Alibaba Cloud Cloud Monitor

Best Practices

Issue: 20190430

MORE THAN JUST CLOUD |

<u>Legal disclaimer</u>

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

- 1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
- 2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
- 3. The content of this document may be changed due to product version upgrades , adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
- 4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults " and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity , applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

- 5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified , reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates . The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
- 6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
-	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning informatio n, supplementary instructions, and other content that the user must understand.	• Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus , page names, and other UI elements.	Click OK.
Courier font	It is used for commands.	Run the cd / d C :/ windows command to enter the Windows system folder.
Italics	It is used for parameters and variables.	bae log list instanceid Instance_ID
[] or [a b]	It indicates that it is a optional value, and only one item can be selected.	ipconfig [-all -t]

Style	Description	Example
{} or {a b}	It indicates that it is a required value, and only one item can be selected.	<pre>swich {stand slave}</pre>

Contents

Legal disclaimer I
Generic conventions I
1 Create an alarm template1
2 Receive alarm notifications in a DingTalk group
3 Monitor the intranet12
4 Create an alert rule for pods in Container Service for
Kubernetes16
5 Best practice for querying monitoring data through APIs 21
6 Process ECS status change events

1 Create an alarm template

This topic illustrates how to use application groups and alarm templates to manage cloud resource alarm rules for various services more efficiently. These tactics are especially important for those who need to monitor and manage resources across several Alibaba Cloud products and regions and who need to modify alarm rules for these resources in a timely manner.

Purposes

- Configuring alarm rules for application groups rather than for single instances can improve efficiency by greatly reducing the time required to configure alarm rules.
 - By setting the resource range of an alarm rule to application group, your alarm rule will be effective for all resources within the target application group, and the number of resources monitored can expand as your services are scaled outward. After initial configuration, you can move specific resources into or out of the application group easily. You can also modify the alarm rule directly so to make changes effective to all instances within an application group.
 - Conversely, setting the resource range to instance will make your alarm rule effective for only one instance. Modifications to your alarm rule will also be effective for only one instance. As a consequence, supposing that you set all your alarm rules this way, as the number of your instances increase, managing alarm rules for these instances will become increasingly time consuming and difficult.
- · Using alarm templates can also reduce the time required to configure alarm rules.
 - The monitoring metrics and alarm thresholds of basic services, such as ECS, RDS , and SLB, are set to fixed values during alarm rule configuration. You can create alarm templates easily based on these configurations, and by creating alarm templates with your target metrics and condition thresholds, you can easily apply these templates to alarm rules you configure for an application group, making configuring rules easy even as your services scale outward. Using alarm templates also enables you to easily modify multiple alarm rules at the same time.

Procedures

The following case outlines the procedure that can be applied to the typical back-end services of an e-commerce company. This case serves to illustrate how you can create

application groups and use alarm templates to easily build a service monitoring and alarming system on the cloud, even for growing service requirements.

- 1. Create an alarm template named "EcommerceBackendAlarmTemplate".
 - a. Log on to the CloudMonitor Console.
 - **b.** From the left-side navigation pane, choose Alarm Templates > Alarms.
 - c. On the Alarm Templates page, click Create Alarm Template in the upper-right corner.
 - d. In the displayed dialog box, set the parameters in the Basic Info area.

Create Alarm Template	♣ Back to Ten	aplate List	
1 Basic Info			
* Template	Name :	EcommerceBackendAlarmTemplate	
Desc	ription :	Up to 64 characters is allowed.	

e. In the View Alarm Rules area, click Add Alarm Rule to add the required alarm rules to the alarm template.

2 View Alarm Rules				
* Product :	ECS X ApsaraDB for RDS X		•	
	ECS •			
	Rule Description : (Agent) Host.cpu.total(Recomme	 ▼, 1mins ▼ it alarms1times α ▼ 	>= •	9/0
	Rule Description : (Agent) Host.cpu.total(Recomme	▼, 1mins ▼ it alarms1times α ▼	>= •	% Delete
	Rule Description : (Agent) Host.cpu.total(Recomme	Timins The second s	>= *	% Delete
	+ Add Alarm Rule			
	ApsaraDB for RDS -			
	Rule Description : Average Data Size of Sync Block.	 Imins ▼ it alarms1times ct ▼ 	>= •	Bytes
	Rule Description : Average Data Size of Sync Block.	▼ 1mins ▼ it alarms1times α ▼	>= •	Bytes Delete (A same template already exists.)
	Rule Description : Average Data Size of Sync Block.	Imins ▼ it alarms1times α ▼	>= •	Bytes Delete (A same template already exists.)
	Rule Description : Average Data Size of Sync Block.	Imins ▼ it alarms1times α ▼	>= *	Bytes Delete (A same template already exists.)
	+ Add Alarm Rule			

f. Click OK.

