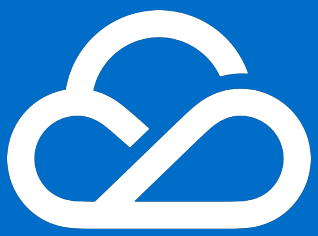
腾讯云自定义监控

快速入门

产品文档



腾讯云

【版权声明】

【商标声明】

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

步骤一：确定需要监控的配置项

自定义监控提供除标准监控外的、用户关心的指标监控。

假设用户需要对机器上的进程级CPU使用率进行监控，且CPU使用率超过80%的进程，需告警通知相关负责人（这里假设用户已经在云监控控制台中配置告警接收组，此告警接收组ID为8888)。本例中使用API进行接口调用（假设用户secretID=AKIDz8krbsJ5yKBZQpn74WFkmLPx3gnPhESA），您也可以通过自定义监控控制台进行操作。

需要使用的配置可以确定为：

- 命名空间(namespace)：proc_monitor（进程监控）
- 指标(metricName)：proc_cpu（进程级CPU使用率）
- 维度(dimensionNames)：proc_name(进程名)、ip(上报机器的IP地址)
- 统计方法(statistics)：取周期内所有上报数据的max值
- 统计周期(period)：每5分钟统计一次数据

步骤二：创建自定义监控配置项

1. 创建自定义命名空间

本例使用API进行创建，用户也可通过自定义监控控制台实现。

注：Signature参数的具体生成步骤请参见接口鉴权

执行以下命令创建命名空间：

```
# 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
```

得到的返回值为：

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

2. 创建自定义指标

在刚刚创建的名字空间（proc_monitor）中创建指标（proc_cpu），通过dimensionNames参数指明定好的维度（proc_name、ip）结构。

本例使用API进行创建，用户也可通过自定义监控控制台实现。

注：Signature参数的具体生成步骤请参见接口鉴权

```
# 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
```

得到的返回值为：

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

3. 为指标创建统计方式

可通过statisticsType参数同时添加多种统计方式和周期的组合。这里添加了一种：统计周期为300s，统计方式为取最大值max。

本例使用API进行创建，用户也可通过自定义监控控制台实现。

注：Signature参数的具体生成步骤请参见接口鉴权

```
#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
```

得到的返回值为：

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

步骤三：上报及查看数据

1. 上报数据

比如在1.2.3.4和1.2.3.5两台机器上，分别上报disk_cleaner和daemon2两个进程的cpu使用率，上报方法如下。

使用get请求，urlencode后，实际数据为：

```
Nonce=41718&Timestamp=1457429445&Region=gz&Namespace=proc_monitor&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC&Signature=s/aiEege8nxOUh79rQ6WqzvEEMc=&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"}]
```

注：Signature参数的具体生成步骤请参见接口鉴权

```
#curl http://receiver.monitor.tencentyun.com:8080/v2/index.php?Action=PutMonitorData
&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"
}
]
```

得到的返回值为：

{“message”：“OK”，“code”：0}

2. 查看数据

本例使用API进行创建，用户也可通过自定义监控控制台实现。

通过调用GetMonitorData接口，查看某个具体对象的监控数据是否已被正常统计。比如查看ip=1.2.3.5&proc_name=daemon2对象在17:35:00之后的数据：

注：Signature参数的具体生成步骤请参见接口鉴权

#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”

得到的返回值为：


```
{
  "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"
  ]
  }
}
```

步骤四：配置告警

1. 创建告警规则

下面针对刚刚的指标和统计方式，创建一个告警规则：对cpu使用率超过80%并持续2个统计周期(period)以上的进程和机器ip进行告警。本例使用API进行创建，用户也可通过自定义监控控制台实现。

注：Signature参数的具体生成步骤请参见接口鉴权

#curl –k “https://monitor.api.qcloud.com/v2/index.php?Action=CreateAlarmRule
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC
&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”

得到的返回值为：

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

这里云API返回了一个告警规则ID：policy-eqzqq79naz，后续很多针对告警规则的操作都需要这个ID。如果忘记了可以通过DescribeAlarmRuleList进行查询。

注意：

- 配置时需绑定告警接收组(receiversId)ID8888，如不填写此参数则不会发送告警信息到任何人。后续通过BindAlarmRuleReceivers接口绑定具体监控对象以接收告警短信和邮件。
- 上述调用没有指定isWild参数，默认创建的是一个非通配规则。非通配规则如要生效，则需要绑定一个具体的告警对象。若加上isWild=1，会创建一个通配规则，则无需绑定具体对象，该规则会对所有的对象生效。

2. 绑定告警规则和监控对象

本例使用API进行创建，用户也可通过自定义监控控制台实现。

注：Signature参数的具体生成步骤请参见接口鉴权

#curl –k “https:// monitor.api.qcloud.com/v2/index.php?Action=BindAlarmRuleObjects
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC
&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”

得到的返回值为：

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

由于之前创建的是非通配规则，这里将对象ip=1.2.3.5&proc_name=daemon2绑定到告警规则，自定义监控系统将针对该对象判断是否告警并在触发告警规则时向ID为8888的接收组发送告警。

步骤五：按照聚合维度统计

当用户想统计某个进程的最高cpu使用率而不需要具体到IP这么细的粒度时，可以将ip维度聚合掉，只根据proc_name来做统计。在原始的上报数据保持不变的情况下，可以通过创建聚合统计来完成这个目标。本例使用API进行创建，用户也可通过自定义监控控制台实现：

#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”

得到的返回值为：

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

继续保持数据上报，一定时间后即可查看具体的聚合维度对象(proc_name=xxx)的数据。

步骤六：拉取实时数据

除了使用GetMonitorData按照时间范围查看数据，还可以调用GetMonitorRealTimeData接口来查看某个对象当前最新的数据，比如要查看上一步中的聚合统计的实时数据。本例使用API进行创建，用户也可通过自定义监控控制台实现：

注：Signature参数的具体生成步骤请参见接口鉴权

#curl –k “https://monitor.api.qcloud.com/v2/index.php?Action=GetMonitorRealtimeData
&SecretId=AKIDlgRMo1j074b1l6nwReIvSk3sO0ssGQlC
&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”

得到以下最新的实时数据，其中updateTime为数据的时间戳：

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

```
}
}
```