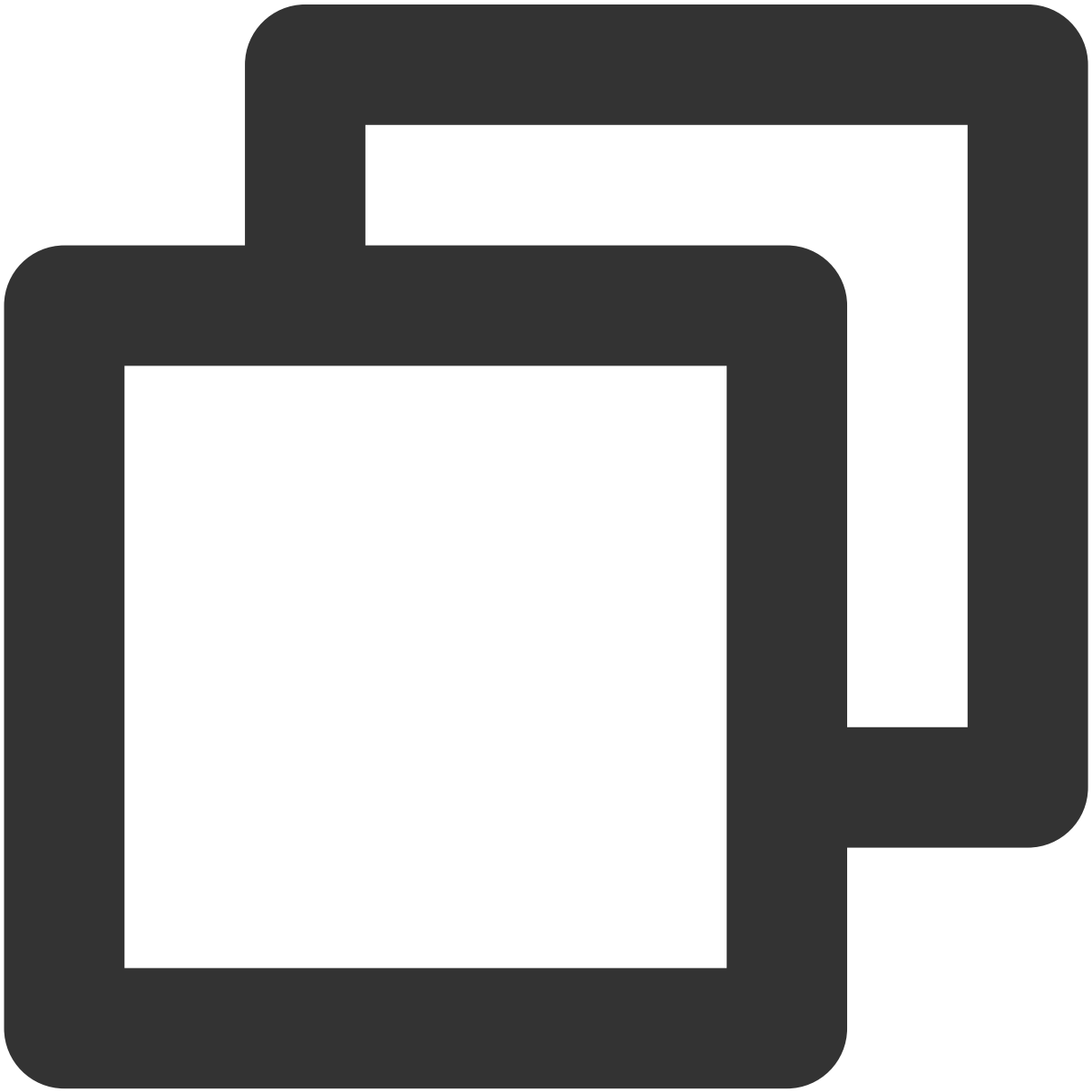
Pushgateway

Although the pull method is recommended in Prometheus, you may still need to use the push method in some scenarios. For more information, please see WHEN TO USE THE PUSHGATEWAY.

TMP is natively integrated with the Pushgateway module, which can directly push data to it. Below is a simple example of pushing data in Go. You need to change the variables of `$IP`, `$PORT`, `$APPID`, and `$TOKEN` to the authentication information of your own instance, which can be queried in the console.

Note:

We strongly recommend you replace `$INSTANCE` with an identifier of the current machine, such as `IP/Hostname`. If this `groupingKey` is not set, when multiple machines report data together, the data entries will overwrite each other.



```
package main

import (
    "fmt"
    "time"

    "github.com/prometheus/client_golang/prometheus"
    "github.com/prometheus/client_golang/prometheus/push"
    "github.com/prometheus/common/expfmt"
)

var completionTime = prometheus.NewGauge(prometheus.GaugeOpts{
    Name: "db_backup_last_completion_timestamp_seconds",
    Help: "The timestamp of the last successful completion of a DB backup.",
})

func do() {
    completionTime.SetToCurrentTime()
}

func ExamplePusher_Push() {
    if err := push.New("http://$IP:$PORT", "db_backup").
        BasicAuth("$APPID", "$TOKEN").
        Collector(completionTime).
        Grouping("instance", "$INSTANCE").
        Grouping("db", "customers").
        Format(expfmt.FmtText).
        Push(); err != nil {
        fmt.Println("Could not push completion time to Pushgateway:", err)
    }
}

func main() {
    do()
    ticker := time.NewTicker(2 * time.Second)
    done := make(chan bool)
    for {
        select {
        case <-done:
            return
        case <-ticker.C:
            ExamplePusher_Push()
        }
    }
}
```

Note:

You can customize the HTTP Client through the `Client` method for the object generated by `push.New`. We recommend you set an appropriate timeout period. In addition, if data is pushed, we recommend you use an async method for calls so as to avoid blocking the primary business process.

For other reporting scenarios, please see Custom Monitoring.

Querying Monitoring Data

Last updated：2024-01-29 16:01:55

Overview

When you need to query monitoring data, you can request data through query APIs.

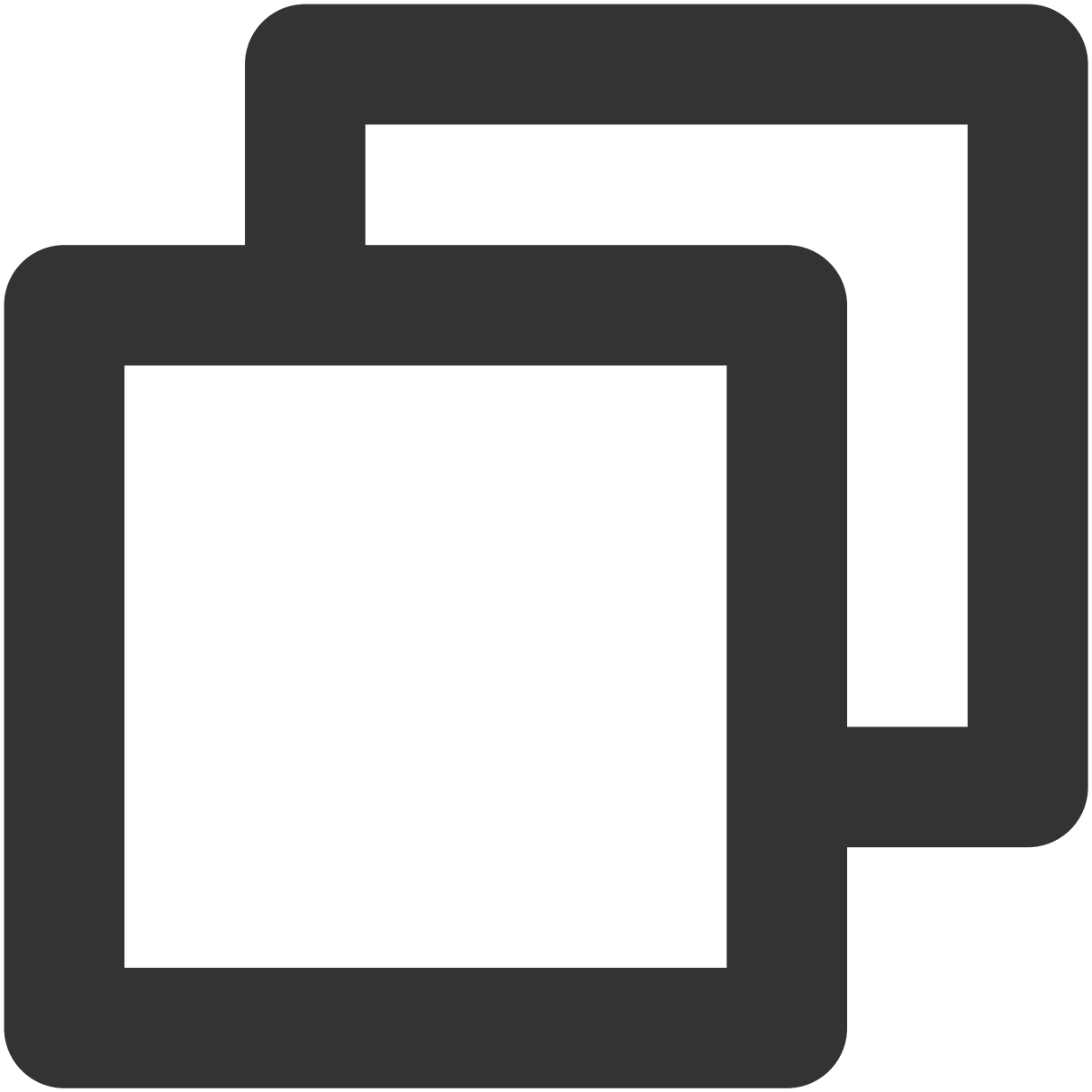
APPID/Token Acquisition Method

TMP uses `APPID` + `Token` to authenticate the access.

`APPID` can be obtained here.

`Token` can be obtained from the basic information of the corresponding TMP instance.

Query APIs



```
GET /api/v1/query
POST /api/v1/query
```

Query Parameters

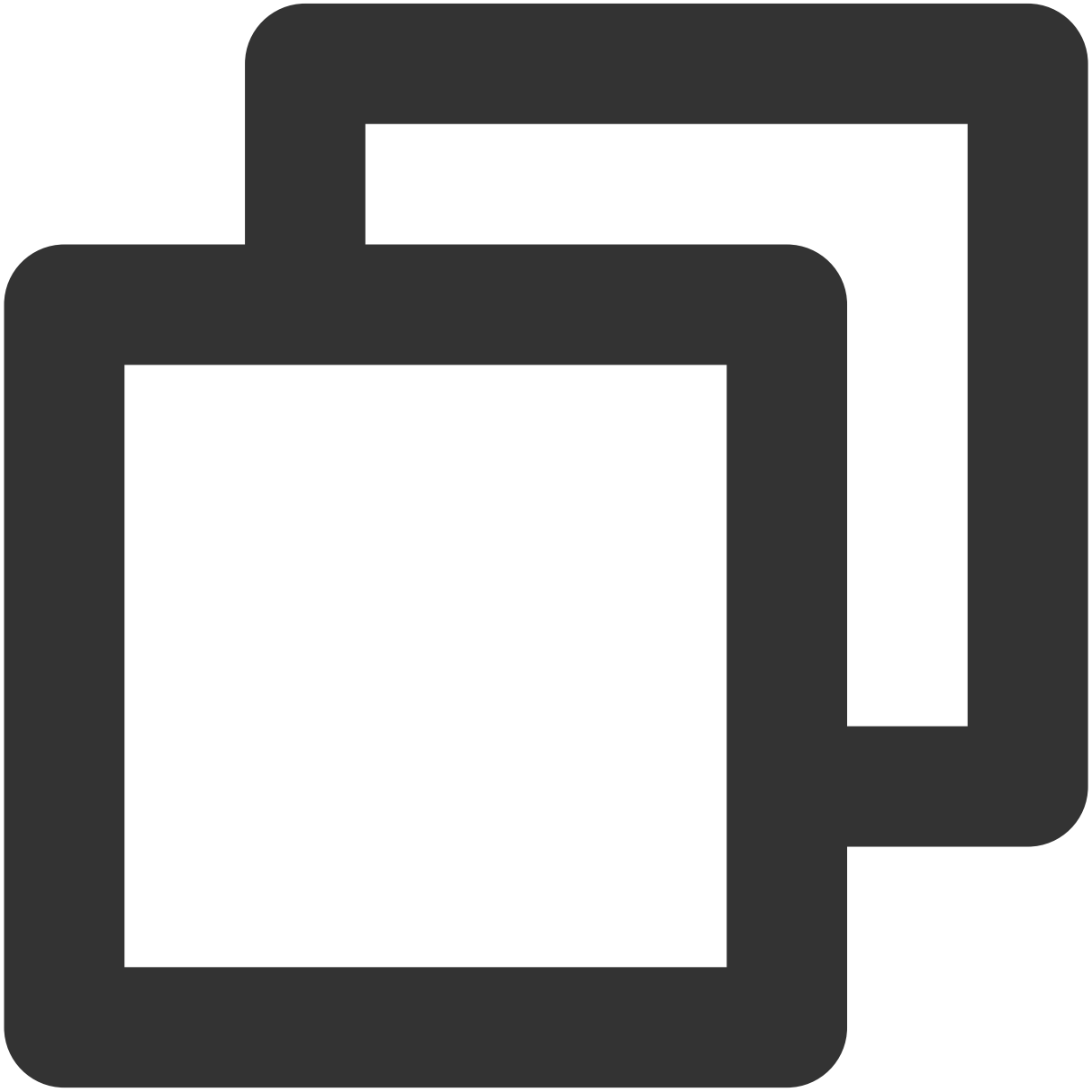
query=<string>: Prometheus: query expression.

time=<rfc3339 | unix_timestamp>: timestamp, which is optional.

timeout=<duration>: detection timeout period, which is optional and specified by the `-query.timeout` parameter by default.

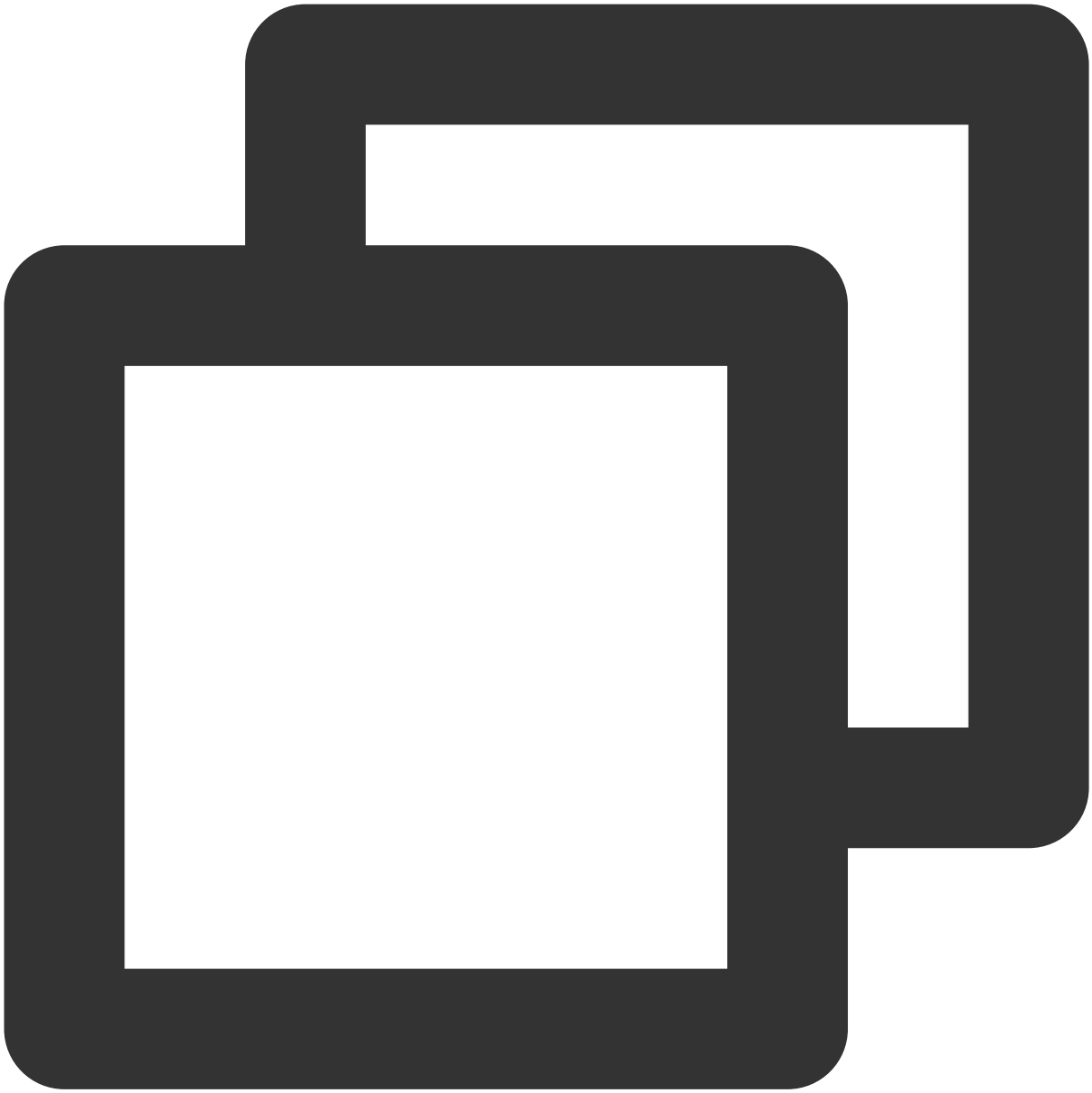
Sample Simple Query

You can use the following sample to query data through an API. The query service address and authentication information can be viewed in the corresponding instance's information in the console:



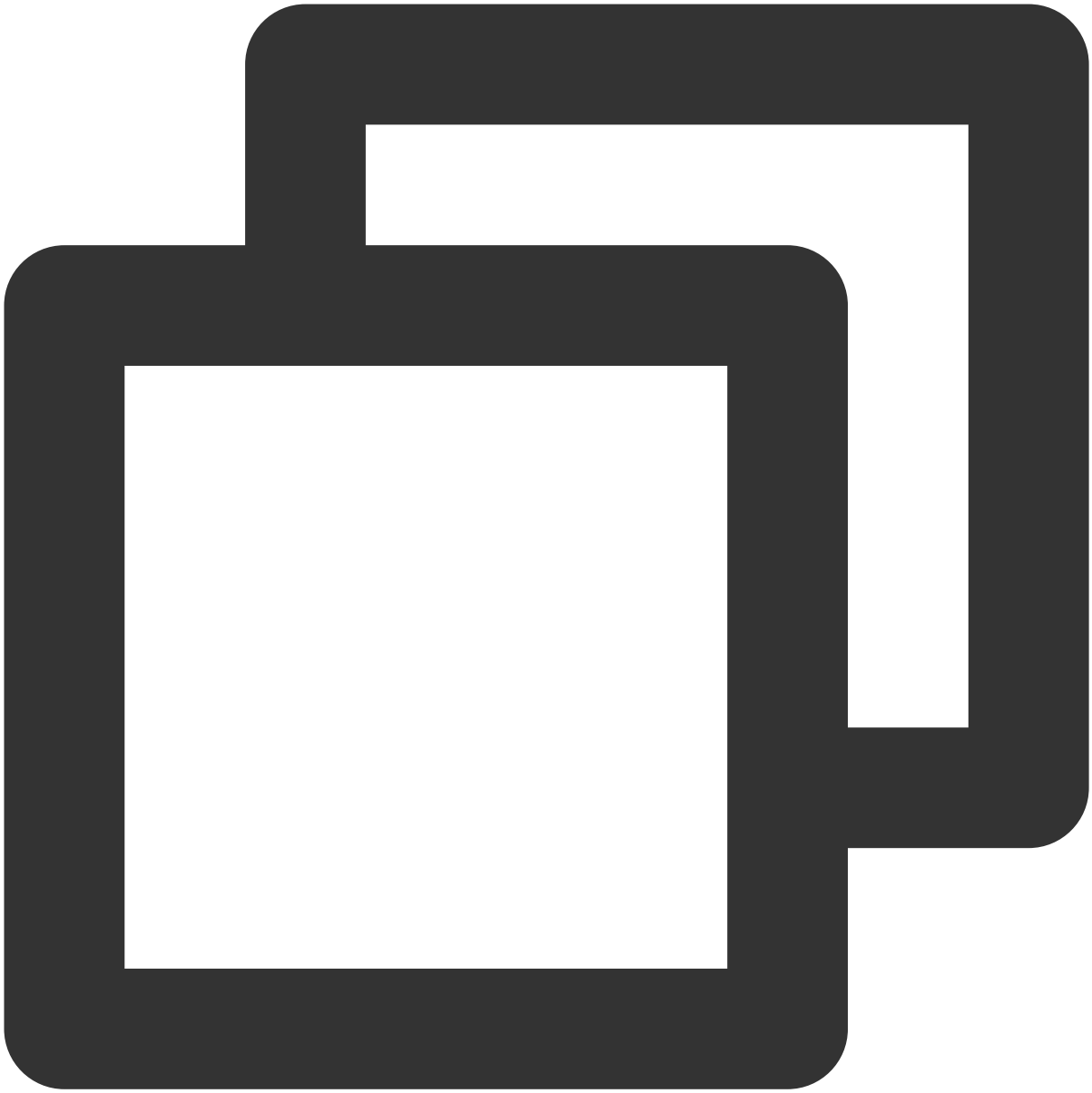
```
curl -u "appid:token" 'http://IP:PORT/api/v1/query?query=up'
```

If the returned status code is 401, please check whether the authentication information is correct.



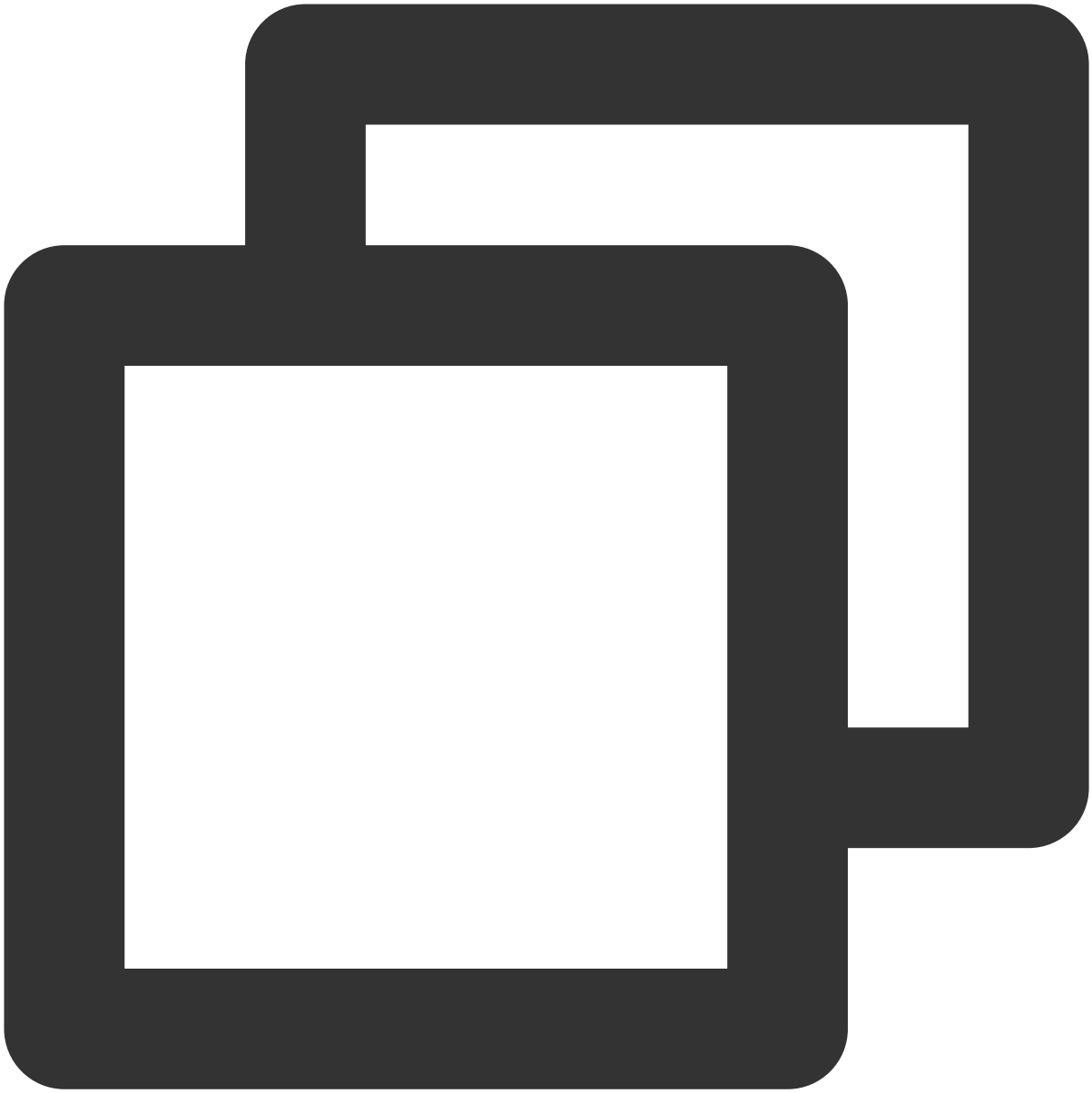
```
< HTTP/1.1 401 Unauthorized
< Content-Length: 0
```

Range Query



```
GET /api/v1/query_range
POST /api/v1/query_range
```

Querying data by time range is the most common query scenario. To do so, you can use the `/api/v1/query_range` API as shown below:



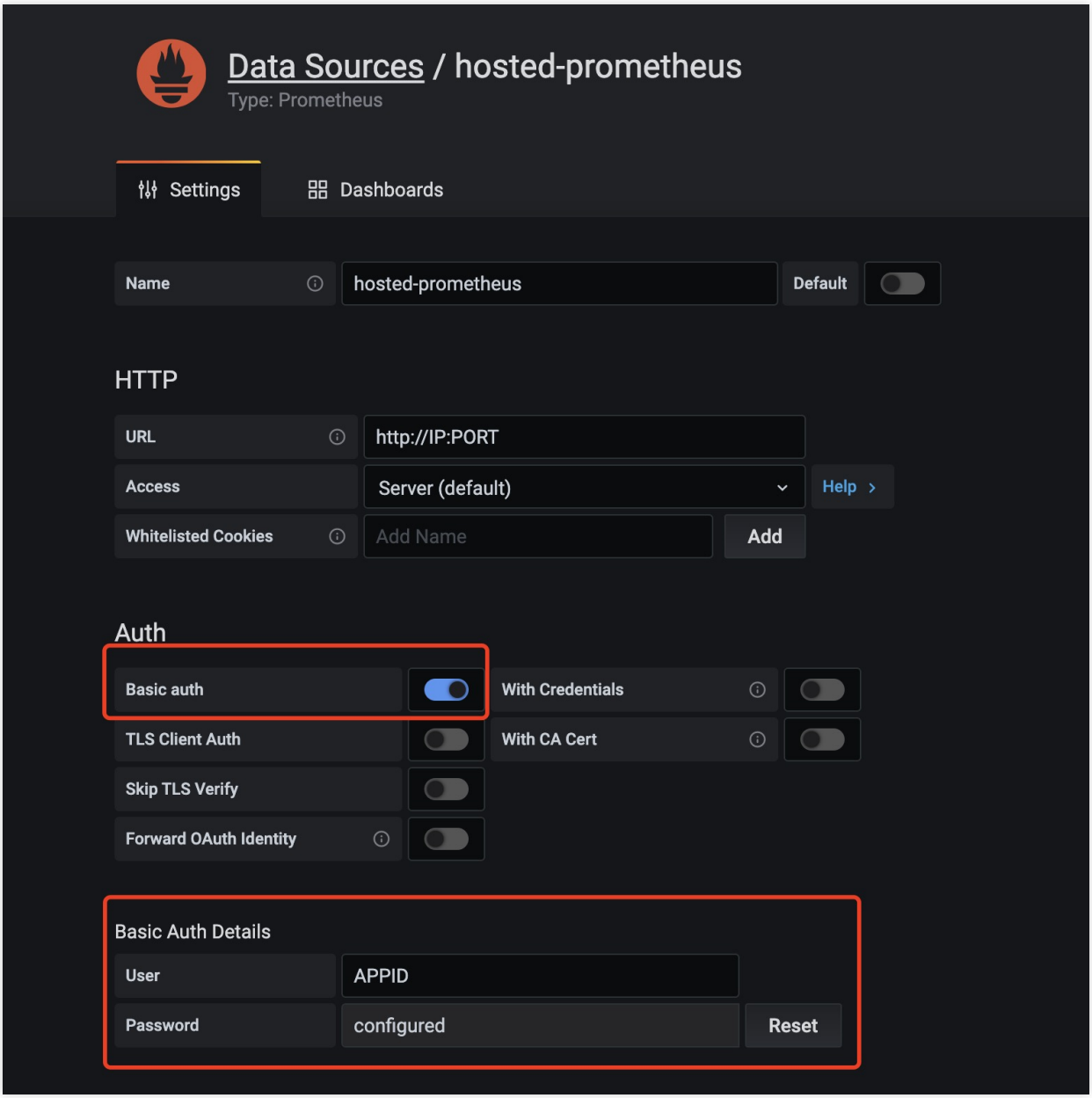
```
$ curl -u "appid:token" 'http://IP:PORT/api/v1/query_range?query=up&start=2015-07-0
{
   "status" : "success",
   "data" : {
      "resultType" : "matrix",
      "result" : [
         {
            "metric" : {
               "__name__" : "up",
               "job" : "prometheus",
               "instance" : "localhost:9090"
```

```
            },
            "values" : [
               [ 1435781430.781, "1" ],
               [ 1435781445.781, "1" ],
               [ 1435781460.781, "1" ]
            ]
         },
         {
            "metric" : {
               "__name__" : "up",
               "job" : "node",
               "instance" : "localhost:9091"
            },
            "values" : [
               [ 1435781430.781, "0" ],
               [ 1435781445.781, "0" ],
               [ 1435781460.781, "1" ]
            ]
         }
      ]
   }
}
```

Adding Data Source to Self-Built Grafana

You can add TMP as a data source in your self-built Grafana, and then you can view data in Grafana, provided that they are in the same VPC and can access each other over the network.

Enable the `BasicAuth` authentication method and enter the corresponding authentication information as shown below:



Instance Diagnosis

Last updated：2024-07-30 18:14:31

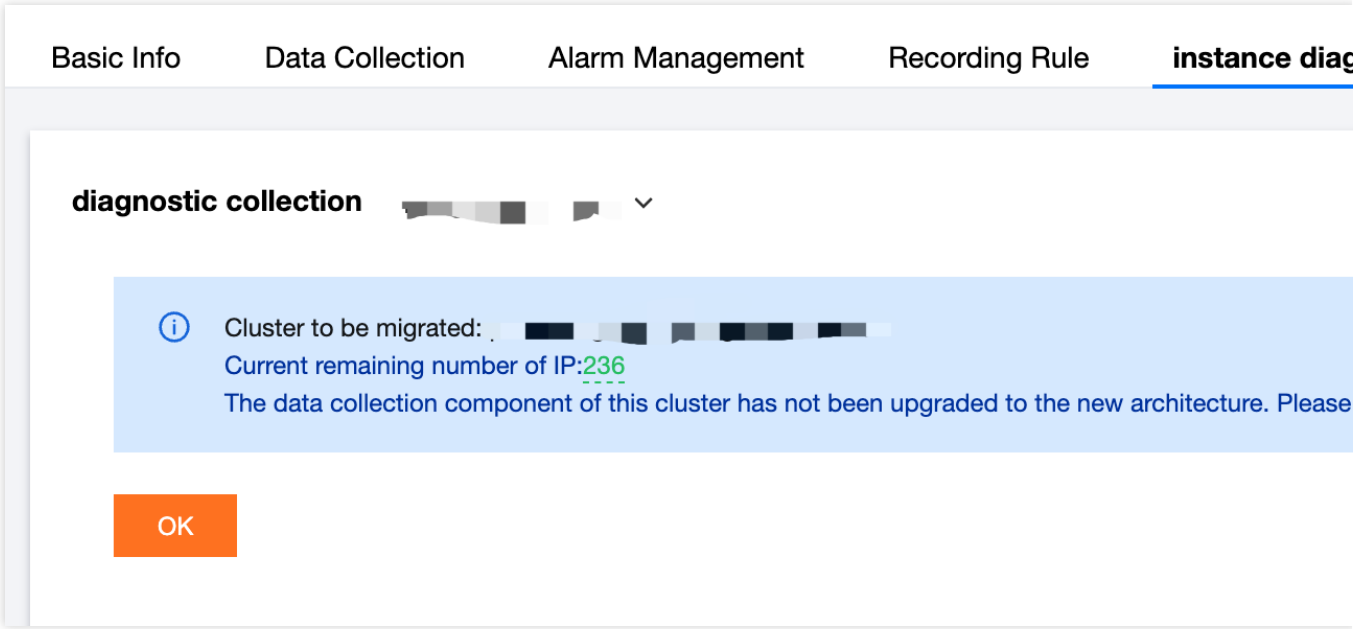
Background

To enhance the user experience with the Prometheus monitoring collection terminal, we now offer a new collection terminal architecture. The upgraded new architecture supports instance diagnosis, system health checks, and improves the resource utilization rate of the collection agent and the stability of metric collection. This document will guide users in upgrading the old collection architecture to the new one via the console instance diagnosis page and obtain detailed information about the current instance collection and storage for a better experience.