- 2. Create an alarm contact and an alarm contact group.
 - a. Log on to the CloudMonitor Console.
 - b. From the left-side navigation pane, choose Alarms > Alarm Contacts.
 - c. On the Alarm Contact Management page, click Create Alarm Contact in the upper-right corner. In the displayed dialog box, enter your phone number and email.

To ensure that you can receive and verify alarm notifications in a timely manner , the system will send verification codes to your phone and email.

- d. Click the Alarm Contact Group tab.
- e. In the upper-right corner, click Create Alarm Contact Group.
- f. In the displayed dialog box, enter the group name and select the contacts that you want to add to the group.

- 3. Create an application group and apply the alarm template. Here, we create an application group named "InventoryManagementOnlineEnvironment" and use the created alarm template "EcommerceBackendAlarmTemplate".
 - a. Log on to the CloudMonitor Console.
 - b. In the left-side navigation pane, click Application Groups.
 - c. On the Application Groups page, click Create Group in the upper-right corner.
 - d. In the Basic Information area, set Product Group Name and Contact Group.

The contact group is the alarm contact group for receiving alarm notifications.

te Group	
Basic Infomation	
Product Group Name	
InventoryManagementOnlineEnvironment	
Contact Group	
Default Contact Group Quickly create a contact group	
MonitorAlarm	
Select Template	
EcommerceBackendAlarmTemplate Create Alarm Template	
 Warning (Phone+Email ID+ Ali WangWang+DingTalk Robot) Info (Email ID+ Ali WangWang+DingTalk Robot) Initialize Agent Installation @ 	
Add Instance dynamically	
 Ø Dynamic rules for ECS instances Ø Dynamic rules Ø All rules Ø Any rule 	
() instance created in future according with this rule would be added to group	
Instance Name Contain Enter(case insensitive)	
+Add Rules	
Add Product	

e. In the MonitorAlarm area, set Select Template and Notification Methods and enable Initialize Agent Installation. The selected template is used to initialize alarm rules for the instances in the group. After new instances are created, a CloudMonitor agent will be automatica lly installed to collect monitoring data.

f. In the Add Instance dynamically area, add at most three dynamic rules with the relationship of AND or OR. Then, click Add Product to customize dynamic rules for RDS and SLB.

Generally, the cloud resources used for inventory management are a server, database, and SLB resource. You can customize dynamic rules to add ECS instances. An ECS instance name can be matched through the condition of Contain, start with, or end with. The instances conforming to the dynamic rules will be added to the specified application group (including instances to be created in the future).

g. Click Create Application Group.

The instances conforming to the dynamic rules are added to the created application group, which can be viewed on the Basic Information page of the application group.

2 Receive alarm notifications in a DingTalk group

This topic discusses how you can receive alarm notifications in a DingTalk group. Among the alarm notification methods available on CloudMonitor, one option is to send alarm notifications to a DingTalk group.

To add this notification method to existing alarm rules, you only need to add the webhook address of the DingTalk robot to your contacts. You do need to modify any other configurations in your alarm rules.

After the webhook address is added to an existing contact, all alarm notifications that were previously sent by email or SMS can also be received by the DingTalk group.

Create a DingTalk robot (desktop version)

1. Open the DingTalk group in which you want to receive alarm notifications.



2. In the upper-right corner of the group page, click the Group Settings icon and choose ChatBot.



3. In the ChatBot window, click Custom.



1 0 Robot details \times < Ď & ••• up Settings 2 Custom L 4 Brief Info: Use the Chatbot API to push any service message that you need to -DingTalk Message preview: VIP monitoring alarm ChatBot (d) The failure rate of sending message is higher than 5%, module 202, network type 4G. @Yinan Urgent task. Plan reminder ChatBot [P3][Online][Early plan] - The number of the mobile home tab shows downgrade - Operator: Xumo Add Cancel Ŧ

4. In the Robot details window, click Add.

5. In the Add Robot window, enter a name for the robot. For example, you can name the robot "CloudMonitor alarm notification". Once you have entered a name, click Finished.

Add Robot		×					
	Eait profile picture						
ChatBot Name :	CloudMonitor alarm notification						
Add to Group :							
	Cancel						

6. Click Copy and then click Finished.

		N. – – ×
.	Add Robot	×
Ŧ		up Settings
8		
٩	1. Add robot~	4
•	2. Set up webhook, dick setting instruction and check how to make robot effective	ad
	webhook : https://oapi.dingtalk.com/robot/send?access_token= Copy	
==		
	Finished Setting ins	
		Send

Add a DingTalk robot to your alarm contacts

You can add the webhook address of the created DingTalk robot to your alarm contacts so that you can receive alarm notifications from the DingTalk group where you created the robot.

- 1. Log on to the CloudMonitor Console.
- 2. From the left-side navigation pane, choose Alarms > Alarm Contacts.

