TencentCloud Managed Service for Prometheus

Practical Tutorial

Product Documentation

Contents

Practical Tutorial
Migration from Self-Built Prometheus

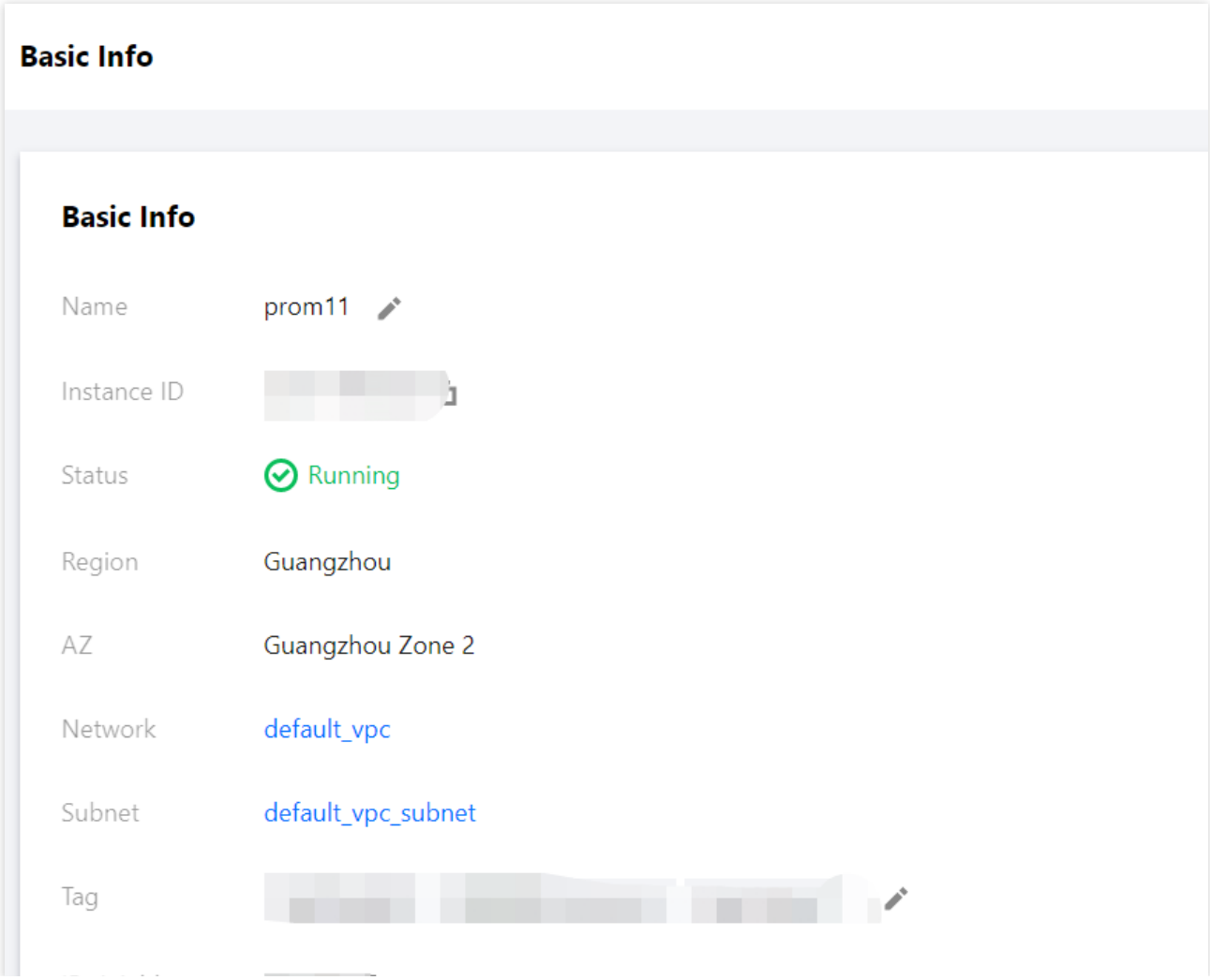
Last updated：2024-01-29 15:55:07

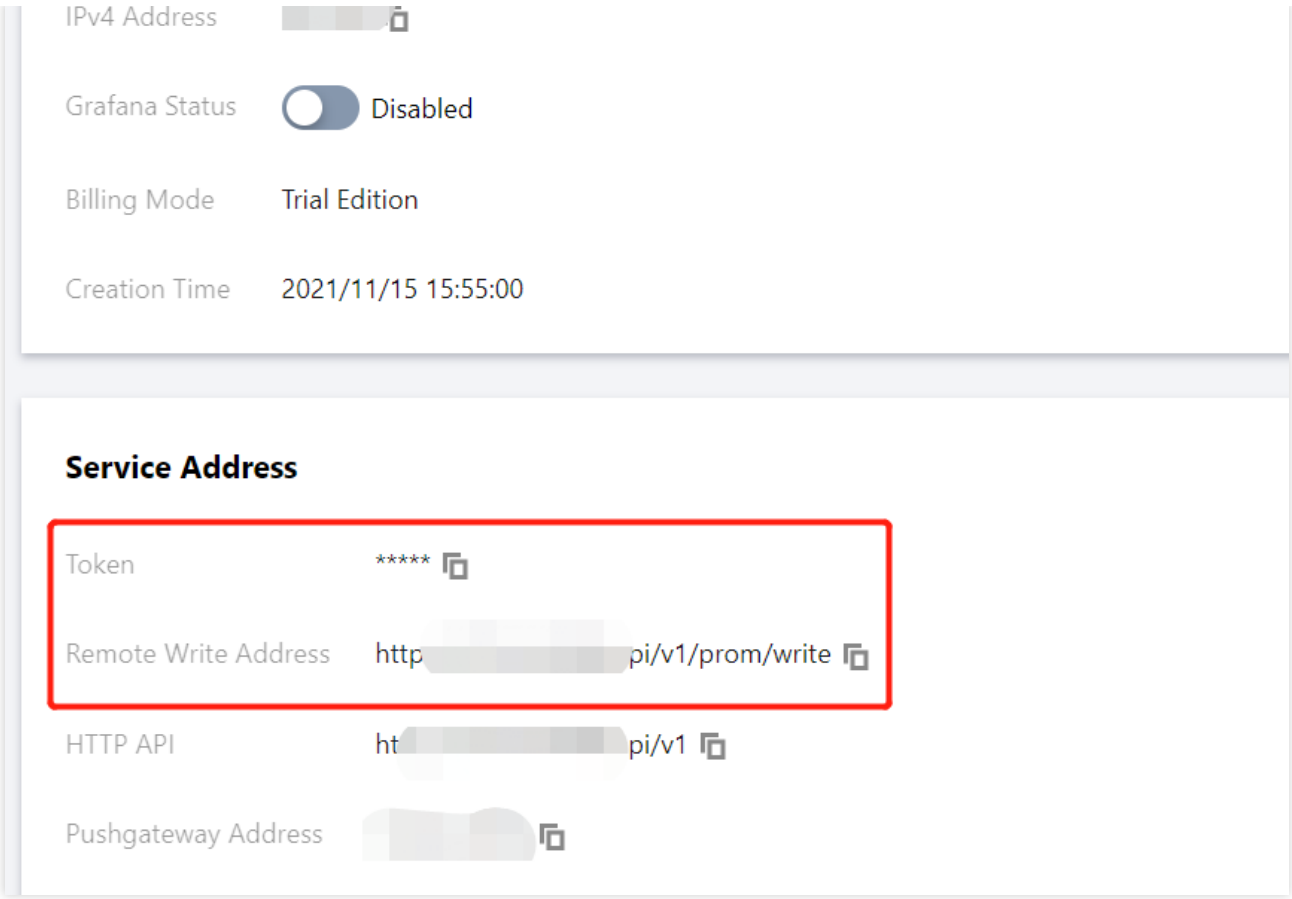
Overview

You can quickly migrate from your self-built Prometheus service to TMP.

Directions

Prometheus itself supports remote write to an external storage; therefore, you can add a remote write configuration pointing to TMP in the configuration file of your self-built Prometheus. The steps are as follows:
1. Get the remote write address and token of TMP through the basic information of the instance as follows:

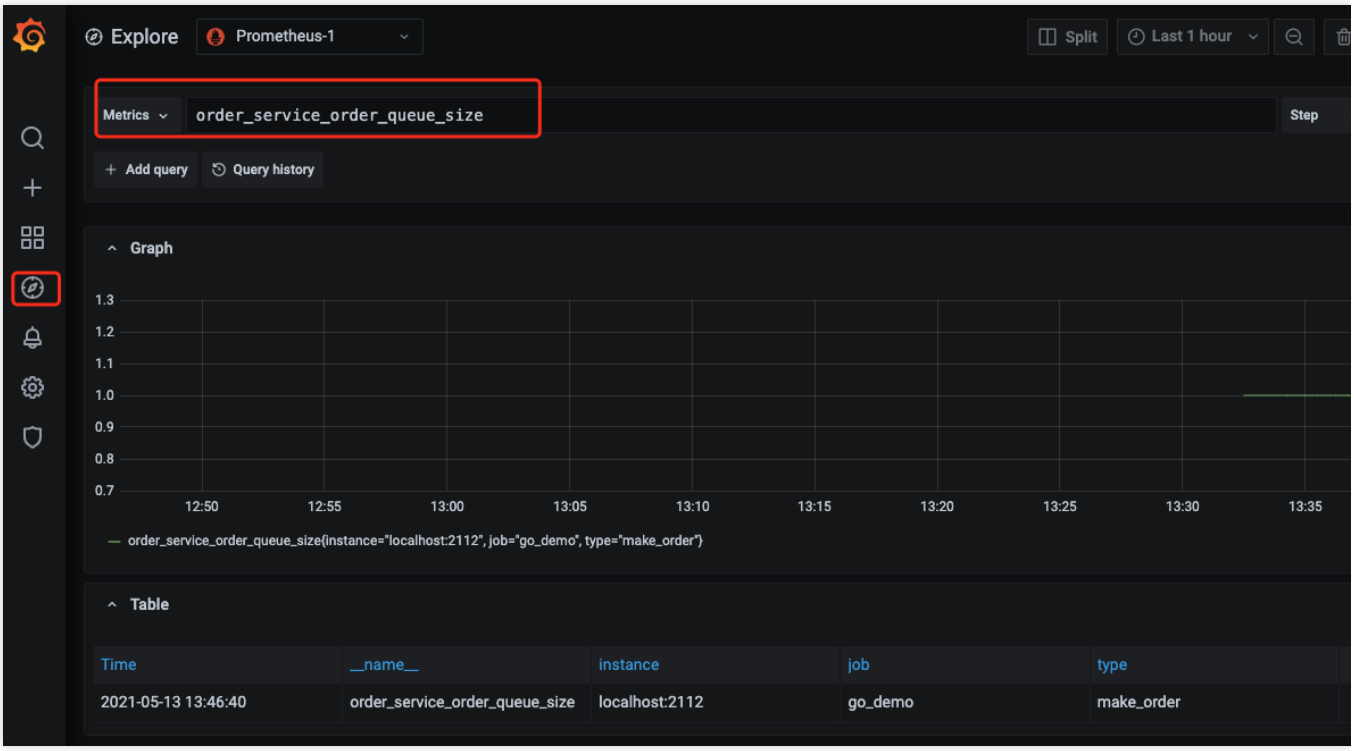




2. Modify `prometheus.yml` and restart Prometheus. The specific configuration is as follows. For more information on the remote write configuration parameters, please see remote_write.

```
remote_write:

  - name: cm_prometheus # Remote write name
    url: http://ip:port/api/v1/prom/write  # Get the remote write address from the
    remote_timeout: 30s # Set according to the actual situation
    bearer_token: k32*****trR # Get the token information from the basic informatio
```

3. Open the Grafana service that comes with TMP and use `Explore` to verify whether the data is written successfully as shown below. You can also customize Grafana monitoring dashboards.



4. You can also use Prometheus APIs for self-built visualization. For more information, please see Monitoring Data Query.

Custom Integration with CVM

Last updated：2024-01-29 15:55:07

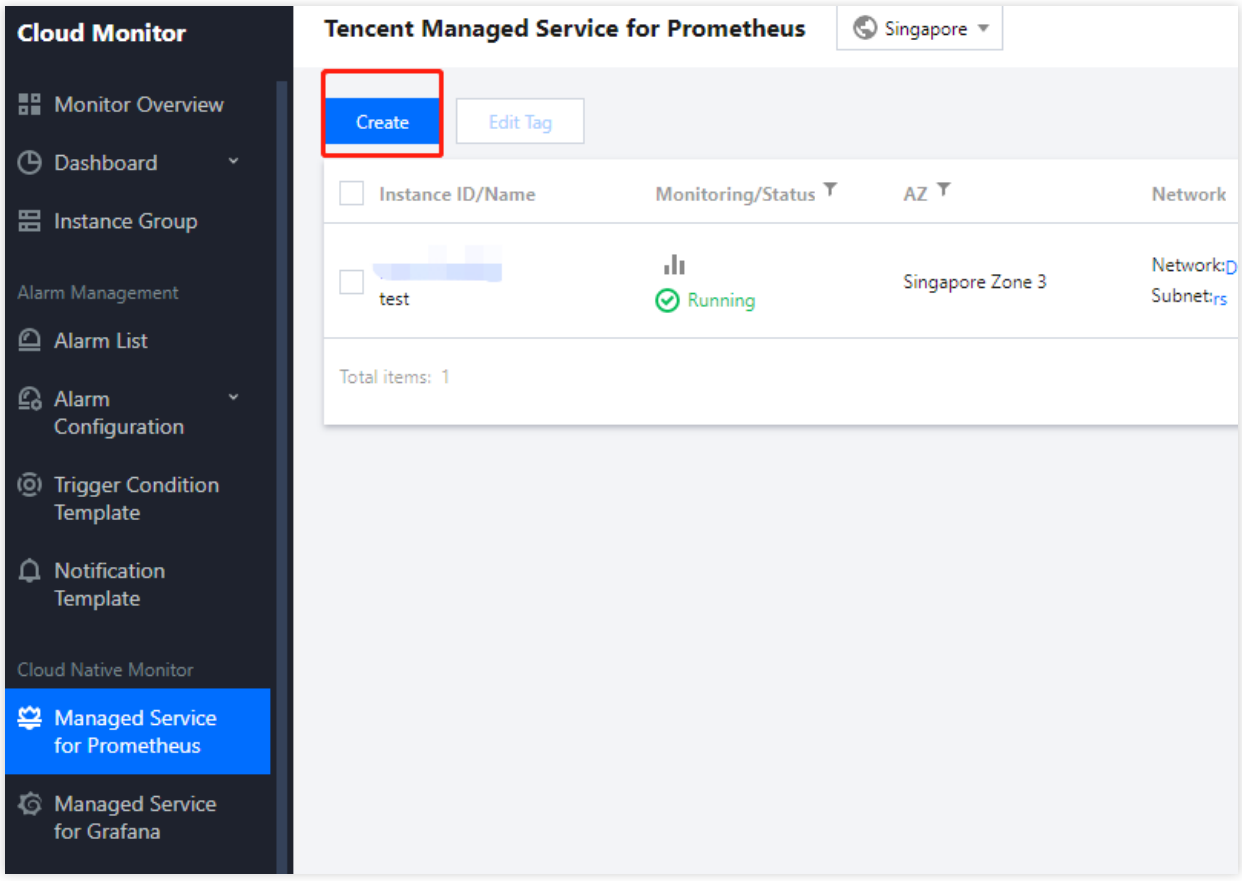
This document describes how to integrate CVM with TMP.