Directions

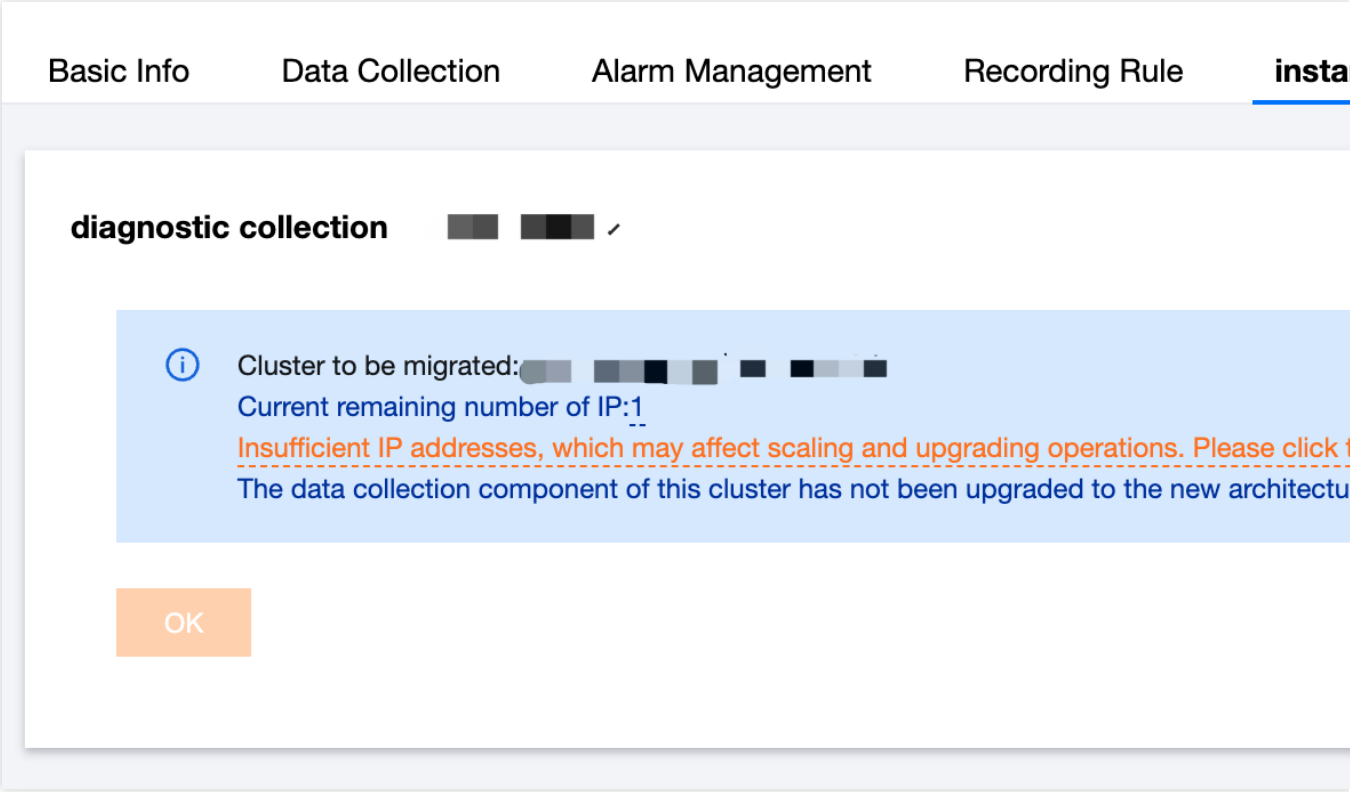
Upgrading to the New Architecture

1. Log in to TMP Console.
2. In the Prometheus instance list, click **Instance ID/Name**.
3. In the Prometheus management center, click **instance diagnostics** on the top navigation bar, and select the corresponding **collection cluster**.
Click **OK** to upgrade to the new architecture.



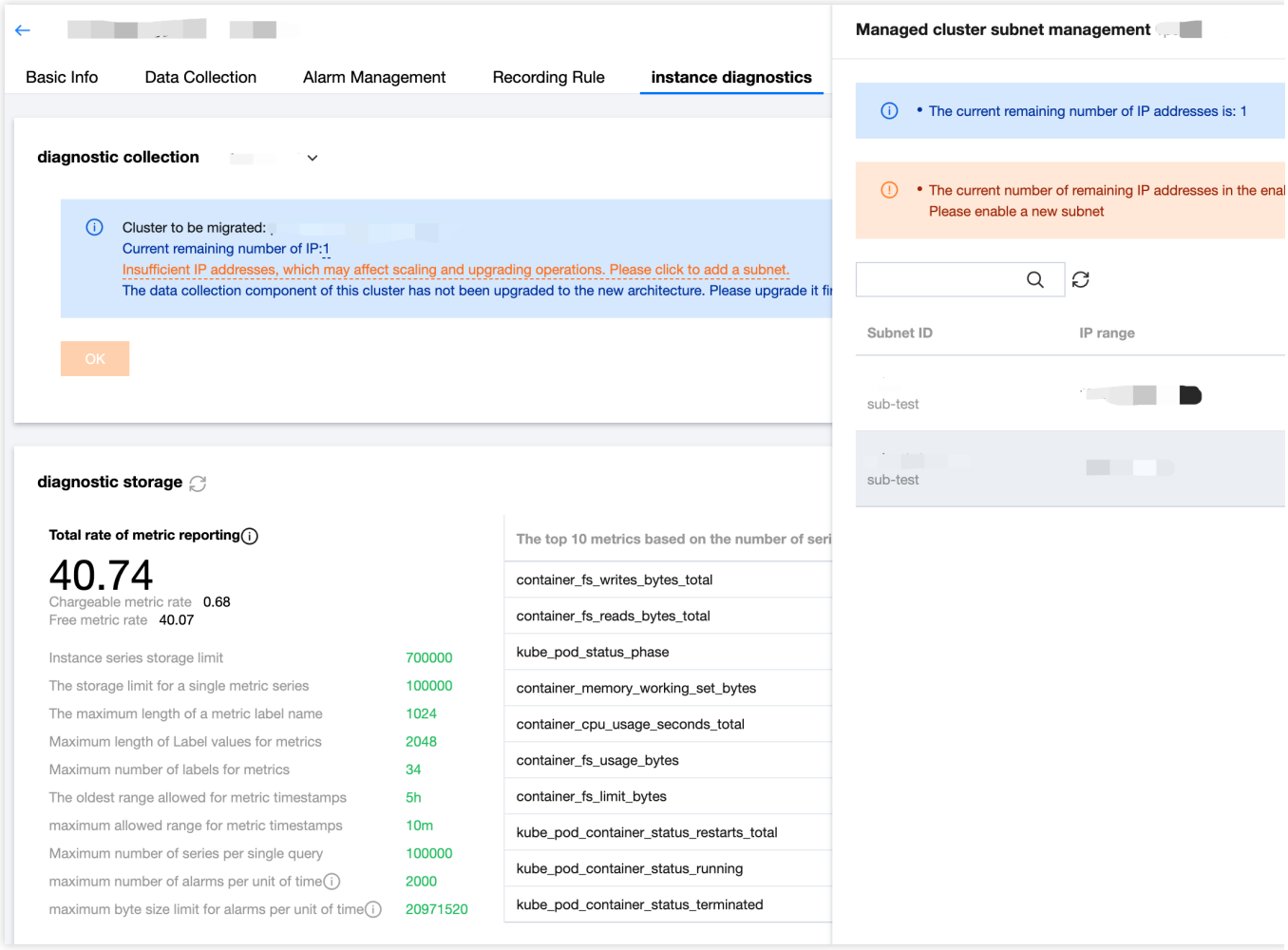
Note:

The upgrade process is expected to take 5 minutes and may experience a 1-2 minute metric interruption.
If the number of IPs in the managed cluster of the instance is less than 10, a risk warning will appear. To prevent the issue of insufficient IPs from causing component upgrade failures, please increase the available IPs as guided below.



Adding Subnet

1. On the instance diagnosis page, click subnet information to enter the **Managed Cluster Subnet Management** page.



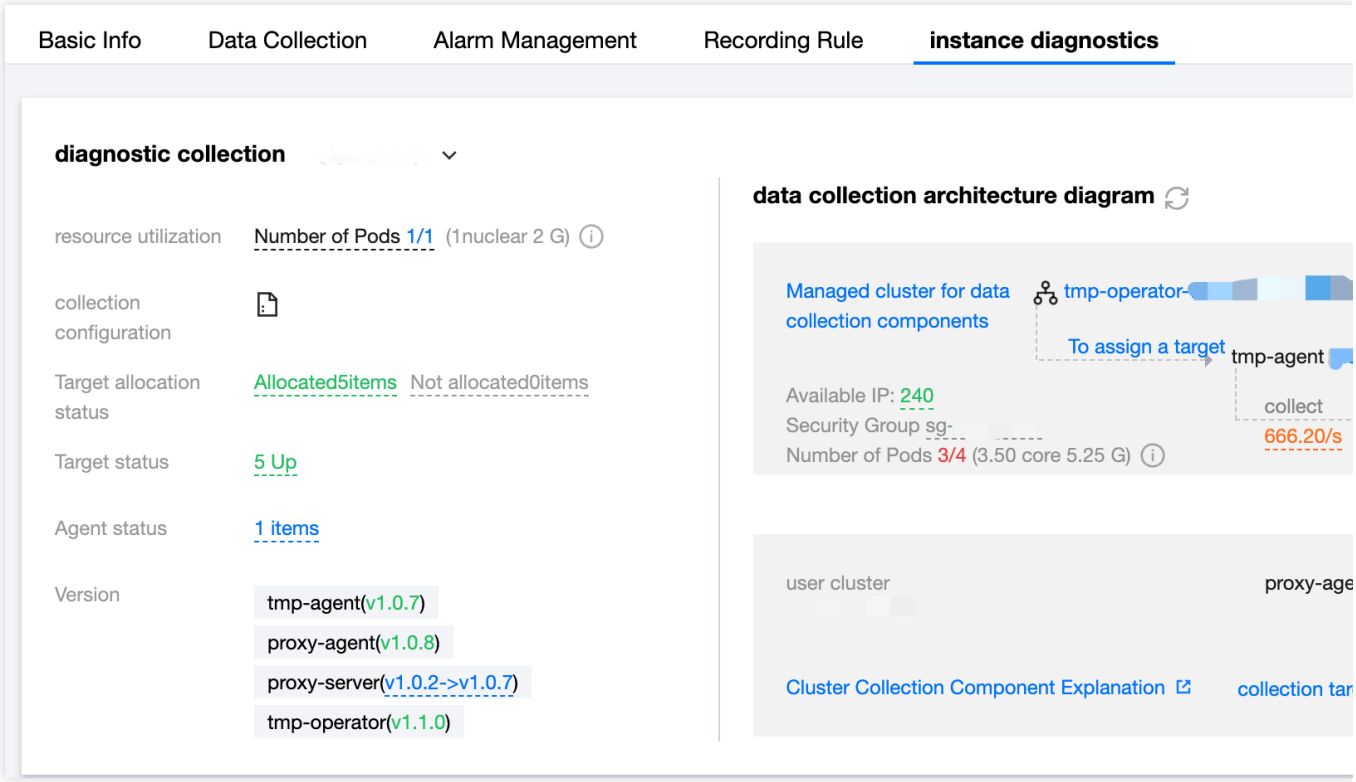
2. The page displays the subnets that have already been added to the managed cluster and those that have not in the current VPC. You can click **Enable Subnet** to enable subnets with sufficient remaining IPs as per your plan.
3. If there are no available subnets, please Add Subnet first and then enable it on the current page.

Instance Diagnosis

After the architecture is upgraded, the instance diagnosis page will include content on both collection and storage, helping users understand the operating status of Prometheus collection and storage to locate issues more quickly.

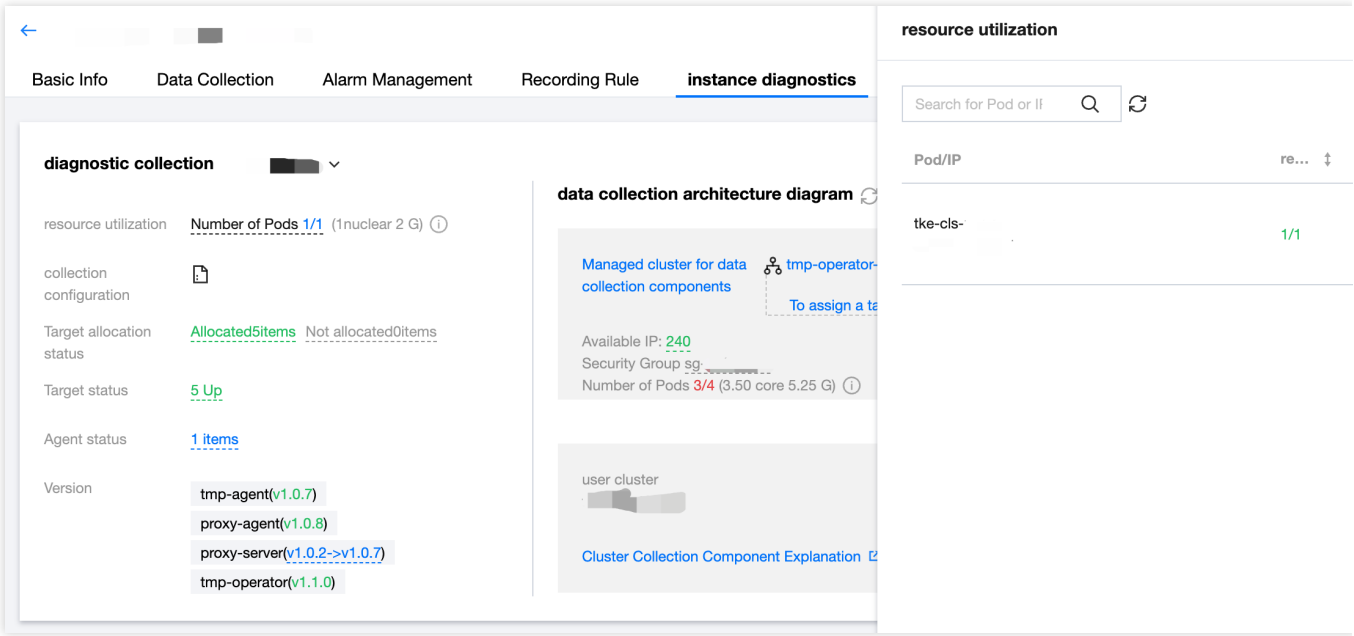
Diagnostic Collection

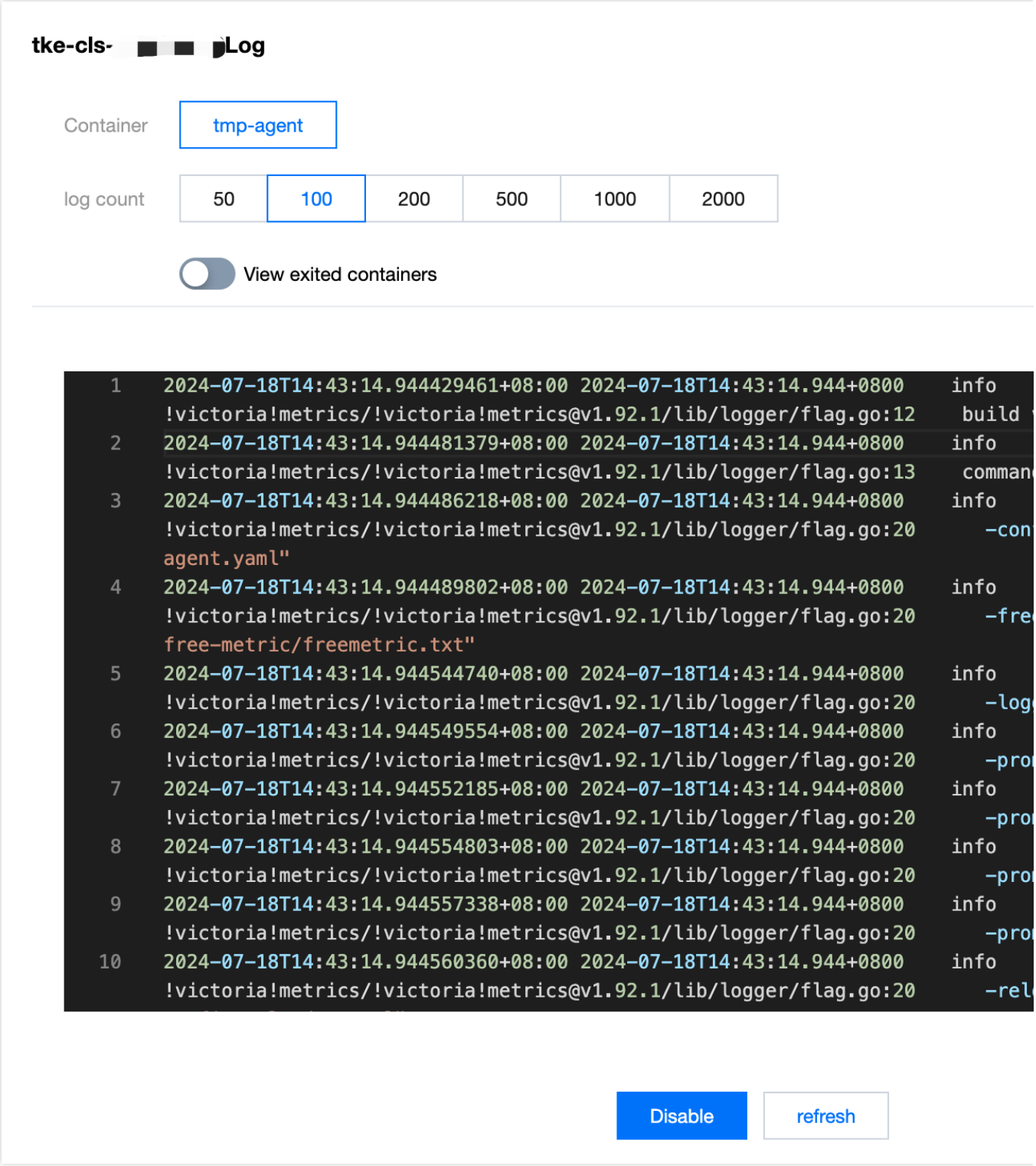
As shown in the following figure, collection diagnosis includes resource occupancy of the corresponding collection, collection configuration, target allocation status, target status, agent status, component version, and the collection architecture diagram.



Resource Utilization

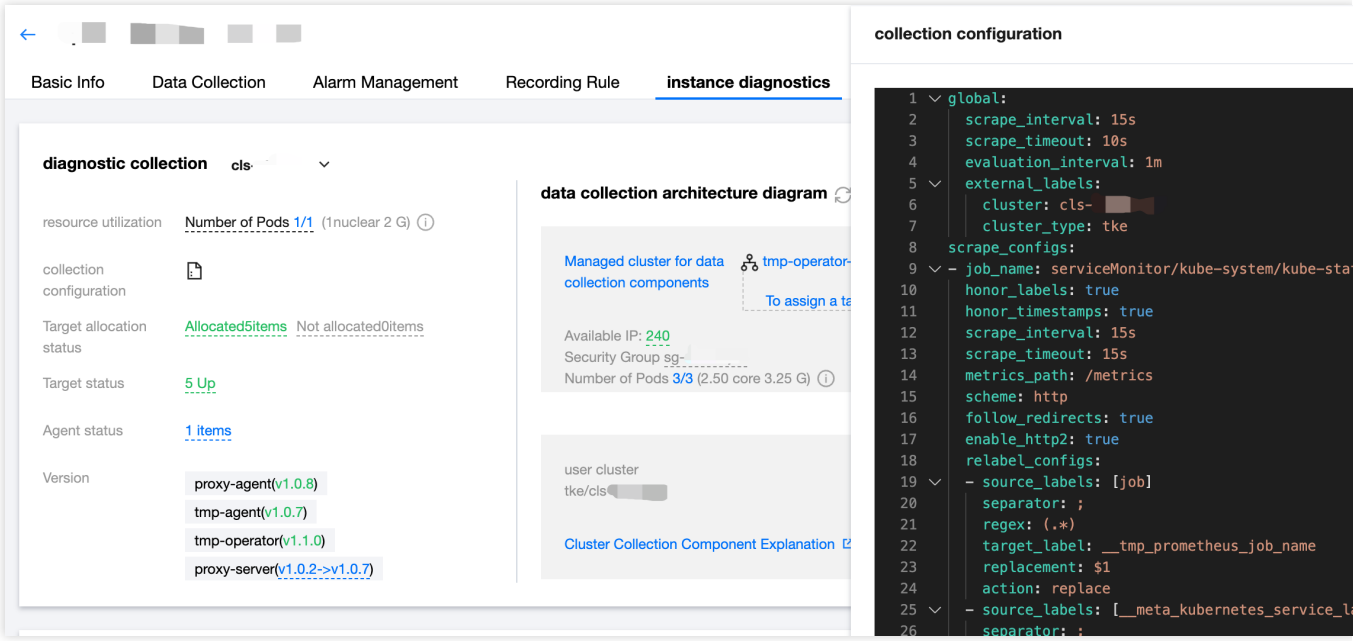
Resource occupancy of the collection shard displays information including CPU, memory limit and occupancy, and inbound and outbound traffic. Click **View logs** to see the logs of the Pod, facilitating the review of running details and troubleshooting of exceptions.





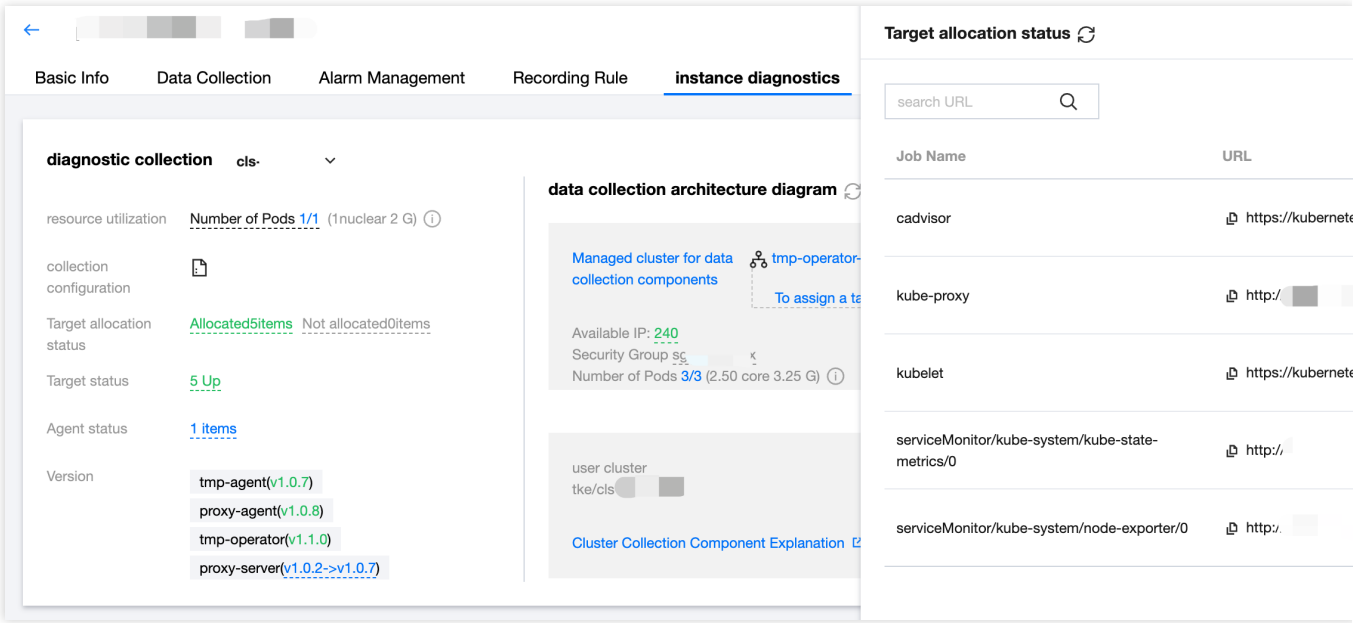
Collection Configuration

It displays the specific collection configuration of the current collection.



Target Allocation Status

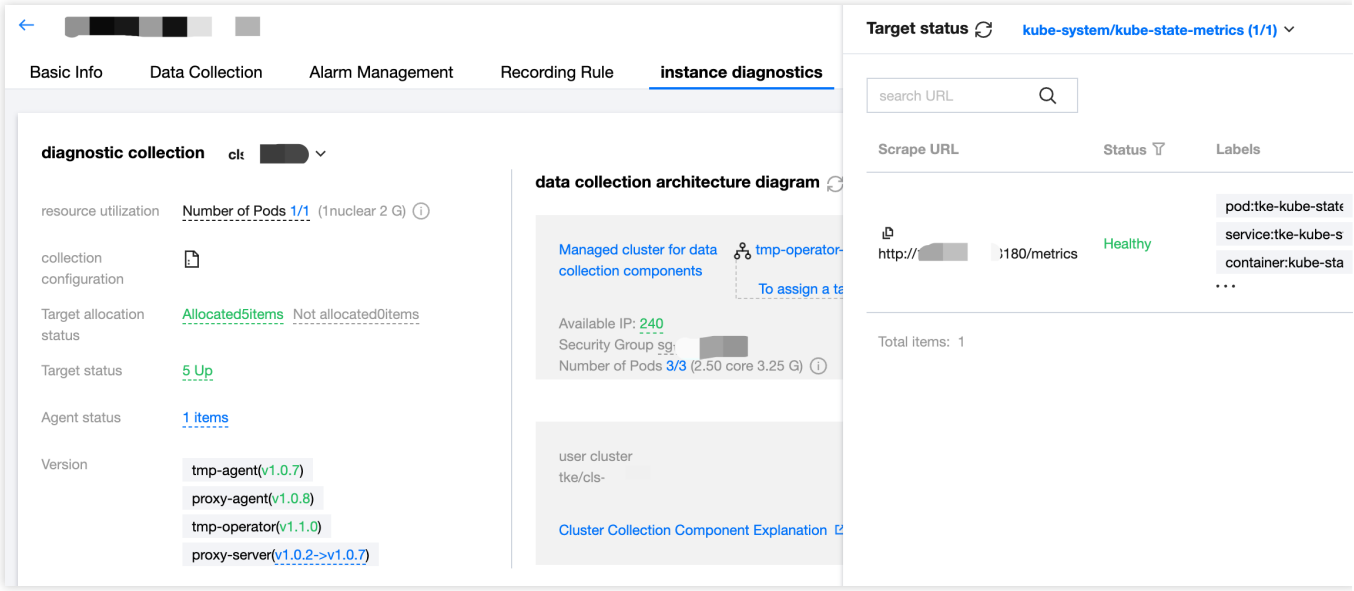
It shows the URL of the collection target, the name of the collection job to which the target belongs, and which collection shard is currently collecting it.



Target Status

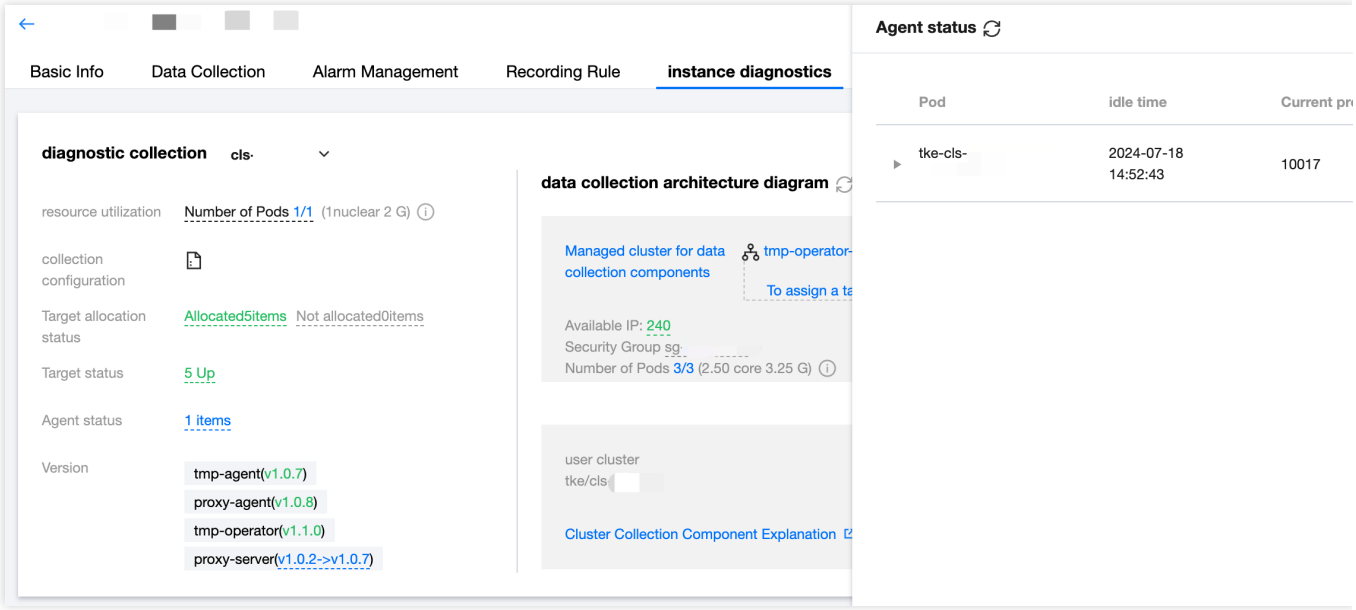
You can filter active collection targets based on the collection job, and obtain information about the corresponding target such as status, Labels, and Discovered Labels. The target status is "Healthy" in normal conditons, but if it is "abnormal" and there has already been the last scrape time, you can troubleshoot based on the error message on the

far right. Common issues may include the target itself being unhealthy, incorrect permission configuration, and network errors.



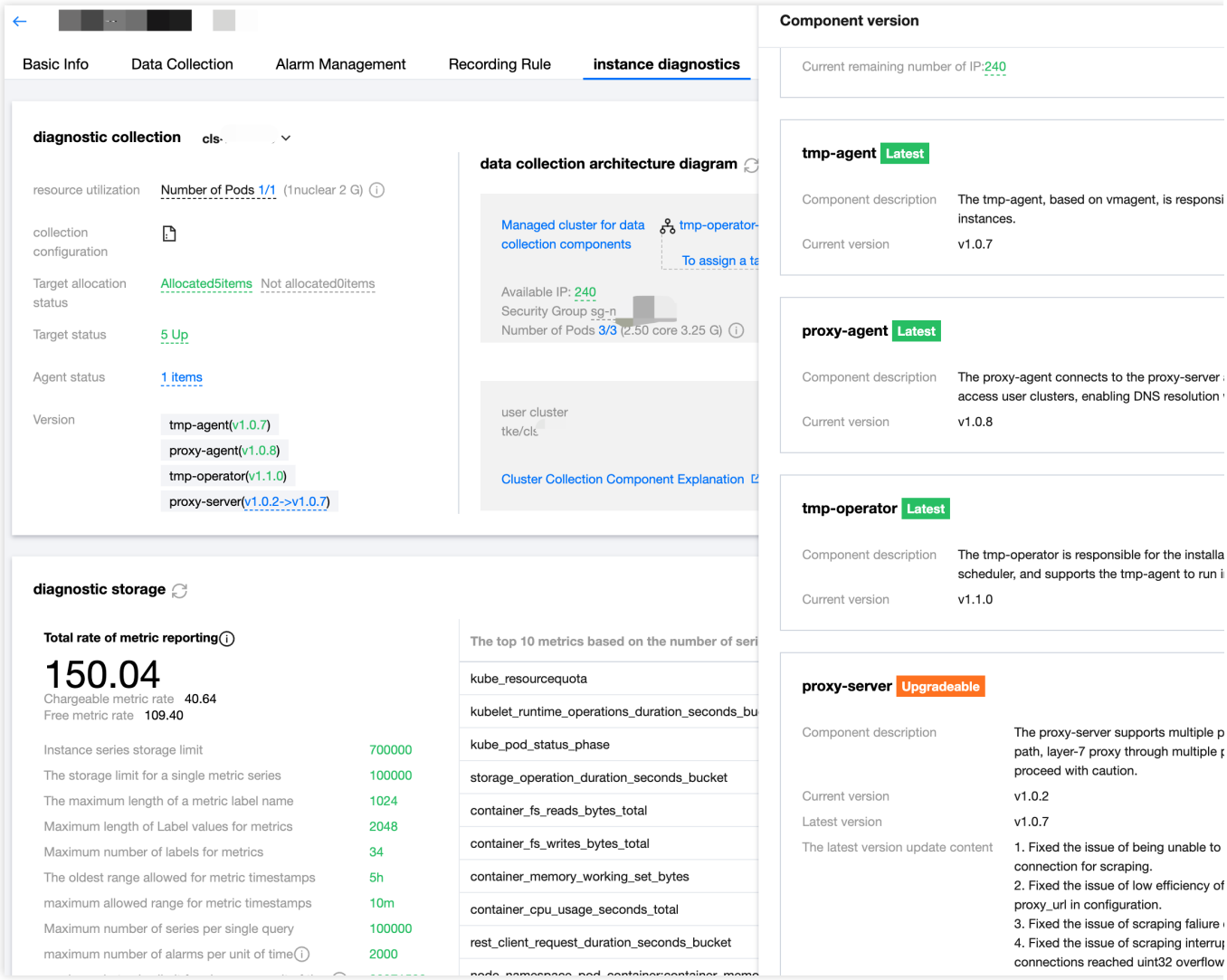
Agent Status

It shows the running status of the collection shard agent. Click **View logs** to see the logs of the Pod to understand running details and troubleshoot exceptions.