Ala	Alarm Contact Management								
Ala	arm Contacts	Alarm Contact Group							
All	Ŧ	Enter the name, phor	ne number, email or Ali Wangwang of	the Search			Refresh	Create Alarm Contact	
	Name	Phone	Email	Ali WangWang	DingTalk Robot	Alarm Group		Actions	
	099	13426206595	kunudangsikalibaba-ins.com		https://wapi.dingtalk.com/robut/send?access_token=7d4951Se8ebf 21106488x9xc4xb3d247771305x5281L5x620x4968f15268da19	anna, 204825.		Edit Delete	
	hq	13521831-079	changjie.hcj@albaba-inc.com			hqi		Edit Delete	

3. On the Alarm Contact Management page, find the target contact and click Edit. In the Set Alarm Contact window, add the webhook address of the DingTalk robot.

Alternatively, click Create Alarm Contact to create a contact to use the DingTalk robot.

Set Alarm Contact			\times
Name:	The name must be 2-40 characters, can include English letters, numbers, ., and underscores, and should start with a Chinese or English character.		
Phone:	13426200585	Send verification code.	
Verification code:	Fill in the phone verification code.		
Email ID:	kun.dangsikalibaba-inc.com	Send verification code.	
Verification code:	Fill in the E-mail verification code.		
Ali WangWang:			
DingTalk Robot:	https://oapi.dingtalk.com/robot/send?access_token=7d49515e8ebf2: How to get the DingTalk robot address		
		Save	Cancel

3 Monitor the intranet

This topic describes how to use CloudMonitor to monitor the intranet and internal VPC services, specifically helping you closely manage the availability of ECS services, the connectivity of ECS to RDS and Redis instances, and the responsiveness of SLB instances in a VPC.

Principle

Before you can begin to monitor the intranet, you will need to install a CloudMonit or agent on your server and create monitoring tasks in the CloudMonitor console , selecting the server on which the agent is installed as the detection source, and configure the target URL or port. The principle behind these prerequisites and the procedure that follows is to allow the detection source to send an HTTP request or a Telnet request through the agent and collect the response time and status codes, so that this data can be sent to CloudMonitor for alarm and visualization monitoring purposes.



Monitor the intranet

- Prerequisites
 - The CloudMonitor agent has been installed on the detection source.
 - You have created an application group and added the detection source to the group.
- Procedure
 - 1. Log on to the CloudMonitor Console.
 - 2. In the left-side navigation pane, click Application Groups.
 - 3. On the Application Groups page, click the application group for which you want to create an availability monitoring task.
 - 4. In the left-side navigation pane, click Availability Monitoring.
 - 5. In the upper-right corner, click Create Configuration.
 - To monitor the responsiveness of a local ECS process in a VPC, select the target ECSs to be monitored as Target Server and enter the addresses in *localhost : port / path* format as Detection Target.
 - To monitor the responsiveness of the SLB in a VPC, select an ECS that is located in the VPC as Target Server and enter the SLB address as Detection Target.
 - To monitor the responsiveness of the RDS or Redis used in the ECS backend in a VPC, add the RDS or Redis in the VPC to the application group, select

the corresponding ECS as Target Server, and select RDS or Redis instances as Detection Target.

Create Availability Monitoring		×
1 Monitoring Configurations		
* Task Name :		
	Enter 4 to 50 characters. Only English letters, numbers, underlines, and Chinese characters are allowed.	
* Target Server :	🖉 All	
	ESS-asg-test001 CMS-grafana-Test ESS-asg-cmstest AegisTest-1 VerieTest 2	
* Detection Type :	URL or IP address	
* Detection Target :	HTTP(S) V E.g: http://localhost:8081/check_health.htm	
* Request Method :	HEAD GET POST	
Advanced Configuration 🔻		
2 Alarm Configuration		
Status Code :	Continue for v greater than v 400	Status Code Description
Response Time :	Continue for v greater than v 500	millisecond
Notification Method :	Text Message + Email + DingTalk + TradeManager	
	Email + DingTalk + TradeManager	
Advanced Configuration 🔻		
	The detection period is 1 minute. When the above alarm configurations are met, any server will send an alarm notification to the contact group associated with the application group.	
		OK Cancel

6. Click OK. Then, you can view the detection results in the corresponding monitoring chart of the task. If detection fails, you will receive an alarm notification.

de	no	n Group								
6	Features 🔗 How to monit	tor local servi	ice availability							
Ente	er a task name to perform a	ı fuzzy query		Search					$\mathcal C$ Refresh	Create Configuration
	Task Name/Task ID	Status	Detection Type	Detection Target	Unhealthy Hosts	Unhealthy Agents	Hosts	Availability 🙆	Average latency	Actions
	demo / 435317	Enable	TELNET	telnet://rm-m5eu2e74zixpy14v1.sqlserver.rds.aliyuncs.co m:3433	7 unit(s)	0 unit(s)	6 unit(s)	0.00%	127303 millisecond	Monitoring Charts Disable Modify Delete
	BatchDelete BatchEna	able Ba	atchDisable					Total	1 10 •	< 1 > »

7. Click Monitoring Charts of a task to view the monitoring details.



4 Create an alert rule for pods in Container Service for Kubernetes

This topic describes how to create an alert rule for one or more pods in Container Service for Kubernetes.