Purchasing a TMP Instance

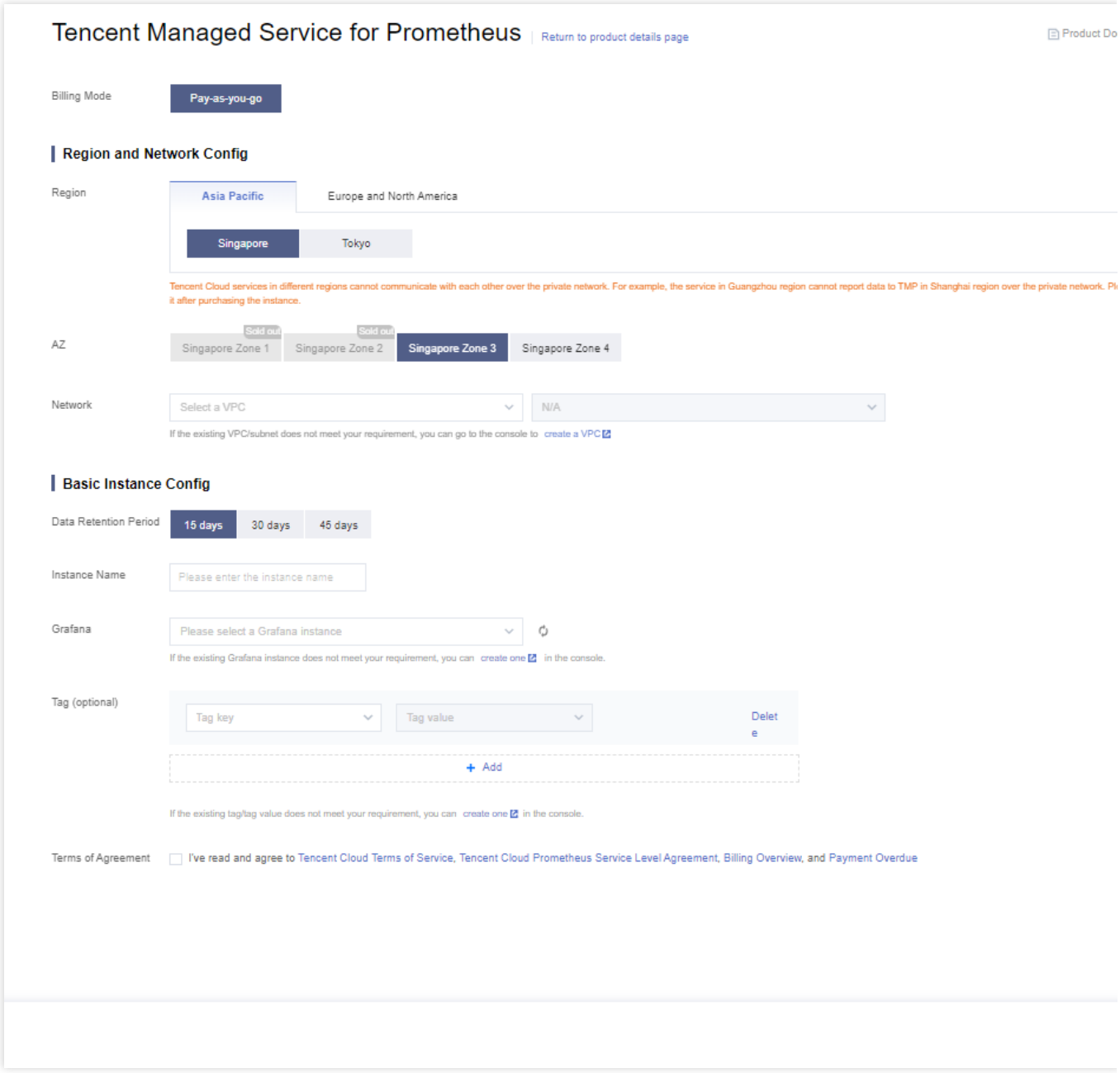
Note:

The purchased TMP instance must be in the same VPC as the monitored CVM instance for network connectivity.

1. Log in to the TMP console and click **Create** to purchase a TMP instance.



2. On the purchase page, select the target instance specification and network. Make sure that the TMP and CVM instances have the same VPC IP range so that data can be collected. Select the instance specification based on your reported data volume.



3. Click **Buy Now** and make the payment.

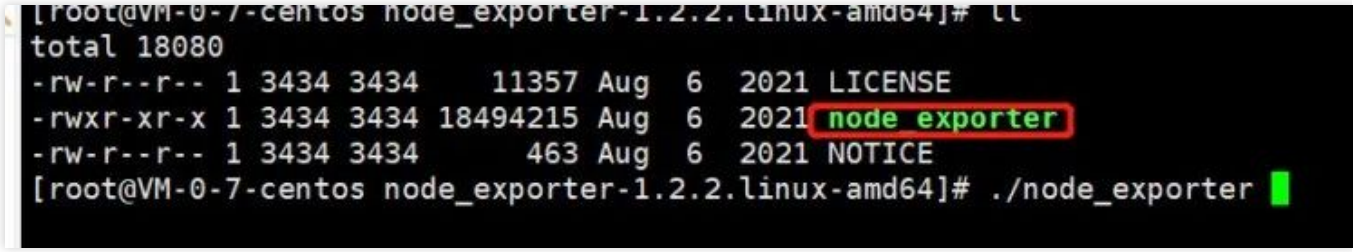
Integrating CVM Basic Metrics

1. Download and install Node Exporter.

Download and install Node Exporter (used to collect basic metric data) in the target CVM instance. Click here or run the following command for download:

```
wget https://github.com/prometheus/node_exporter/releases/download/v1.3.1/node_expo
```

The file directory is as follows:

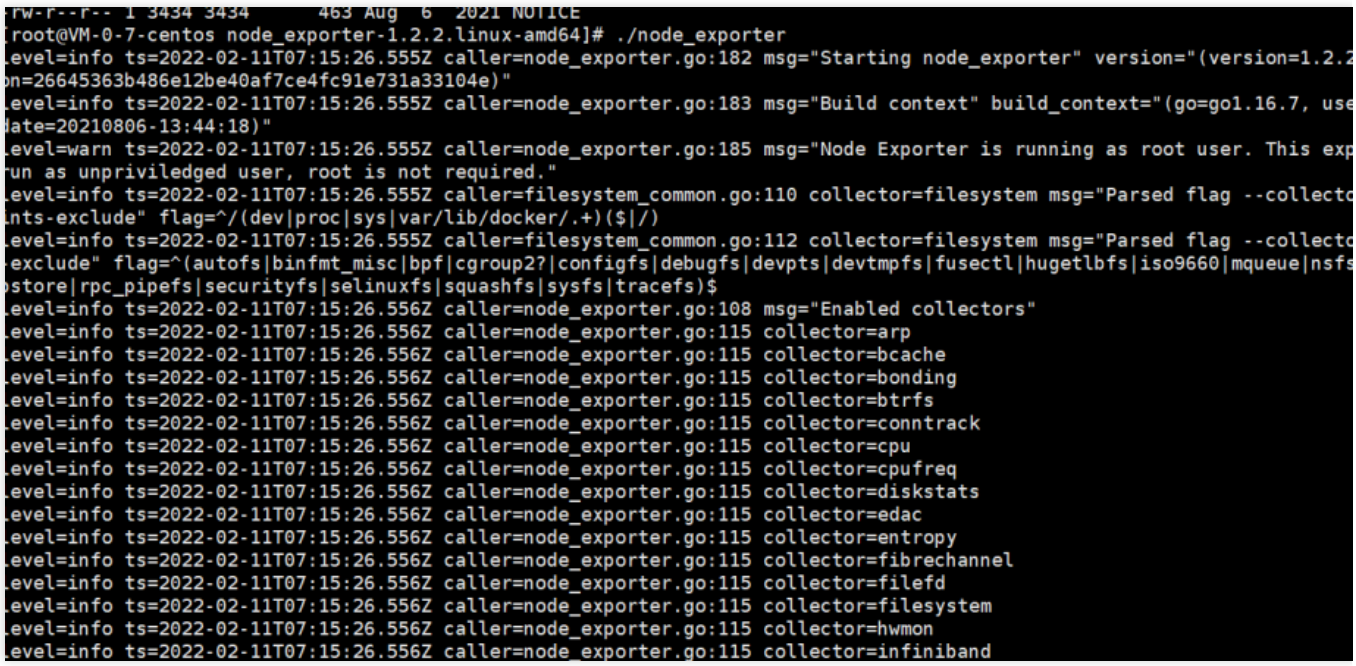


2. Run Node Exporter to collect basic monitoring data.

2.1 Go to the target folder and run Node Exporter.

```
cd node_exporter-1.3.1.linux-amd64
./node_exporter
```

If the following result is displayed, basic monitoring data has been collected successfully.



2.2 Run the following command to expose the basic monitoring data to port 9100:

```
curl 127.0.0.1:9100/metrics
```

You can see the following metric monitoring data that is exposed after the command is executed.

```
[root@VM-0-7-centos node_exporter-1.2.2.linux-amd64]# clear
[root@VM-0-7-centos node_exporter-1.2.2.linux-amd64]# curl 127.0.0.1:9100/metrics
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 0
go_gc_duration_seconds{quantile="0.25"} 0
go_gc_duration_seconds{quantile="0.5"} 0
go_gc_duration_seconds{quantile="0.75"} 0
go_gc_duration_seconds{quantile="1"} 0
go_gc_duration_seconds_sum 0
go_gc_duration_seconds_count 0
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 7
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.16.7"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 2.344136e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 2.344136e+06
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 4562
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 1362
# HELP go_memstats_gc_cpu_fraction The fraction of this program's available CPU time used by the GC sinc
# TYPE go_memstats_gc_cpu_fraction gauge
```

3. Add a scrape task.

Log in to the TMP console, select **Integration Center** > **CVM**, and configure the information in **Task Configuration** as prompted.

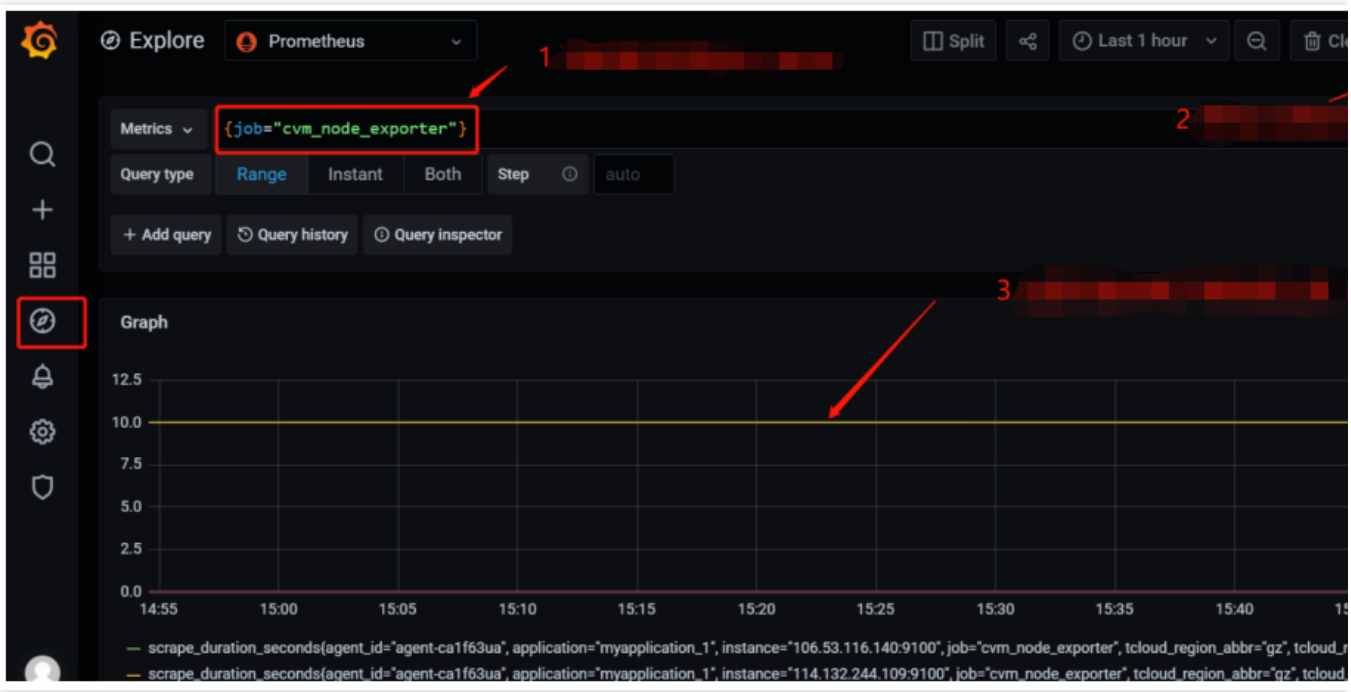
Below is a sample configuration of a scrape task:

```
job_name: example-job-name
metrics_path: /metrics
cvm_sd_configs:
- region: ap-guangzhou
  ports:
  - 9100
  filters:
  - name: tag:Sample tag key
    values:
    - Sample tag value
relabel_configs:
- source_labels: [__meta_cvm_instance_state]
  regex: RUNNING
  action: keep
- regex: __meta_cvm_tag_(.*)
  replacement: $1
  action: labelmap
- source_labels: [__meta_cvm_region]
  target_label: region
  action: replace
```

4. Check whether data is reported successfully.

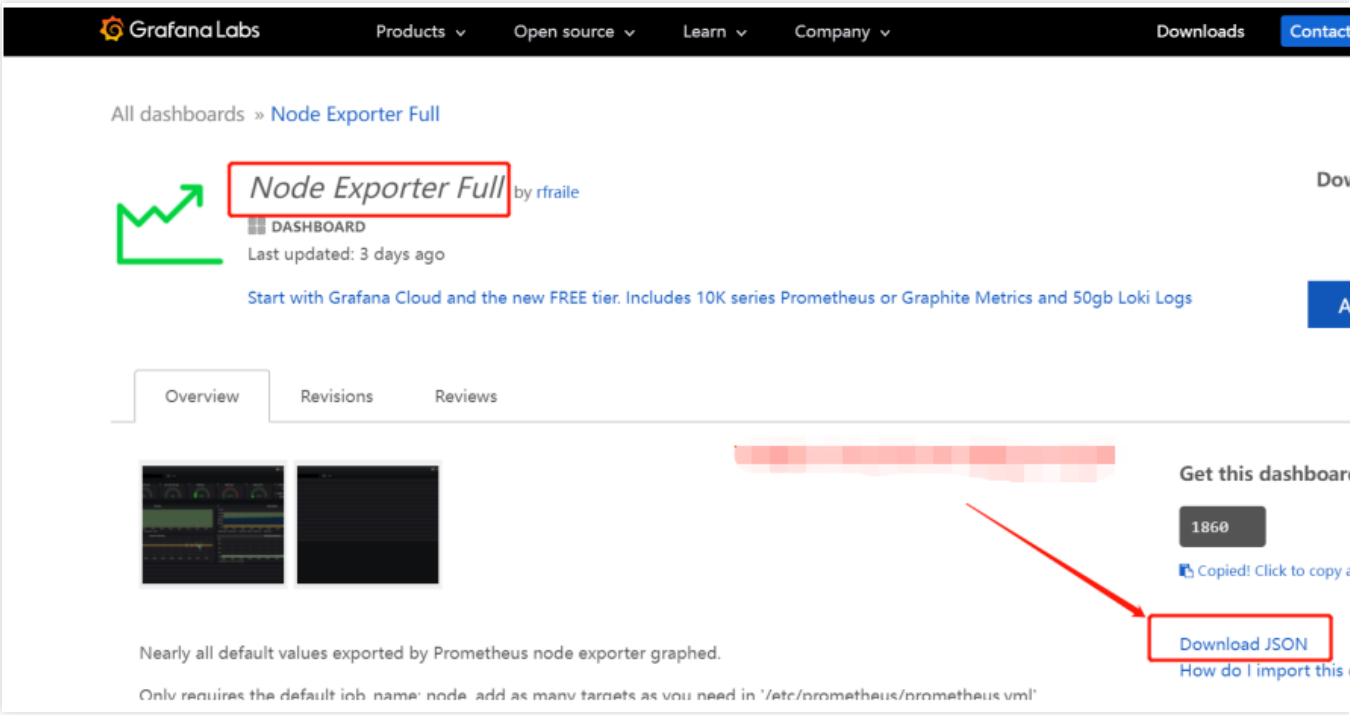
Log in to the TMP console and click the Grafana icon to enter Grafana.