Component Version

It shows the component version information of the current collection. On the **Component version** page, it displays IP quantity check and the current version, latest version, and component description, upgrade description of each component version.



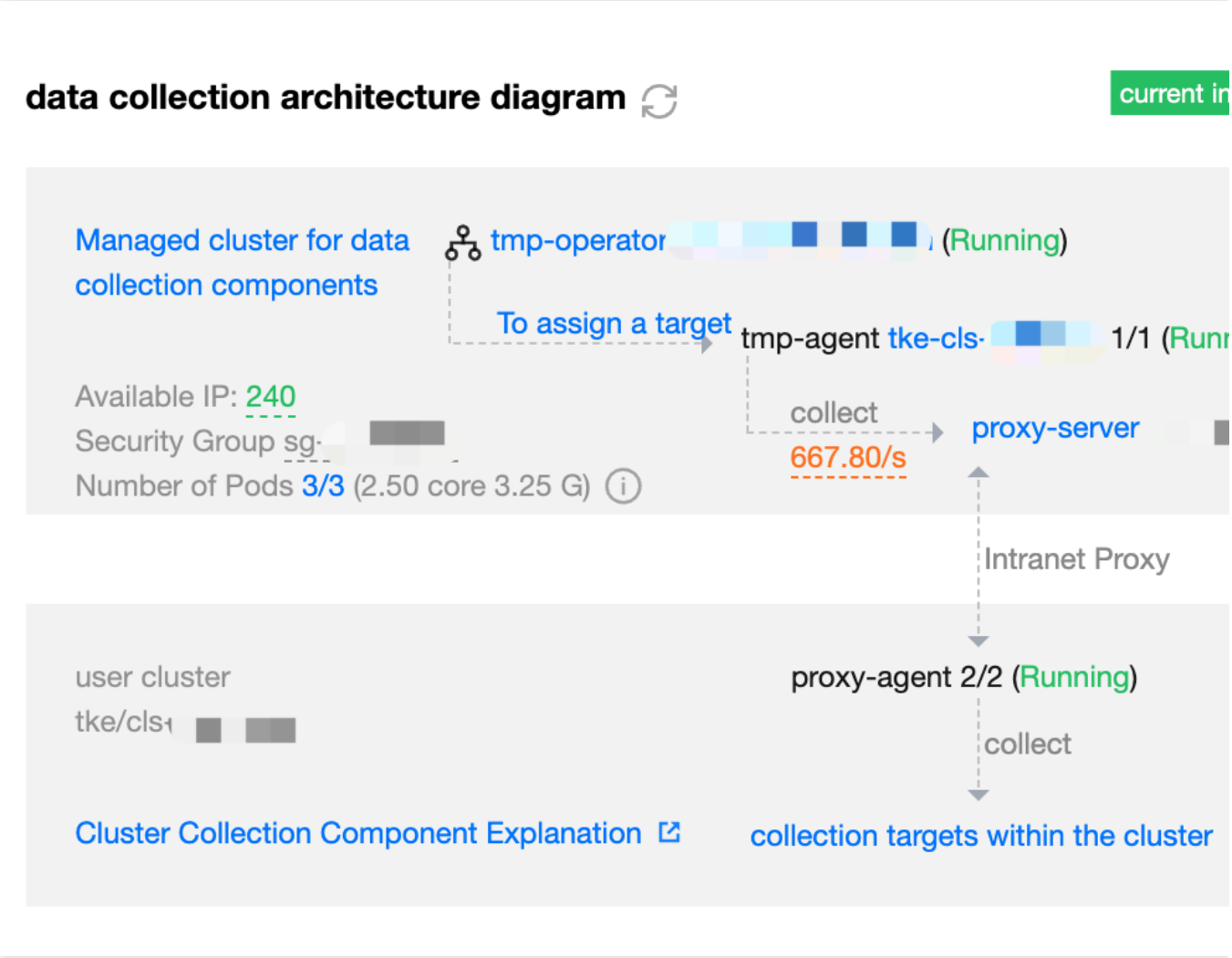
Note:

Please try to keep the collection component in the **latest version**, which can be upgraded through the **Upgrade** of the corresponding component.

Components tmp-operator, tmp-agent, and proxy-agent can be upgraded without impact under normal circumstances. During the upgrade of the proxy-server component, there will be collection breakpoints. The breakpoint duration is the time for eks to activate the component Pod, and it will affect the collection of the entire Prometheus instance (including the integration center and container clusters). Please operate cautiously.

Data Collection Architecture Diagram

The collection architecture diagram provides information about the current collection architecture.



Collection Component Managed Cluster:

Available IP count. When the number of IPs is insufficient, an **Insufficient IP Count** alert will appear. Click to enter the **Managed Cluster Subnet Management** page. For specific operations, see Adding Subnet.

Managed cluster's security group and its pass-through requirements during normal operation. Some network issues during the collection may be caused by the security group not allowing pass-through.

The number of Pods and resource utilization related to the current collection

Running status of the collection scheduling component tmp-operator

Collection target allocation status

Running status of the collection shard component tmp-agent

Metric collection rate and inbound bandwidth

Running status of the proxy component proxy-server

Metric write rate and outbound bandwidth

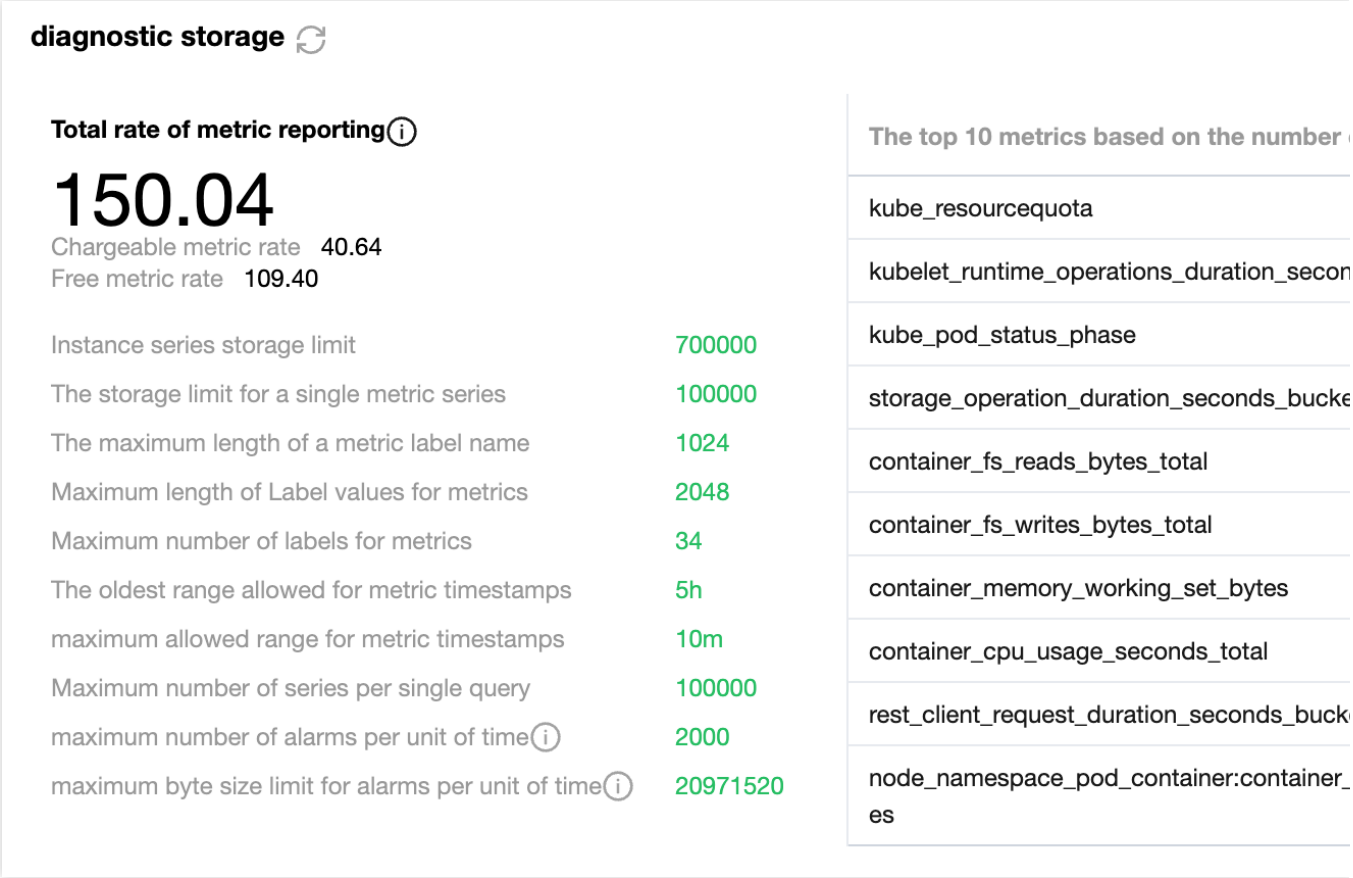
Write target status, including data overwrite of the current instance's corresponding storage and user configuration.

User Cluster:

Status of the public network proxy CLB (if enabled)
Running status of the proxy component proxy-agent;
Status of the collection target within the cluster

Storage Diagnosis

As shown in the figure, the storage diagnosis includes storage-related status and limitations.



Parameter Description

Parameter	Description
Total rate of metric reporting	Includes free metrics and paid metrics.
Instance series storage limit	The number of active series exceeding this value will cause the corresponding instance series discarded due to limit exceeded.
The storage limit for a single metric series	Different labels for the same metric name constitute different series. Exceeding this value will result in discard due to limit exceeded.
The maximum length of a metric label name	Exceeding this value will result in discard due to invalid length.

Maximum number of labels for metrics	Exceeding this value will result in discard due to invalid length.
The oldest range allowed for metric timestamps	Indicates the oldest timestamp acceptable in a single series (out of order not allowed).
maximum allowed range for metric timestamps	Indicates the latest timestamp acceptable in a single series (out of order not allowed).
Maximum number of series per single query	Indicates the maximum number of series involved in data query. It is recommended to shorten the range query time or use instant query in suitable scenarios.
Maximum number of alarms per unit of time	Indicates the maximum number of alarms triggered within 5 minutes.
Maximum byte size limit for alarms per unit of time	Indicates the total size limit of fields (such as lable and annotation) for alarms triggered within 5 minutes.
The top 10 metrics based on the number of series	Same metric name and different label keys are considered different series. Storage is subject to the upper limit for single metric series, and a large quantity can cause high cardinality issue.

TKE Metrics

Free Metrics in Pay-as-You-Go Mode

Last updated：2024-01-29 16:01:55

For pay-as-you-go instances with a storage period of more than 15 days, storage fees for their free metrics will be charged based on the excessive storage period.

Configuration File	Metric Name
node-exporter	node_boot_time_seconds
node-exporter	node_context_switches_total
node-exporter	node_cpu_seconds_total
node-exporter	node_disk_io_now
node-exporter	node_disk_io_time_seconds_total
node-exporter	node_disk_io_time_weighted_seconds_total
node-exporter	node_disk_read_bytes_total
node-exporter	node_disk_read_time_seconds_total
node-exporter	node_disk_reads_completed_total
node-exporter	node_disk_write_time_seconds_total
node-exporter	node_disk_writes_completed_total
node-exporter	node_disk_written_bytes_total
node-exporter	node_filefd_allocated
node-exporter	node_filesystem_avail_bytes
node-exporter	node_filesystem_free_bytes
node-exporter	node_filesystem_size_bytes
node-exporter	node_load1
node-exporter	node_load15
node-exporter	node_load5

node-exporter	node_memory_Buffers_bytes
node-exporter	node_memory_Cached_bytes
node-exporter	node_memory_MemAvailable_bytes
node-exporter	node_memory_MemFree_bytes
node-exporter	node_memory_MemTotal_bytes
node-exporter	node_netstat_TcpExt_ListenDrops
node-exporter	node_netstat_Tcp_ActiveOpens
node-exporter	node_netstat_Tcp_CurrEstab
node-exporter	node_netstat_Tcp_InSegs
node-exporter	node_netstat_Tcp_OutSegs
node-exporter	node_netstat_Tcp_PassiveOpens
node-exporter	node_network_receive_bytes_total
node-exporter	node_network_transmit_bytes_total
node-exporter	node_sockstat_TCP_alloc
node-exporter	node_sockstat_TCP_inuse
node-exporter	node_sockstat_TCP_tw
node-exporter	node_sockstat_UDP_inuse
node-exporter	node_sockstat_sockets_used
node-exporter	node_uname_info
cadvisor	container_cpu_usage_seconds_total
cadvisor	container_fs_limit_bytes
cadvisor	container_fs_reads_bytes_total
cadvisor	container_fs_usage_bytes
cadvisor	container_fs_writes_bytes_total
cadvisor	container_memory_working_set_bytes

cadvisor	container_network_receive_bytes_total
cadvisor	container_network_receive_packets_dropped_total
cadvisor	container_network_receive_packets_total
cadvisor	container_network_transmit_bytes_total
cadvisor	container_network_transmit_packets_dropped_total
cadvisor	container_network_transmit_packets_total
cadvisor	machine_cpu_cores
cadvisor	machine_memory_bytes
kubelet	kubelet_cgroup_manager_duration_seconds_count
kubelet	kubelet_node_config_error
kubelet	kubelet_node_name
kubelet	kubelet_pleg_relist_duration_seconds_bucket
kubelet	kubelet_pleg_relist_duration_seconds_count
kubelet	kubelet_pleg_relist_interval_seconds_bucket
kubelet	kubelet_pod_start_duration_seconds_count
kubelet	kubelet_pod_worker_duration_seconds_count
kubelet	kubelet_running_containers
kubelet	kubelet_running_pods
kubelet	kubelet_runtime_operations_duration_seconds_bucket
kubelet	kubelet_runtime_operations_errors_total
kubelet	kubelet_runtime_operations_total
kubelet	process_cpu_seconds_total
kubelet	process_resident_memory_bytes
kubelet	rest_client_request_duration_seconds_bucket
kubelet	rest_client_requests_total

kubelet	storage_operation_duration_seconds_bucket
kubelet	storage_operation_duration_seconds_count
kubelet	storage_operation_errors_total
kubelet	volume_manager_total_volumes
kube-state-metrics	kube_job_status_succeeded
kube-state-metrics	kube_job_status_failed
kube-state-metrics	kube_job_status_active
kube-state-metrics	kube_node_status_capacity_cpu_cores
kube-state-metrics	kube_node_status_capacity_memory_bytes
kube-state-metrics	kube_node_status_allocatable_cpu_cores
kube-state-metrics	kube_node_status_allocatable_memory_bytes
kube-state-metrics	kube_pod_info
kube-state-metrics	kube_pod_owner
kube-state-metrics	kube_pod_status_phase
kube-state-metrics	kube_pod_container_status_waiting
kube-state-metrics	kube_pod_container_status_running
kube-state-metrics	kube_pod_container_status_terminated
kube-state-metrics	kube_pod_container_status_restarts_total
kube-state-metrics	kube_pod_container_resource_requests_cpu_cores
kube-state-metrics	kube_pod_container_resource_requests_memory_bytes
kube-state-metrics	kube_pod_container_resource_limits_cpu_cores
kube-state-metrics	kube_pod_container_resource_limits_memory_bytes
kube-state-metrics	kube_replicaset_owner
kube-state-metrics	kube_statefulset_status_replicas
kube-controller-manager	rest_client_request_duration_seconds_bucket

kube-controller-manager	rest_client_requests_total
kube-controller-manager	workqueue_adds_total
kube-controller-manager	workqueue_depth
kube-controller-manager	workqueue_queue_duration_seconds_bucket
kube-apiserver	apiserver_current_inflight_requests
kube-apiserver	apiserver_current_inqueue_requests
kube-apiserver	apiserver_init_events_total
kube-apiserver	apiserver_longrunning_gauge
kube-apiserver	apiserver_registered_watchers
kube-apiserver	apiserver_request_duration_seconds_bucket
kube-apiserver	apiserver_request_duration_seconds_sum
kube-apiserver	apiserver_request_duration_seconds_count
kube-apiserver	apiserver_request_filter_duration_seconds_bucket
kube-apiserver	apiserver_request_filter_duration_seconds_sum
kube-apiserver	apiserver_request_filter_duration_seconds_count
kube-apiserver	apiserver_request_total
kube-apiserver	apiserver_requested_deprecated_apis
kube-apiserver	apiserver_response_sizes_bucket
kube-apiserver	apiserver_response_sizes_sum
kube-apiserver	apiserver_response_sizes_count
kube-apiserver	apiserver_selfrequest_total
kube-apiserver	apiserver_tls_handshake_errors_total
kube-apiserver	apiserver_watch_events_sizes
kube-apiserver	apiserver_watch_events_sizes_bucket
kube-apiserver	apiserver_watch_events_sizes_sum

kube-apiserver	apiserver_watch_events_sizes_count
kube-apiserver	apiserver_watch_events_total

Recommended Common Metrics for TKE

Last updated：2024-01-29 16:01:56

Based on the usage of most users, it is recommend that you configure the following commonly used TKE metrics.

Note

The following are paid metrics. For the billing method of metrics, see Billing Mode and Resource Usage.

Configuration File	Metric Name	Metric Description
kubelet	kubelet_running_container_count	kubelet_running_container_ Number of containers curre running
kubelet	kubelet_running_pod_count	kubelet_running_pod_cour Number of pods currently r
kube-state-metrics	kube_pod_container_info	Information about a contair pod.
kube-state-metrics	kube_deployment_status_replicas	The number of replicas per deployment.
kube-state-metrics	kube_deployment_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_pod_start_time	Start time in unix timestamp pod.
kube-state-metrics	kube_pod_status_ready	Describes whether the pod ready to serve requests.
kube-state-metrics	kube_node_info	Information about a cluster
kube-state-metrics	kube_node_status_condition	The condition of a cluster n
kube-state-metrics	kube_deployment_status_replicas_updated	The number of updated rep per deployment.
kube-state-metrics	kube_deployment_status_replicas_available	The number of available re per deployment.
kube-state-metrics	kube_node_status_capacity_pods	The total pod resources of node.

kube-state-metrics	kube_pod_container_status_ready	Describes whether the cont readiness check succeede
kube-state-metrics	kube_deployment_spec_replicas	Number of desired pods for deployment.
kube-state-metrics	kube_pod_status_scheduled_time	Unix timestamp when pod r into scheduled status
kube-state-metrics	kube_node_status_allocatable_pods	The pod resources of a noc are available for scheduling
kube-state-metrics	kube_pod_container_resource_limits	The number of requested li resource by a container.
kube-state-metrics	node_filefd_maximum	File descriptor statistics: maximum.
kube-state-metrics	kube_pod_container_resource_requests	The number of requested r resource by a container.
kube-state-metrics	kube_namespace_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_deployment_status_replicas_unavailable	The number of unavailable replicas per deployment.
kube-state-metrics	kube_pod_created	Unix creation timestamp
kube-state-metrics	kube_pod_container_status_waiting_reason	Describes the reason the container is currently in wa state.
kube-state-metrics	kube_daemonset_status_desired_number_scheduled	The number of nodes that s be running the daemon poc
kube-state-metrics	kube_pod_restart_policy	Describes the restart policy use by this pod.
kube-state-metrics	kube_deployment_metadata_generation	Sequence number represe specific generation of the d state.
kube-state-metrics	kube_statefulset_status_update_revision	Indicates the version of the StatefulSet used to genera Pods in the sequence [repli updatedReplicas].

kube-state-metrics	kube_node_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_statefulset_replicas	Number of desired pods fo StatefulSet.
kube-state-metrics	kube_statefulset_status_observed_generation	The generation observed b StatefulSet controller.
kube-state-metrics	kube_pod_container_status_last_terminated_reason	Describes the last reason t container was in terminate
kube-state-metrics	kube_replicaset_spec_replicas	Number of desired pods fo ReplicaSet.
kube-state-metrics	kube_statefulset_created	Unix creation timestamp
kube-state-metrics	kube_statefulset_status_replicas_current	The number of current repl per StatefulSet.
kube-state-metrics	kube_statefulset_status_current_revision	Indicates the version of the StatefulSet used to genera Pods in the sequence [0].
kube-state-metrics	kube_statefulset_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_deployment_created	Unix creation timestamp
kube-state-metrics	kube_namespace_created	Unix creation timestamp
kube-state-metrics	kube_daemonset_status_number_ready	The number of nodes that s be running the daemon poc have one or more of the da pod running and ready.
kube-state-metrics	kube_deployment_status_observed_generation	The generation observed b deployment controller.
kube-state-metrics	kube_endpoint_info	Information about endpoint
kube-state-metrics	kube_statefulset_status_replicas_updated	The number of updated rep per StatefulSet.

kube-state-metrics	kube_statefulset_metadata_generation	Sequence number represe specific generation of the d state for the StatefulSet.
kube-state-metrics	kube_secret_created	Unix creation timestamp
kube-state-metrics	kube_endpoint_address_not_ready	Number of addresses not r in endpoint
kube-state-metrics	kube_secret_type	Type about secret.
kube-state-metrics	kube_deployment_spec_paused	Whether the deployment is paused and will not be proc by the deployment controlle
kube-state-metrics	kube_pod_container_status_terminated_reason	Describes the reason the container is currently in terminated state.
kube-state-metrics	kube_statefulset_status_replicas_ready	The number of ready replic StatefulSet.
kube-state-metrics	kube_endpoint_address_available	Number of addresses avail endpoint.
kube-state-metrics	kube_secret_info	Information about secret.
kube-state-metrics	kube_service_info	Information about service.
kube-state-metrics	kube_node_status_allocatable	The allocatable for differen resources of a node that ar available for scheduling.
kube-state-metrics	kube_endpoint_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_deployment_status_condition	The current status conditio deployment.
kube-state-metrics	kube_endpoint_created	Unix creation timestamp
kube-state-metrics	kube_replicaset_labels	Kubernetes labels converte Prometheus labels.

kube-state-metrics	kube_replicaset_metadata_generation	Sequence number represe specific generation of the d state.
kube-state-metrics	kube_namespace_status_phase	kubernetes namespace sta phase.
kube-state-metrics	kube_service_created	Unix creation timestamp
kube-state-metrics	kube_configmap_created	Unix creation timestamp
kube-state-metrics	kube_secret_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_deployment_spec_strategy_rollingupdate_max_surge	Maximum number of replica can be scheduled above th desired number of replicas a rolling update of a deploy
kube-state-metrics	kube_configmap_metadata_resource_version	Resource version represen specific version of the confi
kube-state-metrics	kube_pod_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_replicaset_status_replicas	The number of replicas per ReplicaSet.
kube-state-metrics	kube_node_created	Unix creation timestamp
kube-state-metrics	kube_service_spec_type	Type about service.
kube-state-metrics	kube_secret_metadata_resource_version	Resource version represen specific version of secret.
kube-state-metrics	kube_configmap_info	Information about configma
kube-state-metrics	kube_replicaset_status_observed_generation	The generation observed b ReplicaSet controller.
kube-state-metrics	kube_service_labels	Kubernetes labels converte Prometheus labels.

kube-state-metrics	kube_replicaset_created	Unix creation timestamp
kube-state-metrics	kube_deployment_spec_strategy_rollingupdate_max_unavailable	Maximum number of unava replicas during a rolling upd a deployment.
kube-state-metrics	kube_replicaset_status_ready_replicas	The number of ready replic ReplicaSet.
kube-state-metrics	kube_replicaset_status_fully_labeled_replicas	The number of fully labeled replicas per ReplicaSet.
kube-state-metrics	kube_pod_status_scheduled	Describes the status of the scheduling process for the
kube-state-metrics	kube_storageclass_created	Unix creation timestamp
kube-state-metrics	kube_daemonset_status_number_misscheduled	The number of nodes runni daemon pod but are not supposed to.
kube-state-metrics	kube_storageclass_labels	Kubernetes labels converte Prometheus labels.
kube-state-metrics	kube_node_status_capacity	The capacity for different resources of a node.
kube-state-metrics	kube_daemonset_status_current_number_scheduled	The number of nodes runni least one daemon pod and supposed to.
kube-state-metrics	kube_storageclass_info	Information about storagec
kube-state-metrics	kube_node_spec_unschedulable	Whether a node can sched new pods.
kube-state-metrics	kube_daemonset_status_number_available	The number of nodes that s be running the daemon poc have one or more of the da pod running and available.
kube-state-metrics	kube_daemonset_labels	Kubernetes labels converte Prometheus labels.
kube-state-	kube_daemonset_created	Unix creation timestamp

metrics		
kube-state-metrics	kube_daemonset_status_number_unavailable	The number of nodes that s be running the daemon poc have none of the daemon p running and available
kube-state-metrics	kube_daemonset_metadata_generation	Sequence number represe specific generation of the d state.
kube-state-metrics	kube_mutatingwebhookconfiguration_info	Information about the MutatingWebhookConfigur
kube-state-metrics	kube_mutatingwebhookconfiguration_created	Unix creation timestamp.
kube-state-metrics	kube_mutatingwebhookconfiguration_metadata_resource_version	Resource version represen specific version of the MutatingWebhookConfigur
kube-state-metrics	kube_daemonset_updated_number_scheduled	The total number of nodes are running updated daem
node-exporter	node_filesystem_files_free	Filesystem total free file no
node-exporter	node_filesystem_files	Filesystem total file nodes.
node-exporter	node_sockstat_UDP_mem_bytes	Number of UDP sockets in mem_bytes.
node-exporter	node_nf_conntrack_entries_limit	Maximum size of connectic tracking table.
node-exporter	node_memory_Shmem_bytes	Memory information field Shmem_bytes.
node-exporter	node_netstat_Tcp_RetransSegs	Statistic TcpRetransSegs.
node-exporter	node_sockstat_TCP_mem_bytes	Number of TCP sockets in mem_bytes.
node-exporter	node_network_info	Non-numeric data from /sys/class/net/<iface>

node-exporter	node_filesystem_readonly	Filesystem read-only status
node-exporter	node_exporter_build_info	A metric with a constant '1' labeled by version
node-exporter	node_network_iface_link_mode	iface_link_mode value of /sys/class/net/<iface>.
node-exporter	node_network_receive_packets_total	Network device statistic receive_packets.
node-exporter	node_network_transmit_packets_total	Network device statistic transmit_packets.
node-exporter	node_memory_Mlocked_bytes	Memory information field Mlocked_bytes.
node-exporter	node_network_iface_id	iface_id value of /sys/class/net/<iface>.
node-exporter	node_memory_WritebackTmp_bytes	Memory information field WritebackTmp_bytes.
node-exporter	kube_service_status_load_balancer_ingress	Service load balancer ingre status
node-exporter	node_vmstat_pgpgout	/proc/vmstat information fie pgpgout.
node-exporter	node_nf_conntrack_entries	Number of currently allocat flow entries for connection tracking.
node-exporter	node_memory_Inactive_file_bytes	Memory information field Inactive_file_bytes.
node-exporter	node_memory_SwapFree_bytes	Memory information field SwapFree_bytes.
node-exporter	node_sockstat_TCP_mem	Number of TCP sockets in mem.
node-exporter	node_memory_Slab_bytes	Memory information field Slab_bytes.
node-exporter	node_network_transmit_errs_total	Network device statistic transmit_errs.

node-exporter	node_memory_Active_bytes	Memory information field Active_bytes.
node-exporter	node_procs_blocked	Number of processes block waiting for I/O to complete.
node-exporter	node_sockstat_UDP_mem	Number of UDP sockets in mem.
node-exporter	node_timex_maxerror_seconds	Maximum error in seconds.
node-exporter	node_memory_Inactive_bytes	Memory information field Inactive_bytes.
node-exporter	node_network_receive_errs_total	Network device statistic receive_errs.
node-exporter	node_memory_Unevictable_bytes	Memory information field Unevictable_bytes.
node-exporter	node_memory_KernelStack_bytes	Memory information field KernelStack_bytes.
node-exporter	node_procs_running	Number of processes in ru state.
node-exporter	node_memory_SwapTotal_bytes	Memory information field SwapTotal_bytes.
node-exporter	node_netstat_IpExt_OutOctets	Statistic IpExtOutOctets.
node-exporter	node_memory_Active_file_bytes	Memory information field Active_file_bytes.
node-exporter	node_memory_SwapCached_bytes	Memory information field SwapCached_bytes.
node-exporter	node_netstat_Icmp_InMsgs	Statistic IcmpInMsgs.
node-exporter	node_forks_total	Total number of forks.
node-exporter	node_sockstat_RAW_inuse	Number of RAW sockets in inuse.

node-exporter	node_time_seconds	System time in seconds sir epoch (1970).
node-exporter	node_vmstat_pgpgin	/proc/vmstat information fie pgpgin.
node-exporter	node_memory_Mapped_bytes	Memory information field Mapped_bytes.
node-exporter	node_memory_SUnreclaim_bytes	Memory information field SUnreclaim_bytes.
node-exporter	node_memory_HardwareCorrupted_bytes	Memory information field HardwareCorrupted_bytes
node-exporter	node_memory_PageTables_bytes	Memory information field PageTables_bytes.
node-exporter	node_netstat_Udp6_InDatagrams	Statistic Udp6InDatagrams
node-exporter	node_netstat_Icmp_OutMsgs	Statistic IcmpOutMsgs.
node-exporter	node_netstat_Udp6_NoPorts	Statistic Udp6NoPorts.
node-exporter	node_memory_AnonPages_bytes	Memory information field AnonPages_bytes.
node-exporter	node_memory_Committed_AS_bytes	Memory information field Committed_AS_bytes.
node-exporter	node_netstat_TcpExt_ListenOverflows	Statistic TcpExtListenOver
node-exporter	node_netstat_UdpLite_InErrors	Statistic UdpLiteInErrors.
node-exporter	node_entropy_available_bits	Bits of available entropy.
node-exporter	node_memory_Inactive_anon_bytes	Memory information field Inactive_anon_bytes.
node-exporter	node_vmstat_pswpin	/proc/vmstat information fie pswpin.

node-exporter	node_memory_AnonHugePages_bytes	Memory information field AnonHugePages_bytes.
node-exporter	node_memory_SReclaimable_bytes	Memory information field SReclaimable_bytes.
node-exporter	node_netstat_IpExt_InOctets	Statistic IpExtInOctets.
node-exporter	node_netstat_Udp_NoPorts	Statistic UdpNoPorts.
node-exporter	node_timex_sync_status	Is clock synchronized to a r server (1 = yes).
node-exporter	node_memory_CommitLimit_bytes	Memory information field CommitLimit_bytes.
node-exporter	node_memory_VmallocChunk_bytes	Memory information field VmallocChunk_bytes.
node-exporter	node_netstat_Udp_InDatagrams	Statistic UdpInDatagrams.
node-exporter	node_netstat_Icmp6_InErrors	Statistic Icmp6InErrors.
node-exporter	node_netstat_Icmp6_OutMsgs	Statistic Icmp6OutMsgs.
node-exporter	node_netstat_UdpLite6_InErrors	Statistic UdpLite6InErrors.
node-exporter	node_netstat_TcpExt_SyncookiesSent	Statistic TcpExtSyncookies
node-exporter	node_netstat_Tcp_InErrs	Statistic TcpInErrs.
node-exporter	node_intr_total	Total number of interrupts serviced.
node-exporter	node_timex_offset_seconds	Time offset in between loca system and reference clock
node-exporter	node_memory_Bounce_bytes	Memory information field Bounce_bytes.

node-exporter	node_memory_Writeback_bytes	Memory information field Writeback_bytes.
node-exporter	node_netstat_Udp_OutDatagrams	Statistic UdpOutDatagram
node-exporter	node_netstat_Icmp6_InMsgs	Statistic Icmp6InMsgs.
node-exporter	node_netstat_Ip6_OutOctets	Statistic Ip6OutOctets.
node-exporter	node_netstat_Ip_Forwarding	Statistic IpForwarding.
node-exporter	node_sockstat_TCP_orphan	Number of TCP sockets in orphan.
node-exporter	node_netstat_Ip6_InOctets	Statistic Ip6InOctets.
node-exporter	node_netstat_TcpExt_SyncookiesFailed	Statistic TcpExtSyncookiesFailed.
node-exporter	node_netstat_Udp_InErrors	Statistic UdpInErrors.
node-exporter	node_vmstat_pgmajfault	/proc/vmstat information fie pgmajfault.
node-exporter	node_network_transmit_drop_total	Network device statistic transmit_drop.
node-exporter	node_vmstat_pswpout	/proc/vmstat information fie pswpout.
node-exporter	node_network_up	Value is 1 if operstate is 'up
node-exporter	node_memory_NFS_Unstable_bytes	Memory information field NFS_Unstable_bytes.
node-exporter	node_memory_VmallocTotal_bytes	Memory information field VmallocTotal_bytes.
node-exporter	node_sockstat_FRAG_inuse	Number of FRAG sockets i inuse.