Background information

CloudMonitor provides an additional alert function for Container Service for Kubernetes. This function monitors metrics such as CPU utilization and inbound bandwidth of Container Service, providing you with information about the usage of Container Service. After you deploy Container Service, CloudMonitor automatica lly involves Container Service into its monitoring system. You can log on to the CloudMonitor console and access the Container Services for Kubernetes page to view detailed monitoring data. After you configure an alert rule for a metric, you can receive alert notification when the metric data exceeds the defined threshold.

Prerequisites

We recommend that you perform the following operations before you create an alert rule for Container Service for Kubernetes: 1. Deploy Container Service. 2. Create application groups, alert contacts, and alert contact groups in the CloudMonitor console.

Procedure

Precautions

- Monitoring data is stored for up to 31 days.
- You can view the monitoring data for up to 14 consecutive days.

Create an alert template for Container Service for Kubernetes

- 1. Log on to the CloudMonitor console.
- 2. In the left-side navigation pane, choose Alarms > Alarm Templates. The Alarm Templates page is displayed.

3. Click Create Alarm Templates in the upper-right corner. The Create Alarm Template page is displayed.

Create Alarm Template			×
Basic Infomation			
 Template Name 			
The name must be within 30 characte	rs and can contain numbe		
Description			
Up to 64 characters is allowed.	<i>h</i>		
Rule Rules such as heartbeat alarm in alarm	ı template have been migrated to event mo	nitoring. Introduction to Cloud Products Events	
Kubernetes 🗸 ·····			
Rule Name	Rule Description	Resource Description	
+Add Rules			
Products			
			Add Cancel

- 4. Configure template information: Set the service name to Kubernetes and configure related metrics.
- 5. Click Add.

Apply the template to a Kubernetes application group

- 1. Log on to the CloudMonitor console.
- 2. In the left-side navigation pane, choose Alarms > Alarm Templates. The Alarm Templates page is displayed.

3. Click Apply to Group in the Actions column corresponding to the just-created alert template for Container Service for Kubernetes.

Apply Template to Group		\times
▲ Note		
Select a group	Select -	
Muted	24 h 🔹	
Effective Period	00:00 • 23:59 •	
HTTP CallBack	for example: http://alart.aliyun.com:8080/callback	
Option	Group instance priority Template instance precedence	
When an alarm template is applied, if there is no such instance in the group, the rules in the		
	template are ignored.	
	OK Clos	se

4. Select the group for which you want to create an alert rule, and click OK.

Subsequent operations

Alert notification for the Kubernetes application group is automatically sent to Default Contact Group. If you want all alerts for the Kubernetes application group to be sent to the same contact group, directly change the contacts in Default Contact Group.

Directly modify a contact group

- 1. Log on to the CloudMonitor console.
- 2. In the left navigation pane, choose Alarms > Alarm Contacts. The Alarm Contacts page is displayed.
- 3. Click the Alarm Contact Group tab. The Alarm Contact Group tab is displayed.

Alarm Contact M	lanagement						
Alarm Contacts	Alarm Contact Group						
						Refresh	Create Alarm Contact Group
83583					Subscribe V	Weekly Report:	► × ▲
Name		Phone	Email	Ali WangWang	DingTalk Robot		Actions
****		-	-	10000	-		Edit Delete
214		1102020000	kasamplatistar incom	-	https://ospi.dingtalk.com/tobol/wend? access_tokes=764951568ebf21106a88a9cc4b53c247771305d529b1	58520+4960*15268	Edit Delete
100.004		0401030	ula (LANK) (LANK) NORT	-			Edit Delete

4. Click the modification icon. On the Edit Group page that appears, change the contacts and click OK.

If you want alerts for different Kubernetes applications to be sent to different contact groups, you must modify the associated alert contact groups for each Kubernetes application group.

Modify the associated contact group of an application group

- 1. Log on to the CloudMonitor console.
- 2. In the left-side navigation pane, click Application Groups. The Application Groups page is displayed.
- 3. Find the Kubernetes application group and click the group name. The group details page is

displayed.	<	k8s-cb2391c1d62cc433dafd435984e3d9792-worker ← to Application Group	
	Group Resource	Basic Information	
	Dashboards	Product Group Name: k8s-cb2391c1d62cc433dafd435984e3d9792-worker	
	Fault List	Contact Group: ② 云账号报警联系人	
	Event Monitor	Group Instances	
	Availability Monitor	ECS I Add Product	
	Log Monitoring		
	Custom Monitoring	Enter	
	Alarm Logs	Instance Name Health Status 🗞 Resource Description	CP
	Alarm Rule	node=0002+k8s-for-cs- cb2391c1062ce433dafa935994e3d9792 OK 192-598.0.25 i-2zesqR2pwopW80/kpwgb	2.5

4. Click the modification icon. On the page that appears, change the contact group and click OK. All alerts for the Kubernetes application group will be sent to the new alert contact group.

