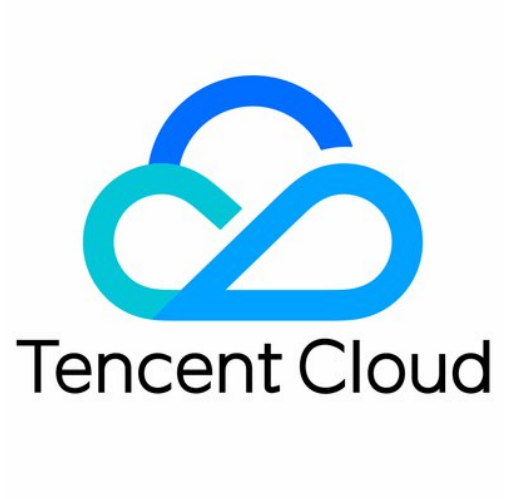
Custom Cloud Monitor

Quick Start

Product Documentation



Contents

Quick Start
Step 1: Confirm monitoring items

Last updated : 2017-08-09 17:00:04

In addition to the standard monitoring, CCM also monitors the metric you have interest in.

For example, a user needs to monitor the process CPU utilization on the CVM, and notify relevant admin of the process with 80% or higher CPU utilization (suppose that the user has configured the alarm receiving group with the ID of 8888 in Cloud Monitor console). In this example, we use API for API calling (suppose that user ID=AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA). You can also operate in CCM console.

Configurations required are as follows:

- Namespace (namespace): proc_monitor (process monitoring)
- Metric (metricName): proc_cpu (process CPU utilization)
- Dimension: (dimensionNames): proc_name (process name), ip (reporting machine IP)
- Statistical method (statistics): Take max value from all the reported data within statistical period
- Statistical period (period): Calculate data once every 5 minutes

Step 2: Create custom monitoring items

Last updated : 2020-06-07 19:04:21

1. Create Custom Namespace

In this example, we use API for creation. Users can also create in CCM console.

> Note: For more information on how to generate Signature parameter, please see API Authentication

Execute the following commands to create a namespace:

```
# curl -k https://monitor.api.qcloud.com/v2/index.php?Action=CreateNamespace
&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA
&Nonce=54579
&Timestamp=1457427062
&Region=gz
&namespace=proc_monitor
&Signature=K%2FX6J6hnjTIE25QK8klMZMJWDGk%3D
```

The following values will be returned:

```
# {"code": 0,"message": ""}
```

2. Create Custom Metric

Create a metric (proc_cpu) in the namespace (proc_monitor) you just created, and specify the dimension (proc_name, ip) structure that is already configured using the parameter dimensionNames.

In this example, we use API for creation. Users can also create in CCM console.

> Note: For more information on how to generate Signature parameter, please see API Authentication

```
# curl -k https://monitor.api.qcloud.com/v2/index.php?Action=CreateMetric
&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA
&Nonce=15945
&Timestamp=1457428021
&Region=gz
&namespace=proc_monitor
&metricName=proc_cpu
&metricCname=%E8%BF%9B%E7%A8%8Bcpu%E4%BD%BF%E7%94%A8%E7%8E%87
&unit=percent
&dimensionNames.0=proc_name
&dimensionNames.1=ip
&Signature=8w0Nwaxb6ZNh3ROPmHruaVz1meE%3D
```

The following values will be returned:

```
# { "code": 0, "message": ""}
```

3. Create Statistical Type for Metric

You can add multiple sets of statistical method and period using the parameter statisticsType. Here, we add one set: statistical period is 300 sec, and statistical method is max.

In this example, we use API for creation. Users can also create in CCM console.