Search for `{job="cvm_node_exporter"}` in **Explore** to see whether there is data, and if so, data is reported successfully.

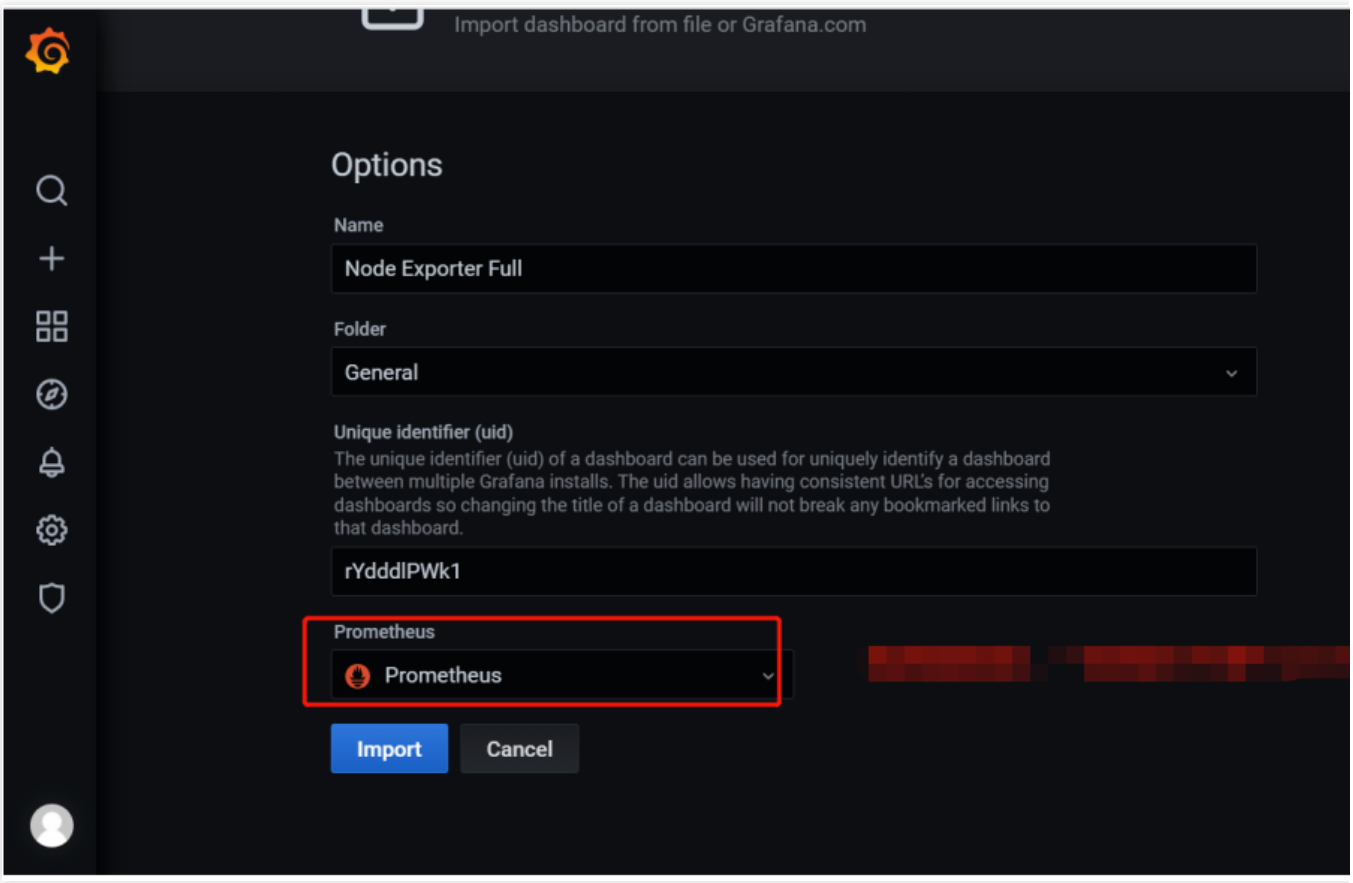


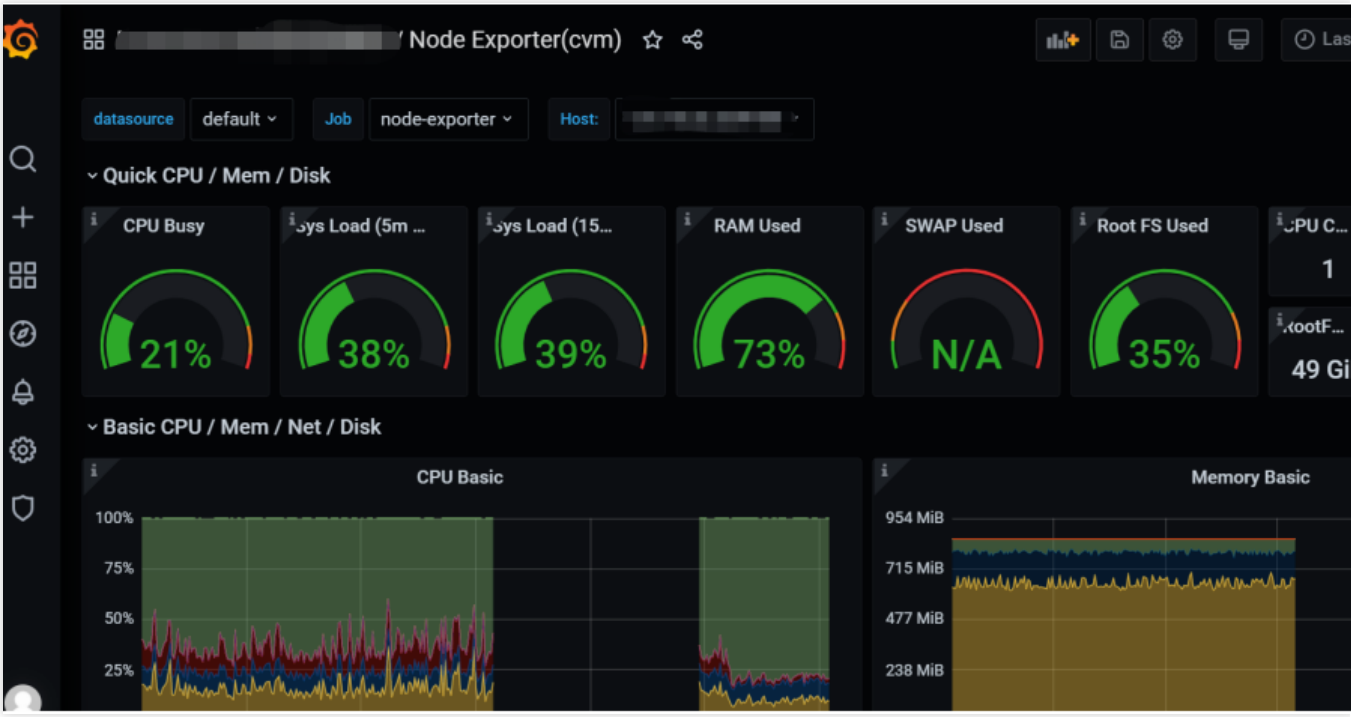
5. Configure the dashboard page: Every product has some existing JSON files that can be directly imported into the dashboard.

5.1 **Download a dashboard file**: Go to the **Dashboard** page, search for `node_exporter`, and select the latest dashboard for download.



5.2 **Import a JSON file into the dashboard**: Log in to the TMP console, select **Basic Info** > **Grafana Address** to enter Grafana. In the Grafana console, select **Create** > **Import** and upload the dashboard file in **Upload JSON file**.





Integrating CVM Metrics at the Business Layer

Prometheus provides four metric types for different monitoring scenarios: Counter, Gauge, Histogram, and Summary. The Prometheus community provides SDKs for multiple programing languages, which are basically similar in usage and mainly differ in the syntax. The following uses Go as an example to describe how to report custom monitoring metrics. For detailed directions of other metric types, see Custom Monitoring.

Counter

A metric in Counter type increases monotonically and will be reset after service restart. You can use counters to monitor the numbers of requests, exceptions, user logins, orders, etc.

1. You can use a counter to monitor the number of orders as follows:

```
package order

import (
 "github.com/prometheus/client_golang/prometheus"
 "github.com/prometheus/client_golang/prometheus/promauto"
)

// Define the counter object to be monitored
var (
 opsProcessed = promauto.NewCounterVec(prometheus.CounterOpts{
     Name: "order_service_processed_orders_total",
```

```
        Help: "The total number of processed orders",
    }, []string{"status"}) // Processing status
)

// Process the order
func makeOrder() {
    opsProcessed.WithLabelValues("success").Inc() // Success
    // opsProcessed.WithLabelValues("fail").Inc() // Failure

    // Order placement business logic
}
```

For example, you can use the `rate()` function to get the order increase rate:

```
rate(order_service_processed_orders_total[5m])
```

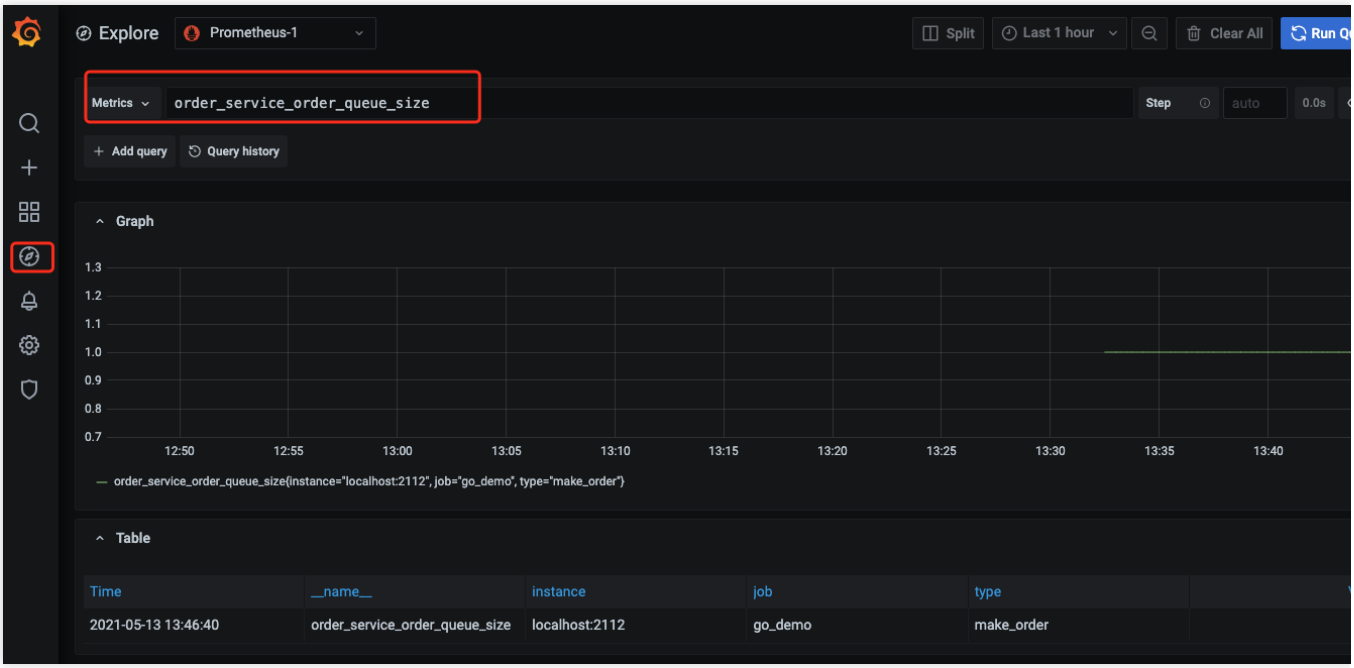
2. Expose Prometheus metrics:

Use `promhttp.Handler()` to expose the metric tracking data to the HTTP service.

```
package main

import (
    "net/http"

    "github.com/prometheus/client_golang/prometheus/promhttp"
)

func main() {
        // Business code

        // Expose Prometheus metrics in the HTTP service
        http.Handle("/metrics", promhttp.Handler())

        // Business code
}
```

3. Collect data:

After the tracking of custom metrics for your business is completed and the application is released, you can use Prometheus to collect the monitoring metric data. After the collection is completed, wait a few minutes and then you can view the business metric monitoring data in Grafana integrated in TMP.



TKE Monitoring

Last updated：2024-01-29 15:55:08

Background

As we all know, Prometheus is the best monitoring tool for container scenarios. However, self-building Prometheus is too expensive for small and medium-sized enterprises with limited Ops manpower, and it is likely to hit performance bottlenecks for large enterprises with rapid business development. Therefore, using Prometheus managed on the cloud has become the first choice for more and more cloud companies. For how to use the managed Prometheus to monitor TKE, see Tencent Kubernetes Engine (TKE).