You can use OpenAPI to modify the contact groups of multiple Kubernetes applicatio n groups.

Modify contact groups through OpenAPI

- 1. Log on to OpenAPI Explorer.
- In the left-side cloud service list, click CloudMonitor. In the search box, enter UpdateMyGr oups. Click the API in the search result. The API call page is displayed.
- 3. Set GroupId to the group ID to be modified.

4. Set ContactGro ups to the contact group to which alert notification is to be sent.

5. Click Initiate a Call.

5 Best practice for querying monitoring data through APIs

Background information

Large-sized enterprises typically have their own O&M systems. When moving businesses to the cloud, these enterprises must integrate the cloud resource monitoring data with their existing systems. This topic describes how to use CloudMonitor APIs to query the monitoring data of various services and integrate Apsara Stack monitoring data into your existing systems.

Prerequisites

Before you query monitoring data through APIs, we recommend that you learn about the three types of APIs provided by CloudMonitor to query monitoring data:

- QueryProjectMeta: queries which services can be monitored by CloudMonitor. For more information, see *#unique_8*.
- QueryMetricMeta: queries which metrics are available for a monitored service. For more information, see *#unique_*9.
- QueryMetricList and QueryMetricLast: query monitoring data by service and metric. For more information, see *#unique_10* and *#unique_11*.

Procedure

Precautions

- QueryMetricList and QueryMetricLast allow you to query data for a certain metric for all your instances. To query data for multiple metrics, you can create multiple threads, or create a single thread to cyclically obtain data of different metrics.
- QueryMetricList supports up to 20 queries per second (QPS) and QueryMetricLast supports up to 30 QPS.
- QueryMetricLast is applicable to scenarios where you need to regularly obtain the most recent monitoring data. The time window automatically slides forward. The latest data is obtained in each period.
- There may be a gap between the occurrence of an event and the time when it is monitored. The gap varies with service. We recommend that you extend the time window to 5 to 10 minutes when using QueryMetricLast to query the latest data.

- Data that is obtained every few seconds is stored for 7 days. Data that is obtained every few minutes is stored for 31 days.
- You do not need to specify Dimensions to query the data of all your instances.

Case study

The following example demonstrates how to use QueryMetricLast to query the latest monitoring data and use QueryMetricList to query the monitoring data in a specified time range.

```
undefined
package
          main
import (
   " encoding / json "
   " testing "
   " github . com / aliyun / alibaba - cloud - sdk - go / services /
cms "
)
// TestQueryM etricLast
func
       TestQueryM etricLast ( t * testing . T ) {
       // For more informatio n about how
                                                   to
                                                         obtain
  the
        AccessKey pair, see
                                 https :// yq . aliyun . com /
articles / 608581 .
    cmsClient , _ := cm
    "< Specify the
    ""</pre>
                    := cms . NewClientW ithAccessK ey (
                                                         sure ,
                          region. If you are not
        cn - hangzhou >",
                                            // Region
                                                        ID
enter
       "< your access
                          key
                               id >",
                                                    // AccessKey
ID
       "< your
                 Access
                          Key
                               Secret >") // AccessKey
                                                         Secret
   // Obtain the
                     latest monitoring data
                                                 for
                                                     the
specified
           instance in the specified time
                                                  range .
    request := cms . CreateQuer yMetricLas tRequest ()
   // Use
            QueryProje ctMeta
                                and QueryMetri cMeta
                                                         to
         project and metric.
obtain
    request . Metric = " cpu_total "
   // Period is set to 60 , indicating that monito
ata is obtained every 60 seconds . The value
                                                        monitoring
  data
                                                             of
period
         varies
                  with
                        metric . Most
                                         metrics are
                                                         polled
                                                                 at
  a 60 - second period.
    request . Period = " 60 "
           start time of query data
   // The
    request . StartTime = " 2018 - 11 - 29
                                             11 : 00 : 00 "
          end time of query data
   // The
    request . EndTime = " 2018 - 11 - 29
                                           12 : 00 : 00 "
   // The number of response data records displayed
                                                                 per
  page for this query. Up to 1,000
                                                  data
                                                         records
    be returned for each
request . Length = " 100 "
can be returned
                                  query .
   // The page cursor . If
                                the returned
                                                  result
                                                          contains
  cursor != "", the current
You must use cursors to
                                 page is not
                                                 the
                                                         last
                                                               page
               use cursors to query
                                            the
                                                  next
                                                         page .
    request . Cursor = ""
    t. Log (" start query ")
```

```
for {
         response , _ := cmsClient . QueryMetri cLast ( request )
// response code
          if response . Code ! = " 200 " || ! response . IsSuccess
 () {
 t . Fatalf (" code :% s , success :% v , msg :% s ,
httpstatus :% d , response :% v \ n ", response . Code , response
. IsSuccess (), response . Message , response . GetHttpSta tus
 (), response )
         }
          var
                datapoints [] map [ string ] interface {}
          err
                := json . Unmarshal ([] byte ( response . Datapoints ),
 & datapoints )
          if err ! = nil {
               t . Fatal ( err )
         }
          t. Logf (" batch count % d \ n ", len ( datapoints ))
for _, v := range datapoints {
   t. Log ( v )
         }
   // If the returned result contains cursor =="", the
current page is the last page. Exit the loop and
end this round of query.
    if response. Cursor == "" {
       t.Log (" end of loop ")
               break
         }
       //\ If there are more pages , assign the cursor the query condition for the next page in the
 as
 next
         request .
          request . Cursor = response . Cursor
          t . Logf (" cursor :% s \ n ", response . Cursor )
    }
    t . Log (" finished ")
}
// TestBatchQ ueryMultiI nstance
// You can use QueryMetri cList to query the monitoring
data of a specified instance in a specified time
 range .
// You can specify multiple instances
                                                         for batch guery.
// To obtain the monitoring data for multiple instances
   over a specified time range, you can specify
 multiple instances for a query. Up
                                                         to
                                                               10 instances
 can be specified at a
                                     time .
 func TestBatchQ ueryMultiI nstance ( t * testing . T ) {
 // For more informatio n about how to obtain the
 AccessKey pair, see <a href="https://yq.aliyun.com/articles/">https://yq.aliyun.com/articles/</a>
 608581
         Client , _ := cms . NewClientW ithAccessK ey (
"< Specify the region . If you are not
      cmsClient ,
                                                                       sure ,
          cn - hangzhou >",
                                                       // Region ID
 enter
                                key id >",
         "< your access
                                                                // AccessKey
 ID
         "< your Access
                                Key Secret >") // AccessKey Secret
```