node-exporter	node_memory_Dirty_bytes	Memory information field Dirty_bytes.
node-exporter	node_netstat_Udp6_InErrors	Statistic Udp6InErrors.
node-exporter	node_netstat_TcpExt_SyncookiesRecv	Statistic TcpExtSyncookies
node-exporter	node_netstat_Udp6_OutDatagrams	Statistic Udp6OutDatagran
node-exporter	node_memory_HugePages_Rsvd	Memory information field HugePages_Rsvd.
node-exporter	node_arp_entries	ARP entries by device
node-exporter	node_network_carrier	carrier value of /sys/class/net/<iface>.
node-exporter	node_timex_pps_stability_exceeded_total	Pulse per second count of stability limit exceeded eve
node-exporter	node_network_receive_compressed_total	Network device statistic receive_compressed.
node-exporter	node_network_transmit_carrier_total	Network device statistic transmit_carrier.
node-exporter	node_memory_DirectMap2M_bytes	Memory information field DirectMap2M_bytes.
node-exporter	node_memory_Hugepagesize_bytes	Memory information field Hugepagesize_bytes.
node-exporter	node_network_address_assign_type	address_assign_type value /sys/class/net/<iface>.
node-exporter	node_network_receive_multicast_total	Network device statistic receive_multicast.
node-exporter	node_network_transmit_compressed_total	Network device statistic transmit_compressed.
node-exporter	node_memory_DirectMap4k_bytes	Memory information field DirectMap4k_bytes.

node-exporter	node_network_transmit_queue_length	transmit_queue_length valu /sys/class/net/<iface>.
node-exporter	node_memory_HugePages_Free	Memory information field HugePages_Free.
node-exporter	node_network_receive_frame_total	Network device statistic receive_frame.
node-exporter	node_memory_HugePages_Total	Memory information field HugePages_Total.
node-exporter	node_network_flags	flags value of /sys/class/net/<iface>.
node-exporter	node_network_receive_fifo_total	Network device statistic receive_fifo.
node-exporter	node_scrape_collector_duration_seconds	node_exporter: Duration of collector scrape.
node-exporter	node_network_speed_bytes	speed_bytes value of /sys/class/net/<iface>.
node-exporter	node_sockstat_UDPLITE_inuse	Number of UDPLITE socke state inuse.
node-exporter	node_cpu_guest_seconds_total	Seconds the cpus spent in (VMs) for each mode.
node-exporter	node_filesystem_device_error	Whether an error occurred getting statistics for the giv device.
node-exporter	node_scrape_collector_success	node_exporter: Whether a collector succeeded.
node-exporter	node_network_transmit_fifo_total	Network device statistic transmit_fifo.
node-exporter	node_vmstat_pgfault	/proc/vmstat information fie pgfault.
node-exporter	node_network_device_id	device_id value of /sys/class/net/<iface>.
node-exporter	node_network_protocol_type	protocol_type value of /sys/class/net/<iface>.

node-exporter	node_network_receive_drop_total	Network device statistic receive_drop.
node-exporter	node_timex_estimated_error_seconds	Estimated error in seconds
node-exporter	node_disk_writes_merged_total	The number of writes merg
node-exporter	node_network_transmit_colls_total	Network device statistic transmit_colls.
node-exporter	node_timex_tick_seconds	Seconds between clock tic
node-exporter	node_textfile_scrape_error	1 if there was an error oper reading a file
node-exporter	node_network_iface_link	iface_link value of /sys/class/net/<iface>.
node-exporter	node_disk_reads_merged_total	The total number of reads merged.
node-exporter	node_timex_status	Value of the status array bi
node-exporter	node_netstat_Icmp_InErrors	Statistic IcmpInErrors.
node-exporter	node_memory_Active_anon_bytes	Memory information field Active_anon_bytes.
node-exporter	node_timex_pps_frequency_hertz	Pulse per second frequenc
node-exporter	node_network_mtu_bytes	mtu_bytes value of /sys/class/net/<iface>.
node-exporter	node_timex_tai_offset_seconds	International Atomic Time (offset.
node-exporter	node_timex_pps_jitter_total	Pulse per second count of j limit exceeded events.
node-exporter	node_timex_pps_jitter_seconds	Pulse per second jitter.

node-exporter	node_network_net_dev_group	net_dev_group value of /sys/class/net/<iface>.
node-exporter	node_network_dormant	dormant value of /sys/class/net/<iface>.
node-exporter	node_timex_pps_calibration_total	Pulse per second count of calibration intervals.
node-exporter	node_timex_pps_shift_seconds	Pulse per second interval duration.
node-exporter	node_timex_pps_error_total	Pulse per second count of calibration errors.
node-exporter	node_memory_VmallocUsed_bytes	Memory information field VmallocUsed_bytes.
node-exporter	node_timex_frequency_adjustment_ratio	Local clock frequency adjustment.
node-exporter	node_sockstat_FRAG_memory	Number of FRAG sockets i memory.
node-exporter	node_memory_HugePages_Surp	Memory information field HugePages_Surp.
node-exporter	node_timex_loop_time_constant	Phase-locked loop time co
node-exporter	node_timex_pps_stability_hertz	Pulse per second stability

Container Monitoring Chart Metrics

Last updated：2024-07-30 18:14:31

Cluster Monitoring Overview

Chart Name	Query Statement
CPU Requests Commitment	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster"}) / sum(kube_
CPU Limits Commitment	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster"}) / sum(kube_nod
Memory Utilisation	1 - sum(:node_memory_MemAvailable_bytes:sum{cluster="$cluster"}) / sum(node_memory
Memory Requests Commitment	sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster"}) / sum(ku
Memory Limits Commitment	sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster"}) / sum(kube_
Node Count	count(kube_node_info{cluster="$cluster"})
Pod Count	count(kube_pod_info{cluster="$cluster"})

Node Request CPU Average Percent	avg(sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster"})by (node
Node Request Memory Average Percent	avg(sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster"})by (node)/sum(kube_node_status_capacity_memory_bytes{cluster="$cluster"})by(node))
API Server Success Request Percent	sum(irate(apiserver_request_total{cluster="$cluster",code=~"20.*",verb=~"GET|LIST"}[5m] [5m]))
Namespace Overview	count(kube_pod_info{cluster="$cluster"}) by (namespace)
	count(kube_service_info{cluster="$cluster"}) by(namespace)
	count(kube_pod_container_info{cluster="$cluster"}) by(namespace)
	count(kube_configmap_info{cluster="$cluster"}) by(namespace)
	count(kube_secret_info{cluster="$cluster"}) by(namespace)
	count(kube_deployment_created{cluster="$cluster"}) by (namespace)
	count(kube_statefulset_created{cluster="$cluster"}) by (namespace)
	count(kube_job_created{cluster="$cluster"}) by (namespace)
	count(kube_cronjob_created{cluster="$cluster"}) by (namespace)
	count(kube_pod_status_ready{cluster="$cluster",condition="false"}==1) by(namespace) - (by(namespace) or vector(0)) or count(kube_pod_status_ready{cluster="$cluster",condition

	count(kube_deployment_status_replicas_ready{cluster="$cluster"}<kube_deployment_spe
	count(kube_statefulset_status_replicas_ready{cluster="$cluster"}<kube_statefulset_replica
	count(kube_daemonset_status_number_unavailable{cluster="$cluster"}>0)by(namespace)
	count(kube_job_status_failed{cluster="$cluster"} == 1) by (namespace)
	count(kube_daemonset_created{cluster="$cluster"}) by (namespace)
	count(kube_persistentvolumeclaim_info{cluster="$cluster"}) by (namespace)
CPU Usage	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{clus
CPU Quota	sum(kube_pod_owner{cluster="$cluster"}) by (namespace)
	count(avg(namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster"}) by (wo
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{clus
	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster"}) by (namespa

	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{clus sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster"}) by (namespa
	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster"}) by (namespace)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{clus sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster"}) by (namespace)
Memory Usage (working_set)	sum(container_memory_working_set_bytes{cluster="$cluster", container!="", container!="F
Memory Requests	sum(kube_pod_owner{cluster="$cluster"}) by (namespace)
	count(avg(namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster"}) by (wor
	sum(container_memory_rss{cluster="$cluster", container!="", container!="POD"}) by (name
	sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster"}) by (name
	sum(container_memory_rss{cluster="$cluster", container!="", container!="POD"}) by (name sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster"}) by (name
	sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster"}) by (namespa
	sum(container_memory_rss{cluster="$cluster", container!="", container!="POD"}) by (name sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster"}) by (namespa
Node Memory Usage (Top 10)	sum(label_replace(topk(10, 1-(node_memory_MemAvailable_bytes{cluster="$cluster"} / no "instance", "(.*)"))by(node_ip)

Node CPU Usage (Top 10)	topk(10, sum(label_replace(1 - sum(rate(node_cpu_seconds_total{cluster="$cluster",mode sum(rate(node_cpu_seconds_total{cluster="$cluster"}[1m])) by (instance),"host_ip","$1","in
Node Disk Usage (Top 10)	topk(10, sum(label_replace(1- node_filesystem_free_bytes{cluster="$cluster",mountpoint="/"}/node_filesystem_size_bytes (.*)"))by(host_ip))
Node Network In (Top 10)	topk(10, sum(label_replace(max(irate(node_network_receive_bytes_total{cluster="$cluster
Node Network Out (Top 10)	topk(10, sum(label_replace(max(irate(node_network_transmit_bytes_total{cluster="$cluste
Node Sockets Count(Top 10)	topk(10, sum(label_replace(max(node_sockstat_TCP_alloc{cluster="$cluster"}) by (instanc
Container Memory Usage(Top10)	topk(10, sum (container_memory_working_set_bytes{cluster="$cluster",container !="",cont
Container Memory Usage/Limit(Top10)	topk(10, avg(container_memory_working_set_bytes{cluster="$cluster",container!=""}/(cont namespace))
Container CPU Usage(Top10)	topk(10, sum(rate(container_cpu_usage_seconds_total{cluster="$cluster",container !="",co
Container Network	topk(10, sum(irate(container_network_receive_bytes_total{cluster="$cluster",image!="",co
	-topk(10, sum(irate(container_network_transmit_bytes_total{cluster="$cluster",image!="",c
Container Memory Usage/Limit (Top 10)	topk(10, avg(container_memory_working_set_bytes{cluster="$cluster",container!=""}/(cont namespace))
Container CPU Usage (Top 10)	topk(10, sum(irate(container_cpu_usage_seconds_total{cluster="$cluster",container!="",co
Container Socket Count(Top 10)	topk(10, sum(container_sockets{cluster="$cluster",container!=""}) by (container,pod,names

Cluster Namespace Dashboard

Chart Name	Query Statement
CPU Usage	sum(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace=~"$name
CPU Usage/Request(%)	sum(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace=~"$name [2m]))/sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",nam
CPU Usage/Limit(%)	sum(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace=~"$name [2m]))/sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster",namesp
CPU Request	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",namespace
CPU Limit	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster",namespace=~'
Cluster Available	sum(sum(kube_node_status_capacity{resource="cpu",cluster="$cluster",namespace=~
StatefulSet Created	count(kube_statefulset_created{cluster="$cluster",namespace="$namespace"}) or on()
Pod Created	count(kube_pod_info{cluster="$cluster",namespace="$namespace"}) or on() vector(0)
Containers	count(kube_pod_container_info{cluster="$cluster",namespace="$namespace"}) or on()
DaemonSet Created	count(kube_daemonset_created{cluster="$cluster",namespace="$namespace"}) or on()
Job Created	count(kube_job_info{cluster="$cluster",namespace="$namespace"})or on() vector(0)
Job Active	count(kube_job_status_active{cluster="$cluster",namespace="$namespace"}==1)or on

Cron Job Created	count(kube_cronjob_created{cluster="$cluster",namespace="$namespace"}) or on() vec
Cron Job Active	count(kube_cronjob_status_active{cluster="$cluster",namespace="$namespace"}==1) (
Unbound PVC	count(kube_persistentvolumeclaim_status_phase{phase!="Bound", cluster="$cluster",n
PersistentVolumeClaim Created	count(kube_persistentvolumeclaim_info{cluster="$cluster",namespace="$namespace"})
Service Created	count(kube_service_info{cluster="$cluster",namespace="$namespace"}) or on() vector(
LoadBalancer Created	count(kube_service_spec_type{type="LoadBalancer", cluster="$cluster",namespace="$
Ingress Created	count(kube_ingress_info{cluster="$cluster",namespace="$namespace"})or on() vector((
ConfigMap Created	count(kube_configmap_info{cluster="$cluster",namespace="$namespace"})
Secret Created	count(kube_secret_info{cluster="$cluster",namespace="$namespace"}) or on() vector(0
PVC Storage Requests Total	sum(kube_persistentvolumeclaim_resource_requests_storage_bytes{cluster="$cluster"
Pod NotReady	count(kube_pod_status_ready{condition="false", cluster="$cluster",namespace="$name by(namespace) or vector(0)) or count(kube_pod_status_ready{condition="false", cluster
Pod UnSchedulable	count(kube_pod_status_unschedulable{cluster="$cluster",namespace="$namespace"})
Deployment NotReady	count(sum(kube_deployment_status_replicas_ready{cluster="$cluster",namespace="$r vector(0)

Daemonset NotReady	count(kube_daemonset_status_number_unavailable{cluster="$cluster",namespace="$r
Job Failed	count(kube_job_status_failed{cluster="$cluster",namespace="$namespace"} == 1)
CPU Usage	sum(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace=~"$name
CPU Quota	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{c
	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespac
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{c sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespac
	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{c sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="
Memory Usage	sum(container_memory_working_set_bytes{cluster="$cluster",namespace=~"$namespa
Memory Usage/Request(%)	sum(container_memory_working_set_bytes{cluster="$cluster",namespace=~"$namespa unit="byte", resource="memory"}) or on() vector(0)
Memory Usage/Limit(%)	sum(container_memory_working_set_bytes{cluster="$cluster",namespace=~"$namespa unit="byte", resource="memory"}) or on() vector(0)
Memory Request	sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",names
Memory Limit	sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster",namespac

Cluster Available	sum(sum(kube_node_status_capacity{resource="memory"}) by (node) + sum(kube_noc
Memory Usage (w/o cache)	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespa
	scalar(kube_resourcequota{cluster="$cluster", namespace="$namespace", type="hard"
	scalar(kube_resourcequota{cluster="$cluster", namespace="$namespace", type="hard"
Memory Quota	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespa
	sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster", names
	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespa sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",names
	sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespac
	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespa
	sum(container_memory_rss{cluster="$cluster", namespace="$namespace",container!='
	sum(container_memory_cache{cluster="$cluster", namespace="$namespace",containe
	sum(container_memory_swap{cluster="$cluster", namespace="$namespace",container
Containers	group by (image, container, container_id, pod) (kube_pod_container_info{cluster="$clus
	group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nar (kube_pod_container_status_running{cluster="$cluster",namespace="$namespace"})

group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai (kube_pod_container_status_restarts_total{cluster="$cluster",namespace="$namespac
group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai max(irate(container_cpu_usage_seconds_total{cluster="$cluster",namespace="$names
group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai (max(irate(container_cpu_usage_seconds_total{container!="",container!="POD",cluster (max(container_spec_cpu_quota{container!="",container!="POD",cluster="$cluster",nan
group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai (kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",namespace="$
group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai (max(irate(container_cpu_usage_seconds_total{container!="",container!="POD",cluster (kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",namespace="$
group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai (kube_pod_container_resource_limits{resource="cpu",cluster="$cluster",namespace="$
group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai !="",container!="POD",cluster="$cluster",namespace=~"$namespace"}) by (pod,contain
group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nai (max(container_memory_working_set_bytes{container!="",container!="POD",cluster="$!="",container!="POD",cluster="$cluster",namespace=~"$namespace"}) by (pod, contair

	group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nar (kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",namespace
	group by (container, container_id, pod) (kube_pod_container_info{cluster="$cluster",nar !="",container!="POD",cluster="$cluster",namespace=~"$namespace"}) by (pod,contain

API Server (Independent Cluster)

Chart Name	Query Statement
Availability > 99.000%	1 - ((sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_count{cluster="$cluster"}[5m])) - sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_bucket{le="1", cluster="$cluster"}[5m]))) + sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube-apiserver",code=~"5.. cluster="$cluster"}[5m]) or vector(0))) / sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube- apiserver",cluster="$cluster"}[5m]))
ErrorBudget > 99.000%	100 * (1 - ((sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_count{cluster="$cluster"}[5m])) - sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_bucket{le="1", cluster="$cluster"}[5m])))

	+ sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube-apiserver",code=~"5.. cluster="$cluster"}[5m]) or vector(0))) / sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube-apiserver",cluster="$cluster"}[5m])) -0.990000)
Read Availability	1 - ((sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_count{verb=~"LIST|GET", cluster="$cluster"}[5m]) - sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_bucket{verb=~"LIST\|GET",le="1", cluster="$cluste [5m]))) + sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube-apiserver",verb=~"LIST|GET",code=~"5..", cluster="$cluster"}[5m]) or vector(0))) / sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube-apiserver",verb=~"LIST\|GET", cluster="$cluster"}[5m]))
Read SLI - Requests	sum by (code) (rate(apiserver_request_total{job="kube-apiserver",verb=~"LIST|GET",cluster="$cluster"}[5m]))
Read SLI - Errors	sum by (resource) (rate(apiserver_request_total{job="kube-apiserver",verb=~"LIST|GET",code=~"5..",cluster="$cluster"}[5m]))/ sum by (resource) (rate(apiserver_request_total{job="kube-apiserver",verb=~"LIST\|GET",cluster="$cluster"}[5m]))
Read SLI - Duration	histogram_quantile(0.99, sum by (le, resource,cluster,cluster_type) (rate(apiserver_request_duration_seconds_bucket{job="kube-apiserver",verb=~"LIST|GET",cluster="$cluster"}[5m]))) > 0
Write Availability	1 - ((sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_count{verb=~"POST|PUT|PATCH|DELETE", cluster="$cluster"}[5m])) - sum by (cluster,cluster_type) (increase(apiserver_request_duration_seconds_bucket{verb=~"POST|PUT|PATCH|DELETE",le="

	cluster="$cluster"}[5m]))) + sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube-apiserver",verb=~"POST\|PUT\|PATCH\|DELETE",code=~"5..", cluster="$cluster"}[5m]) or vector(0))) / sum by (cluster,cluster_type) (increase(apiserver_request_total{job="kube-apiserver",verb=~"POST|PUT|PATCH|DELETE", cluster="$cluster"}[5m]))
Write SLI - Requests	sum by (code) (rate(apiserver_request_total{job="kube-apiserver",verb=~"POST|PUT|PATCH|DELETE",cluster="$cluster"}[5m]))
Write SLI - Errors	sum by (resource) (rate(apiserver_request_total{job="kube-apiserver",verb=~"POST|PUT|PATCH|DELETE",code=~"5..",cluster="$cluster"}[5m]))/ sum by (resource) (rate(apiserver_request_total{job="kube-apiserver",verb=~"POST\|PUT\|PATCH\|DELETE",cluster="$cluster"}[5m]))
Write SLI - Duration	histogram_quantile(0.99, sum by (le, resource,cluster,cluster_type) (rate(apiserver_request_duration_seconds_bucket{job="kube-apiserver",verb=~"POST|PUT|PATCH|DELETE",cluster="$cluster"}[5m]))) > 0
Work Queue Add Rate	sum(rate(workqueue_adds_total{job="kube-apiserver", instance=~"$instance", cluster=~"$cluster"}[5m])) by (instance, name)
Work Queue Depth	sum(rate(workqueue_depth{job="kube-apiserver", instance=~"$instance", cluster=~"$cluster"}[5m])) by (instance, name)
Work Queue Latency	histogram_quantile(0.99, sum(rate(workqueue_queue_duration_seconds_bucket{job="kube-apiserver", instance=~"$instance", cluster=~"$cluster"}[5m])) by (instance, name, le))
Memory	process_resident_memory_bytes{job="kube-apiserver",instance=~"$instance", cluster=~"$cluster"}
CPU usage	rate(process_cpu_seconds_total{job="kube-apiserver",instance=~"$instance", cluster=~"$cluster"}[5m])

Controller Manager (Independent Cluster)

Chart Name	Query Statement	Metrics Us

Up	sum(up{cluster=~"$cluster",job="kube-controller-manager"})	up
Work Queue Add Rate	sum(rate(workqueue_adds_total{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance"}[5m])) by (instance, name)	workqueue
Work Queue Depth	sum(rate(workqueue_depth{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance"}[5m])) by (instance, name)	workqueue
Work Queue Latency	histogram_quantile(0.99, sum(rate(workqueue_queue_duration_seconds_bucket{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance"}[5m])) by (instance, name, le))	workqueue
Kube API Request Rate	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance",code=~"2.."}[5m]))	rest_client
	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance",code=~"3.."}[5m]))	rest_client
	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance",code=~"4.."}[5m]))	rest_client
	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance",code=~"5.."}[5m]))	rest_client
Post Request Latency 99th Quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance", verb="POST"}[5m])) by (verb, url, le))	rest_client
Get Request Latency 99th Quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{cluster=~"$cluster",job="kube-controller-manager", instance=~"$instance", verb="GET"}[5m])) by (verb, url, le))	rest_client
Memory	process_resident_memory_bytes{cluster=~"$cluster",job="kube-controller-manager",instance=~"$instance"}	process_re
CPU usage	rate(process_cpu_seconds_total{cluster=~"$cluster",job="kube-controller-manager",instance=~"$instance"}[5m])	process_c

Kubelet

Chart Name	Query Statement
Up	sum(up{cluster="$cluster", job="kubelet"})
Running Pods	sum(kubelet_running_pods{cluster="$cluster", job="kubelet", instance=~"$instance"})
Running Container	sum(kubelet_running_containers{cluster="$cluster", job="kubelet", instance=~"$instance"})
Actual Volume Count	sum(volume_manager_total_volumes{cluster="$cluster", job="kubelet", instance=~"$instance", state="actual_state_of_world"})
Desired Volume Count	sum(volume_manager_total_volumes{cluster="$cluster", job="kubelet", instance=~"$instance",state="desired_state_of_world"})
Config Error Count	sum(rate(kubelet_node_config_error{cluster="$cluster", job="kubelet", instance=~"$instance"}[5m]))
Operation Rate	sum(rate(kubelet_runtime_operations_total{cluster="$cluster",job="kubelet",instance=~"$instance"}[(operation_type, instance)
Operation Error Rate	sum(rate(kubelet_runtime_operations_errors_total{cluster="$cluster",job="kubelet",instance=~"$inst (instance, operation_type)
Operation duration 99th quantile	histogram_quantile(0.99, sum(rate(kubelet_runtime_operations_duration_seconds_bucket{cluster="$cluster",job="kubelet",ins [5m])) by (instance, operation_type, le))
Pod Start Rate	sum(rate(kubelet_pod_start_duration_seconds_count{cluster="$cluster",job="kubelet",instance=~"$ (instance)
	sum(rate(kubelet_pod_worker_duration_seconds_count{cluster="$cluster",job="kubelet",instance=~ by (instance)
Pod Start Duration	histogram_quantile(0.99, sum(rate(kubelet_pod_start_duration_seconds_count{cluster="$cluster",job="kubelet",instance=~"$ (instance, le))
	histogram_quantile(0.99,

	sum(rate(kubelet_pod_worker_duration_seconds_bucket{cluster="$cluster",job="kubelet",instance= by (instance, le))
Storage Operation Rate	sum(rate(storage_operation_duration_seconds_count{cluster="$cluster",job="kubelet",instance=~"$ (instance, operation_name, volume_plugin)
Storage Operation Error Rate	sum(rate(storage_operation_errors_total{cluster="$cluster",job="kubelet",instance=~"$instance"}[5m operation_name, volume_plugin)
Storage Operation Duration 99th quantile	histogram_quantile(0.99, sum(rate(storage_operation_duration_seconds_bucket{cluster="$cluster", instance=~"$instance"}[5m])) by (instance, operation_name, volume_plugin, le))
Cgroup manager operation rate	sum(rate(kubelet_cgroup_manager_duration_seconds_count{cluster="$cluster", job="kubelet", insta [5m])) by (instance, operation_type)
Cgroup manager 99th quantile	histogram_quantile(0.99, sum(rate(kubelet_cgroup_manager_duration_seconds_bucket{cluster="$c job="kubelet", instance=~"$instance"}[5m])) by (instance, operation_type, le))
PLEG relist rate	sum(rate(kubelet_pleg_relist_duration_seconds_count{cluster="$cluster", job="kubelet", instance=~' (instance)
PLEG relist interval	histogram_quantile(0.99, sum(rate(kubelet_pleg_relist_interval_seconds_bucket{cluster="$cluster",job="kubelet",instance=~" (instance, le))
PLEG relist duration	histogram_quantile(0.99, sum(rate(kubelet_pleg_relist_duration_seconds_bucket{cluster="$cluster",job="kubelet",instance=~ (instance, le))
RPC Rate	sum(rate(rest_client_requests_total{cluster="$cluster",job="kubelet", instance=~"$instance",code=~'
	sum(rate(rest_client_requests_total{cluster="$cluster",job="kubelet", instance=~"$instance",code=~'
Request duration 99th quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{cluster="$cluster" instance=~"$instance"}[5m])) by (instance, verb, url, le))
Memory	process_resident_memory_bytes{cluster="$cluster",job="kubelet",instance=~"$instance"}

CPU usage	rate(process_cpu_seconds_total{cluster="$cluster",job="kubelet",instance=~"$instance"}[5m])
Goroutines	go_goroutines{cluster="$cluster",job="kubelet",instance=~"$instance"}

Proxy(Non-default Installation Component)

Chart Name	Query Statement
Up	sum(up{job="kube-proxy"})
Rules Sync Rate	sum(rate(kubeproxy_sync_proxy_rules_duration_seconds_count{job="kube-proxy", instance=~"$instance"}[5m]))
Rule Sync Latency 99th Quantile	histogram_quantile(0.99,rate(kubeproxy_sync_proxy_rules_duration_seconds_bucket{job="kube proxy", instance=~"$instance"}[5m]))
Network Programming Rate	sum(rate(kubeproxy_network_programming_duration_seconds_count{job="kube-proxy", instance=~"$instance"}[5m]))
Network Programming Latency 99th Quantile	histogram_quantile(0.99, sum(rate(kubeproxy_network_programming_duration_seconds_bucket{job="kube-proxy", instance=~"$instance"}[5m])) by (instance, le))
Kube API Request Rate	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"2.."}[5m]))
	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"3.."}[5m]))
	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"4.."}[5m]))
	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"5.."}[5m]))
Post Request Latency 99th Quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{job="kube-proxy",instance=~"$instance",verb="POST"}[5m])) by (verb, url, le))
Kube API	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"2.."}[5m]))

Request Rate	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"3.."}[5m]))
	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"4.."}[5m]))
	sum(rate(rest_client_requests_total{job="kube-proxy", instance=~"$instance",code=~"5.."}[5m]))
Post Request Latency 99th Quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{job="kube-proxy",instance=~"$instance",verb="POST"}[5m])) by (verb, url, le))
Get Request Latency 99th Quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{job="kube-proxy", instance=~"$instance", verb="GET"}[5m])) by (verb, url, le))
Memory	process_resident_memory_bytes{job="kube-proxy",instance=~"$instance"}
CPU usage	rate(process_cpu_seconds_total{job="kube-proxy",instance=~"$instance"}[5m])

Scheduler(Independent Cluster)

Chart Name	Query Statement	Metrics Us
Up	sum(up{cluster=~"$cluster", job="kube-scheduler"})	up
Kube API Request Rate	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance",code=~"2.."}[5m]))	rest_client
	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance",code=~"3.."}[5m]))	rest_client
	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance",code=~"4.."}[5m]))	rest_client
	sum(rate(rest_client_requests_total{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance",code=~"5.."}[5m]))	rest_client
Post Request Latency 99th Quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance", verb="POST"}[5m])) by (verb, url, le))	rest_client

Get Request Latency 99th Quantile	histogram_quantile(0.99, sum(rate(rest_client_request_duration_seconds_bucket{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance", verb="GET"}[5m])) by (verb, url, le))	rest_client
Memory	process_resident_memory_bytes{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance"}	process_re
CPU usage	rate(process_cpu_seconds_total{cluster=~"$cluster",job="kube-scheduler", instance=~"$instance"}[5m])	process_c

Cluster Node Monitoring Details

Chart Name	Query Statement
Server resource overview table	node_uname_info{job=~"$job", cluster=~"$cluster"} - 0
	node_memory_MemTotal_bytes{job=~"$job",cluster=~"$cluster"} - 0
	count(node_cpu_seconds_total{job=~"$job",mode='system',cluster=~"$cluster"}) by (instance)
	sum(time() - node_boot_time_seconds{job=~"$job",cluster=~"$cluster"})by(instance)
	max((node_filesystem_size_bytes{job=~"$job",cluster=~"$cluster",fstype=~"ext.?|xfs"}- node_filesystem_free_bytes{job=~"$job",cluster=~"$cluster",fstype=~"ext.?|xfs"}) *100/(node_filesy {job=~"$job",cluster=~"$cluster",fstype=~"ext.?|xfs"}+ (node_filesystem_size_bytes{job=~"$job",cluster=~"$cluster",fstype=~"ext.?|xfs"}- node_filesystem_free_bytes{job=~"$job",cluster=~"$cluster",fstype=~"ext.?|xfs"}))))by(instance)
	(1 - avg(irate(node_cpu_seconds_total{job=~"$job",mode="idle",cluster=~"$cluster"}[5m])) by (insta
	(1 - (node_memory_MemAvailable_bytes{job=~"$job",cluster=~"$cluster"} / (node_memory_MemTotal_bytes{job=~"$job",cluster=~"$cluster"}))) * 100

	node_load5{job=~"$job",cluster=~"$cluster"}
	max(irate(node_disk_written_bytes_total{job=~"$job",cluster=~"$cluster"}[5m])) by (instance)
	max(irate(node_network_receive_bytes_total{job=~"$job",cluster=~"$cluster"}[5m])*8) by (instance
	max(irate(node_network_transmit_bytes_total{job=~"$job",cluster=~"$cluster"}[5m])*8) by (instanc
	node_load5{job=~"$job",cluster=~"$cluster"}
Overall total load and average CPU utilization	count(node_cpu_seconds_total{job=~"$job",cluster=~"$cluster", mode='system'})
	sum(node_load5{job=~"$job",cluster=~"$cluster"})
	avg(1 - avg(irate(node_cpu_seconds_total{job=~"$job",mode="idle",cluster=~"$cluster"}[5m])) by (i
Overall total memory and average memory utilization	sum(node_memory_MemTotal_bytes{job=~"$job",cluster=~"$cluster"})
	sum(node_memory_MemTotal_bytes{job=~"$job",cluster=~"$cluster"} - node_memory_MemAvailable_bytes{job=~"$job",cluster=~"$cluster"})
	(sum(node_memory_MemTotal_bytes{job=~"$job",cluster=~"$cluster"} - node_memory_MemAvailable_bytes{job=~"$job",cluster=~"$cluster"}) / sum(node_memory_MemTotal_bytes{job=~"$job",cluster=~"$cluster"}))*100
Overall total disk and average disk utilization	sum(avg(node_filesystem_size_bytes{job=~"$job",cluster=~"$cluster",fstype=~"xfs|ext.*"})by(device
	sum(avg(node_filesystem_size_bytes{job=~"$job",cluster=~"$cluster",fstype=~"xfs|ext.*"})by(device sum(avg(node_filesystem_free_bytes{job=~"$job",cluster=~"$cluster",fstype=~"xfs|ext.*"})by(device

	(sum(avg(node_filesystem_size_bytes{job=~"$job",cluster=~"$cluster",fstype=~"xfs|ext.*"})by(devic sum(avg(node_filesystem_free_bytes{job=~"$job",cluster=~"$cluster",fstype=~"xfs|ext.*"})by(device *100/(sum(avg(node_filesystem_avail_bytes{job=~"$job",cluster=~"$cluster",fstype=~"xfs|ext.*"})by (sum(avg(node_filesystem_size_bytes{job=~"$job",fstype=~"xfs\|ext.*"})by(device,instance)) - sum(avg(node_filesystem_free_bytes{job=~"$job",cluster=~"$cluster",fstype=~"xfs|ext.*"})by(device
Running time	avg(time() - node_boot_time_seconds{instance=~"$node",cluster=~"$cluster"}) 75
CPU cores	count(node_cpu_seconds_total{cluster=~"$cluster",instance=~"$node", mode='system'})
Total memory	sum(node_memory_MemTotal_bytes{cluster=~"$cluster",instance=~"$node"})
Total CPU utilization	100 - (avg(irate(node_cpu_seconds_total{instance=~"$node",mode="idle",cluster=~"$cluster"}[5m]
Memory utilization	(1 - (node_memory_MemAvailable_bytes{instance=~"$node",cluster=~"$cluster"} / (node_memory_MemTotal_bytes{instance=~"$node",cluster=~"$cluster"})))* 100
Maximum partition utilization	(node_filesystem_size_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint= node_filesystem_free_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint=' /(node_filesystem_avail_bytes {cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoir (node_filesystem_size_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint= node_filesystem_free_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint='
CPU iowait	avg(irate(node_cpu_seconds_total{cluster=~"$cluster",instance=~"$node",mode="iowait"}[5m])) * 1
Available space for each partition	node_filesystem_size_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint !
	node_filesystem_avail_bytes {cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint
	(node_filesystem_size_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint node_filesystem_free_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint ! *100/(node_filesystem_avail_bytes {cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",moun