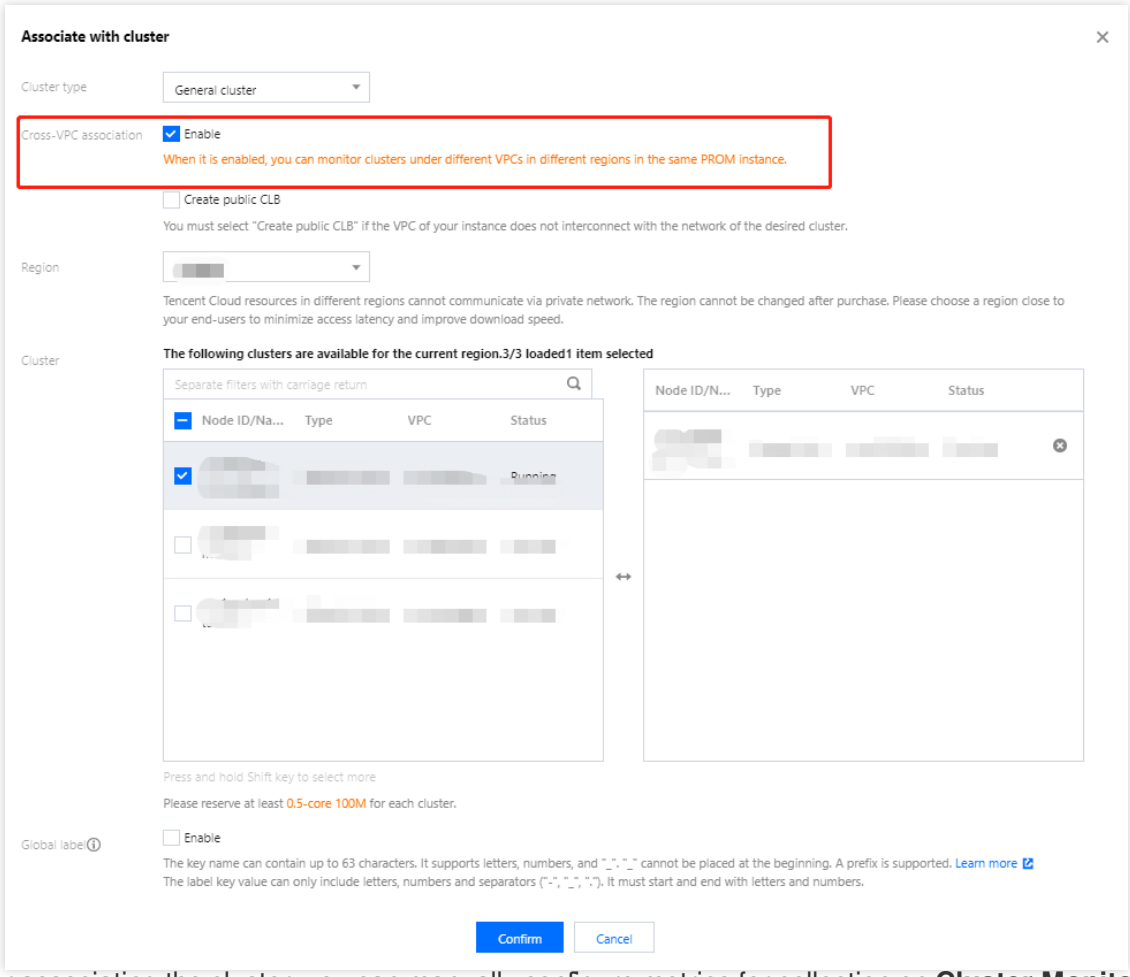
Directions

Step 1. Purchasing an instance

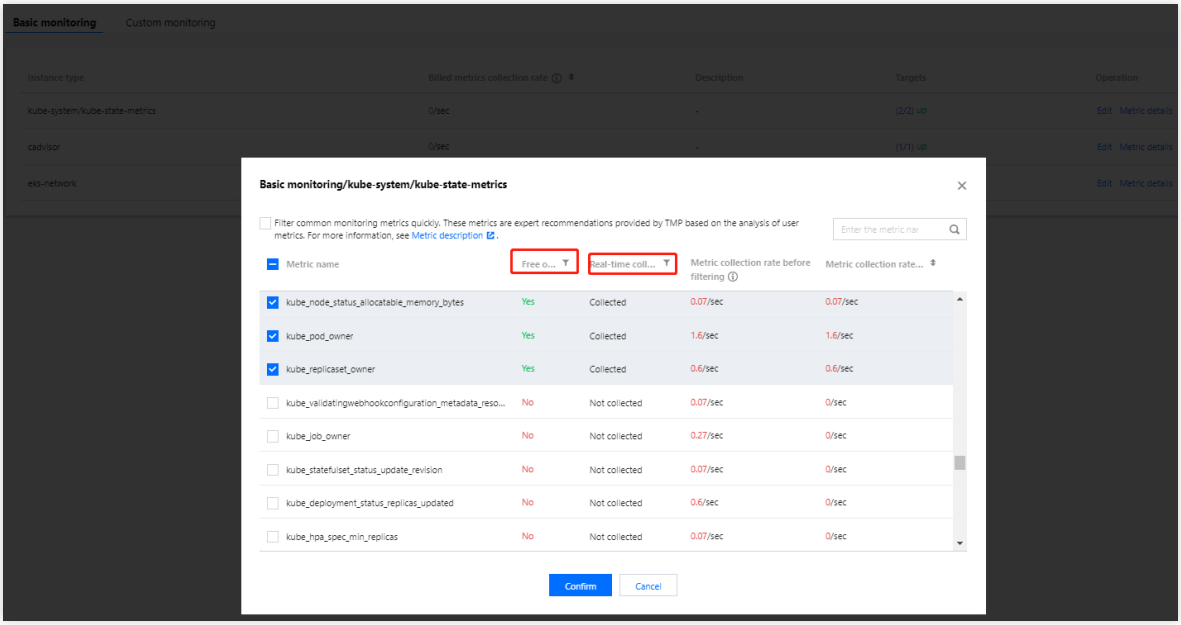
1. Log in to the TMP console.
2. Click **Create**, select the purchase region, storage duration and select the Grafana instance to be associated based on your needs. If there is no Grafana instance, see Creating Instance to create one. You need to create an instance and complete the purchase.
3. After completing the configuration, click **Buy Now**. For more information on billing rules, see Pay-as-You-Go Description.

Step 2. Integrate with TKE

1. After creating the instance, click the **ID**/**Name** of the target instance in the instance list to enter the instance details page.
2. On the left sidebar, click **Integrate with TKE** > **Associate Cluster**.
3. Select the cluster that needs to be associated in the pop-up window. A total of 4 types of clusters are supported: standard cluster, elastic cluster, registered cluster, and edge cluster. The clusters can be across VPCs. If different VPCs are not interconnected, you need to create a public network CLB instance.

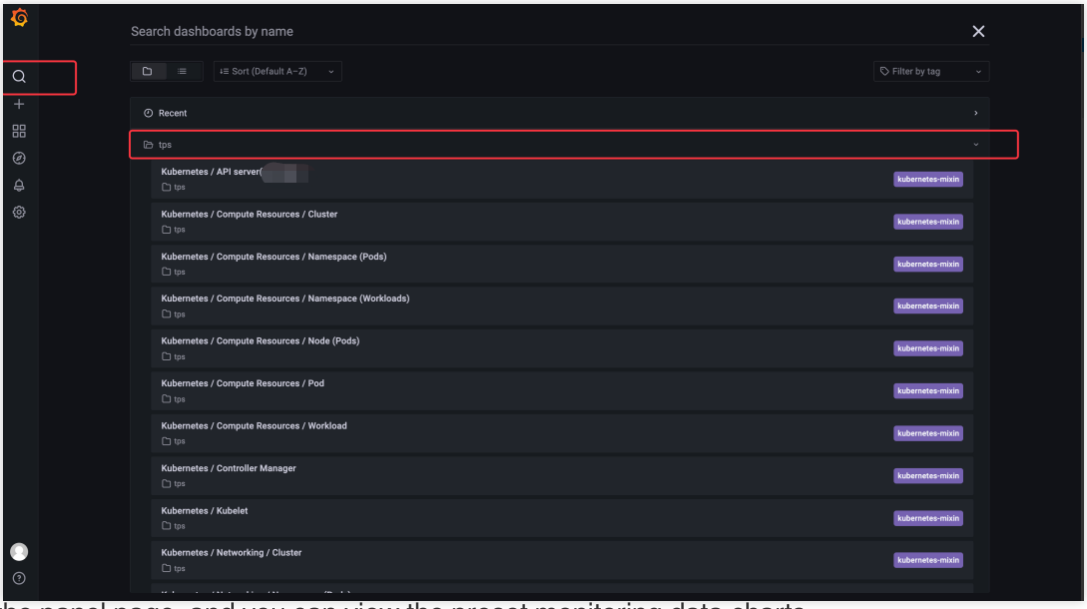


4. After associating the cluster, you can manually configure metrics for collection on **Cluster Monitoring** > **Data Collection Configuration**, view the default free basic collection metrics, and add or reduce the metrics as needed.

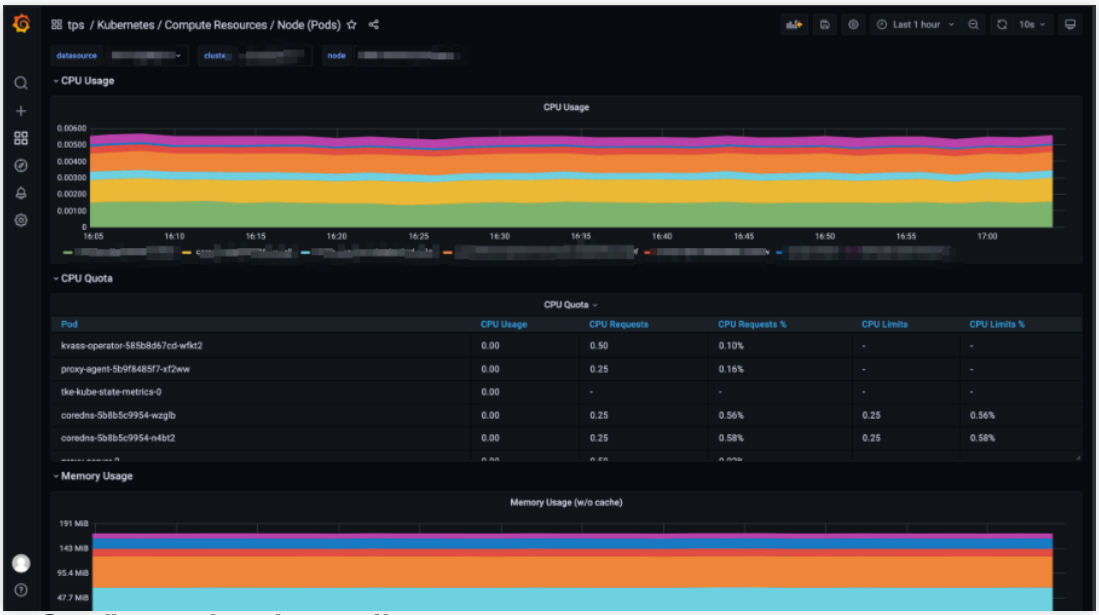


Step 3. View monitoring data in Grafana

1. Click the Grafana icon to the right of the instance in the instance list to enter the Grafana service platform.
2. In the dashboard search list, TKE-related monitoring panels are preset by default, and click a panel name.

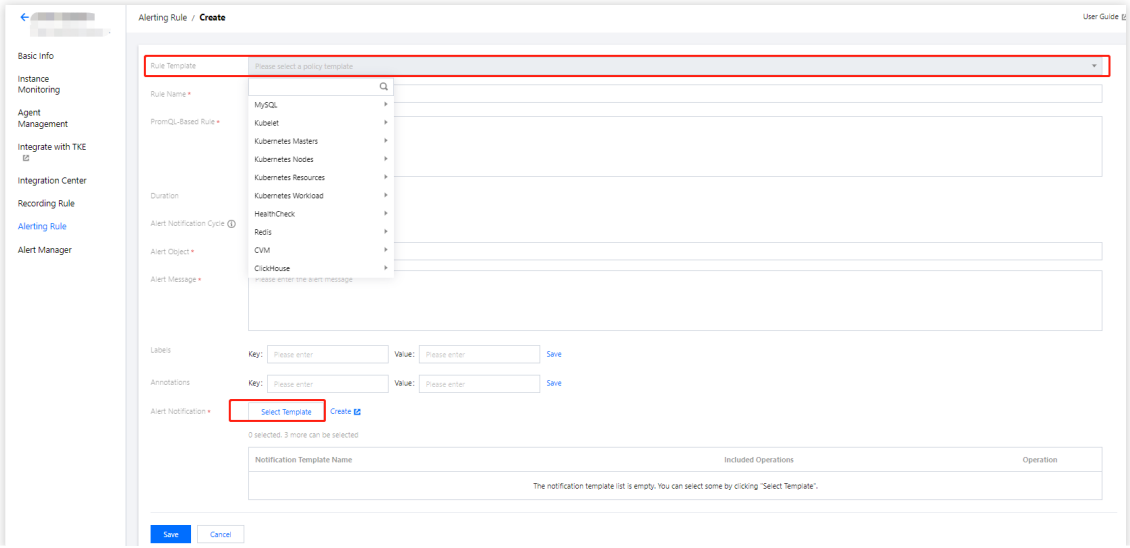


Enter the panel page, and you can view the preset monitoring data charts.



Step 4. Configure the alert policy

On the Prometheus instance details page, click **Alerting Rule**, and you can select a preset template type without manual configuration. For alert notifications, you can select existing notification templates on the TCOP to quickly configure alerts.