// Query the monitoring data for the specified instance over the specified time range .
 request := cms . CreateQuer yMetricLis tRequest () // Use QueryProje ctMeta and QueryMetri cMeta to obtain
project and metric . request . Project = " acs_ecs_da shboard " request. Metric = "cpu_total" // Period is set to 60, indicating that monitoring data is obtained every 60 seconds. The value of eriod varies with metric. Most metrics are polled at period period varies with metric. Most metrics are potted a 60 - second period. // The start time of query data request. StartTime = " 2018 - 11 - 25 11 : 00 : 00 " // The end time of query data request. EndTime = " 2018 - 11 - 29 12 : 00 : 00 " // The number of response data records displayed page for this query. Up to 1,000 data records can be returned for each query. request. Length = " 10 " request. Dimensions = "[{' instanceId ':' AY14061311 per request . Length = " 10 "
request . Dimensions = "[{' instanceId ':' AY14061311
3321409088 '},{' instanceId ':' AY14061311 320829774f '},{'
instanceId ':' AY14061311 330190512d '}]"
// The page cursor . If the returned result contains
cursor != "", the current page is not the last page
. You must use cursors to query the next page .
request . Cursor = "" t. Log (" start query ") for { response , _ := cmsClient . QueryMetri cList (request)
// Response code if response . Code ! = " 200 " || ! response . IsSuccess () { t . Fatalf (" code :% s , success :% v , msg :% s , httpstatus :% d , response :% v \ n ", response . Code , response . IsSuccess (), response . Message , response . GetHttpSta tus (), response) } datapoints [] map [string] interface {} var err := json . Unmarshal ([] byte (response . Datapoints), & datapoints) if err ! = nil { t . Fatal (err) } t . Logf (" batch count % d \ n ", len (datapoints)) for _ , v := range datapoints { t.Log(v) } // If the returned result contains cursor =="", the current page is the last page. Exit the loop and end this round of query. if response .Cursor == "" { t. Log ("end of loop") break } // If there are more pages , assign the cursor the query condition for the next page in the as next request. request . Cursor = response . Cursor

```
t.Logf (" cursor :% s \ n ", response . Cursor )
}
t.Log (" finished ")
}
```

6 Process ECS status change events

This topic describes how CloudMonitor automatically processes ECS status change events by using MNS message queues.

Overview

An ECS instance status change event is triggered when the instance status changes . Specifically, a status change event can indicate changes resulting from operations on the console, the usage of APIs or SDKs, automatic scaling, detection of overdue payments, system exceptions, and more.

To automate the processing of ECS status change events, CloudMonitor provides two methods: function calculation formulas and MNS message queues. This topic describes three best practice cases that use MNS message queues.

Preparations

- · Create a message queue.
 - 1. Log on to the <u>MNS Console</u>.
 - 2. On the Queue List page, select the target region, and click Create Queue in the upper-right corner.

New Queue		×
* Queue Name 💿 :		
* Region :	China (Hangzhou)	
Long-polling Wait Time (s) 📀 :		
Invisibility Timeout (s) 💿 :		
Maximum Message Size (Byte) 💿 :		
Message Retention Period (s) 📀 :		
Message Delay (s) 💿 :		
Enable Logging :		
	OK Can	cel

3. In the New Queue dialog box, enter the queue name (for example, ecs-cmsevent) and other required information, and then click OK.