	(node_filesystem_size_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint node_filesystem_free_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint !~
CPU utilization	avg(irate(node_cpu_seconds_total{cluster=~"$cluster",instance=~"$node",mode="system"}[5m])) b
	avg(irate(node_cpu_seconds_total{cluster=~"$cluster",instance=~"$node",mode="user"}[5m])) by (
	avg(irate(node_cpu_seconds_total{cluster=~"$cluster",instance=~"$node",mode="iowait"}[5m])) by
	(1 - avg(irate(node_cpu_seconds_total{cluster=~"$cluster",instance=~"$node",mode="idle"}[5m])) b
Memory information	node_memory_MemTotal_bytes{cluster=~"$cluster",instance=~"$node"}
	node_memory_MemTotal_bytes{cluster=~"$cluster",instance=~"$node"} - node_memory_MemAvailable_bytes{cluster=~"$cluster",instance=~"$node"}
	node_memory_MemAvailable_bytes{cluster=~"$cluster",instance=~"$node"}
	(1 - (node_memory_MemAvailable_bytes{cluster=~"$cluster",instance=~"$node"} / (node_memory_MemTotal_bytes{cluster=~"$cluster",instance=~"$node"}))) * 100
Network bandwidth usage per second	irate(node_network_receive_bytes_total{cluster=~"$cluster",instance=~'$node',device=~"$device"}
	irate(node_network_transmit_bytes_total{cluster=~"$cluster",instance=~'$node',device=~"$device"
System average load	node_load1{cluster=~"$cluster",instance=~"$node"}
	node_load5{cluster=~"$cluster",instance=~"$node"}
	node_load15{cluster=~"$cluster",instance=~"$node"}

	sum(count(node_cpu_seconds_total{cluster=~"$cluster",instance=~"$node", mode='system'}) by (by(instance)
Disk read/write capacity per second	irate(node_disk_read_bytes_total{cluster=~"$cluster",instance=~"$node"}[5m])
	irate(node_disk_written_bytes_total{cluster=~"$cluster",instance=~"$node"}[5m])
Disk utilization	(node_filesystem_size_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint node_filesystem_free_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint !~ *100/(node_filesystem_avail_bytes {cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",moun (node_filesystem_size_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint node_filesystem_free_bytes{cluster=~"$cluster",instance=~'$node',fstype=~"ext.*|xfs",mountpoint !~
	node_filesystem_files_free{cluster=~"$cluster",instance=~'$node',fstype=~"ext.?|xfs"} / node_filesystem_files{cluster=~"$cluster",instance=~'$node',fstype=~"ext.?|xfs"}
Disk read/write rate (IOPS)	irate(node_disk_reads_completed_total{cluster=~"$cluster",instance=~"$node"}[5m])
	irate(node_disk_writes_completed_total{cluster=~"$cluster",instance=~"$node"}[5m])
	node_disk_io_now{cluster=~"$cluster",instance=~"$node"}
I/O operation time percentage per second	irate(node_disk_io_time_seconds_total{cluster=~"$cluster",instance=~"$node"}[5m])
Time taken for each IO read/write	irate(node_disk_read_time_seconds_total{cluster=~"$cluster",instance=~"$node"}[5m]) / irate(node_disk_reads_completed_total{instance=~"$node"}[5m])
	irate(node_disk_write_time_seconds_total{cluster=~"$cluster",instance=~"$node"}[5m]) / irate(node_disk_writes_completed_total{cluster=~"$cluster",instance=~"$node"}[5m])

	irate(node_disk_io_time_seconds_total{cluster=~"$cluster",instance=~"$node"}[5m])
	irate(node_disk_io_time_weighted_seconds_total{cluster=~"$cluster",instance=~"$node"}[5m])
Network socket connection information	node_netstat_Tcp_CurrEstab{cluster=~"$cluster",instance=~'$node'}
	node_sockstat_TCP_tw{cluster=~"$cluster",instance=~'$node'}
	node_sockstat_sockets_used{cluster=~"$cluster",instance=~'$node'}
	node_sockstat_UDP_inuse{cluster=~"$cluster",instance=~'$node'}
	node_sockstat_TCP_alloc{cluster=~"$cluster",instance=~'$node'}
	irate(node_netstat_Tcp_PassiveOpens{cluster=~"$cluster",instance=~'$node'}[5m])
	irate(node_netstat_Tcp_ActiveOpens{cluster=~"$cluster",instance=~'$node'}[5m])
	irate(node_netstat_Tcp_InSegs{cluster=~"$cluster",instance=~'$node'}[5m])
	irate(node_netstat_Tcp_OutSegs{cluster=~"$cluster",instance=~'$node'}[5m])
	irate(node_netstat_Tcp_RetransSegs{cluster=~"$cluster",instance=~'$node'}[5m])
Open file descriptors (left)/context switches per second (right)	node_filefd_allocated{cluster=~"$cluster",instance=~"$node"}
	irate(node_context_switches_total{cluster=~"$cluster",instance=~"$node"}[5m])
	(node_filefd_allocated{cluster=~"$cluster",instance=~"$node"}/node_filefd_maximum{cluster=~"$cl *100

Node Pod Monitoring

Chart Name	Query Statement
Pods	count(kube_pod_info{node=~"$node"})
Pod Request Memory	sum(kube_pod_container_resource_requests_memory_bytes{node=~"$node"})by(node)
Pod Request CPU Cores	sum(kube_pod_container_resource_requests_cpu_cores{node=~"$node"})by(node)
CPU Usage	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluste node=~"$node", container!="POD", container!=""}) by (pod)
CPU Quota	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluste node=~"$node", container!="POD", container!=""}) by (pod)
	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", node=~"$node"}) by (pod)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluste node=~"$node", container!="POD", container!=""}) by (pod) / sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", node=~"$node"}) by (pod)
	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", node=~"$node"}) by (pod)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluste node=~"$node", container!="POD", container!=""}) by (pod) / sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", node=~"$node"}) by (pod)

Memory Usage	sum(node_namespace_pod_container:container_memory_working_set_bytes{cluster="$cluster", node=~"$node", container!="", container!="POD"}) by (pod)
Memory Quota	sum(node_namespace_pod_container:container_memory_working_set_bytes{cluster="$cluster", node=~"$node",container!="", container!="POD"}) by (pod)
	sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster", node=~"$node"}) by (pod)
	sum(node_namespace_pod_container:container_memory_working_set_bytes{cluster="$cluster", node=~"$node",container!="", container!="POD"}) by (pod) / sum(kube_pod_container_resource_requests_memory_bytes{node=~"$node"}) by (pod)
	sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", node=~"$node"}) by (poc
	sum(node_namespace_pod_container:container_memory_working_set_bytes{cluster="$cluster", node=~"$node",container!="", container!="POD"}) by (pod) / sum(kube_pod_container_resource_limits_memory_bytes{node=~"$node"}) by (pod)
Pod List	group (kube_pod_info{host_ip="$node"})by(created_by_kind, created_by_name,host_network,pod_ip,pod,priority_class,namespace)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() max(kube_pod_status_phase{}==1) by (pod, phase)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() sum(container_memory_working_set_bytes) by (pod)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() sum(rate(container_cpu_usage_seconds_total{image!=""}[5m])) by (pod)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() max(time()-kube_pod_start_time by (pod)

	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) max(kube_pod_status_ready{condition="true" by (pod) or on() vector(0)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() max(rate(container_network_receive_bytes_total{image!=""}[5m])) by (pod) or on() vector(0)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() max(rate(container_network_transmit_bytes_total{image!=""}[5m])) by (pod) or on() vector(0)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() max(rate(container_fs_reads_bytes_total{container!="POD", container!=""}[5m])) by (pod) or on() vector(0)
	min(kube_pod_info{host_ip="$node"})by(pod) * on(pod) group_right() max(rate(container_fs_writes_bytes_total{container!="POD", container!=""}[5m])) by (pod) or on() vector(0)

Workload Monitoring Overview

Chart Name	Query Statement
CPU Usage	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
	scalar(kube_resourcequota{cluster="$cluster", namespace="$namespace", type="hard",resource="requests.cpu"})

	scalar(kube_resourcequota{cluster="$cluster", namespace="$namespace", type="hard",resource="limits.cpu"})
CPU Quota	count(namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type) /sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",

	workload_type="$type"}) by (workload, workload_type)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type) /sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
Memory Usage	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", container!="", container!="POD"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
	scalar(kube_resourcequota{cluster="$cluster", namespace="$namespace", type="hard",resource="requests.memory"})
	scalar(kube_resourcequota{cluster="$cluster", namespace="$namespace", type="hard",resource="limits.memory"})
Memory Quota	count(namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)
	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", container!="", container!="POD"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}

) by (workload, workload_type)

sum(
  kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",
namespace="$namespace"}
* on(namespace,pod)
  group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload_type="$type"}
) by (workload, workload_type)

sum(
    container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace",
container!="", container!="POD"}
  * on(namespace,pod)
    group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload_type="$type"}
) by (workload, workload_type)
/sum(
  kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",
namespace="$namespace"}
* on(namespace,pod)
  group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload_type="$type"}
) by (workload, workload_type)

sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$namespace"} * on(namespace,pod)   group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)

sum(
    container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace",
container!="", container!="POD"}
  * on(namespace,pod)
    group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload_type="$type"}

	) by (workload, workload_type) /sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload_type="$type"}) by (workload, workload_type)

Deployment

Chart Name	Query Statement
Age	time() - max(kube_deployment_created{cluster="$cluster",namespace="$namespace",deplo
Replicas(Pods)-Request	max(kube_deployment_spec_replicas{deployment="$workload",cluster="$cluster",namespa
Replicas(Pods)-Ready	max(kube_deployment_status_replicas_ready{deployment="$workload",cluster="$cluster",r
Replica Trend	max(kube_deployment_spec_replicas{deployment="$workload",cluster="$cluster",namespa
	max(kube_deployment_status_replicas{deployment="$workload",cluster="$cluster",namesp
	min(kube_deployment_status_replicas_ready{deployment="$workload",cluster="$cluster",na pod)
	min(kube_deployment_status_replicas_available{deployment="$workload",cluster="$cluste (instance, pod)
	min(kube_deployment_status_replicas_updated{deployment="$workload",cluster="$cluster" (instance, pod)
	min(kube_deployment_status_replicas_unavailable{deployment="$workload",cluster="$clus (instance, pod)
CPU Usage	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster= container!="POD", container!=""}

	* on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)
CPU Quota	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster= container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)
	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$na * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster= container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod) /sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$na * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)
	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$name * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster= container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe

	namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod) /sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$name * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)
CPU Limit-Total	sum(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() sum(kube_pod_container_resource_limits_cpu_cores{resource="cp cluster="$cluster",namespace="$namespace"}) by (pod))
CPU Request-Total	sum(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() sum(kube_pod_container_resource_requests_cpu_cores{resource= cluster="$cluster",namespace="$namespace"}) by (pod))
CPU Info	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset)

	* on(pod) group_right() max(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace="$namespace (pod, container)
	max(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",namespace="$ by (pod, container))by(container)
	max(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster",namespace="$nam (pod, container))by(container)
CPU Usage/Limit (%)	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace="$namespace (pod, container) / max by(container, pod) (kube_pod_container_resource_limits_cpu_cores{ cluster="$cluster",namespace="$namespace"})

CPU Usage/Request(%)	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace="$namespace (pod, container) / max by(container, pod) (kube_pod_container_resource_requests_cpu_co cluster="$cluster",namespace="$namespace"})
CPU User Time(%)	avg(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() (max(rate(container_cpu_user_seconds_total{cluster="$cluster",namespace="$namespace' (pod,container) / max(rate(container_cpu_user_seconds_total{cluster="$cluster",namespace="$namespace" [5m])+rate(container_cpu_system_seconds_total{cluster="$cluster",namespace="$namespa (pod,container))) by (pod,container)
Memory Usage	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)
Memory Quota	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="deployment"}) by (pod)

	sum(
	  kube_pod_container_resource_requests_memory_bytes{cluster="$cluster", namespace=
	* on(namespace,pod)
	  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
	namespace="$namespace", workload="$workload", workload_type="deployment"}
	) by (pod)

	sum(
	  container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co
	* on(namespace,pod)
	  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
	namespace="$namespace", workload="$workload", workload_type="deployment"}
	) by (pod)
	/sum(
	  kube_pod_container_resource_requests_memory_bytes{cluster="$cluster", namespace=
	* on(namespace,pod)
	  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
	namespace="$namespace", workload="$workload", workload_type="deployment"}
	) by (pod)

	sum(
	  kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$na
	* on(namespace,pod)
	  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
	namespace="$namespace", workload="$workload", workload_type="deployment"}
	) by (pod)

	sum(
	  container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co
	* on(namespace,pod)
	  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
	namespace="$namespace", workload="$workload", workload_type="deployment"}
	) by (pod)
	/sum(
	  kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$na
	* on(namespace,pod)
	  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
	namespace="$namespace", workload="$workload", workload_type="deployment"}
	) by (pod)

Memory Limit-Total	sum(label_replace(
	  max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl
	uid, pod, pod_ip, node),
	  "replicaset",
	  "$1",
	  "created_by_name",

	"(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() sum(container_spec_memory_limit_bytes{cluster="$cluster",namesp
Memory Request-Total	sum(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() sum(kube_pod_container_resource_requests_memory_bytes{resou cluster="$cluster",namespace="$namespace"}) by (pod))
Memory Info	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max by(container, pod) (container_memory_working_set_bytes{clus container!="", image!="", container!="POD"})
	max(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max by(container, pod) (kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",namespace="$
	max(label_replace(

	max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max by(container, pod) (kube_pod_container_resource_limits_memory_bytes{cluster="$cluster",namespace="$nam
Memory Usage/Limit(%)	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max by(container, pod) (container_memory_working_set_bytes{clus container!="", image!="", container!="POD"})/max by(container, pod) (kube_pod_container_resource_limits_memory_bytes{resource="memory", cluster="$cluste
Memory Usage/Request(%)	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max by(container, pod) (container_memory_working_set_bytes{clus container!="", image!="", container!="POD"})/max by(container, pod) (kube_pod_container_resource_requests_memory_bytes{resource="memory", cluster="$clu
Sockets	sum(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name",

	"(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() sum(container_sockets{cluster="$cluster",namespace="$namespac
Network In	sum(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() sum(rate(container_network_receive_bytes_total{cluster="$cluster",
Network Out	sum(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() sum(rate(container_network_transmit_bytes_total{cluster="$cluster"
Network Errors	sum(label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() (sum(container_network_receive_errors_total{cluster="$cluster",nan sum(container_network_transmit_errors_total{cluster="$cluster",namespace="$namespace'
Network IO	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node),

	"replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(rate(container_network_receive_bytes_total{cluster="$cluster",
	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(rate(container_network_transmit_bytes_total{cluster="$cluster"
File System Read	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(rate(container_fs_reads_bytes_total{cluster="$cluster",namesp container!=""}[5m])) by (pod,container)
File System Write	label_replace(max(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Repl uid, pod, pod_ip, node), "replicaset", "$1", "created_by_name", "(.+)") * on(replicaset) group_left() max(kube_replicaset_owner{cluster="$cluster",namespace="$namespace",owner_kind="De (replicaset) * on(pod) group_right() max(rate(container_fs_writes_bytes_total{cluster="$cluster",namesp container!=""}[5m])) by (pod,container)

StatefulSet

Chart Name	Query Statement
Generation	max(kube_statefulset_metadata_generation{cluster="$cluster",namespace="$namespace",
Replicas(Pods)-Request	max(kube_statefulset_replicas{statefulset="$workload",cluster="$cluster",namespace="$na
Replicas(Pods)-Ready	max(kube_statefulset_status_replicas_ready{statefulset="$workload",cluster="$cluster",nan
Age	time() - max(kube_statefulset_created{cluster="$cluster",namespace="$namespace",statefu
Replica Trend	max(kube_statefulset_replicas{statefulset="$workload",cluster="$cluster",namespace="$na
	max(kube_statefulset_status_replicas{statefulset="$workload",cluster="$cluster",namespac pod)
	min(kube_statefulset_status_replicas_ready{statefulset="$workload",cluster="$cluster",nam (instance, pod)
	min(kube_statefulset_status_replicas_available{statefulset="$workload",cluster="$cluster",r (instance, pod)
	min(kube_statefulset_status_replicas_updated{statefulset="$workload",cluster="$cluster",n (instance, pod)
CPU Usage	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster= namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod)
CPU Quota	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster= namespace="$namespace", container!="POD", container!=""} * on(namespace,pod)

```
  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
namespace="$namespace", workload="$workload", workload_type="statefulset"}
) by (pod)
```

```
sum(
  kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$na
 * on(namespace,pod)
  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
namespace="$namespace", workload="$workload", workload_type="statefulset"}
) by (pod)
```

```
sum(
  node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster=
namespace="$namespace", container!="POD", container!=""}
 * on(namespace,pod)
  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
namespace="$namespace", workload="$workload", workload_type="statefulset"}
) by (pod)
/sum(
  kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$na
 * on(namespace,pod)
  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
namespace="$namespace", workload="$workload", workload_type="statefulset"}
) by (pod)
```

```
sum(
  kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$name
 * on(namespace,pod)
  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
namespace="$namespace", workload="$workload", workload_type="statefulset"}
) by (pod)
```

```
sum(
  node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster=
namespace="$namespace", container!="POD", container!=""}
 * on(namespace,pod)
  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
namespace="$namespace", workload="$workload", workload_type="statefulset"}
) by (pod)
/sum(
  kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$name
 * on(namespace,pod)
  group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe
namespace="$namespace", workload="$workload", workload_type="statefulset"}
) by (pod)
```

CPU Limit-Total	sum(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind= created_by_name="$workload"}) by (pod) * on(pod) group_right() sum(kube_pod_container_resource_limits_cpu_cores{resource="cp cluster="$cluster",namespace="$namespace"}) by (pod))
CPU Request-Total	sum(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind= created_by_name="$workload"}) by (pod) * on(pod) group_right() sum(kube_pod_container_resource_requests_cpu_cores{resource= cluster="$cluster",namespace="$namespace"}) by (pod))
CPU Info	group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Stat created_by_name="$workload"}) by (pod) * on(pod) group_right() max(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace="$namespace [5m])) by (pod, container,image)
	group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Stat created_by_name="$workload"}) by (pod) * on(pod) group_right() max(kube_pod_container_resource_limits_cpu_cores{cluster="$clus (pod, container,image)
	group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Stat created_by_name="$workload"}) by (pod) * on(pod) group_right() max(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",namespace="$ container,image)
CPU Usage/Limit (%)	group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Stat created_by_name="$workload"}) by (pod) * on(pod) group_right() max(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace="$namespace [5m])) by (pod, container) / max by(container, pod) (kube_pod_container_resource_limits_cp cluster="$cluster",namespace="$namespace"})
CPU Usage/Request(%)	group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Stat created_by_name="$workload"}) by (pod) * on(pod) group_right() max(rate(container_cpu_usage_seconds_total{cluster="$cluster",namespace="$namespace [5m])) by (pod, container) / max by(container, pod) (kube_pod_container_resource_requests cluster="$cluster",namespace="$namespace"})
CPU User Time(%)	avg(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind=' created_by_name="$workload"}) by (pod)

	* on(pod) group_right() (max(rate(container_cpu_user_seconds_total{cluster="$cluster",namespace="$namespace' [5m])) by (pod, container,image) / max(rate(container_cpu_user_seconds_total{cluster="$cluster",namespace="$namespace" [5m])+rate(container_cpu_system_seconds_total{cluster="$cluster",namespace="$namespa [5m])) by (pod,container,image))) by (pod,container,image)
Memory Usage	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod)
Memory Quota	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod)
	sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster", namespace= * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod)
	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod) /sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster", namespace= * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod)
	sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$na * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe

	namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod)
	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", co * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod) /sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$na * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabe namespace="$namespace", workload="$workload", workload_type="statefulset"}) by (pod)
Memory Limit-Total	sum(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind= created_by_name="$workload"}) by (pod) * on(pod) group_right() sum(container_spec_memory_limit_bytes{cluster="$cluster",namespace="$namespace",co (pod))
Memory Request-Total	sum(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind= created_by_name="$workload"}) by (pod) * on(pod) group_right() sum(kube_pod_container_resource_requests_memory_bytes{resou cluster="$cluster",namespace="$namespace"}) by (pod))
Memory Info	avg(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind=' created_by_name="$workload"}) by (pod) * on(pod) group_right() max by(container, pod, image) (container_memory_working_set_bytes{cluster="$cluster",namespace="$namespace", cont container!="POD"}))by (container, pod, image)
	max(avg(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_k created_by_name="$workload"}) by (pod) * on(pod) group_right() max by(container, pod, image) (kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",namespace="$ pod))by(container)
	max(avg(group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_k created_by_name="$workload"}) by (pod) * on(pod) group_right() max by(container, pod) (kube_pod_container_resource_limits_memory_bytes{cluster="$cluster",namespace="$nam pod))by(container)
Memory	group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Stat

Usage/Limit(%)	created_by_name="$workload"}) by (pod) * on(pod) group_right() max by(container, pod) (container_memory_working_set_bytes{cluster="$cluster",namespace="$namespace", cont container!="POD"})/max by(container, pod) (kube_pod_container_resource_limits_memory_ cluster="$cluster",namespace="$namespace"})
Memory Usage/Request(%)	group(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Stat created_by_name="$workload"}) by (pod) * on(pod) group_right() max by(container, pod) (container_memory_working_set_bytes{cluster="$cluster",namespace="$namespace", cont container!="POD"})/max by(container, pod) (kube_pod_container_resource_requests_mem cluster="$cluster",namespace="$namespace"})
Sockets	sum(sum(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind=" created_by_name="$workload"}) by (pod) * on(pod) group_right() sum(container_sockets{cluster="$cluster",namespace="$namespac
Network In	sum(sum(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind=" created_by_name="$workload"}) by (pod) * on(pod) group_right() sum(rate(container_network_receive_bytes_total{cluster="$cluster", by (pod))
Network Out	sum(sum(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind=" created_by_name="$workload"}) by (pod) * on(pod) group_right() sum(rate(container_network_transmit_bytes_total{cluster="$cluster" by (pod))
Network Errors	sum(sum(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind=" created_by_name="$workload"}) by (pod) * on(pod) group_right() (sum(container_network_receive_errors_total{cluster="$cluster",nan sum(container_network_transmit_errors_total{cluster="$cluster",namespace="$namespace"
Network IO	sum(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="Statef created_by_name="$workload"}) by (pod) * on(pod) group_right() max(rate(container_network_receive_bytes_total{cluster="$cluster", by (pod)
	-sum(kube_pod_info{cluster="$cluster",namespace="$namespace",created_by_kind="State created_by_name="$workload"}) by (pod) * on(pod) group_right() max(rate(container_network_transmit_bytes_total{cluster="$cluster" by (pod)

DaemonSet

Chart Name	Query Statement
CPU Usage	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)
CPU Quota	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)
	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod) /sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",

	namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)
	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cluster", namespace="$namespace", container!="POD", container!=""} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod) /sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)
Memory Usage	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", container!="", container!="POD"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)
Memory Quota	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", container!="", container!="POD"} * on(namespace,pod) group_left(workload, workload_type)

```
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload="$workload", workload_type="daemonset"}
) by (pod)
```

```
sum(
kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",
namespace="$namespace"}
  * on(namespace,pod)
    group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload="$workload", workload_type="daemonset"}
) by (pod)
```

```
sum(
    container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace",
container!="", container!="POD"}
  * on(namespace,pod)
    group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload="$workload", workload_type="daemonset"}
) by (pod)
/sum(

kube_pod_container_resource_requests_memory_bytes{cluster="$cluster",
namespace="$namespace"}
  * on(namespace,pod)
    group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload="$workload", workload_type="daemonset"}
) by (pod)
```

```
sum(
kube_pod_container_resource_limits_memory_bytes{cluster="$cluster",
namespace="$namespace"}
  * on(namespace,pod)
    group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
workload="$workload", workload_type="daemonset"}
) by (pod)
```

```
sum(
    container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace",
container!="", container!="POD"}
  * on(namespace,pod)
    group_left(workload, workload_type)
namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace",
```

	workload="$workload", workload_type="daemonset"}) by (pod) /sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$namespace"} * on(namespace,pod) group_left(workload, workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster="$cluster", namespace="$namespace", workload="$workload", workload_type="daemonset"}) by (pod)

Cluster Pod Monitoring

Chart Name	Query Statement
Age	time() - max(kube_pod_created{pod=~"$pod",cluster="$cluster",namespace="$namespace"})
Restart Count-Last 1 Hour	ceil(sum (increase(kube_pod_container_status_restarts_total{pod=~"$pod",cluster="$cluster",nam
Requests-CPU	sum(kube_pod_container_resource_requests_cpu_cores{pod=~"$pod"}) or vector(0)
Requests-Memory	sum(kube_pod_container_resource_requests_memory_bytes{pod=~"$pod"}) or vector(0)
Limits-CPU	sum(kube_pod_container_resource_limits_cpu_cores{pod=~"$pod"}) or vector(0)
Limits-Memory	sum(kube_pod_container_resource_limits_memory_bytes{pod=~"$pod"}) or vector(0)
Containers	group by (image, container,pod) (kube_pod_container_info{cluster="$cluster",namespace="$name
	sum by (container,pod)(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster",nan
	sum by (container,pod)(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster" pod=~"$pod"})

	max by (container,pod)(kube_pod_container_status_running{cluster="$cluster",namespace="$nam
	sum by (container,pod)(kube_pod_container_resource_limits{resource="cpu",cluster="$cluster",na
	sum by (container,pod)(kube_pod_container_resource_limits{resource="memory",cluster="$cluster pod=~"$pod"})
	max by (container,pod)(kube_pod_container_status_restarts_total{cluster="$cluster",namespace='
CPU Usage (%)	max(irate(container_cpu_usage_seconds_total{pod=~"$pod",container!="",container!="POD",cluste [1m])) by (container,namespace,pod) / max(container_spec_cpu_quota{pod=~"$pod",container!="",container!="POD",cluster="$cluster",na (container,namespace,pod) or on() vector(0)
CPU Usage By Cores	max(irate(container_cpu_usage_seconds_total{pod=~"$pod",container!="",container!="POD",cluste [1m])) by (pod,container,namespace)or on() vector(0)
CPU Load (10s)	max(container_cpu_load_average_10s{namespace=~"$namespace", pod=~"$pod", container!="", 1000)by(pod,container)
CPU Throttled Percent	max (rate (container_cpu_cfs_throttled_seconds_total{image!="", container!="", cluster="$cluster",r pod=~"$pod"}[1m])) by (container,pod) / max (rate (container_cpu_cfs_periods_total{image!="", cor cluster="$cluster",namespace=~"$namespace", pod=~"$pod"}[1m])) by (container,pod) or on() vect
CPU Quota	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cl pod="$pod", container!="POD", container!=""}) by (container)
	sum(kube_pod_container_resource_requests_cpu_cores{cluster="$cluster", namespace="$names
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cl pod="$pod", container!="POD", container!=""}) by (container) / sum(kube_pod_container_resource_ namespace="$namespace", pod="$pod"}) by (container)
	sum(kube_pod_container_resource_limits_cpu_cores{cluster="$cluster", namespace="$namespac
	sum(node_namespace_pod_container:container_cpu_usage_seconds_total:sum_rate{cluster="$cl pod="$pod", container!="POD", container!=""}) by (container) / sum(kube_pod_container_resource_ namespace="$namespace", pod="$pod"}) by (container)

Memory Usage (WSS)	max(container_memory_working_set_bytes{pod=~"$pod",container !="",container!="POD",cluster= by (pod,namespace,container)
Memory Usage	max(container_memory_usage_bytes{pod=~"$pod",container !="",container!="POD",cluster="$clus (pod,namespace,container)
Memory Usage (RSS)	max(container_memory_rss{pod=~"$pod",container !="",container!="POD",cluster="$cluster",name (pod,namespace,container) or on() vector(0)
Memory Cache	max(container_memory_cache{pod=~"$pod",container !="",container!="POD",cluster="$cluster",na (pod,namespace,container)
Usage WSS/Limit (%)	(max(container_memory_working_set_bytes{pod=~"$pod",container !="",container!="POD",cluster= by (pod,namespace,container)/ max(container_spec_memory_limit_bytes{pod=~"$pod",container !="",container!="POD",cluster="$cluster",namespace=~"$namespace"}) by (pod,namespace, contai
Usage/Limit (%)	(max(container_memory_usage_bytes{pod=~"$pod",container !="",container!="POD",cluster="$clu (pod,namespace,container)/ max(container_spec_memory_limit_bytes{pod=~"$pod",container !="",container!="POD",cluster="$cluster",namespace=~"$namespace"}) by (pod,namespace, contai
Usage RSS/Limit (%)	(max(container_memory_rss{pod=~"$pod",container !="",container!="POD",cluster="$cluster",nam (pod,namespace,container)/ sum(container_spec_memory_limit_bytes{pod=~"$pod",container !="",container!="POD",cluster="$cluster",namespace=~"$namespace"}) by (pod,namespace, contai
Memory Failcnt	max (increase(container_memory_failcnt{cluster="$cluster",namespace=~"$namespace", pod=~"$p (pod,container)
Memory Quota	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", pod="$ by (container)
	sum(kube_pod_container_resource_requests_memory_bytes{cluster="$cluster", namespace="$na
	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", pod="$ by (container) / sum(kube_pod_container_resource_requests_memory_bytes{namespace="$name
	sum(kube_pod_container_resource_limits_memory_bytes{cluster="$cluster", namespace="$name (container)
	sum(container_memory_working_set_bytes{cluster="$cluster", namespace="$namespace", pod="$ by (container) / sum(kube_pod_container_resource_limits_memory_bytes{namespace="$namespa

Network Input	max (rate (container_network_receive_bytes_total{image!="",cluster="$cluster",namespace=~"$nar by(pod)
Network Output	max (rate (container_network_transmit_bytes_total{image!="",cluster="$cluster",namespace=~"$na [1m]))by(pod)
Network Input Error (%)	max (increase (container_network_receive_packets_dropped_total{id!="/", cluster="$cluster",name [1m])) by (pod,interface) / max (increase (container_network_receive_packets_total{id!="/", cluster= pod=~"$pod"}[1m])) by (pod,interface)
	max (increase (container_network_receive_errors_total{id!="/", cluster="$cluster",namespace=~"$r (pod,interface) / max (increase (container_network_receive_packets_total{id!="/", cluster="$cluster pod=~"$pod"}[1m])) by (pod,interface)
Network Output Error (%)	max (increase (container_network_transmit_packets_dropped_total{id!="/", cluster="$cluster",nam [1m])) by (pod,interface) / max (increase (container_network_transmit_packets_total{id!="/", cluster pod=~"$pod"}[1m])) by (pod,interface)
	max (increase (container_network_transmit_errors_total{id!="/", cluster="$cluster",namespace=~"$ (pod,interface) / max (increase (container_network_receive_packets_total{id!="/", cluster="$cluster pod=~"$pod"}[1m])) by (pod,interface)
File System Read	max (rate(container_fs_reads_bytes_total{cluster="$cluster",namespace=~"$namespace", pod=~"$ (container,pod)
File System Write	max (rate(container_fs_writes_bytes_total{cluster="$cluster",namespace=~"$namespace", pod=~"$ (container,pod)
Network Socket	max(container_sockets{cluster="$cluster",namespace=~"$namespace", pod=~"$pod", container!="
Process Number	count(container_processes{cluster="$cluster",namespace=~"$namespace", pod=~"$pod", containe

Cluster Network Monitoring

Chart Name	Query Statement
Current Rate of	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~".+"} (namespace))

Bytes Received	
Current Rate of Bytes Transmitted	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~".+" (namespace))
Current Status	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~".+"} (namespace))
	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~".+" (namespace))
	sort_desc(avg(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~".+"}[(namespace))
	sort_desc(avg(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~".+"} (namespace))
	sort_desc(sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~". (namespace))
	sort_desc(sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~" (namespace))
	sort_desc(sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~" (namespace))
	sort_desc(sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster",name [5m])) by (namespace))
Average Rate of Bytes Received	sort_desc(avg(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~".+"}[(namespace))
Average Rate of Bytes Transmitted	sort_desc(avg(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~".+"} (namespace))
Receive Bandwidth	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~".+"} (namespace))
Transmit Bandwidth	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~".+" (namespace))