> Note: For more information on how to generate Signature parameter, please see API Authentication

```
#curl -k https://monitor.api.qcloud.com/v2/index.php?Action=CreateMetricStatisticsType
&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA
&Nonce=13133
&Timestamp=1457428414
&Region=gz
&namespace=proc_monitor
&metricName=proc_cpu
&dimensionNames.0=proc_name
&dimensionNames.1=ip
&statisticsType.0.period=300
&statisticsType.0.statistics=max
&Signature=hcFFCJcHYiLW0PH%2F%2FuJ%2FgFeDRtY%3D
```

The following values will be returned:

```
# { "code": 0, "message": ""}
```

Step 3: Report and check data

Last updated : 2019-08-07 13:50:15

1. Report Data

Report the CPU utilization of processes disk_cleaner and daemon2 on machines 1.2.3.4 and 1.2.3.5 by following the method below.

GET method is used. The data encoded with urlencode is as follows:

```
Nonce=41718&Timestamp=1457429445&Region=gz&Namespace=proc_monitor&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC&Signature=s/aiEege8n
xOUh79rQ6WqzvEEMc=&Data=[{"dimensions": {"ip": "1.2.3.4", "proc_name": "disk_cleaner"}, "value": 30, "metricName": "proc_cpu"}, {"di
mensions": {"ip": "1.2.3.5", "proc_name": "daemon2"}, "value": 20, "metricName": "proc_cpu"}]
```

Note: For more information on how to generate Signature parameter, please see API Authentication

```
#curl http://receiver.monitor.tencentyun.com:8080/report.cgi?Nonce=41718
&Timestamp=1457429445
&Region=gz
&Namespace=proc_monitor
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC
&Signature=s%2FaiEege8nxOUh79rQ6WqzvEEMc%3D
&Data=[
{"dimensions":
{"ip":"1.2.3.4","proc_name":"disk_cleaner"},
"value":30,
"metricName":"proc_cpu"
},
{"dimensions":
{"ip":"1.2.3.5","proc_name":"daemon2"},
"value":20,
"metricName":"proc_cpu"
}
]
```

The following values will be returned:

```
# {"message": "OK", "code": 0}
```

2. View Data

In this example, we use API for creation. Users can also create in CCM console.

Check whether the monitoring data of a specific object is calculated properly by calling the API GetMonitorData. For example, check the data of the object ip=1.2.3.5&proc_name=daemon2: monitored after 17:35:00:

> Note: For more information on how to generate Signature parameter, please see API Authentication

```
#curl -k "https://monitor.api.qcloud.com/v2/index.php?Action=GetMonitorData
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC
&Nonce=34872
&Timestamp=1457431571
&Region=gz
&namespace=proc_monitor
&metricName=proc_cpu
&dimensions.0.name=proc_name
&dimensions.0.value=daemon2
&dimensions.1.name=ip
&dimensions.1.value=1.2.3.5
&period=300
&statistics=max
&startTime=2016-03-08+17%3A35%3A00
&Signature=FacKUqRPhqdEa%2FDvEHHAFAPKj8k%3D"
```

The following values will be returned:

```
{
"code": 0,
"message": "",
"metricName": "proc_cpu",
"startTime": "2016-03-08 17:35:00",
"endTime": "2016-03-08 18:05:00",
"period": "300",
"dataPoints": {
"ip=1.2.3.5&proc_name=daemon2": [
"20.0",
"90.0",
"90.0",
"90.0",
"90.0",
```

```
"90.0",
"90.0"
]
}
}
```

Step 4: Configure alarms

Last updated : 2019-08-07 13:53:01

1. Create Alarm Rules

Now we create an alarm rule based on the metric and statistical type: send an alarm about the process with 80% or higher CPU utilization for 2 statistical periods and the machine ip. In this example, we use API for creation. Users can also create in CCM console.

> Note: For more information on how to generate Signature parameter, please see API Authentication

```
#curl -k "https://monitor.api.qcloud.com/v2/index.php?Action=CreateAlarmRule
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3s00ssGQlC
&Nonce=14971
&Timestamp=1457430090
&Region=gz
&namespace=proc_monitor
&metricName=proc_cpu
&dimensionNames.0=proc_name
&dimensionNames.1=ip
&period=300
&statistics=max
&constancy=2
&threshold=80
&operatorType=>
&Signature=aGftupI7YXRRInk9JT9tru7FzKM%3D
&receiversId=8888"
```

The following values will be returned:

```
# { "code": 0, "message": "", "data": { "alarmRuleId": "policy-eqzqq79naz" } }
```

The cloud API returns an alarm rule ID: policy-eqzqq79naz, which is required for the further operations on the alarm rules. You can query this ID using DescribeAlarmRuleList in case you forget.