- · Create an alarm rule for status change events.
 - 1. Log on to the CloudMonitor Console.
 - 2. In the left-side navigation pane, click Event Monitoring.
 - 3. Switch to the Alarm Rules tab page, and then click Create Event Alerts.

Create / Modify Event Alerts

Basic Infomation

Alarm Rule Name

Combination of alphabets, numbers and unders

Event alert

Evont	Tuno
Eveni	I VDe

System Event Ocustom Event

Product Type

ECS

Event Type

StatusNotification 🗙

Event Level

All Levels 🗙

Event Name

All Events 🗙

Resource Range

All Resources Opplication Groups

Alarm Type

Alarm Notification	n Notification
--------------------	----------------

Contact Group	Delete
GPU监控	•

Notification Method

Warning (Message+Email ID+Ali WangWar

+Add

- MNS queue
- Function service (Best Practises)

URL callback



4. In the Basic Information area, enter a name for the alarm rule, for example, ecstest-rule.

5. In the Event alert area, set the parameters as follows:

- Set Event Type to System Event.
- Set Product Type to ECS and Event Type to StatusNotiifcation, and set other parameters as needed.
- If Resource Range is set to All Resources, change events of any resource will trigger notifications. If Resource Range is set to Application Groups, only change events of the resources within the specified group will trigger notifications.
- 6. In the Alarm Type area, select MNS queue, and then specify Region and Queue (for example, ecs-cms-event).
- 7. Click OK.
- Install Python dependencies.

The following code is tested in Python 3.6. You can use other programming languages, such as Java, as needed.

Use PyPi to install the following Python dependencies:

- aliyun-python-sdk-core-v3 of 2.12.1 or later
- aliyun-python-sdk-ecs of 4.16.0 or later
- aliyun-mns of 1.1.5 or later

Procedure

CloudMonitor sends all status change events of ECS instances to MNS. You can then obtain the notifications from MNS and process them by running code. The following practice sections overview a complete tutorial of the preceding methods.

Practice 1: Records of all ECS creation and release events

Currently, you cannot query instances that have been released on the ECS console . If you need to perform these queries, you need to record the life cycle of all ECS instances in your own database or log through an ECS status change event. Specifical ly, whenever an ECS instance is created, a Pending event will be sent, and whenever an ECS instance is released, a Deleted event will be sent. You can record these two events by performing the following steps: 1. Create a *Conf* file, which must include the MNS endpoint, AccessKeyId and AccessKeySecret of your Alibaba Cloud account, region ID (for example, cnbeijing), and the MNS queue name.

Note:

To view the MNS endpoint, you can log on to the MNS console, and click Get Endpoint on the Queue List page.

```
class Conf :
    endpoint = ' http ://< id >. mns .< region >. aliyuncs . com
/'
    access_key = '< access_key >'
    access_key _secret = '< access_key _secrect >'
    region_id = ' cn - beijing '
    queue_name = ' test '
    vsever_gro up_id = '< your_vserv er_group_i d >'
```

2. Use the MNS SDK to compile an MNS client to receive MNS messages.

```
# -*- coding : utf - 8 -*-
import
          json
                                   import
                                              MNSExcepti onBase
from
      mns . mns_except ion
import
          logging
from
      mns . account
                         import
                                    Account
from . import
                    Conf
        MNSClient ( object ):
class
          __init__ ( self ):
     def
         self . account = Account ( Conf . endpoint , Conf .
access_key , Conf . access_key _secret )
    self . queue_name = Conf . queue_name
    self . listeners = dict ()
          regist_lis tener ( self , listener , eventname ='
     def
Instance : StateChang e '):
                                  self . listeners . keys ():
         if
               eventname in
              self . listeners . get ( eventname ). append ( listener
)
         else :
              self . listeners [ eventname ] = [ listener ]
            run ( self ):
     def
                 = self . account . get_queue ( self . queue_name )
          queue
                 True :
         while
              try :
                  message = queue . receive_me ssage ( wait_secon
ds = 5 )
                  event = json . loads ( message . message_bo dy )
if event [' name '] in self . listeners :
    for listener in self . listeners . get (
event [' name ']):
                           listener . process ( event )
                  queue . delete_mes sage ( receipt_ha ndle =
message . receipt_ha ndle )
              except
                                               as
                       MNSExcepti onBase
                                                     e :
                        e . type == ' QueueNotEx ist ':
                  if
```

The preceding code is used only to pull MNS messages and delete the messages after the listener consumption message is called.