Rate of Received Packets	sort_desc(sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~". (namespace))
Rate of Transmitted Packets	sort_desc(sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~" (namespace))
Rate of Received Packets Dropped	sort_desc(sum(irate(container_network_receive_packets_dropped_total{cluster=~"$cluster",names [5m])) by (namespace))
Rate of Transmitted Packets Dropped	sort_desc(sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster",name [5m])) by (namespace))
Rate of TCP Retransmits out of all sent segments	sort_desc(sum(rate(node_netstat_Tcp_RetransSegs{cluster=~"$cluster"}[5m]) / rate(node_netstat_Tcp_OutSegs{cluster=~"$cluster"}[$interval:$resolution])) by (instance))
Rate of TCP SYN Retransmits out of all retransmits	sort_desc(sum(rate(node_netstat_TcpExt_TCPSynRetrans{cluster=~"$cluster"}[$interval:$resolutic rate(node_netstat_Tcp_RetransSegs{cluster=~"$cluster"}[$interval:$resolution])) by (instance))

Namespace Pods Network Monitoring

Chart Name	Query Statement
Current Rate of Bytes Received	sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$namespace"]
Current Rate of	sum(irate(container_network_transmit_bytes_total{namespace=~"$namespace"}[5m]))

Bytes Transmitted	
Current Status	sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$namespace"} (pod)
	sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$namespace' (pod)
	sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~"$namespac (pod)
	sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~"$namespa (pod)
	sum(irate(container_network_receive_packets_dropped_total{cluster=~"$cluster",namespace=~"$r [5m])) by (pod)
	sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster",namespace=~"$ [5m])) by (pod)
Receive Bandwidth	sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$namespace"} (pod)
Transmit Bandwidth	sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$namespace' (pod)
Rate of Received Packets	sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~"$namespac (pod)
Rate of Transmitted Packets	sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~"$namespa (pod)
Rate of Received Packets Dropped	sum(irate(container_network_receive_packets_dropped_total{cluster=~"$cluster",namespace=~"$r [5m])) by (pod)
Rate of Transmitted Packets Dropped	sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster", namespace=~"$namespace"}[5m])) by (pod)

Namespace Workload Network Monitoring

Chart Name	Query Statement
Current Rate of Bytes Received	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Current Rate of Bytes Transmitted	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$n * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Current Status	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$n * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
	sort_desc(avg(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
	sort_desc(avg(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
	sort_desc(sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~"$ * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
	sort_desc(sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~"

	* on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
	sort_desc(sum(irate(container_network_receive_packets_dropped_total{cluster=~"$cluster",names [5m]) * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
	sort_desc(sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster",name [5m]) * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Average Rate of Bytes Received	sort_desc(avg(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Average Rate of Bytes Transmitted	sort_desc(avg(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Receive Bandwidth	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Transmit Bandwidth	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$n * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Rate of Received Packets	sort_desc(sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~"$ * on (namespace,pod) group_left(workload,workload_type)

	namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Rate of Transmitted Packets	sort_desc(sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~" * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Rate of Received Packets Dropped	sort_desc(sum(irate(container_network_receive_packets_dropped_total{cluster=~"$cluster",names [5m]) * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))
Rate of Transmitted Packets Dropped	sort_desc(sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster",name [5m]) * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload_type="$type"}) by (workload))

Pod Network Monitoring

Chart Name	Query Statement
Current Rate of Bytes Received	sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$namespace", pod=~"$pod"}[5m]))
Current Rate of Bytes Transmitted	sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$namespace' pod=~"$pod"}[5m]))
Receive Bandwidth	sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$namespace", pod=~"$pod"}[5m])) by (pod)
Transmit Bandwidth	sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$namespace' pod=~"$pod"}[5m])) by (pod)
Rate of	sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~"$namespac

Received Packets	pod=~"$pod"}[5m])) by (pod)
Rate of Transmitted Packets	sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~"$namespa pod=~"$pod"}[5m])) by (pod)
Rate of Received Packets Dropped	sum(irate(container_network_receive_packets_dropped_total{cluster=~"$cluster",namespace=~"$r pod=~"$pod"}[5m])) by (pod)
Rate of Transmitted Packets Dropped	sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster",namespace=~"$ pod=~"$pod"}[5m])) by (pod)

Workload Network Monitoring

Chart Name	Query Statement
Current Rate of Bytes Received	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Current Rate of Bytes Transmitted	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$n * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Average Rate of Bytes Received	sort_desc(avg(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Average Rate of Bytes Transmitted	sort_desc(avg(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type)

	namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Receive Bandwidth	sort_desc(sum(irate(container_network_receive_bytes_total{cluster=~"$cluster",namespace=~"$na * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Transmit Bandwidth	sort_desc(sum(irate(container_network_transmit_bytes_total{cluster=~"$cluster",namespace=~"$n * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Rate of Received Packets	sort_desc(sum(irate(container_network_receive_packets_total{cluster=~"$cluster",namespace=~"$ * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Rate of Transmitted Packets	sort_desc(sum(irate(container_network_transmit_packets_total{cluster=~"$cluster",namespace=~" * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Rate of Received Packets Dropped	sort_desc(sum(irate(container_network_receive_packets_dropped_total{cluster=~"$cluster",names [5m]) * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))
Rate of Transmitted Packets Dropped	sort_desc(sum(irate(container_network_transmit_packets_dropped_total{cluster=~"$cluster",name [5m]) * on (namespace,pod) group_left(workload,workload_type) namespace_workload_pod:kube_pod_owner:relabel{cluster=~"$cluster",namespace=~"$namespac workload=~"$workload", workload_type="$type"}) by (pod))

PVC Storage Monitoring

Chart	Query Statement	Metrics Used

Name		
Volume Space Usage	(sum without(instance, node) (kubelet_volume_stats_capacity_bytes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"}) - sum without(instance, node) (kubelet_volume_stats_available_bytes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"}))	kubelet_volume_stats_capacity_bytes
		kubelet_volume_stats_available_bytes
	sum without(instance, node) (kubelet_volume_stats_available_bytes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"})	kubelet_volume_stats_available_bytes
Volume Space Usage	(kubelet_volume_stats_capacity_bytes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"} - kubelet_volume_stats_available_bytes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"}) / kubelet_volume_stats_capacity_bytes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"} * 100	kubelet_volume_stats_capacity_bytes
		kubelet_volume_stats_available_bytes
		kubelet_volume_stats_capacity_bytes
Volume inodes Usage	sum without(instance, node) (kubelet_volume_stats_inodes_used{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"})	kubelet_volume_stats_inodes_used
	(sum without(instance, node) (kubelet_volume_stats_inodes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"}) -	kubelet_volume_stats_inodes

	sum without(instance, node) (kubelet_volume_stats_inodes_used{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"}))	kubelet_volume_stats_inodes_used
Volume inodes Usage	kubelet_volume_stats_inodes_used{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"} / kubelet_volume_stats_inodes{cluster="$cluster", job="kubelet", namespace="$namespace", persistentvolumeclaim="$volume"} * 100	kubelet_volume_stats_inodes_used kubelet_volume_stats_inodes

Data Collection Configuration

Last updated：2024-07-30 18:14:31

Overview

This document describes how to configure monitoring collection items for the associated cluster.

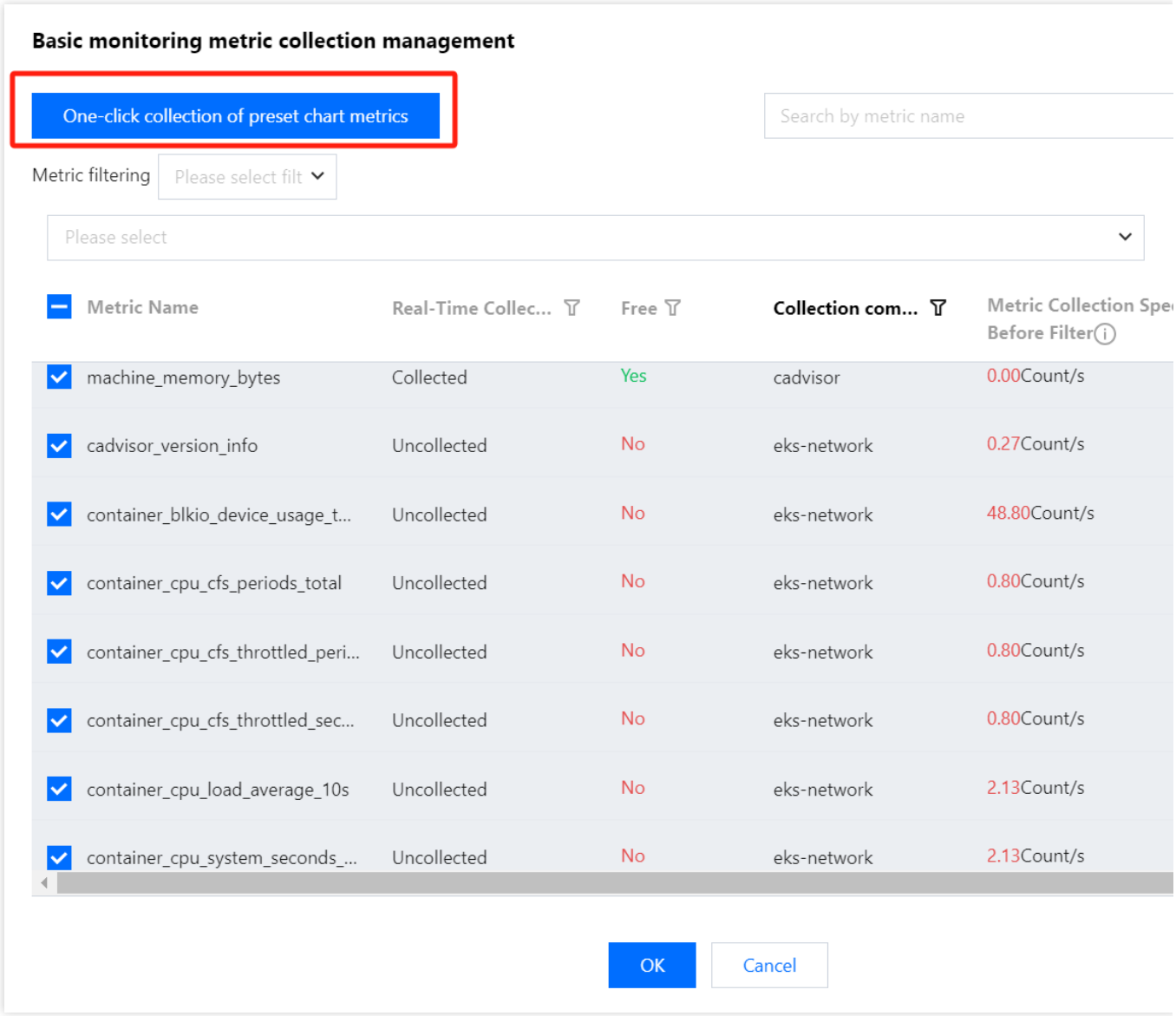
Prerequisites

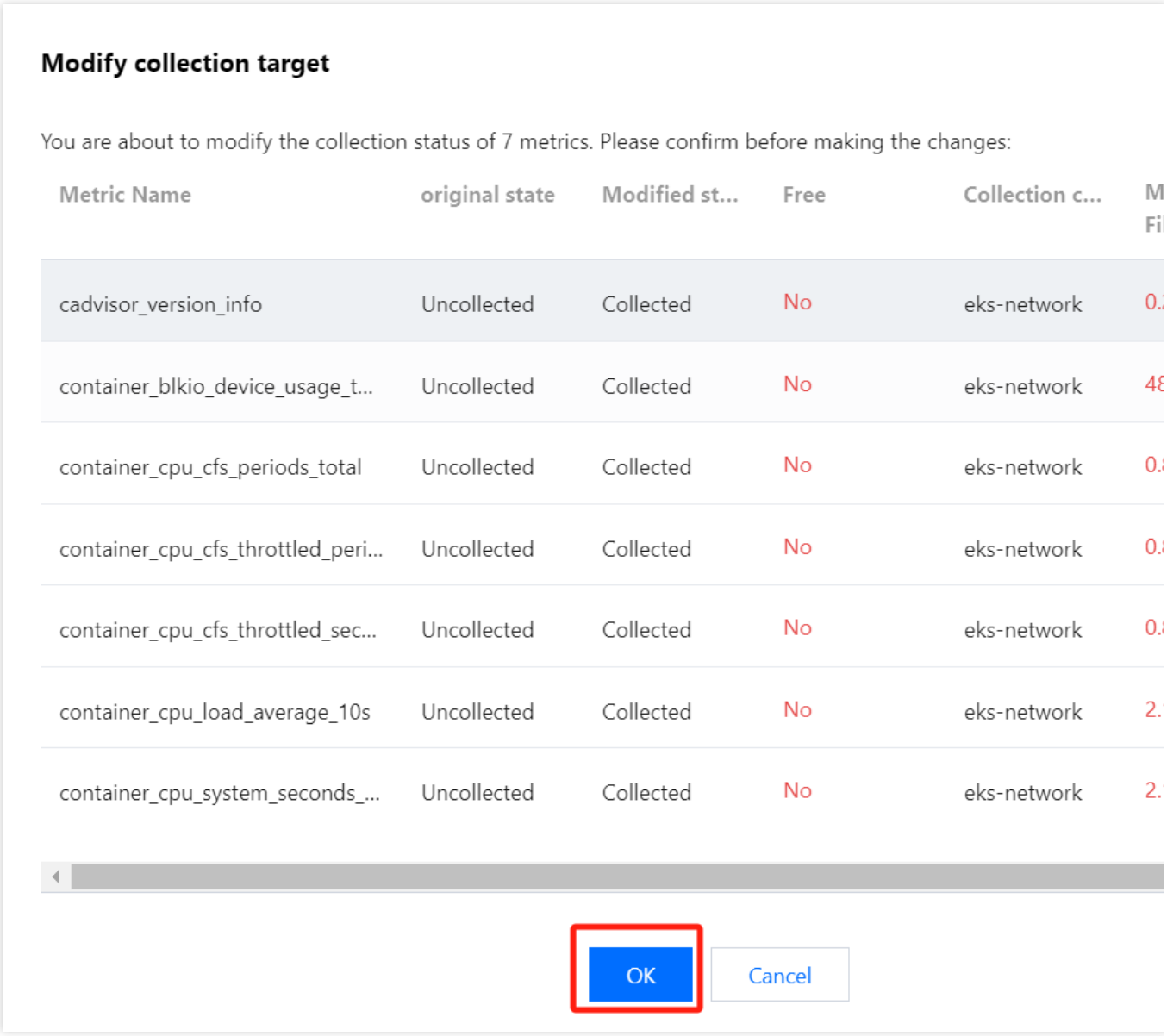
Before configuring monitoring collection items, you need to perform the following operations:
The Prometheus monitoring instance has been successfully created.
The cluster to be monitored has been associated with the corresponding instance.

Directions

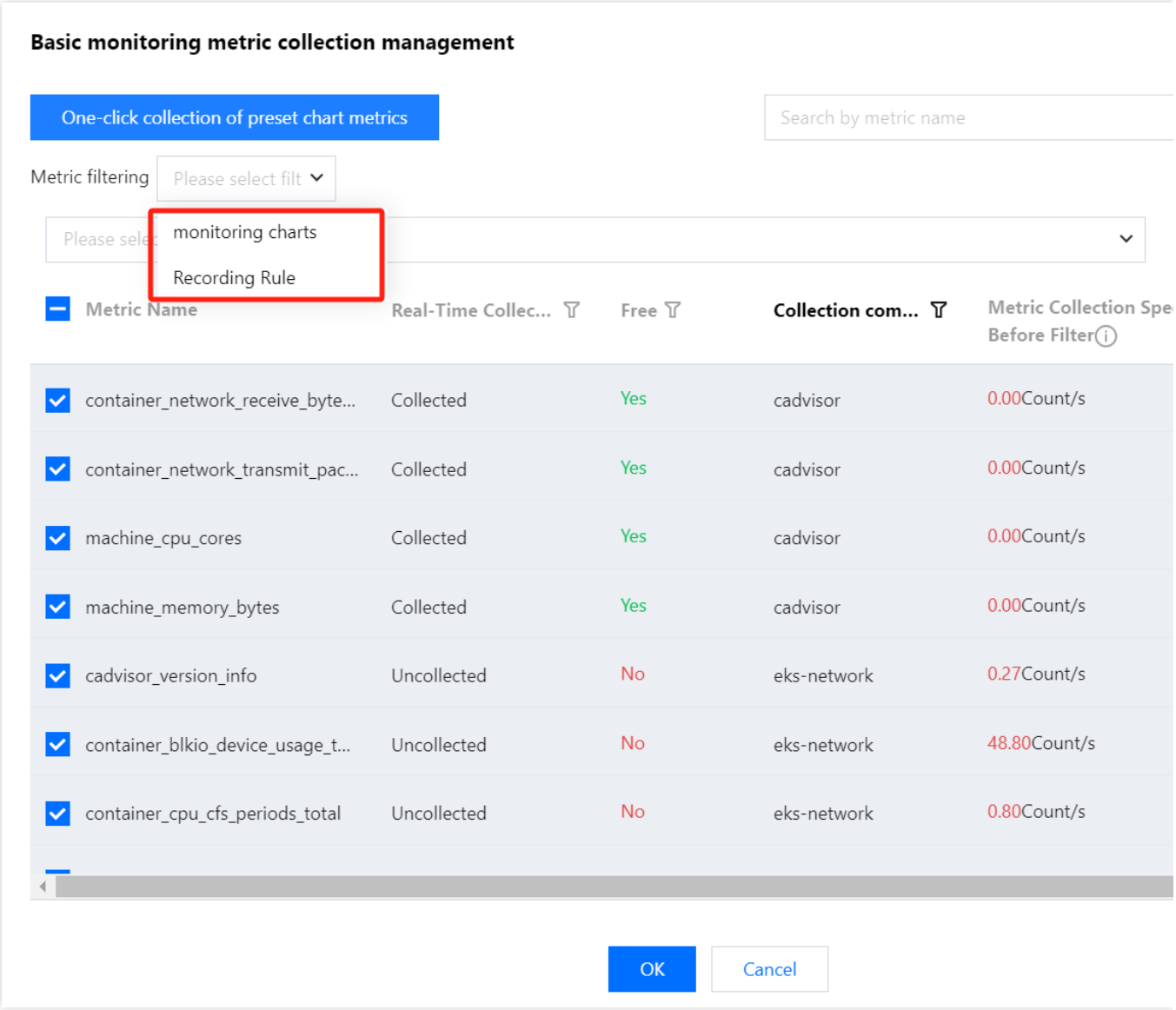
Configuring Data Collection

1. Log in to TMP Console.
2. On the instance list page, select the name of the instance that requires configuring data collection rules and enter its details page.
3. On the **Data Collection > Integration with TKE** page, click **Data Collection Configuration** on the right of the instance to enter the collection configuration list page.
4. Click **Metrics collection management** at the top of the page to pop up the Basic Monitoring Metrics Collection Management page.
If you need to collect all container chart metrics, you can click **One-click collection of preset chart metrics** in the pop-up window, and then click **OK** in the pop-up confirmation window to collect all preset chart metrics.





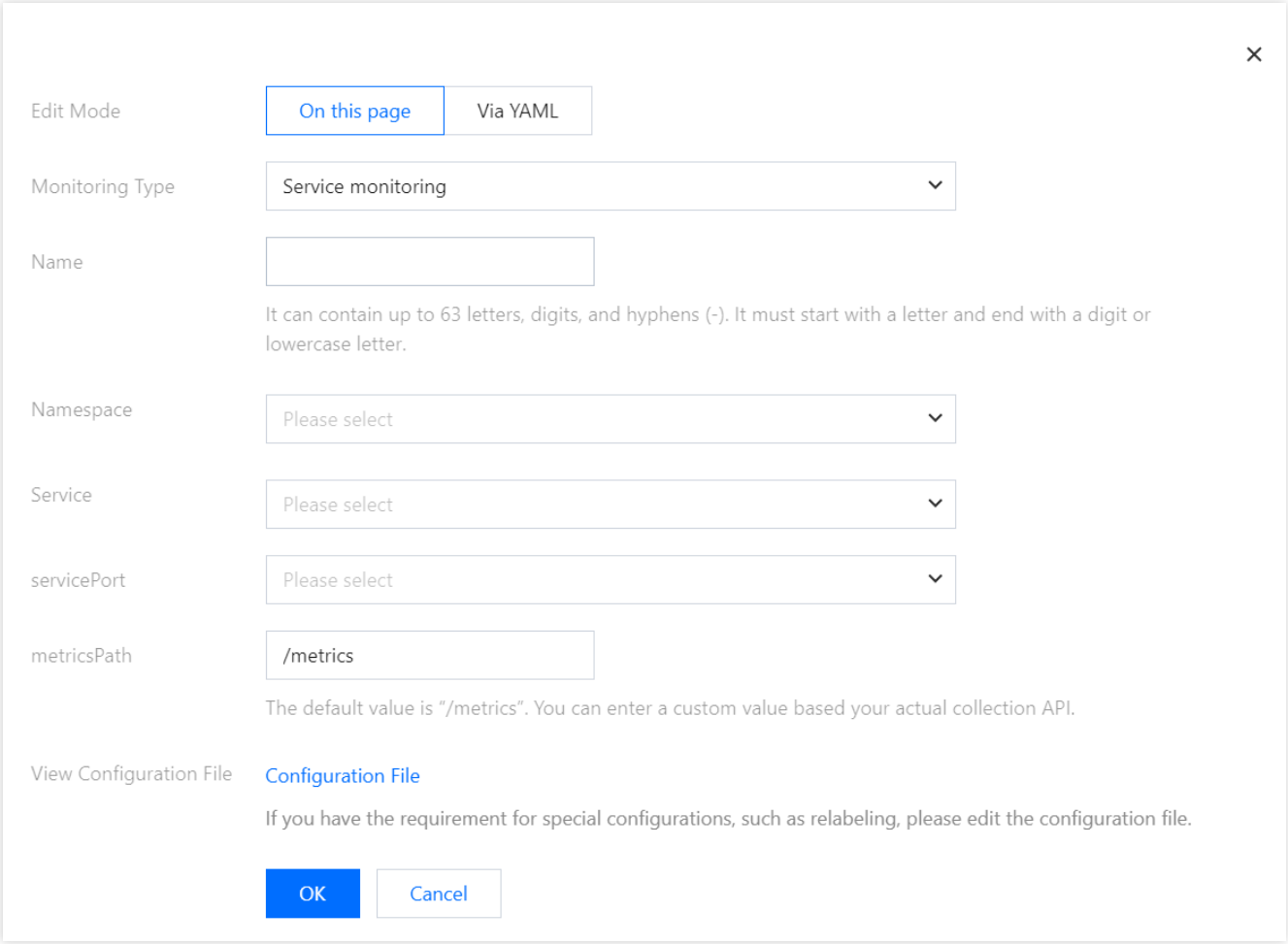
If only part of the metrics need to be collected, you can filter the metrics through, **monitoring charts** and **Recording Rule**, select the metrics you need, and then click **OK**. For details, see the following figure.



5. On the "Data Collection Configuration" page, click **Customize Monitoring Configuration** to add a new data collection configuration. TKE has preset some collection configuration files for collecting standard monitoring data. You can configure new data collection rules to monitor your business data in the following two ways:
Adding Configuration in the Console
Adding Configuration via YAML File

Monitoring Service

1. Click **On this Page**.
2. In the "Create Collection Configuration" pop-up window, fill in the configuration information. For details, see the following figure.



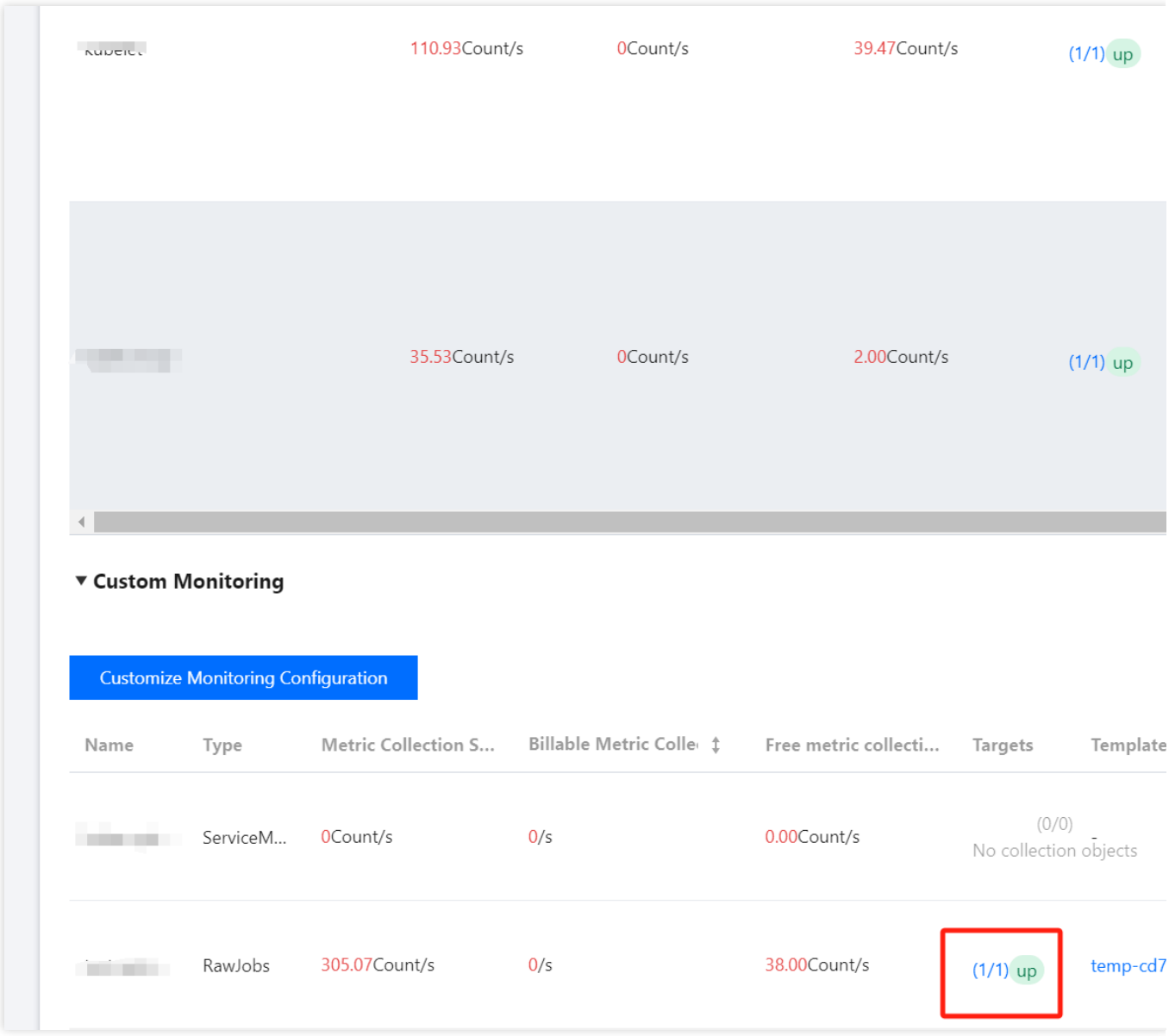
1. Click **Via YAML**.
2. In the pop-up window, select the monitoring type and fill in the corresponding configuration. You can complete the data collection configuration by submitting the corresponding YAML as per community usage.

Workload Monitoring: Corresponding configuration is PodMonitors.

Service Monitoring: Corresponding configuration is ServiceMonitors.

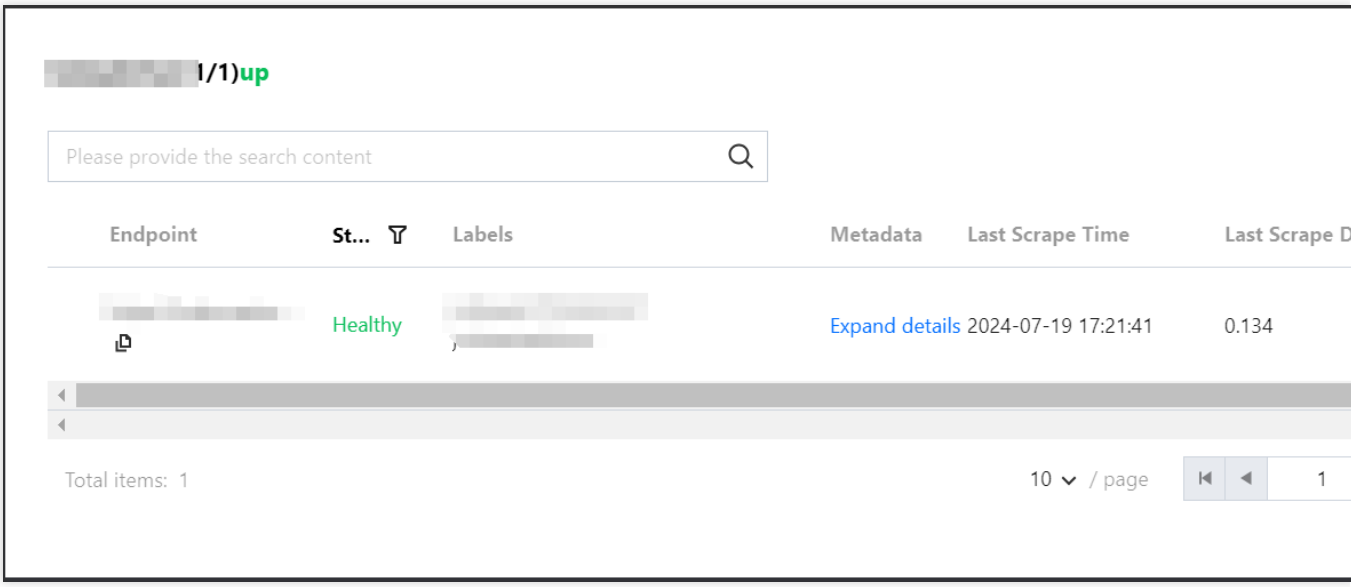
RawJobs Monitoring: Corresponding configuration is RawJobs.

6. Click **OK** to complete the configuration.
7. On the Data Collection Configuration page of the instance, view the status of the collection target. For details, see the following figure.



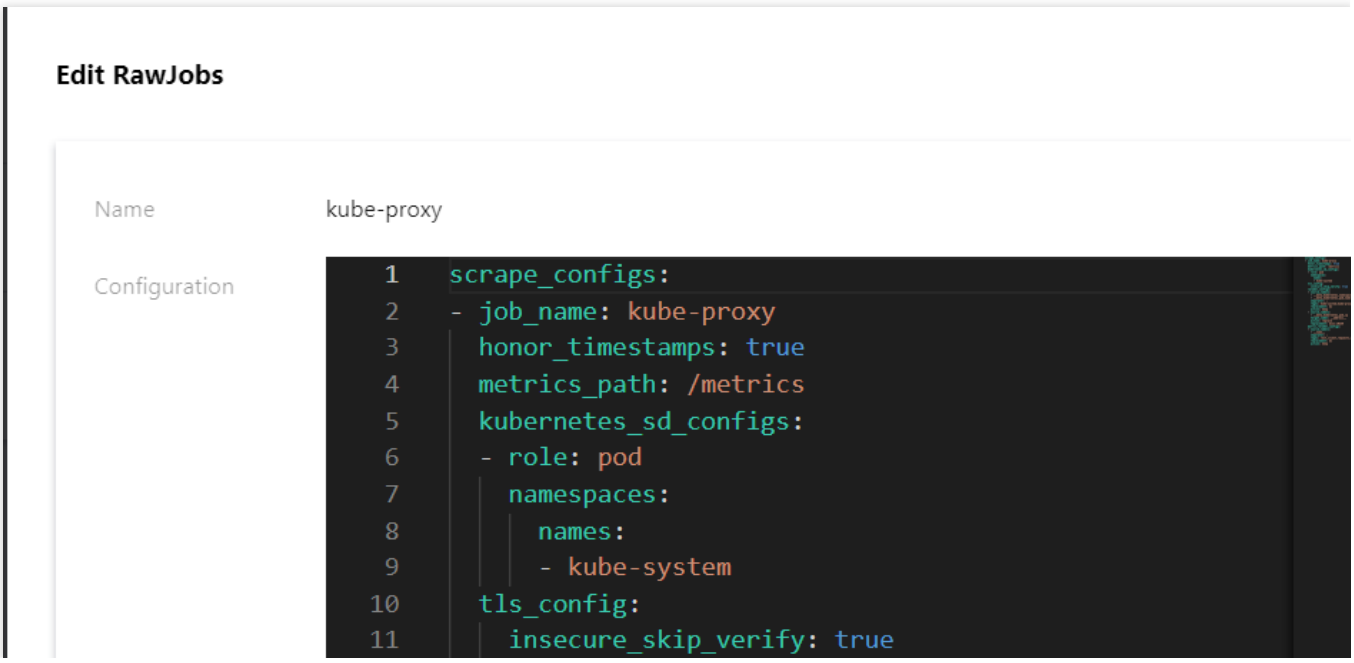
targets(1/1) means (the actual number of targets scraped is 1 / the number of detected collection targets is 1). When the actual number of scraped targets equals the number of detected targets, it is displayed as up, indicating that the current scrape is normal. When the actual number of scraped targets is less than the number of detected targets, it is displayed as down, indicating that some endpoints failed to scrape.