Note:

- You need to bind an alarm receiving group (receiversId) (ID is 8888) during configuration. If this parameter is left empty, the alarm message will not be sent to any user. You can use the API BindAlarmRuleReceivers to bind a specific monitoring object to receive the alarm SMS messages and emails.

- The parameter isWild is not specified in the above calling. A non-wildcard rule is created by default. The non-wildcard will become effective if you bind a specific alarm object. If isWild=1 is specified, a wildcard rule is created. The rule is valid for all the objects without the need to bind a specific object.

2. Bind Alarm Rule to Monitoring Object

In this example, we use API for creation. Users can also create in CCM console.

> Note: For more information on how to generate Signature parameter, please see API Authentication

```
#curl -k ”https:// monitor.api.qcloud.com/v2/index.php?Action=BindAlarmRuleObjects
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3s00ssGQlC
&Nonce=8573
&Timestamp=1457431999
&Region=gz
&alarmRuleId=policy-eqzqq79naz
&dimensions.0.name=ip
&dimensions.0.value=1.2.3.5
&dimensions.1.name=proc_name
&dimensions.1.value=daemon2
&Signature=wxreGK7XUZQtLluaKUbUAwbQbtI%3D”
```

The following values will be returned:

```
# { ”code”: 0, ”message”: ”” }
```

Since the rule we just created is a non-wildcard rule, here we bind the object ip=1.2.3.5&proc_name=daemon2 to the alarm rule. CCM will determine whether to send an alarm to the object and send an alarm to the receiving group (ID is 8888) when triggering the alarm rule.

Step 5: Make statistics by aggregation dimension

Last updated : 2017-07-24 17:31:39

If a user wants to analyze the maximum CPU utilization of a process instead of a specific IP, he/she can aggregate the ip dimension and only use proc_name for analysis. This can be achieved by creating aggregation statistics if the original reported data remains unchanged. In this example, we use API for creation. Users can also create in CCM console:

```
#curl -k "https://monitor.api.qcloud.com/v2/index.php?Action=CreateMetricAggeration
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC
&Nonce=56289
&Timestamp=1457433928
&Region=gz
&namespace=proc_monitor
&metricName=proc_cpu
&dimensionNames.0=proc_name
&statisticsType.0.period=300
&statisticsType.0.statistics=max
&Signature=3DeRgk4acf13QE7ecpUZfn4zkWc%3D"
```

The following values will be returned:

```
# {"code": 0, "message": ""}
```

Keep reporting the data, and view data of the specific aggregation dimension object (proc_name=xxx) after a period of time.

Step 6: Pull real-time data

Last updated : 2019-08-07 13:56:14

You can use the API GetMonitorData to view the data for a specified time range, or use the API GetMonitorRealTimeData to view the latest data of an object. For example, the real-time aggregation data in the previous step. In this example, we use API for creation. Users can also create in CCM console:

> Note: For more information on how to generate Signature parameter, please see API Authentication

```
#curl -k "https://monitor.api.qcloud.com/v2/index.php?Action=GetMonitorRealtimeData
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3s00ssGQlC
&Nonce=23034
&Timestamp=1457434224
&Region=gz
&namespace=proc_monitor
&metricName=proc_cpu
&dimensions.0.name=proc_name
&dimensions.0.value=daemon2
&period=300
&statistics=max
&Signature=mNyoxCKj8DRPdWqX%2Fw4fG%2BOCulA%3D"
```

You will get the latest real-time data as follows. The updateTime is the timestamp.

```
# {
"code": 0,
"message": "",
"data": {
"proc_name=daemon2": {
"value": 90,
"updateTime": "2016-03-08 18:55:00"
}
}
}
```