Enabling Public Network Access for TKE Serverless Cluster

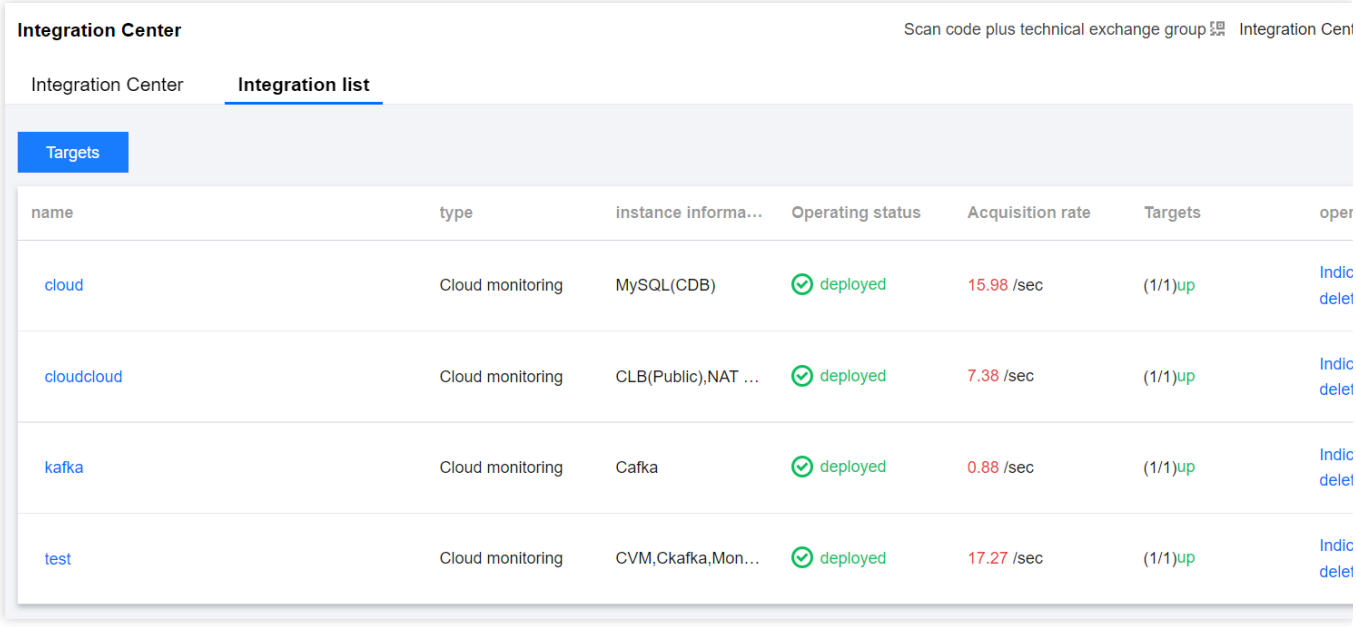
Last updated：2024-01-29 15:55:07

Overview

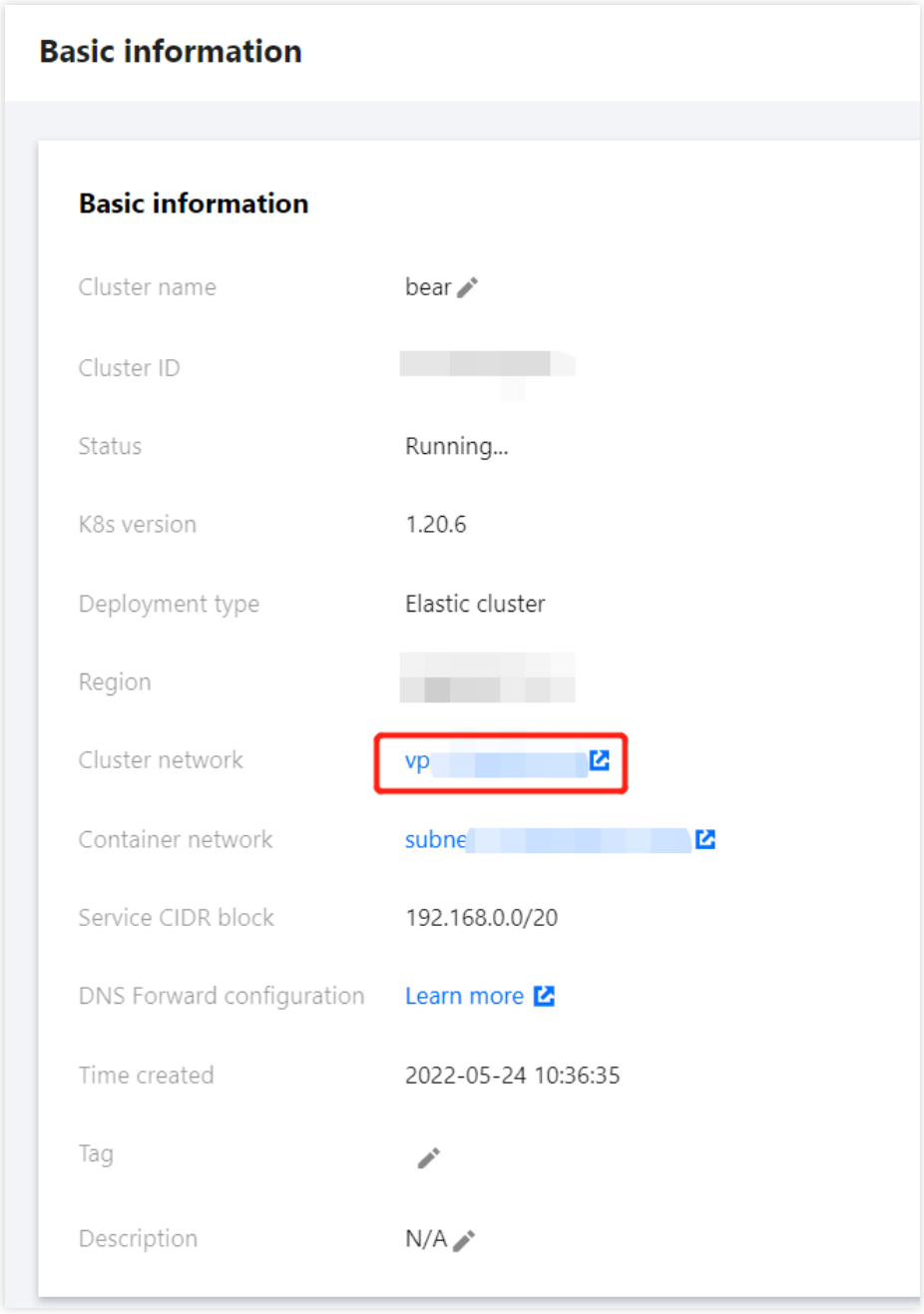
TMP is integrated with CM. When creating an integration, if you select COS, you need to enable public network access for the TKE Serverless cluster of the target CM exporter, as COS doesn't support private network access.

Directions

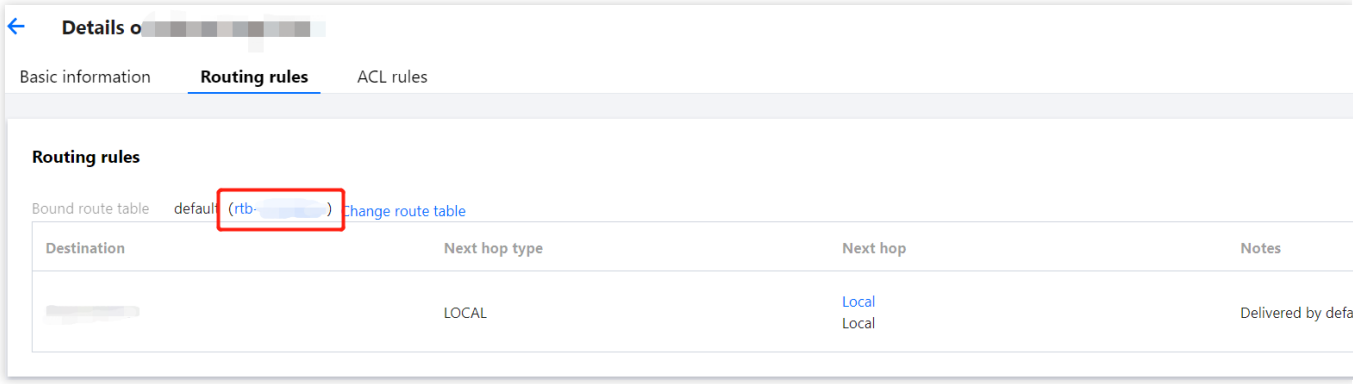
1. Log in to the TMP console.
2. Click the target instance to enter the instance management page. Then, click **Integration Center** > **Integration List**.
3. Click **Log** in the **Operation** column of the line where **Type** is **CM**.



4. On the topbar, switch to the Pod management page. Click the instance name to enter the cluster page.
5. On the **Basic Info** page, click **Container Network**.



6. On the topbar of the container network page, switch to the **Routing Policy** tab. Click the route table link (`rtb-xxx` in the list) to enter the route table page.

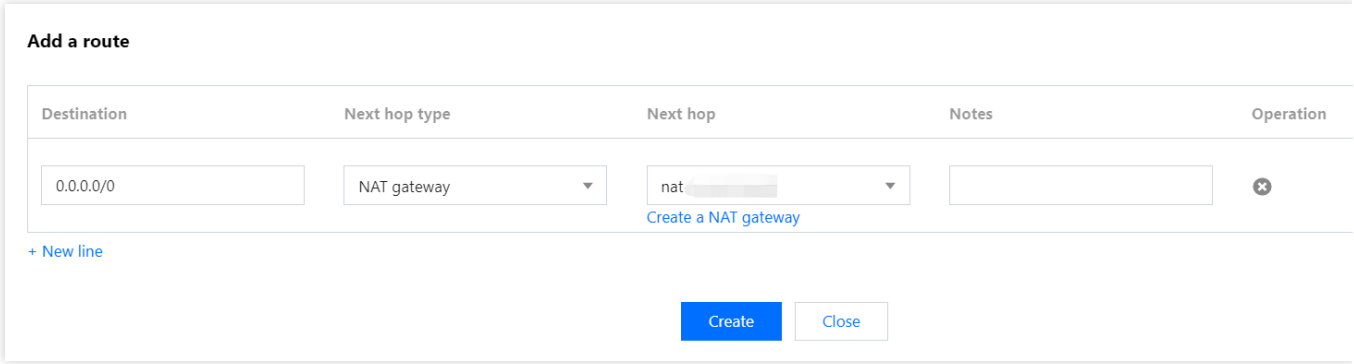


7. On the route table page, click **Create Routing Policy**.

Destination: Enter `0.0.0.0/0` .

Next Hop Type: Select **NAT Gateway**.

Next Hop: Select the target gateway. If there is no gateway, create one as instructed in Getting Started.



8. Click **Create**. After the creation is successful, public network access is enabled for TKE Serverless.

Connecting TMP to Local Grafana

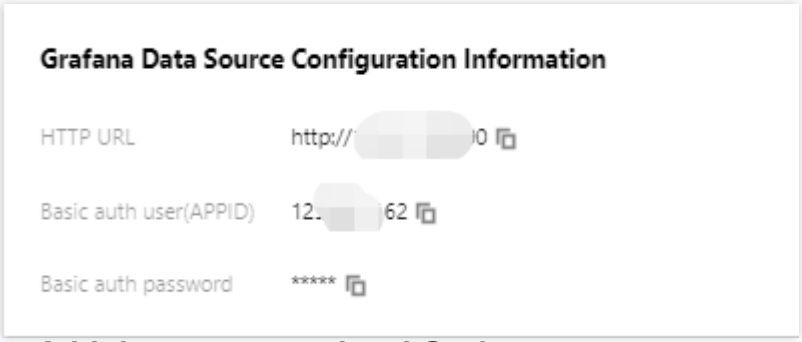
Last updated：2024-01-29 15:55:08

If you need to view the relevant data of TMP in the local Grafana system, you can use the HTTP API provided by TMP to do so. This document describes how to connect TMP data to local Grafana.

Directions

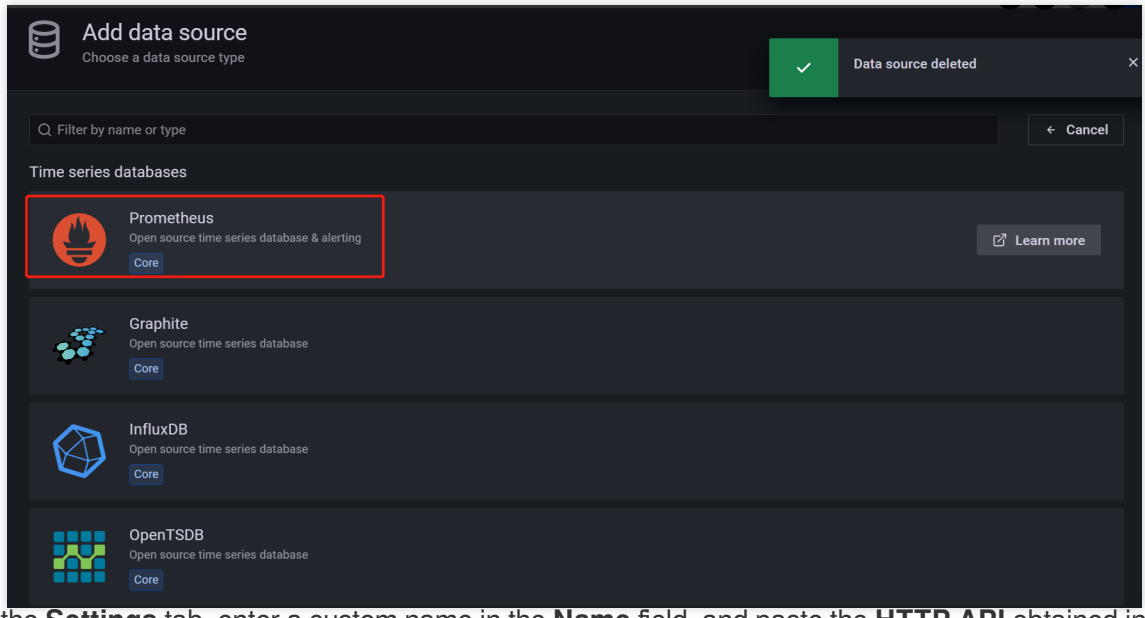
Step 1: Obtain the HTTP API provided by TMP

1. Log in to the TMP console.
2. Click the corresponding pay-as-you-go instance to enter the basic information page of TMP.
3. Get the HTTP API address in the service address module. If you need to improve the security of Grafana data read, you can obtain the authentication token of the TMP instance and fill it in as instructed in step 2.



Step 2. Add data source to local Grafana

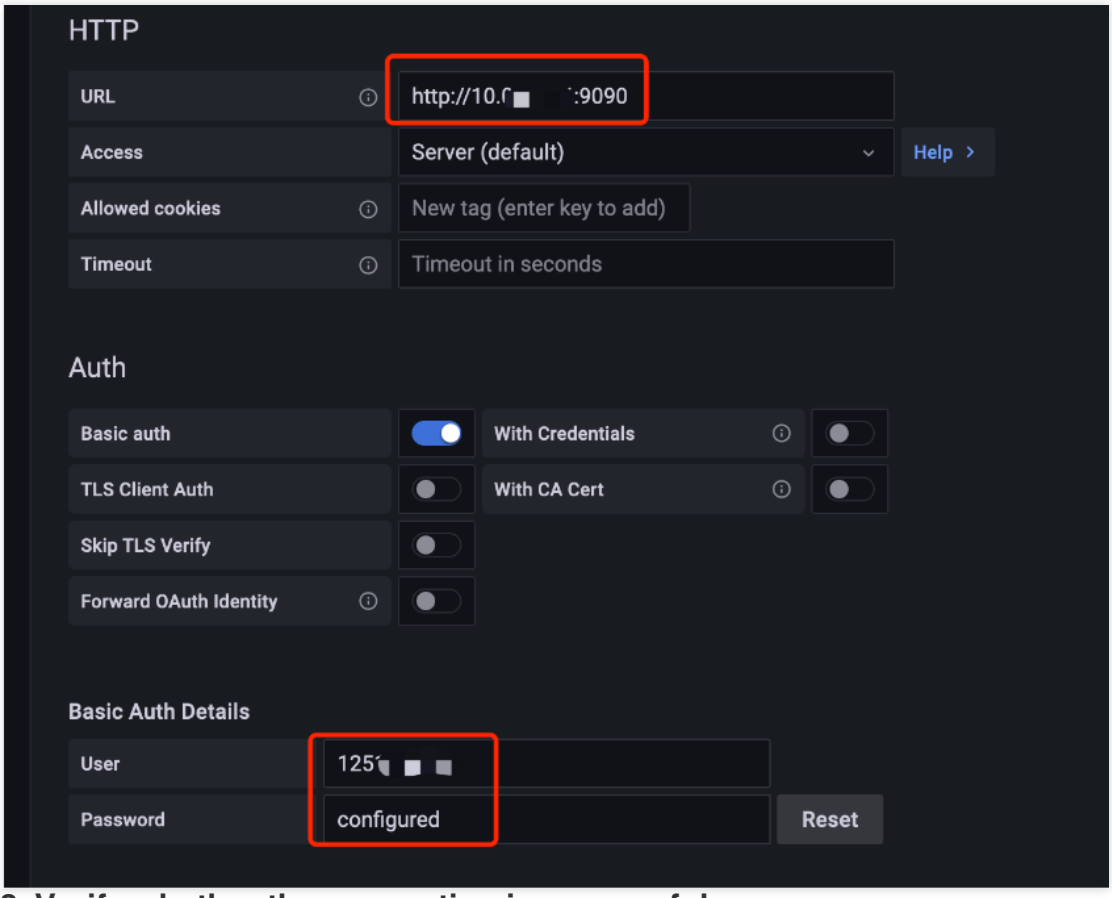
1. Log in to the local Grafana system with **admin account**.
2. Select **Configuration > Data Sources** on the left sidebar (non-admins cannot view this menu).
3. On the **Data Sources** page, click **Add data source**.
4. On the **Add data source** page, select **Prometheus**.