3. Register a listener to use a specified event. When this listener determines that it has received a Pending or Deleted event, it prints a row in the log file.

```
# -*- coding : utf - 8 -*-
import logging
from .mns_client import BasicListe ner

class ListenerLo g ( BasicListe ner ):
    def process ( self , event ):
        state = event [' content '][' state ']
        resource_i d = event [' content '][' resourceId ']
        if state == ' Panding ':
            logging . info ( f ' The instance { resource_i d }
state is { state }')
        elif state == ' Deleted ':
            logging . info ( f ' The instance { resource_i d }
state is { state }')
```

The following Main function can also be used:

```
mns_client = MNSClient ()
mns_client . regist_lis tener ( ListenerLo g ())
mns_client . run ()
```

In your actual scenario, you can store events in your database or use SLS to facilitate search and audit tasks at a later date.

Practice 2: Automatic restart of ECS servers

In some scenarios, ECS servers may shut down unexpectedly. In this case, you need to set automatic restart for the servers.

Use the MNS client in Practice 1 and create a new listener. Then, when the listener receives a Stopped event, the listener executes a Start command on the target ECS server.

-*- coding : utf - 8 -*-

```
import logging
         aliyunsdke cs. request. v20140526
from
                                                                 import
                                                                             StartInsta
nceRequest
from
         aliyunsdkc ore . client
                                               import
                                                            AcsClient
        . mns_client import BasicListe
from
                                                          ner
from . config
                      import Conf
class
           ECSClient ( object ):
           __init__ ( self , acs_client ):
self . client = acs_client
     def
    # Start
                 the ECS
                                   instance
           start_inst ance ( self ,
logging . info ( f ' Start
                                                   instance_i d ):
     def
                                                   instance { instance_i d
} ....')
           request = StartInsta nceRequest . StartInsta nceRequest
()
           request . set_accept _format (' json ')
request . set_Instan ceId ( instance_i
                                                                    d )
           self . client . do_action_ with_excep tion ( request )
           ListenerSt art ( BasicListe ner ):
    __init__ ( self ):
acs_client = AcsClient ( Conf . access_key ,
class
     def
                                                                              Conf .
access_key _secret , Conf . region_id )
        self . ecs_client = ECSClient ( acs_client )
           process ( self , event ):
detail = event [' content ']
instance_i d = detail [' resourceId ']
if detail [' state '] == ' Stopped ':
    self . ecs_client . start_inst ance ( instance_i d )
     def
```

In your actual scenario, after the Start command is executed, you will receive Starting, Running, or Stopped event notifications. In this case, you can proceed with the procedure upon command execution for more detailed O&M with the help of a timer and a counter.

Practice 3: Automatic removal of preemptible instances from SLB before they are released

A release alarm event will be sent five minutes before a preemptible instance is released. During these five minutes, you can run some processes without your services being interrupted. For example, you can manually remove the target preemptible instance from the backend SLB server.

Use the MNS client in Practice 1 and create a new listener. Then, when the listener receives the preemptible instance release alarm, the listener calls an SLB SDK.

-*- coding : utf - 8 -*from aliyunsdkc ore .client import AcsClient
from aliyunsdkc ore .request import CommonRequ est
from .mns_client import BasicListe ner

```
from . config
                   import
                              Conf
       SLBClient ( object ):
class
    def __init__ ( self ):
         self . client = AcsClient ( Conf . access_key , Conf .
access_key _secret , Conf . region_id )
         self. request = CommonRequ est ()
         self . request . set_method (' POST ')
         self . request . set_accept _format (' json ')
self . request . set_versio n (' 2014 - 05 - 15 ')
self . request . set_domain (' slb . aliyuncs . com ')
         self . request . add_query_ param (' RegionId ', Conf .
region_id )
    def remove_vse rver_group _backend_s ervers ( self ,
vserver_gr oup_id , instance_i d ):
         self . request . set_action _name (' RemoveVSer verGroupBa
ckendServe rs ')
         self . request . add_query_ param (' VServerGro
                                                                     upId ',
vserver_gr oup_id )
         self . request . add_query_ param (' BackendSer
"[{' ServerId ':'" +
                                                                     vers ',
                                                                    instance_i
d + "',' Port ':' 80 ',' Weight ':' 100 '}]")
         response = self . client . do_action_ with_excep tion (
self . request )
                    str ( response , encoding =' utf - 8 ')
         return
         ListenerSL B ( BasicListe ner ):
class
         __init__ ( self , vsever_gro up_id ):
self . slb_caller = SLBClient ()
self . vsever_gro up_id = Conf . vsever_gro up_id
    def
         process ( self , event ):
detail = event [' content ']
    def
         instance_i d = detail [' instanceId ']
              detail [' action '] == ' delete ':
self . slb_caller . remove_vse rver_group _backend_s
         if
ervers ( self . vsever_gro up_id , instance_i d )
```

!) Notice:

The event name of the preemptible instance release alarm is Instance:Preemptibl eInstanceInterruption", mns_client.regist_listener(ListenerSLB(Conf.vsever_gro up_id), 'Instance:PreemptibleInstanceInterruption').

In your actual scenario, you need to apply for a new preemptible instance and attach it to SLB to guarantee that your services can run normally.