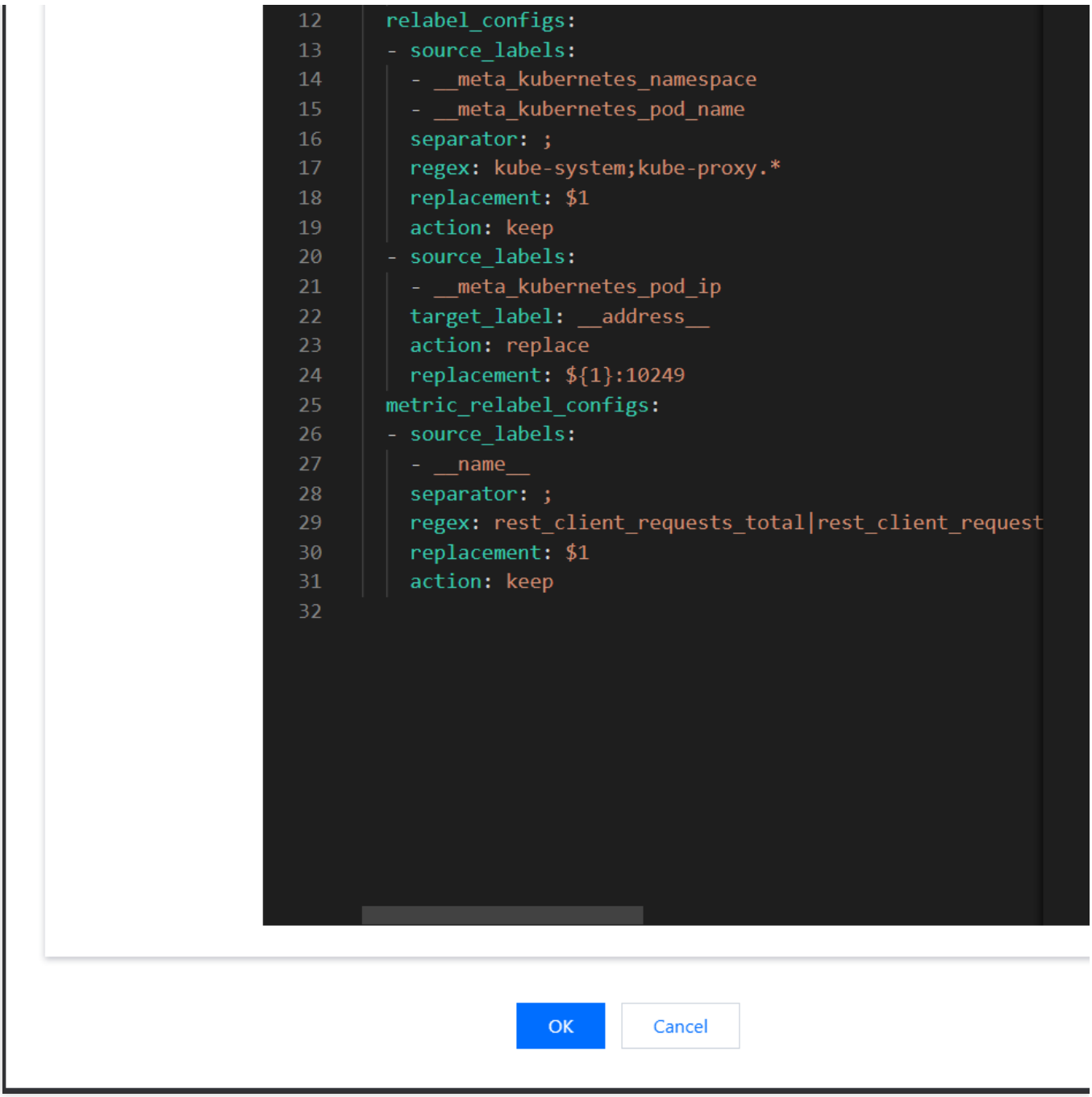
Click the field value **(1/1)** in the above figure to view more details about the collection target. For details, see the following figure.



View Existing Configurations

1. Log in to TMP Console.
2. On the instance list page, select the name of the instance that requires configuring data collection rules and enter its details page.
3. On the **Data Collection > Integration TKE** page, click **Data Collection Configuration** on the right side of the instance to enter the collection configuration list page. Select **Basic Monitoring** or **Custom Monitoring**, then click **Edit** on the right.
4. In the pop-up edit window, view all monitoring objects currently configured in the YAML file. For details, see the following figure.



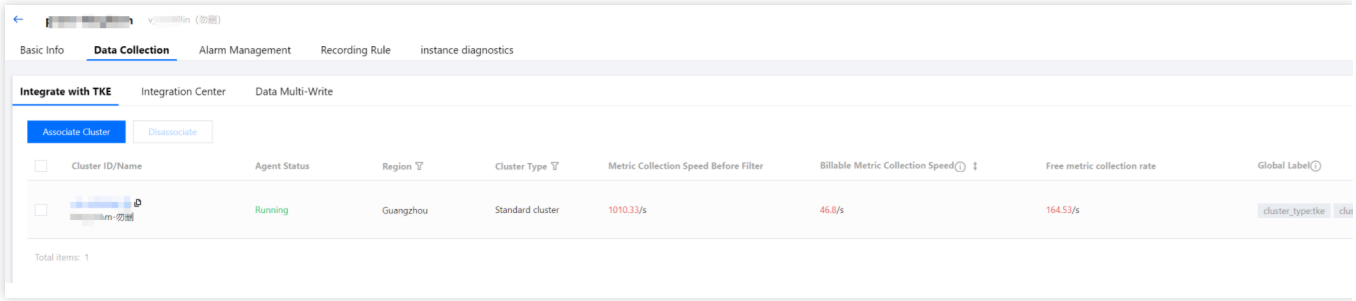


View Collection Targets

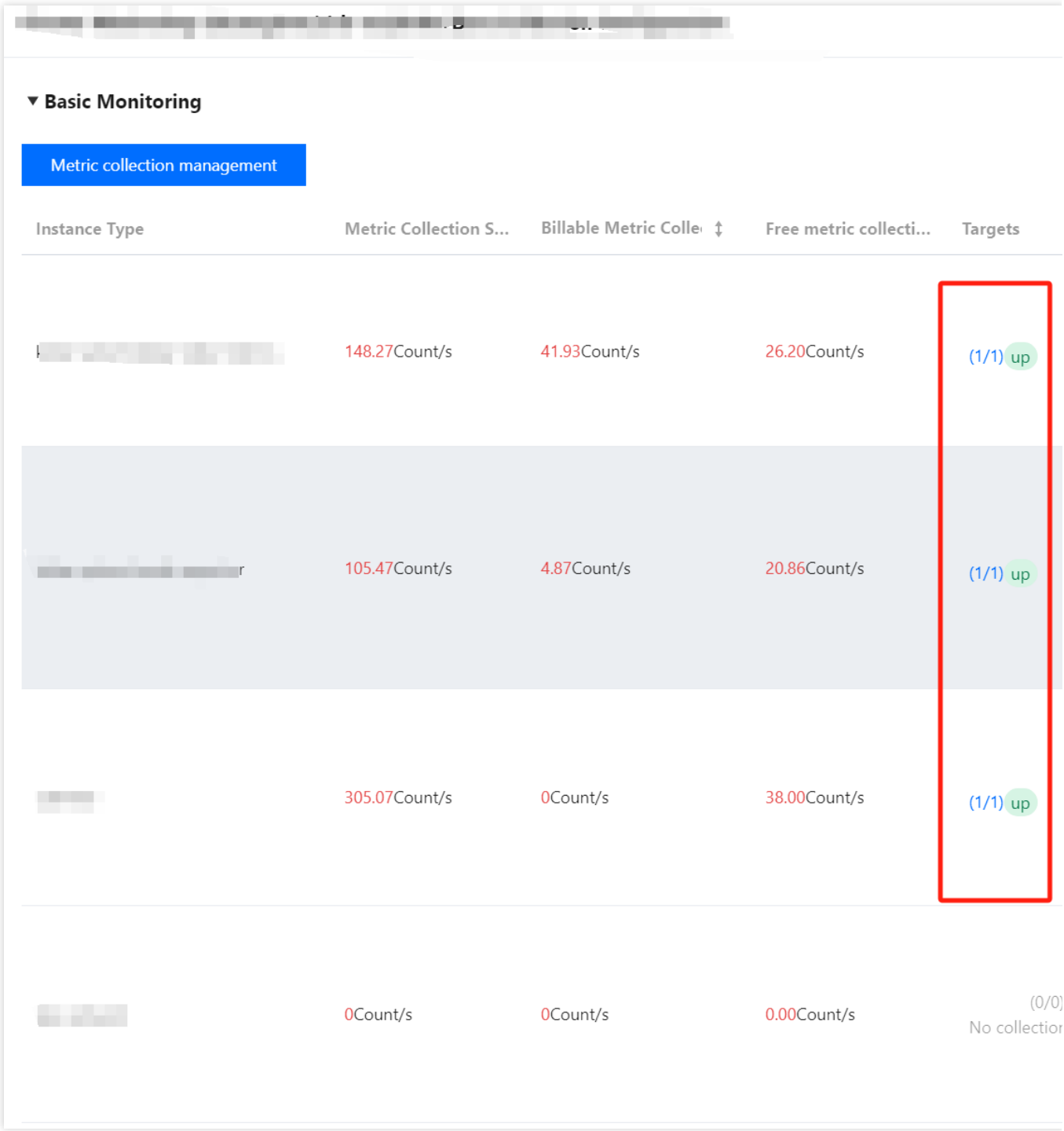
1. Log in to TMP Console.

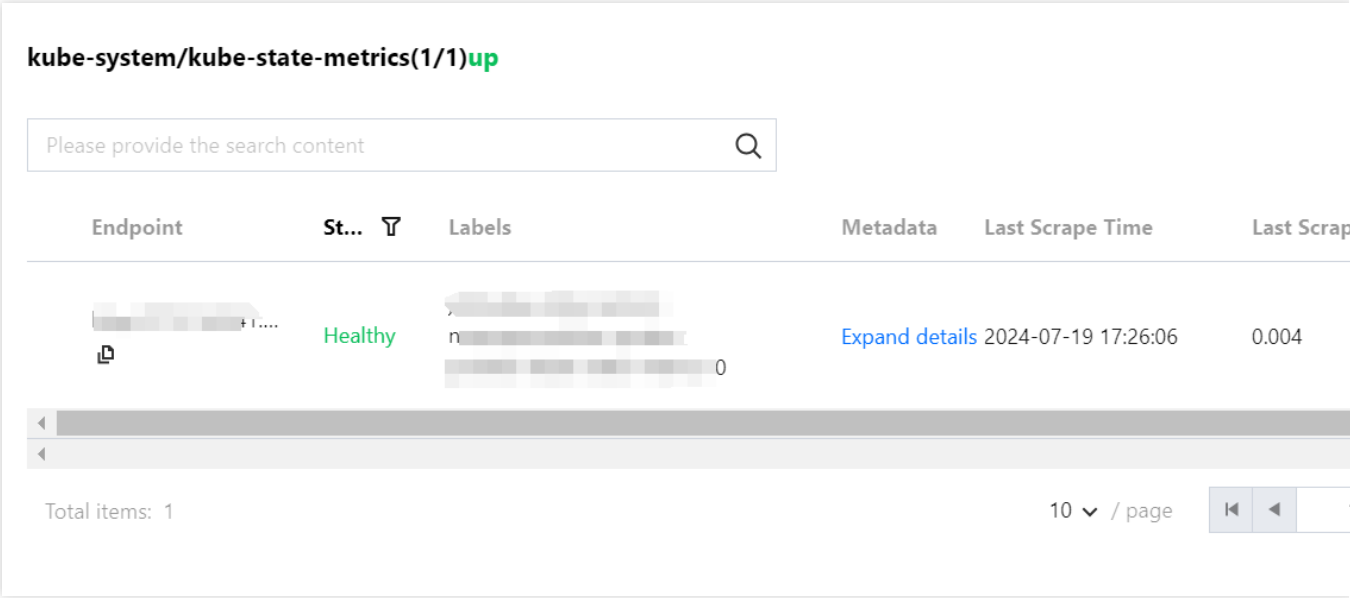
2. On the monitoring instance list page, select the instance name whose targets you want to view and enter the details page of the instance.

3. On the **Data Collection > Integration with TKE** page, click **Data Collection Configuration** on the right side of the instance.



4. In the pop-up window, click the status under targets to jump to the details page of the current data.





Related Operations

Mounting Files to a Collector

When configuring the collection item, if you need to provide some files for the configuration, such as a certificate, you can mount the files to the collector in the following way. The file update will be synchronized to the collector in real time.

prometheus.tke.tencent.cloud.com/scrape-mount = "true"

Add the above label to the configmap under the prom-xxx namespace. All keys will be mounted to the collector's path `/etc/prometheus/configmaps/[configmap-name]/` .

prometheus.tke.tencent.cloud.com/scrape-mount = "true"

Add the above label to the secret under the prom-xxx namespace. All keys will be mounted to the collector's path `/etc/prometheus/secrets/[secret-name]/` .

Configuring Necessary Monitoring Metrics

Last updated：2024-08-08 10:42:17

This document describes how to configure necessary **collection metrics** of TMP to avoid unnecessary expenses.

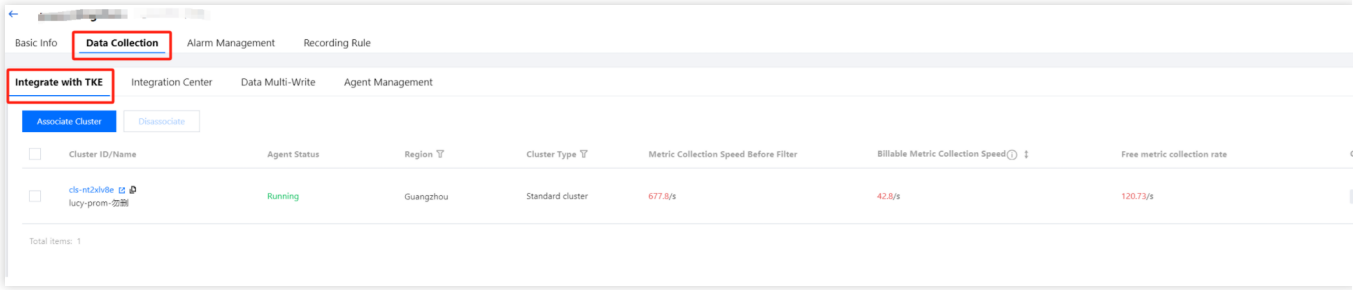
Prerequisites

Before configuring monitoring collection items, you need to perform the following operations:
Create a Prometheus monitoring instance.
Associate the cluster to be monitored with the instance.

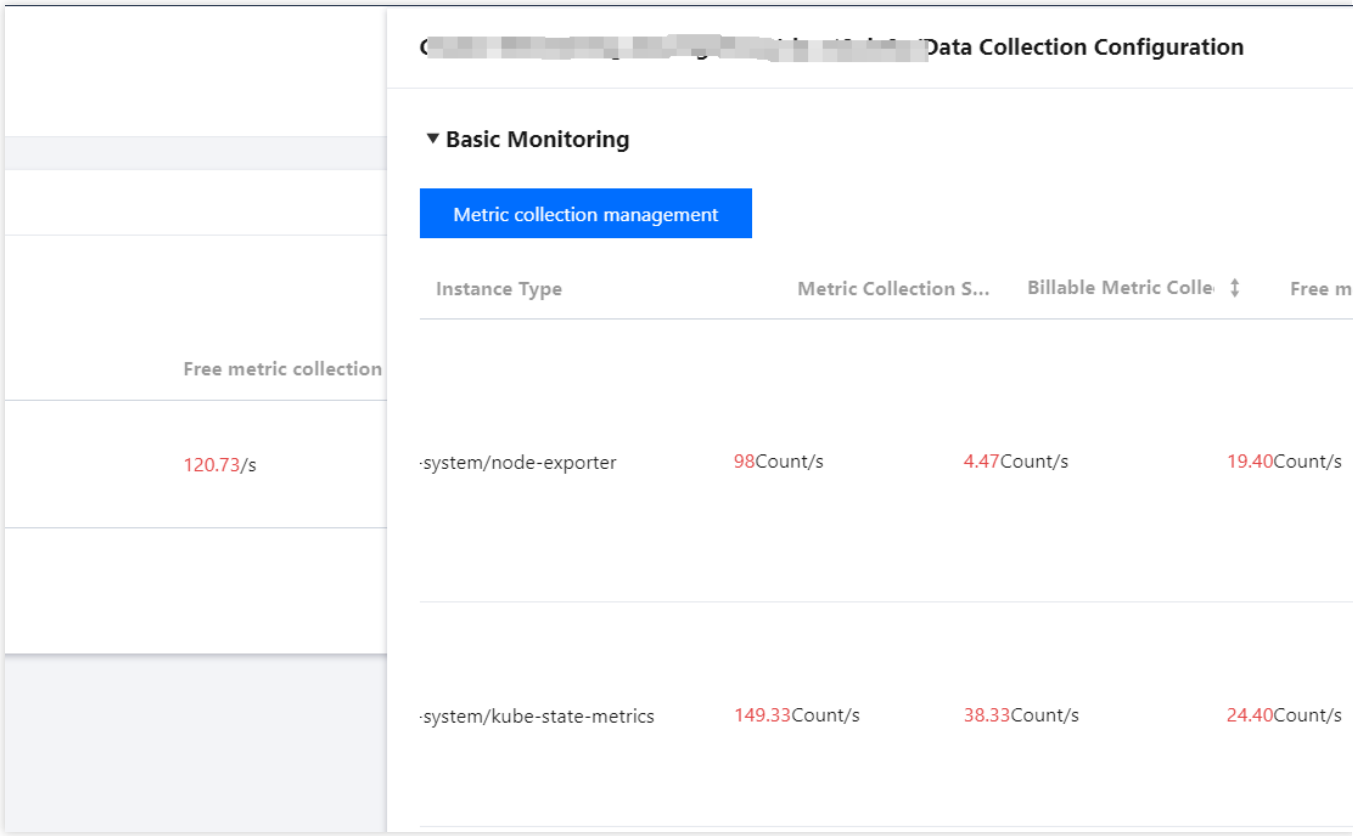
Configuring Necessary Metrics

Configuring Necessary Metrics via Console

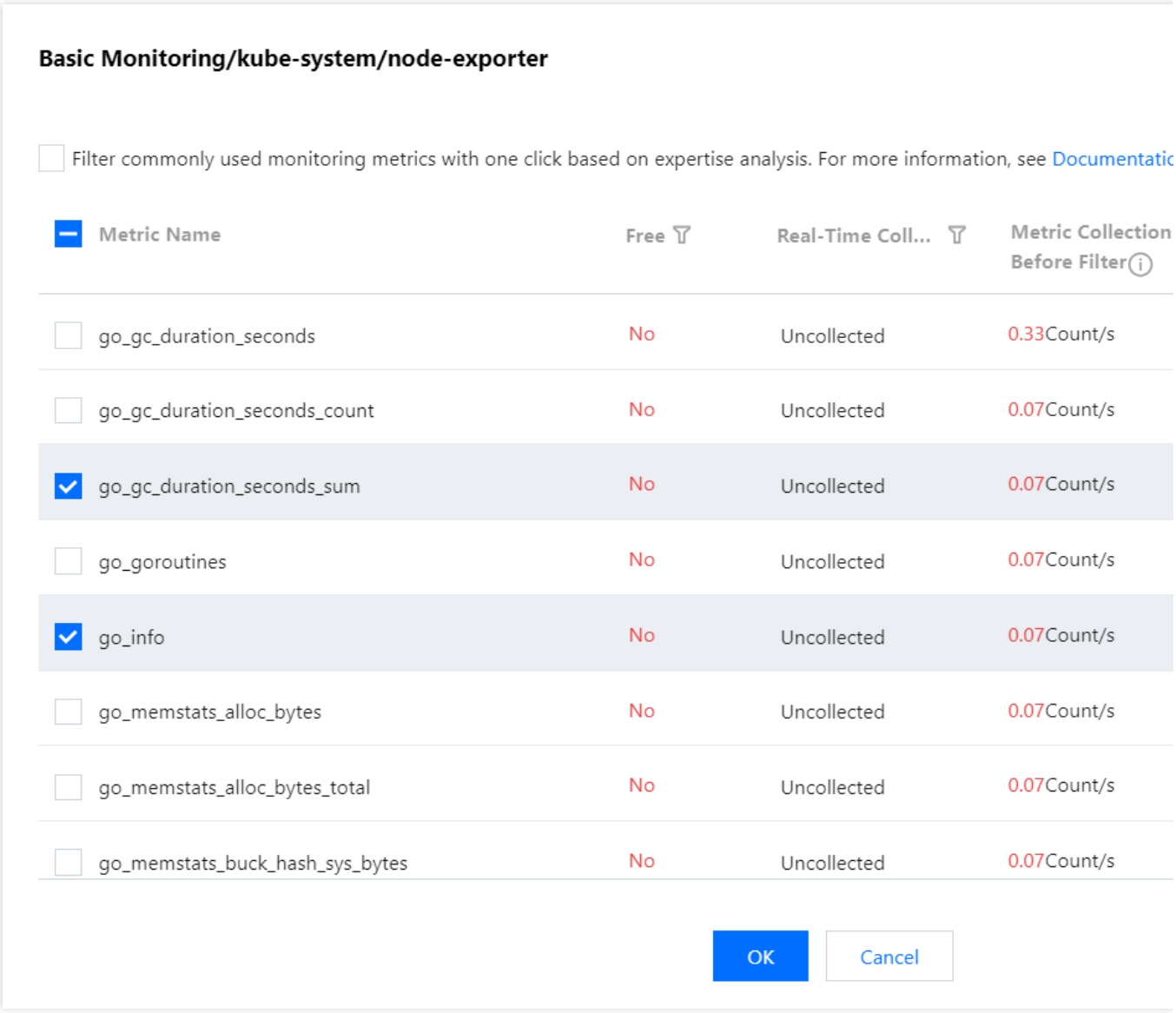
TMP provides over 100 free basic monitoring metrics. For the complete metric list, see Free Metrics in Pay-as-You-Go Mode.
1. Log in to the TCOP console, and click Prometheus Monitoring in the left sidebar.
2. On the instance list page, select the name of the instance that requires configuring data collection rules and enter its details page.
3. On the **Data Collection** page, click **Integration with TKE**, find the target cluster, and click **Data Collection Configuration** in the operation bar, as shown below:



4. In the data collection configuration window, click **Metric Details** in the operation bar to deselect unnecessary metrics, as shown below:



5. On the page below, you can view free and billed metrics. If a metric is selected, corresponding data will be collected. You can select metrics as needed. Only basic monitoring provides free metrics. For all free metrics, see Free Metrics in Pay-as-You-Go Mode. For billing of paid metrics, see TMP Pay-as-You-Go.



Configuring Necessary Metrics via YAML File

The billing of TMP currently is based on the number of monitored data points. To minimize unnecessary expenses, it is recommended that you optimize the collection configuration to collect data of only necessary metrics, thereby reducing the overall data amount. For billing details and the usage of related cloud resources, see Resource Usage and Billing Overview.

The following part describes how to add filters to ServiceMonitor, PodMonitor, and native Job to configure necessary metrics.

1. Log in to the TCOP console, and click Prometheus Monitoring in the left sidebar.
2. On the instance list page, select the name of the instance that requires configuring data collection rules and enter its details page.
3. On the **Data Collection** page, click **Integration with TKE**, find the target cluster, and click **Data Collection Configuration** in the operation bar.

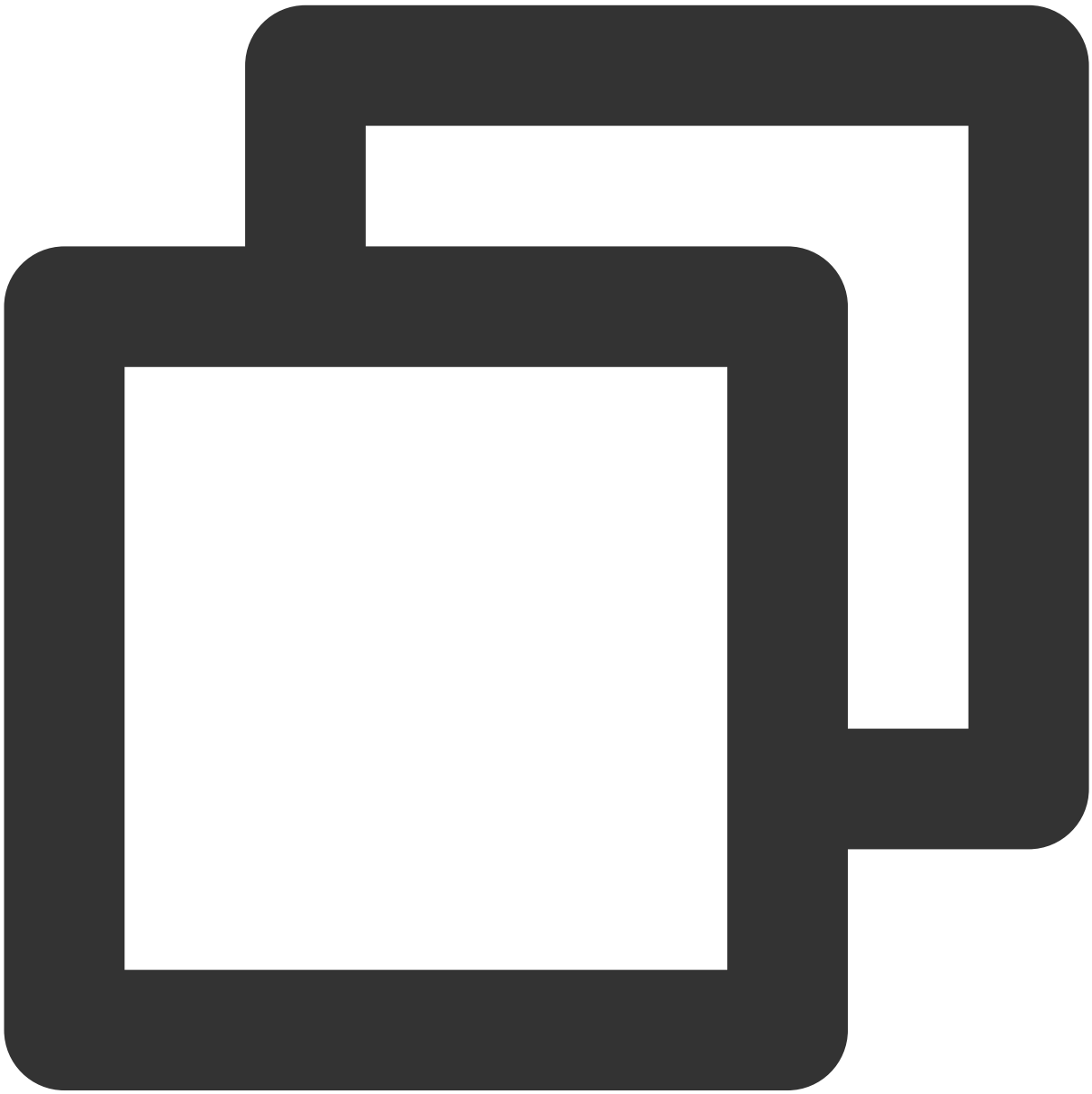
4. Find the target instance, and click **Editing** in the operation bar to view metric details.

ServiceMonitor and PodMonitor

Native Job

The filter configurations of ServiceMonitor and PodMonitor are the same. Take ServiceMonitor as an example.

ServiceMonitor example:

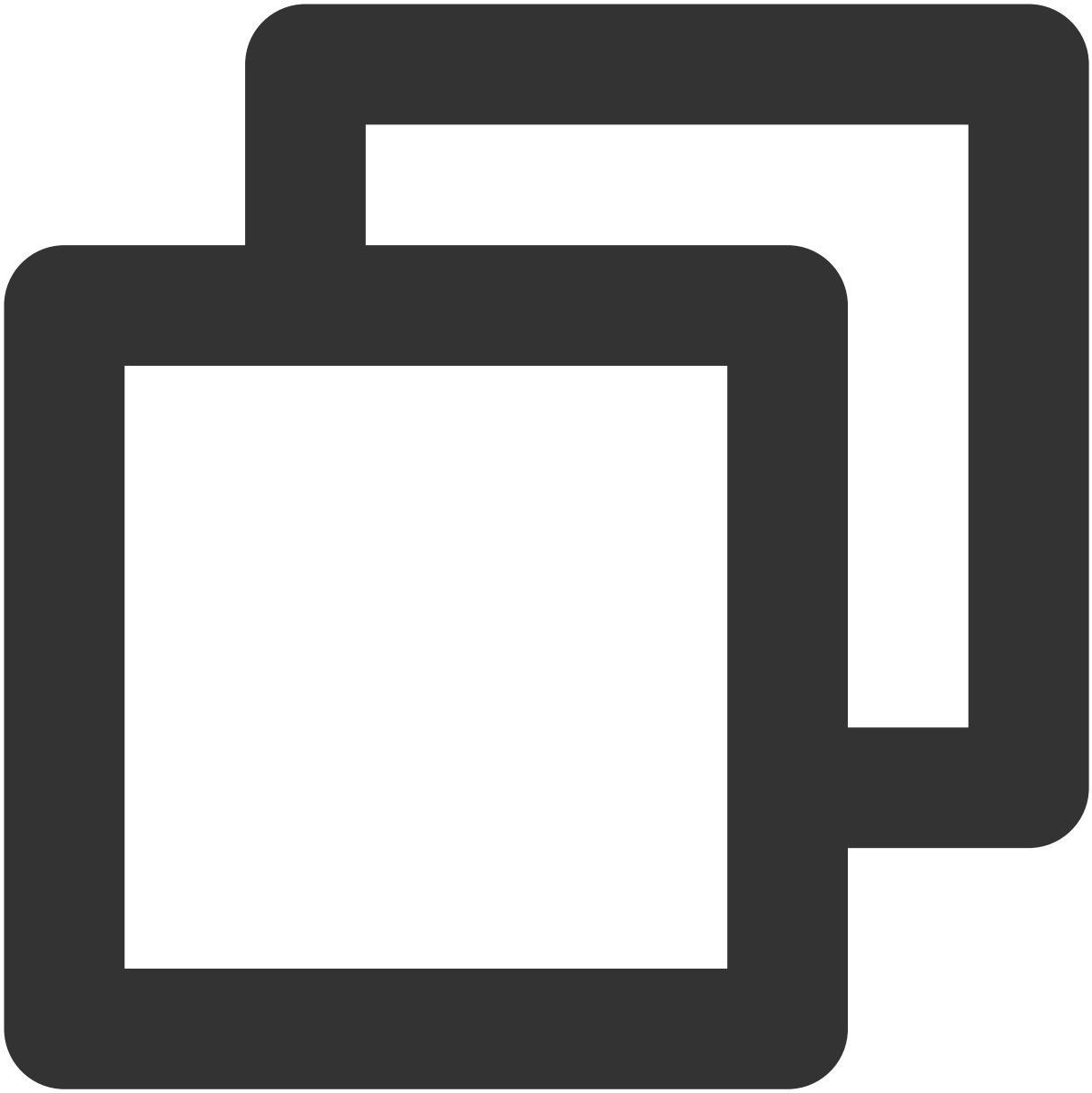


```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
```

```
    app.kubernetes.io/name: kube-state-metrics
    app.kubernetes.io/version: 1.9.7
  name: kube-state-metrics
  namespace: kube-system
spec:
  endpoints:
  - bearerTokenSecret:
      key: ""
    interval: 15s # This parameter indicates the collection interval. You can incre
    port: http-metrics
    scrapeTimeout: 15s # This parameter indicates the collection timeout. TMP requi
  jobLabel: app.kubernetes.io/name
  namespaceSelector: {}
  selector:
    matchLabels:
      app.kubernetes.io/name: kube-state-metrics
```

For example, if you want to collect data of metrics `kube_node_info` and `kube_node_role`, add the configuration with the `metricRelabelings` field to the endpoints list of ServiceMonitor. Note that the field is **`metricRelabelings`**, not `relabelings`.