5. On the **Settings** tab, enter a custom name in the **Name** field, and paste the **HTTP API** obtained in step 1 in the URL field.

6. Toggle on **Basic Auth** in the **Auth** module. In the **Basic Auth Details** module, set **User** same as **Basic auth user** and **Password** as **Basic auth password** obtained in the step 1.

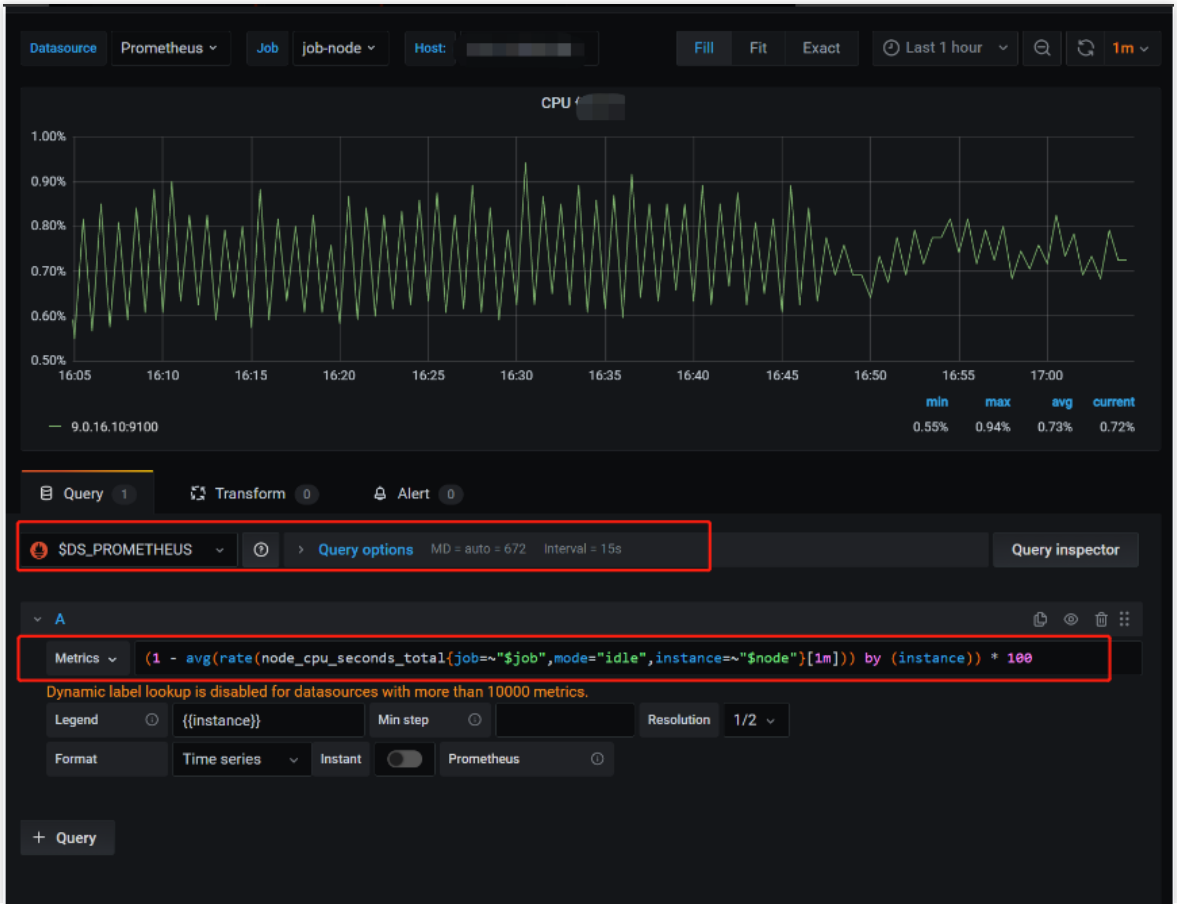
7. Click **Save & test** to complete the settings.



Step 3. Verify whether the connection is successful

Follow the steps below to verify whether TMP is successfully connected to the local Grafana:

1. Log in to your local Grafana system.
2. On the left sidebar, select **+ > Create**.
3. On the **New dashboard** page, click **Add a new panel**.
4. On the **Edit Panel** page, select the data source added in step 2 in the drop-down box on the **Query** page, enter the metric name in the **Metrics** field in the A module and press Enter.
5. If the chart of the corresponding metric can be displayed, the operation is successful. Otherwise, check whether the API address or token entered is correct, and whether the data source has TMP data.



Enabling Public Network Access for Prometheus Instances

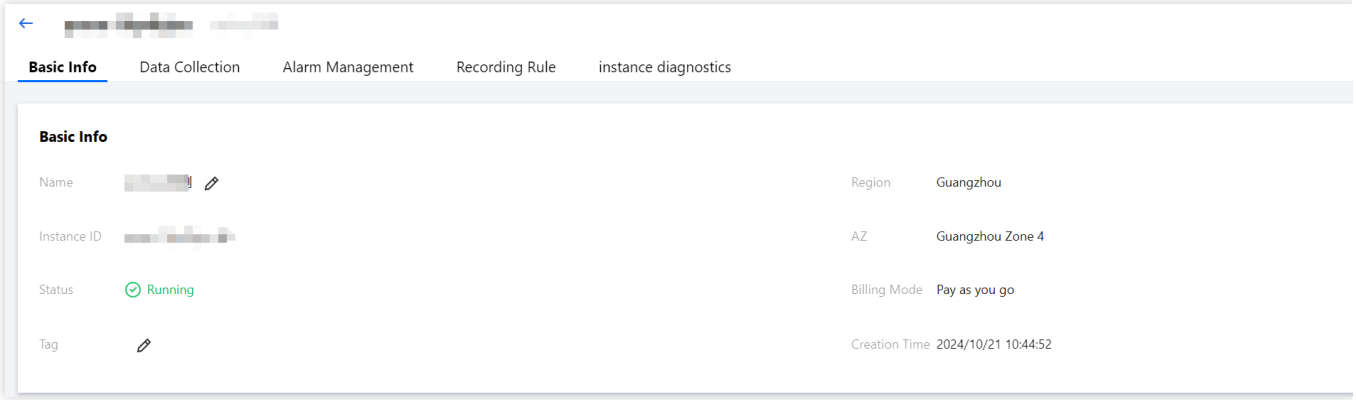
Last updated：2024-11-22 18:16:04

Operation Background

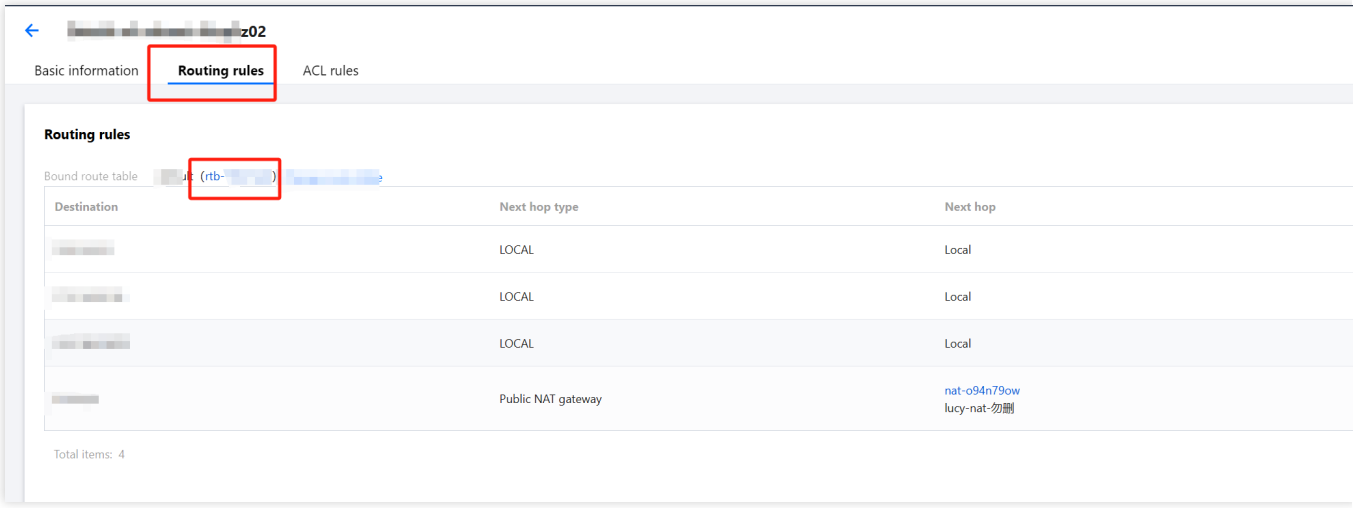
In some scenarios, you may need your Prometheus instance to have the ability to access the public network, for example:
When Prometheus is integrated to monitor COS products on the cloud, since COS products do not support private network access, the Prometheus instance needs to have the ability to access the public network.
The WebHook in the Prometheus alarm policy is used, and this WebHook URL is a public network address.
In such cases, you need to enable public network access for Prometheus (which essentially means enabling public network access for the TKE-managed cluster where the Prometheus instance components are located).

Operations

1. Log in to TMP Console.
2. Click the corresponding instance to enter the Instance Management page. Then, click **Subnet** on the Basic info tab page.



3. In the top menu of the subnet, switch to the **Routing rules** section. Click the route table link (rtb-xxx on the list) to enter the route table interface.

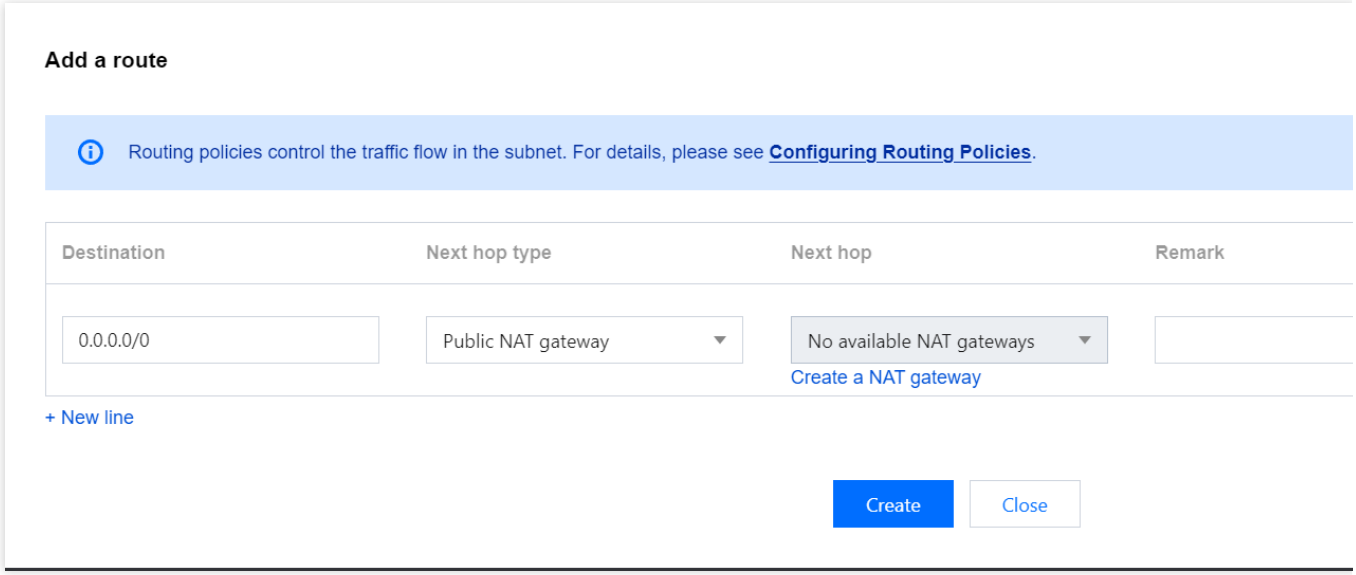


4. On the route table page, click **Add routing policy**.

Destination: Enter 0.0.0.0/0.

Next hop type: select NAT Gateway.

Next hop: select the target gateway. If there is no gateway, please refer to the NAT Gateway guide to create one.



5. After completing the settings, click **Create**. After the creation is successful, public network access is enabled for TKE Serverless.

Configuring a Public Network Address for a Prometheus Instance

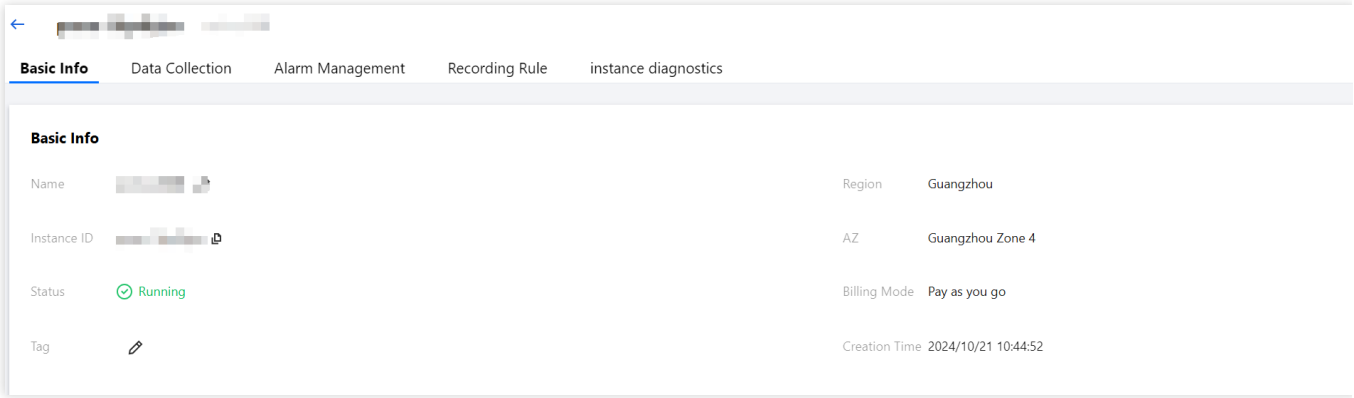
Last updated：2024-11-22 18:15:33

This document introduces how to configure a public network address for Prometheus monitoring.

Practice Steps

Step 1: Purchasing a Prometheus Instance

1. Log in to TMP Console.
2. Click **Create** in the upper left corner to enter the Prometheus purchase page. You can purchase the corresponding instance based on your actual needs. For details, see Creating Instance.
3. After successfully purchasing, click the ID/Name of the created instance to enter the **Basic Info** of the instance details page to obtain the Prometheus IPV4 address.



Step 2: Applying for the Cross-VPC Access Feature of CLB (Beta Feature)

1. Go to the CLB Cross-Region Binding 2.0 and Hybrid Cloud Deployment Beta Application page.
2. Fill in required information and submit the application.
3. You can use the cross-VPC access feature of CLB after the application is approved.

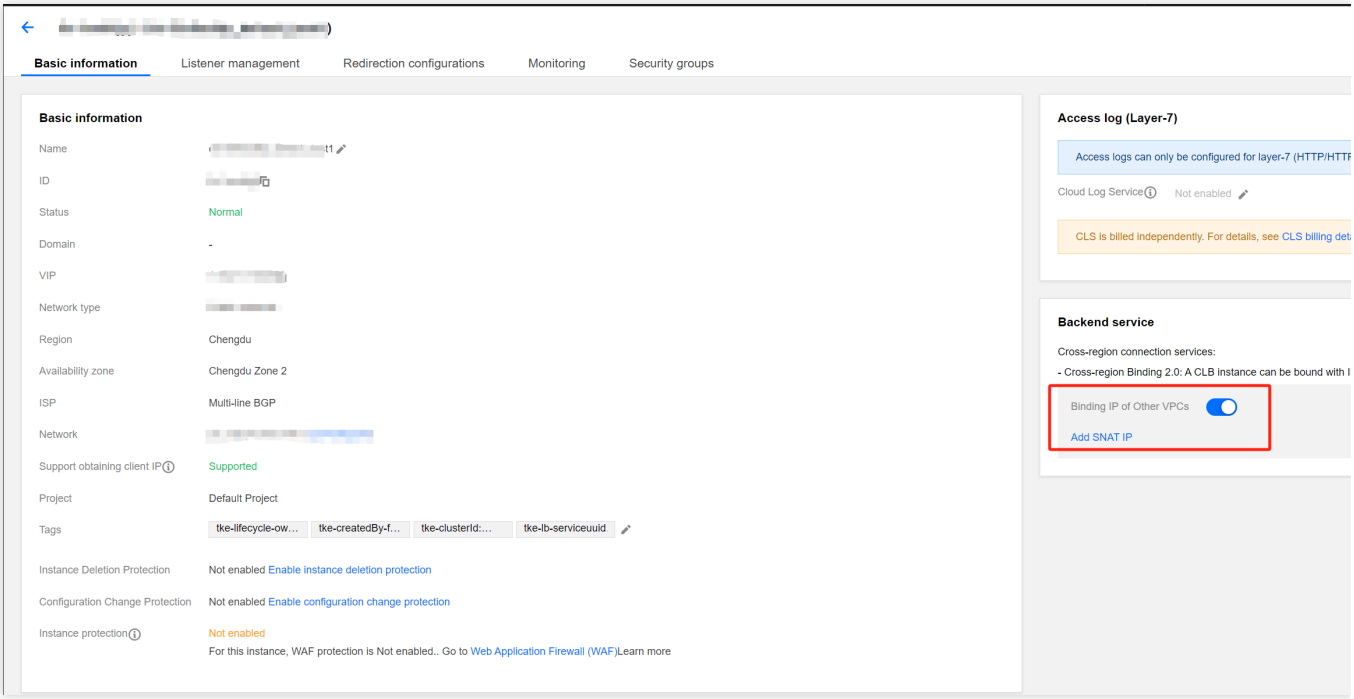
Step 3: Creating a CLB Instance for the Public Network

1. Go to the CLB Console and create a CLB instance.
2. Fill in the information as prompted. For detailed instructions, see Creating a CLB Instance.

Note:

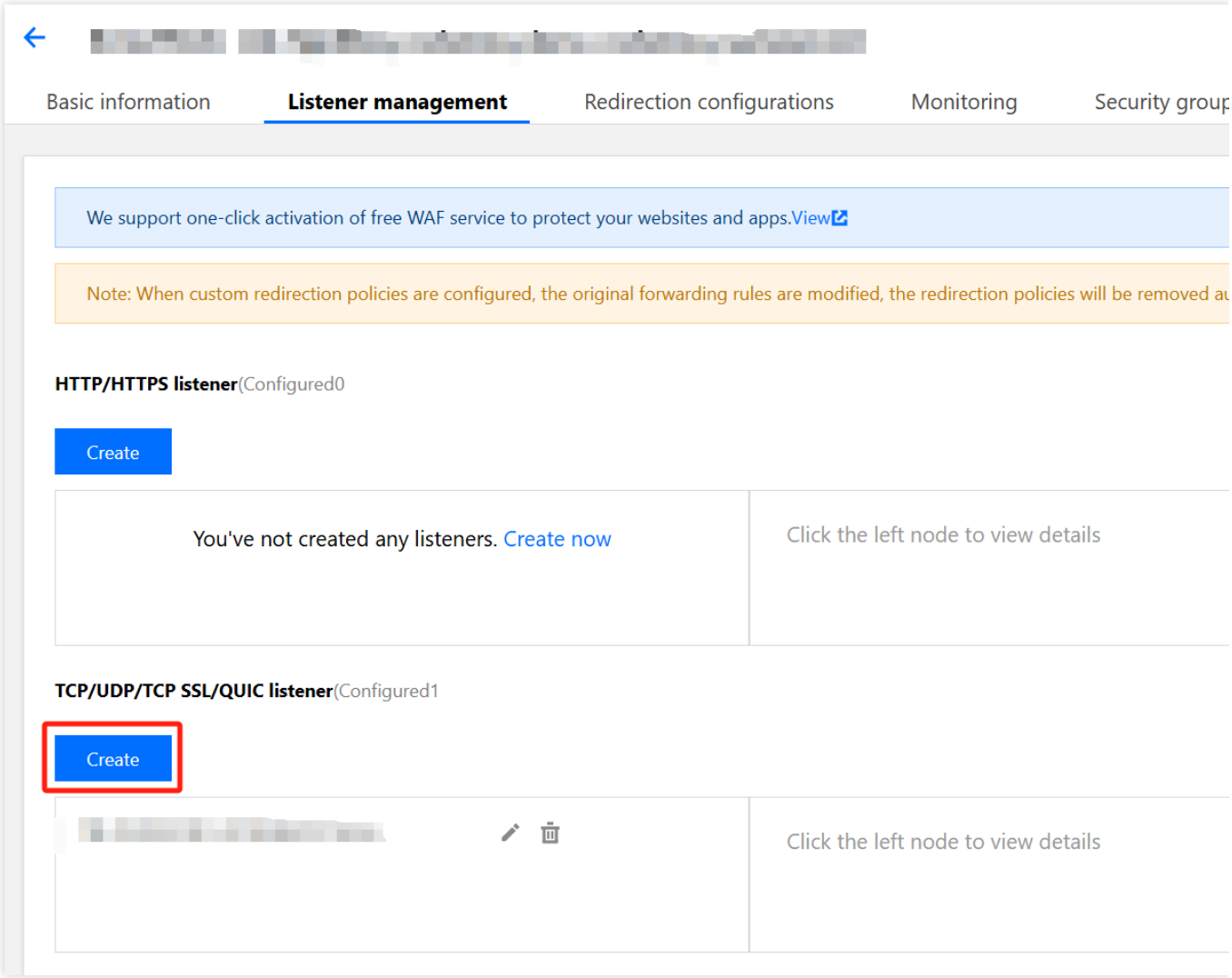
To bind the private IP address of a Prometheus instance, you need to create a CLB instance under the same VPC as this Prometheus instance. If a CLB instance already exists on the public network, no creation is required.

3. After creation, enter the basic information page of the instance and enable the cross-VPC access feature.

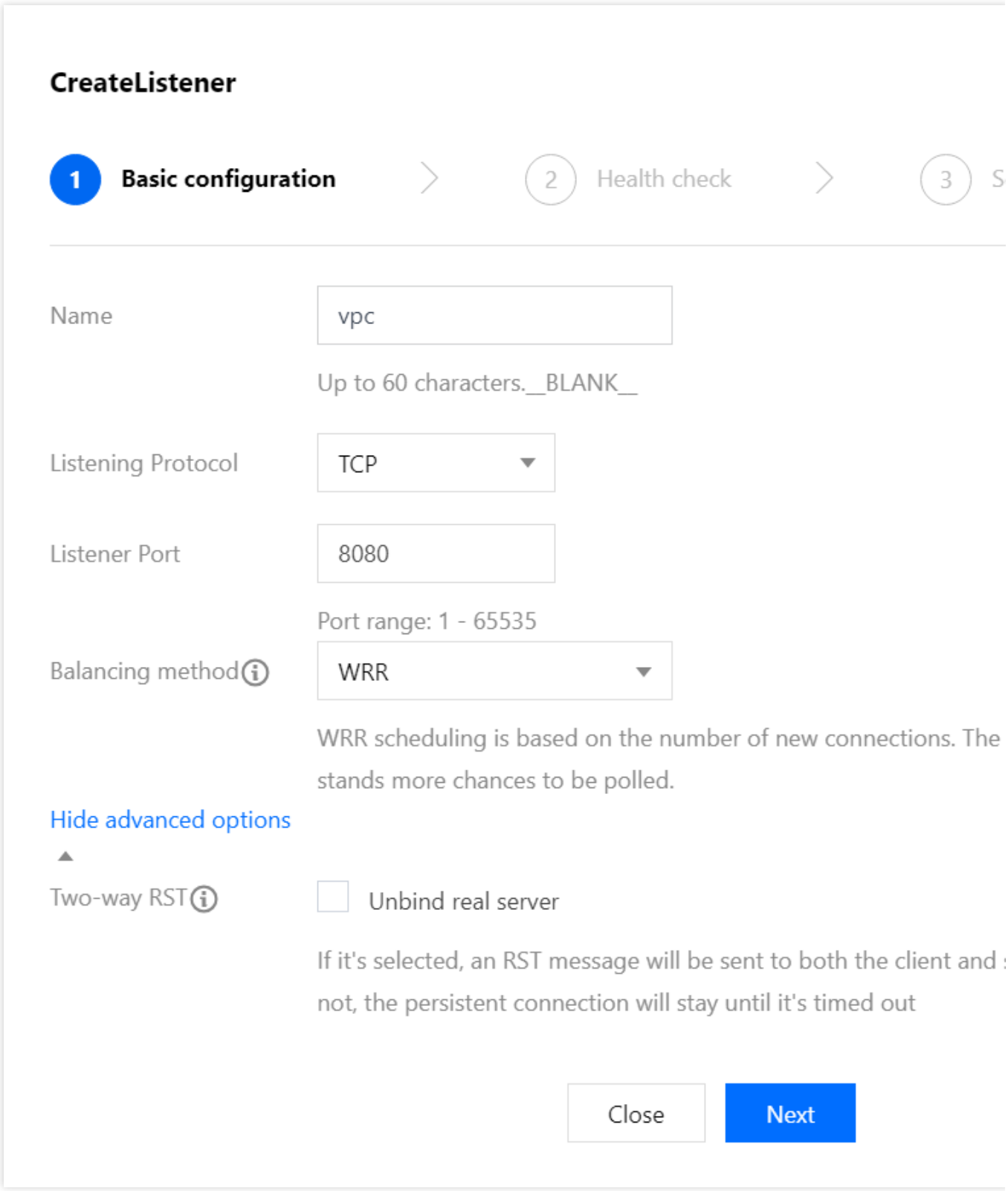


Step 4: Binding the Backend Service

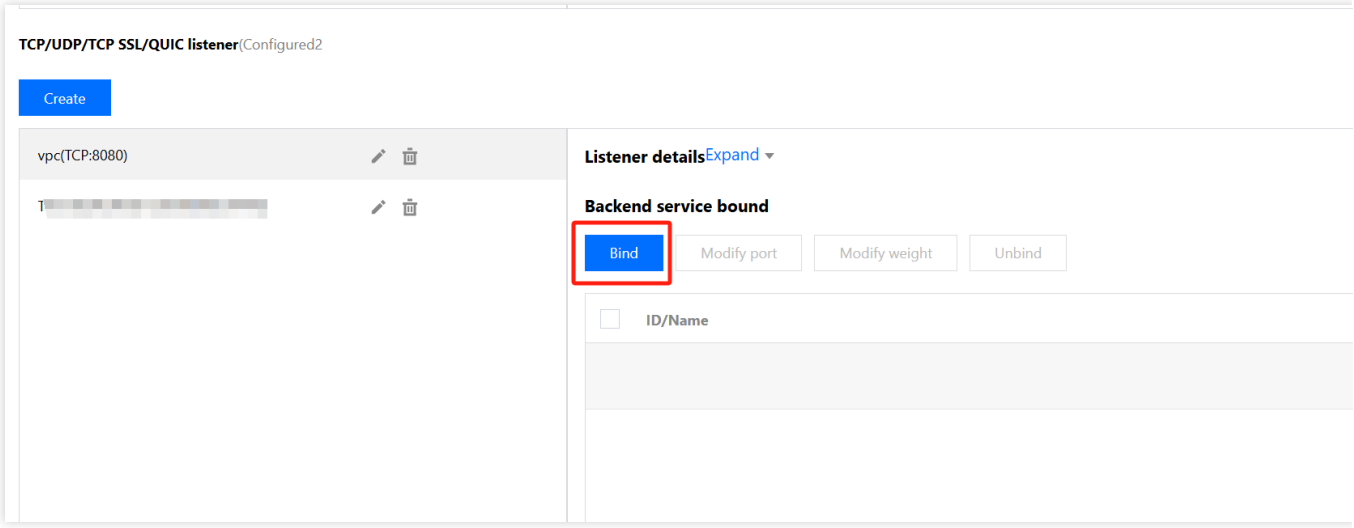
1. Enter the **Listener management** page.
2. Click **Create** under the TCP/UDP/TCP SSL/QUIC listener.



For the operation guide, see CLB Listener.