Example of adding `metricRelabelings`:

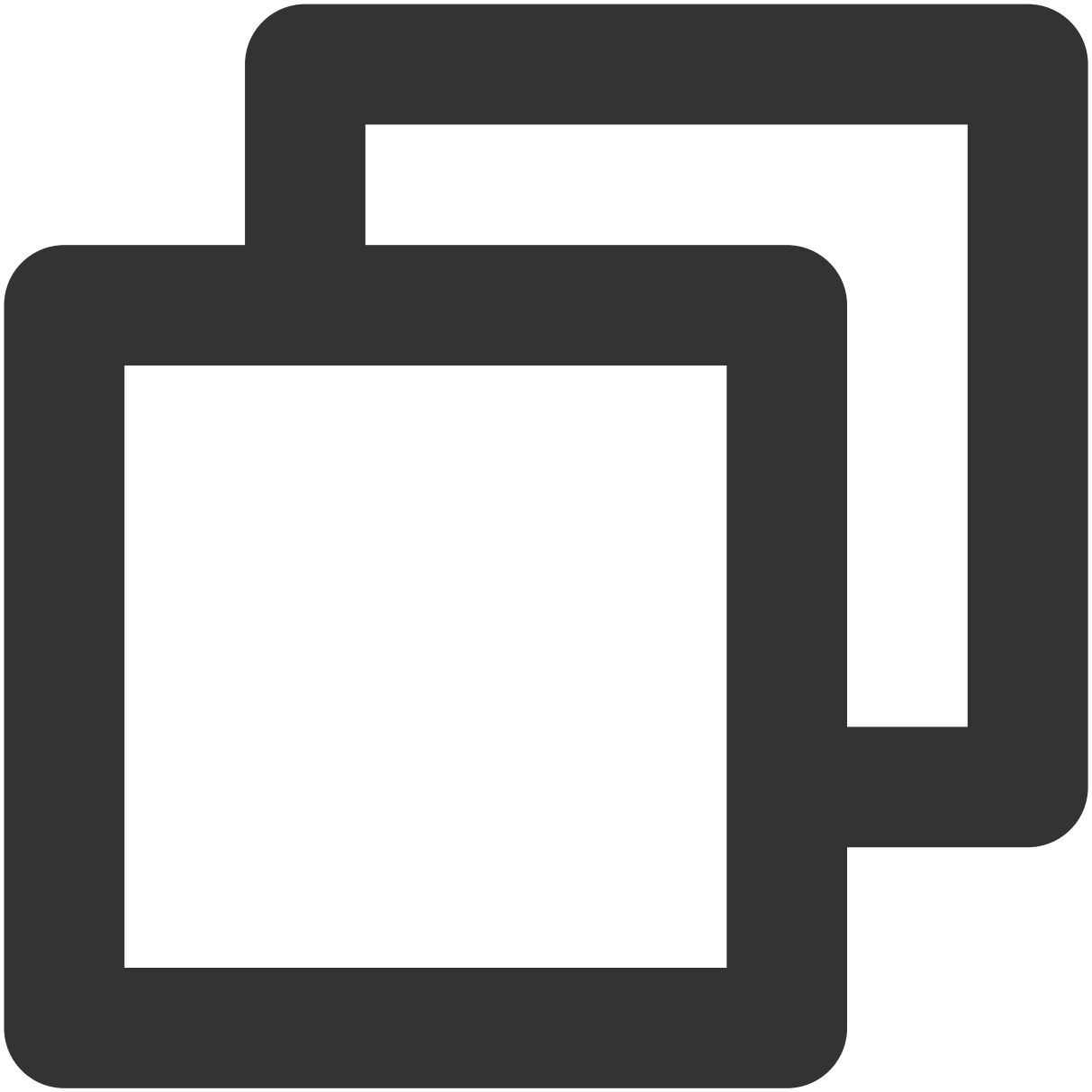


```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    app.kubernetes.io/name: kube-state-metrics
    app.kubernetes.io/version: 1.9.7
  name: kube-state-metrics
  namespace: kube-system
spec:
  endpoints:
  - bearerTokenSecret:
```

```
      key: ""
    interval: 15s # This parameter indicates the collection interval. You can incre
    port: http-metrics
    scrapeTimeout: 15s
    # The following four lines are added:
    metricRelabelings: # Each metric will undergo the following process.
    - sourceLabels: ["__name__"] # The label name to be checked. __name__ refers to
      regex: kube_node_info|kube_node_role # Check whether the label matches this r
      action:  keep # If the metric name meets the above conditions, the metric is
  jobLabel: app.kubernetes.io/name
  namespaceSelector: {}
  selector:
```

If native Jobs of Prometheus are used, filter metrics using the method below.

Job example:

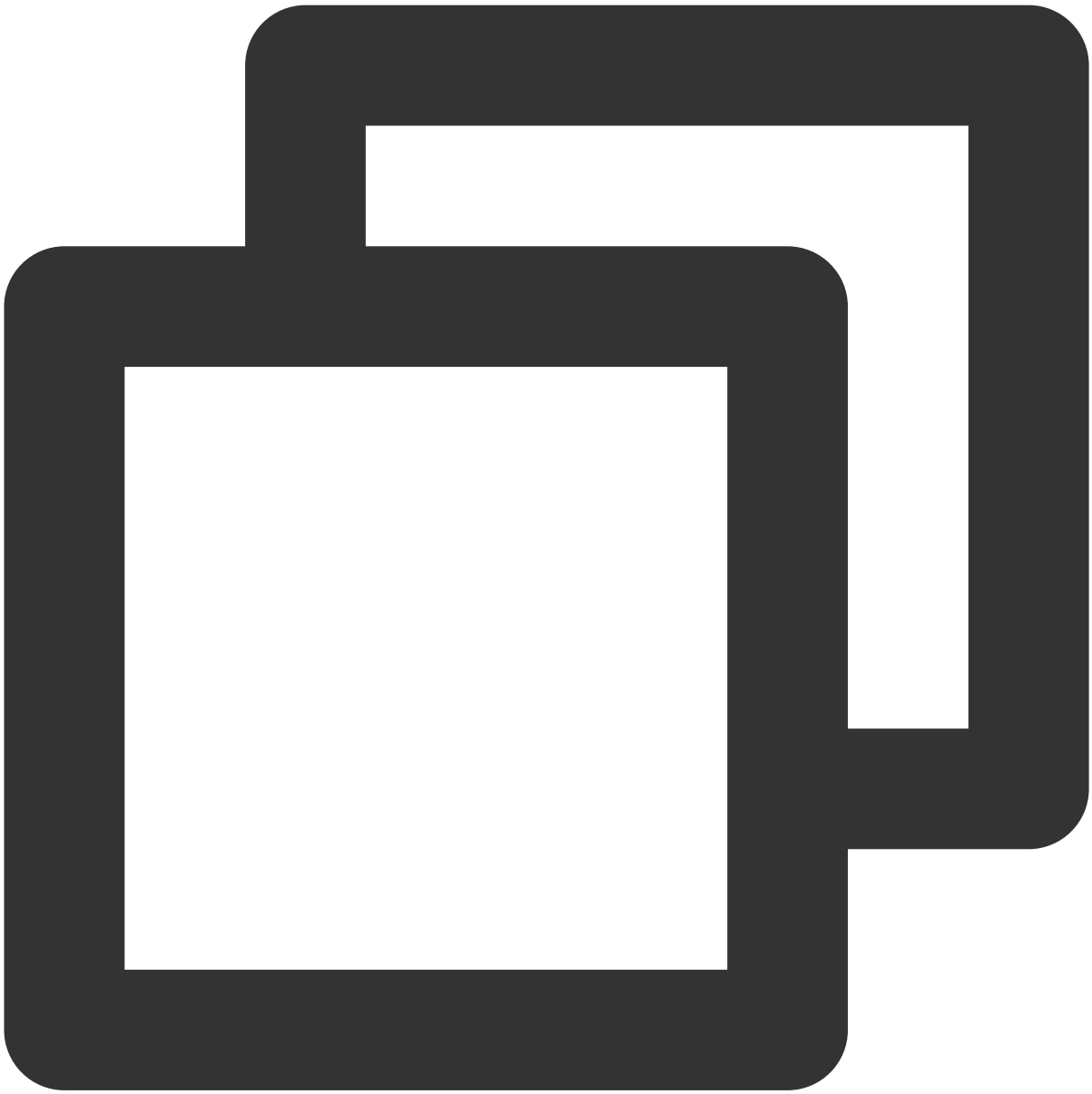


```
scrape_configs:
  - job_name: job1
    scrape_interval: 15s # This parameter indicates the collection interval. You ca
    static_configs:
      - targets:
          - '1.1.1.1'
```

For example, if you want to collect data of metrics `kube_node_info` and `kube_node_role` only, add the configuration with the field `metric_relabel_configs`. Note: The field is **`metric_relabel_configs`**, not

`relabel_configs`.

Example of adding `metric_relabel_configs`:



```
scrape_configs:
  - job_name: job1
    scrape_interval: 15s # This parameter indicates the collection interval. You ca
    static_configs:
    - targets:
      - '1.1.1.1'
    # The following four lines are added:
    metric_relabel_configs: # Each metric will undergo the following process.
```

```
    - source_labels: ["__name__"] # The label name to be checked. __name__ refers t
      regex: kube_node_info|kube_node_role # Check whether the label matches this r
      action:  keep # If the metric name meets the above conditions, the metric is
```

5. Click **Confirm**.

Skipping Data Collection

Skipping Data Collection from Entire Namespace

TMP will manage all ServiceMonitors and PodMonitors in a cluster by default after the cluster is associated. If you want to skip data collection from a namespace, you can add the label `tps-skip-monitor: "true"` to the specified namespace. For label details, see Labels and Selectors.

Skipping Data Collection

TMP collects monitoring data by creating ServiceMonitor and PodMonitor types of CRD resources in a cluster. If you want to skip data collection from specified ServiceMonitor and PodMonitor resources, you can add the label `tps-skip-monitor: "true"` to these CRD resources. For label details, see Labels and Selectors.

Adjusting Collection IntervalAdjusting Collection Interval

Last updated：2024-08-08 10:42:17

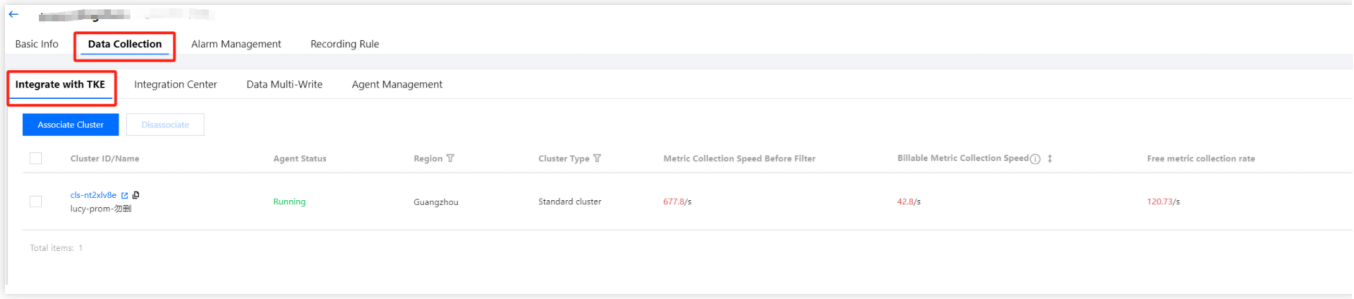
This document describes how to adjust the **sampling interval** of TMP to avoid unnecessary expenses.

Prerequisites

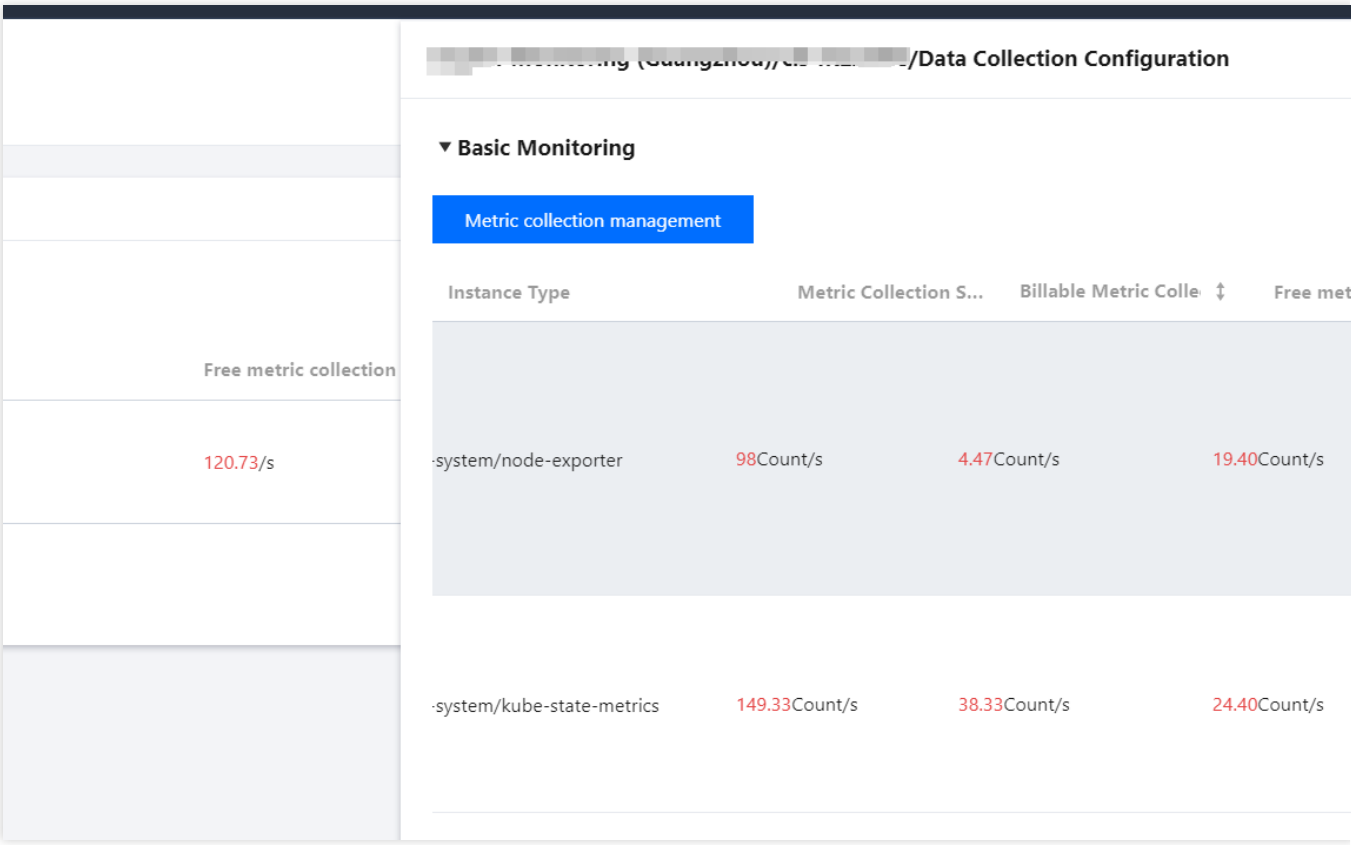
Before configuring monitoring collection items, you need to perform the following operations:
Create a Prometheus monitoring instance.
Associate the cluster to be monitored with the instance.

Adjusting Collection Interval via TCOP Console

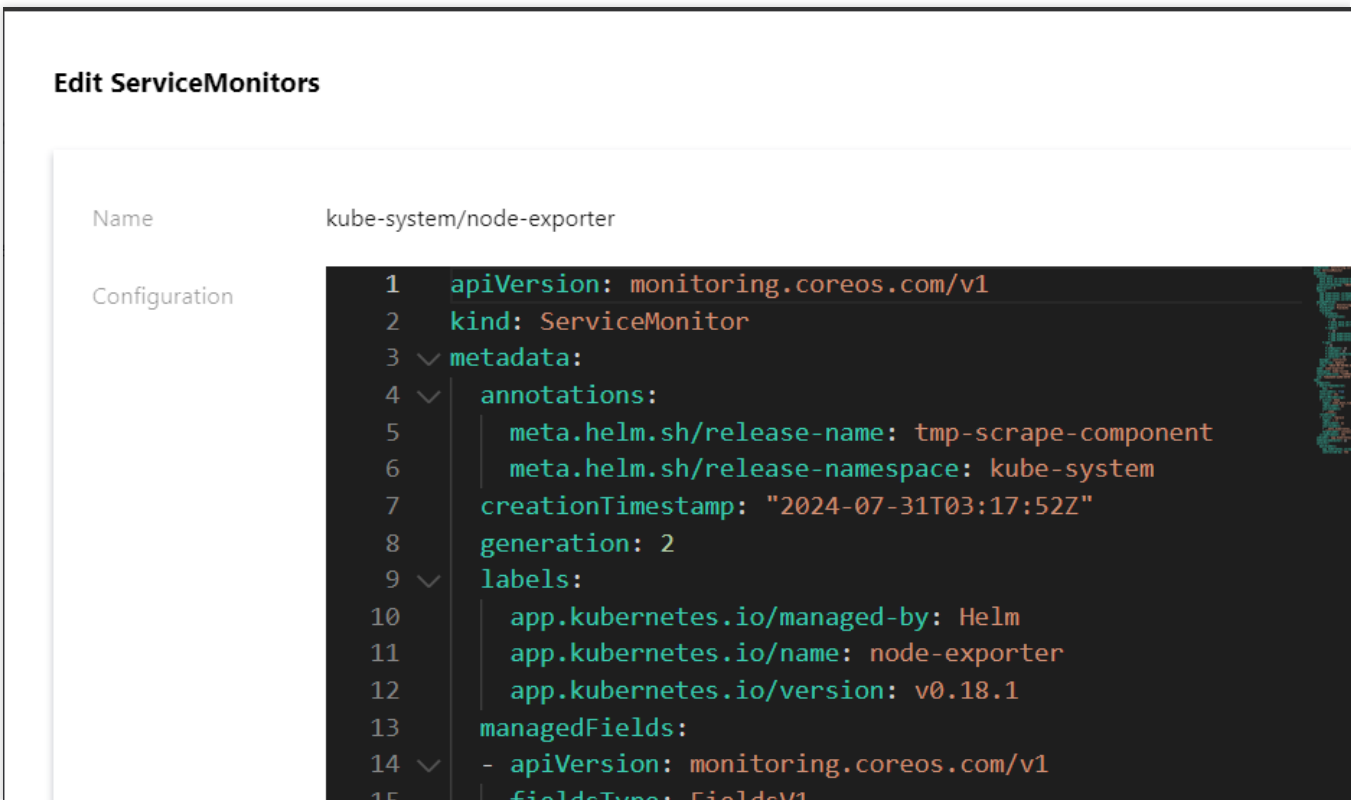
1. Log in to the TCOP console, and click Prometheus Monitoring in the left sidebar.
2. On the instance list page, select the name of the instance that requires configuring data collection rules and enter its details page.
3. On the **Data Collection** page, select **Integrate with TKE** , click **Data Collection Configuration** on the right side of the cluster, as shown in the following figure:



4. In the data collection configuration window, click **Edit** in the operation bar, as shown below:



5. Find the **interval** field, which represents the collection interval. You can increase its value to reduce data storage costs, as shown below:



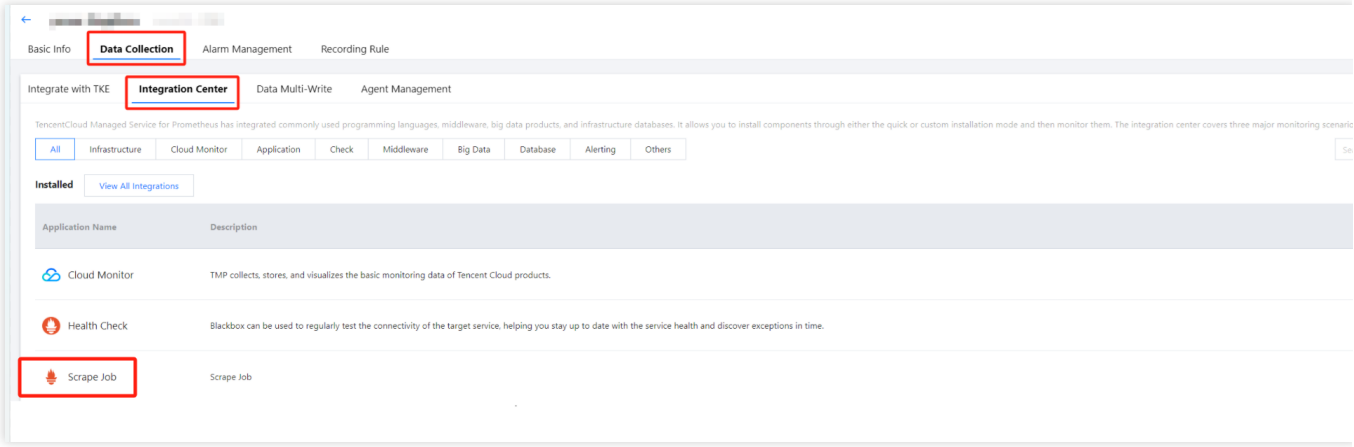
6. Click **Confirm** after completing the configuration.

Adjusting Collection Interval via Integration Center

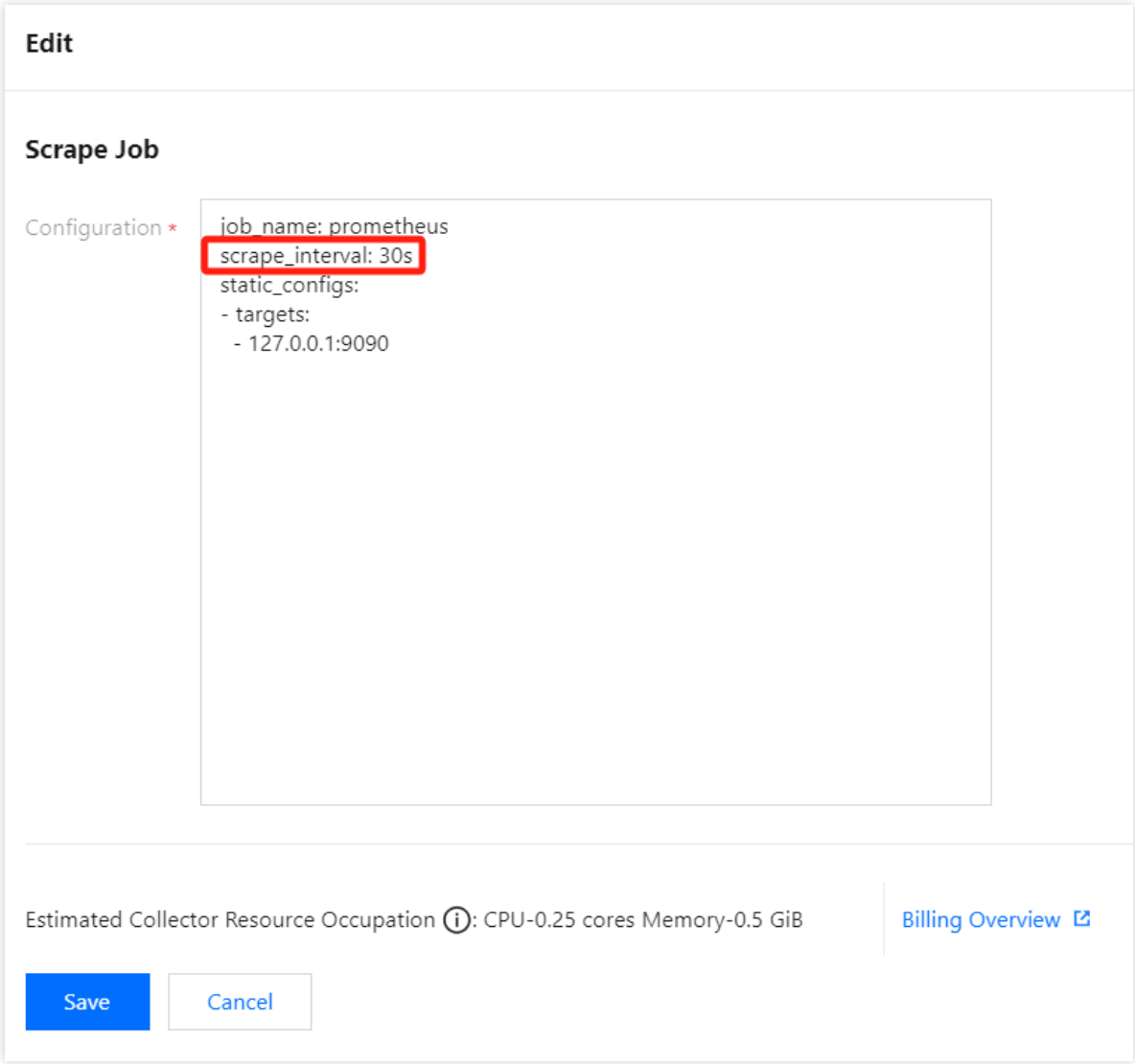
Note:

The Scraping Task application should have been installed and integrated. If it is not installed, click **One-click installation** on the right before you proceed with the following steps.

1. Log in to the TCOP console, and click Prometheus Monitoring in the left sidebar.
2. On the instance list page, select the name of the instance that requires configuring data collection rules and enter its details page.
3. On the **Data Collection** page, click **Integration Center**, find **Scrape Job**, and click **View Integrations** on the right, as shown below:



4. On the page that appears, click any name to enter the collection tasks configuration page. Then, find the **scrape_interval** field, which represents the collection interval. You can increase its value to reduce data storage costs, as shown below:



5. Click **Save** after completing the configuration.

Resource Usage and Billing Overview

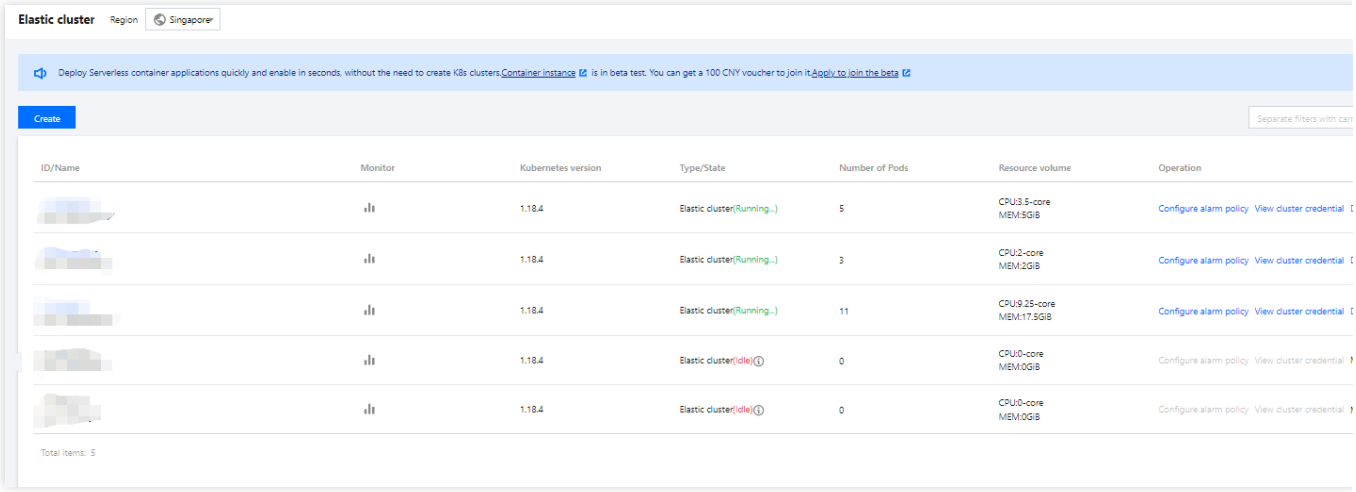
Last updated：2024-01-29 16:01:55

When using the Tencent Managed Service for Prometheus (TMP) service, you may use resources such as **Tencent Kubernetes Engine (TKE) serverless clusters, TencentCloud Managed Service for Grafana (TCMG), and Cloud Load Balancer (CLB)**. This document describes the use cases and billing rules of these resources.

TKE Serverless Cluster

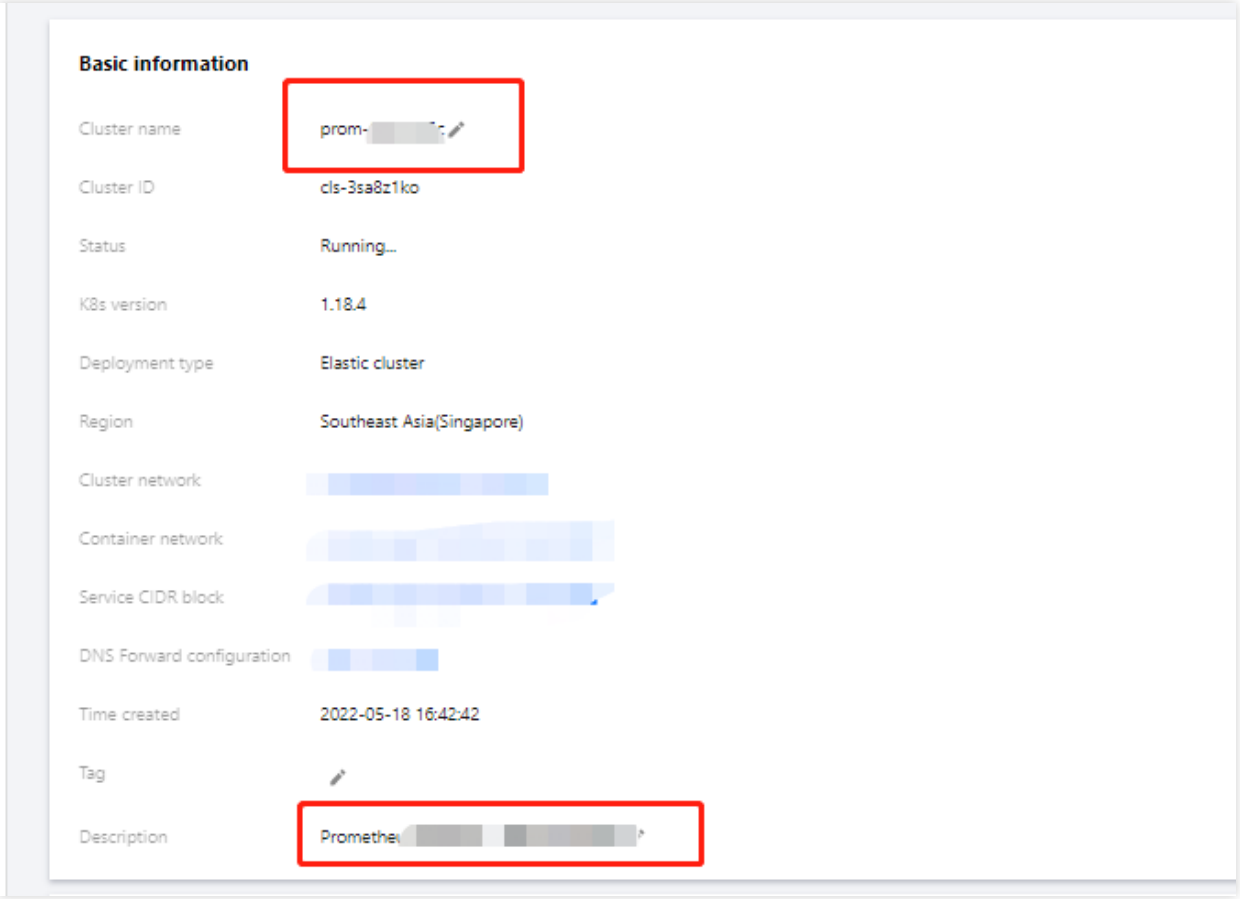
Use cases

You need to create a TKE serverless cluster if you use a TMP-associated cluster to monitor TKE.
A TKE serverless cluster will be automatically created for data collection when you install an integration plugin in the TMP integration center.
If both use cases are required for you, only one TKE serverless cluster will be created and shared. You can view the created clusters on the cluster list page.



Note

The name of the TKE serverless cluster is the TMP **instance ID**, and the cluster description states that **For TMP use only. Do not modify or delete**.



Billing overview

The billing mode is **pay-as-you-go**. For more information, see Product Pricing.

The TKE serverless cluster automatically scales according to the monitoring size. The relationship between the monitoring size and the TKE serverless cluster cost is shown below:

Reported Instantaneous Series	Estimated TKE Serverless Cluster Resources Required	List Price/Day
< 500,000	1.25 cores, 1.6 GiB	0.35 USD
1 million	0.5 cores, 1.5 GiB*2	1.46 USD
5 million	1 core, 3 GiB*3	2.93 USD
20 million	1 core, 6 GiB*5	7.98 USD
30 million	1 core, 6 GiB*8	12.77 USD

Sample TKE serverless cluster costs are as follows:

If the TKE serverless cluster used for a newly initialized TMP instance consumes 1.25 CPU cores and 1.5 GiB memory, then the estimated list price per day will be 0.0319 *24 + 0.0132* 24 = 1.0824 USD.

TCMG

Use cases

When creating a TMP instance, you need to associate it with a TCMG instance in the same region for the visual display of monitoring data collected by TMP. For billing information, see Billing Overview.

CLB

When you use a TMP-associated cluster to monitor TKE, a private network CLB instance will be created under your account for network connectivity between the collector and the cluster.
If you associate an edge cluster or a cluster with no network connection, a public network CLB instance will be created for network connectivity.
To access the TCMG service over the public network, you need to create a public network CLB instance.
These CLB resources will be charged. You can view the resource information of the created public network CLB instances in the CLB console.
Resources are billed based on the actual usage. For billing details, see Network Pricing.

Resource Termination

If you terminate TMP instances in the TMP console, all relevant resources will also be terminated. Tencent Cloud does not repossess TMP instances proactively. If you no longer use TMP, you need to delete the instances promptly to avoid extra charges. For instance termination directions, see Terminating Instance.