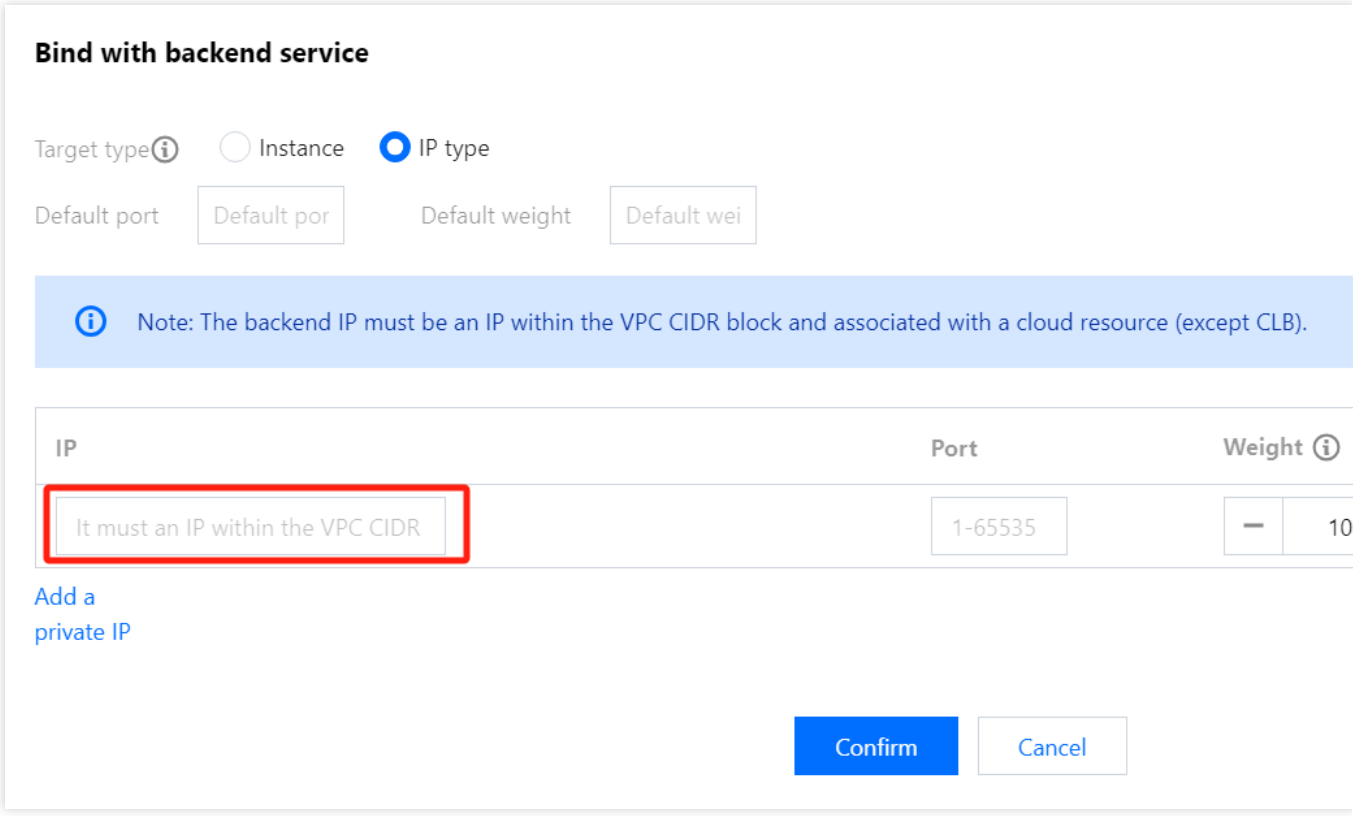
3. After creating the listener, click the listener name. In the sub-window, click **Bind** to bind the backend service.



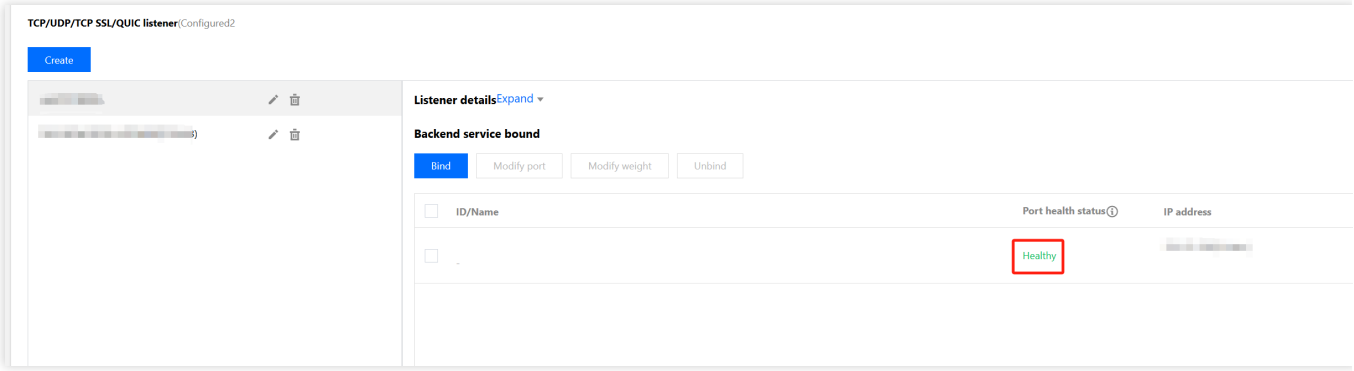
Target Type: Select IP Type.

Default Port: Enter 9090.

IP Address: Enter the IPv4 address of the Prometheus instance obtained in step 1.

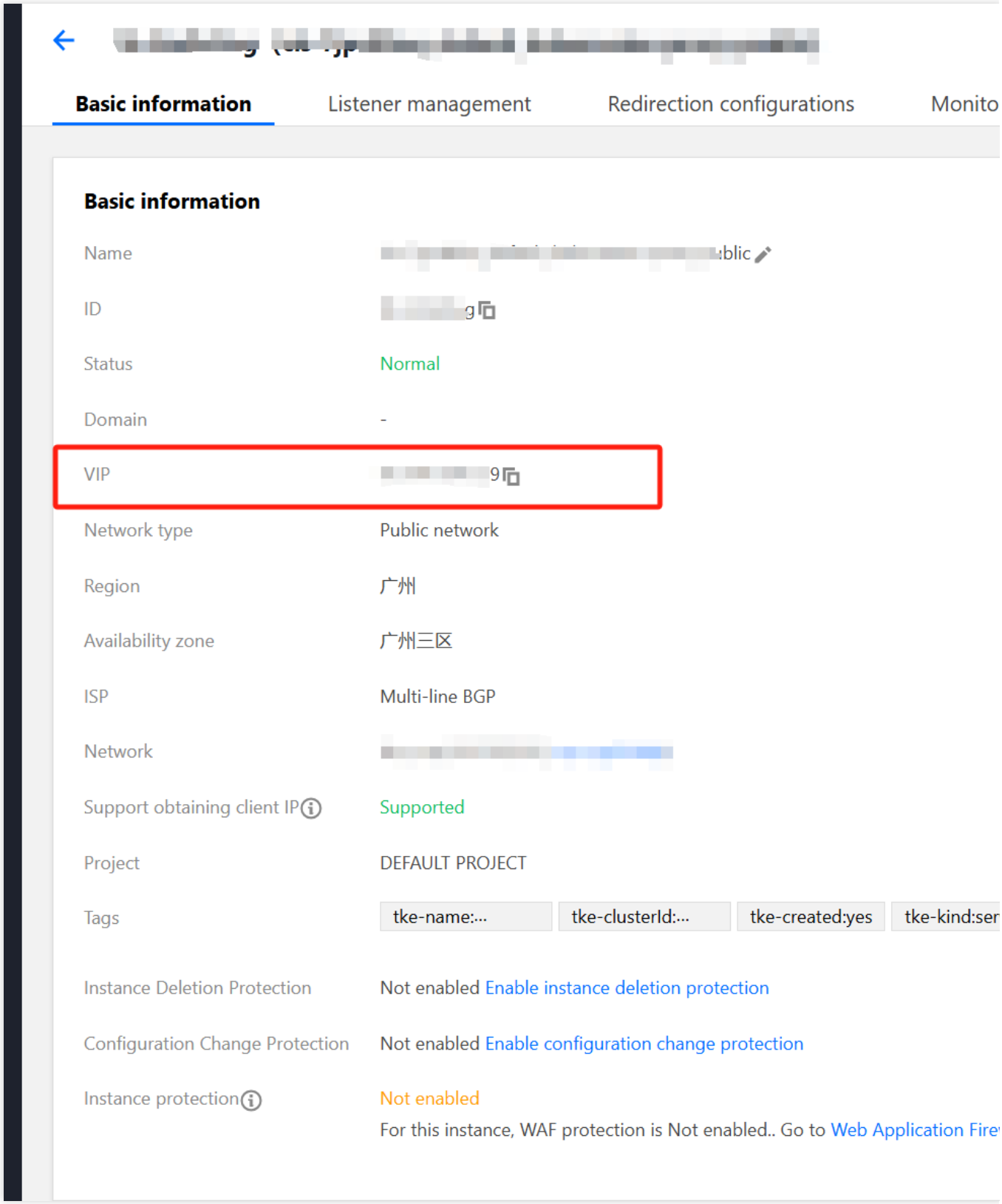


4. Click **Listener Name** to check whether the listening is normal.

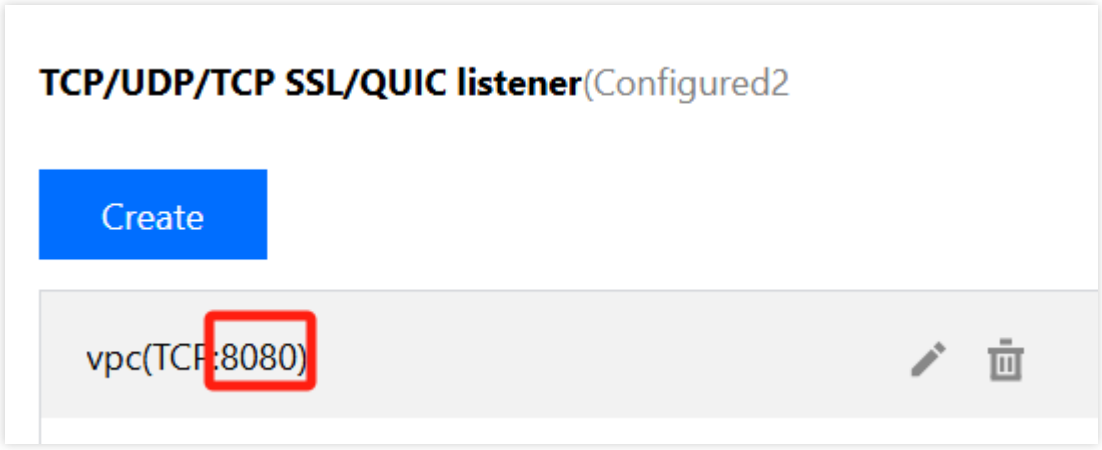


Step 5: Testing Whether the Configuration is Successful

1. Check the public network address of CLB. Assume that the address is 192.168.1.1.



2. Check the port configured for the listener, for example, port 8080.



Based on the above information, it is determined that the public network address for Prometheus forwarding is IP:PORT in the following new address, that is, `192.168.1.1:8080` .

3. Check in the browser or on the machine to see whether UP data can be obtained by using this IP address.

4. HTTP API address:

```
http://IP:PORT/api/v1/query?query=up
```

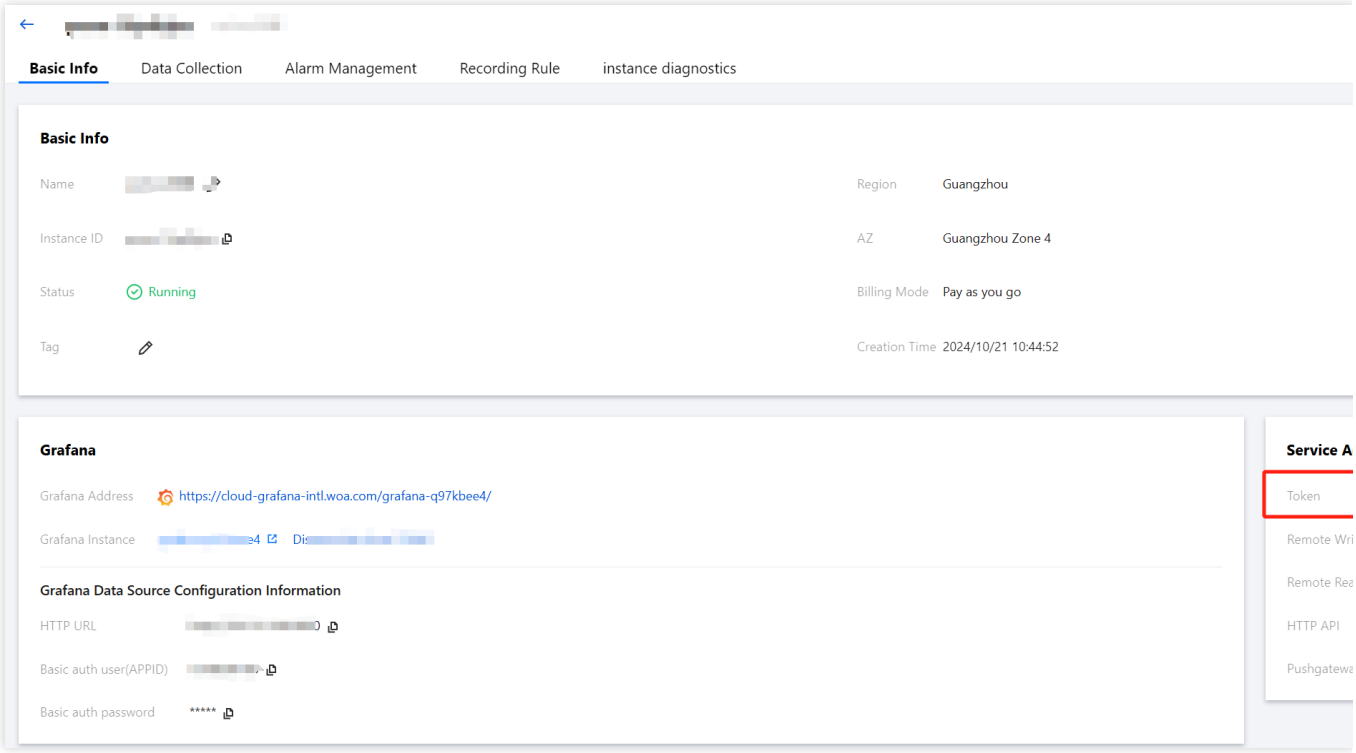
Replace IP:PORT with **Public IP and Port** of CLB as follows:

```
http://81.71.21.123:8080/api/v1/query?query=up
```

5. Access `http://81.71.21.123:8080/api/v1/query?query=up` .

username: Enter your root account ID (app ID).

password: Obtain the token from the basic information page of the instance.



As shown below, the public network address is successfully configured for the Prometheus instance.