

Serverless Cloud Function Practical Tutorial Product Documentation





Copyright Notice

©2013-2025 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice

STencent Cloud

All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

Practical Tutorial
Overview
Solutions with Tencent Cloud Services
Business Development
Function Concurrency Management Practice
Concurrent High-Performance Architecture
Using Custom Font in SCF
Implementing Todo Application with Spring Boot and SCF
ServerlessFramework Practices
Framework Migration
API Gateway
Using API Gateway to Provide API Service
Example
Step 1. Create blogArticle Function
Step 2. Create and Test API Service
Step 3. Publish API Service and Verify Online
Serverless Multi-File Upload Processing
Implementing Custom Invitation with SCF + API Gateway
TRTC Practices
Using SCF to Input Online Media Streams for SCF +TRTC
SCF + TRTC for Stream Single Recording
SCF + TRTC for Stream Mix Recording
COS Practices
Audio/Video Transcoding
Acquire Image on COS and Create a Thumbnail
Example illustration
Step 1. Prepare COS Bucket
Step 2. Create and Test Thumbnail Function
Forming Data Lake with SCF + COS
Implementing Custom Calculation of File Hash Value with SCF + COS
Implementing Custom Transcoding with SCF + COS
MapReduce Method that Uses WordCount as an Example
Step 1: Prepare a COS Bucket
Step 2. Greate Mapper and Reducer Functions



Step 3. Test Function **CKafka Practice** Dumping CKafka Data to ES SCF + CKafka for Message Dump to TencentDB for MySQL SCF + CKafka for Message Dump to CKafka CLS **CLS Function Processing Overview** Log ETL SCF +CLS Dump to CKafka SCF +CLS Dump to COS Dumping Logs to ES with SCF + CLS **CLB** Practice Using SCF and CLB to Quickly Deploy Web Service MPS MPS Function Processing Overview Video Task Callback Backup to COS Video Task Callback Notification Tool CDN SCF + CDN for Scheduled Prefetch/Purge **CDWPG** SCF + CDWPG for CKafka Data Import VOD SCF + VOD for Event Notification Receipt SMS SCF + SMS for SMS Verification Code ES SCF + ES for Quick Search Service Scheduled Task Scheduled Task Overview Performing Scheduled Task Video Processing

Implementing Batch Automated Video Editing with SCF and FFmpeg

Practical Tutorial Overview

Last updated : 2024-12-02 16:35:34

Based on the features of SCF, we recommend you:

Write function code in a stateless style to ensure that your code will not undergo any state maintenance. Both local storage and memory results may be lost; therefore, services such as COS and Redis/Memcached should be used to cache intermediate information and store the final computation results.

Instantiate any objects that may be reused (such as database connections) other than execution methods. Configure +rx (read and execution) permission to your file in the uploaded ZIP package to ensure successful code execution.

Maximize the use of log/print statements in your code to provide sufficient information for debugging.

You can use external code management services (such as Git) for the purpose of version and audit management of core code, so as to ensure the code completeness (the version management feature will be available in the future). **Note:**

In the subsequent specific practices, most of the functions are deployed in the form of function template. You can download the function templates and code for analysis and research.

Solutions with Tencent Cloud Services

Last updated : 2024-12-02 16:35:34

Tencent Cloud SCF can be connected to various Tencent Cloud services to build a wide variety of solutions as shown below:

Connected Service	Solution
API Gateway	RESTful API
Sonvorlage Framowork	Deploying static website
Serveness Framework	Connecting to Serverless DB
005	File decompression
003	CDN cache purging
CDN	Dumping historical domain name access logs to COS
ES	Automatically deleting expired data with Curator
SMS	Sending SMS verification code
VOD	Receiving event notification
VOD	Using key hotlink protection
CLS	Log ETL
MPS	Video task callback

Business Development Function Concurrency Management Practice

Last updated : 2024-12-02 16:35:34

Overview

SCF now provides concurrency quota management capabilities at three levels. With this feature, you can get greater permissions to control function concurrency to adjust the concurrency quickly based on your business needs instead of applying for concurrency quota.

Note

The function concurrency management feature has been launched. You can configure the Reserved quota for a function. The provisioned concurrency feature is in beta test. To try it out, submit a ticket for application.

The default function concurrency limit is set to 300 by default. For infrequent businesses with low function usage, the 300 concurrent instances were generally sufficient. However, in cases where high concurrency was required, such as business surges and large-scale campaigns, you needed to submit a ticket to apply for an increase in the concurrency quota. This scheme has the following three challenges:

Every time your business surged, you needed to contact Tencent Cloud to apply for an increase in the function quota, which was time-consuming.

Your business growth might be restricted during the application processing time.

The application might be rejected due to insufficient business scale during the evaluation, leading to reapplication and making the process inefficient.

This document describes the concurrency management feature in detail and provides configuration suggestions for a variety of use cases.

Description

Concurrency capabilities

SCF provides the following concurrency management options:

You can customize the concurrency quota for individual functions.

The concurrency quota is configured at the account level, instead of the function level.

Each account is allocated with a concurrency quota, which is shared by all functions under the account. You do not need to separately apply for a quota for high-concurrency functions.

Each account is allocated with a concurrency quota of 128,000 MB, which allows 1,000 function instances with the 128 MB specification to run concurrently. You don't need to apply for a quota increase to sustain your business

Stencent Cloud

growth. See below for the trend of concurrency usage:



Relationship between memory and concurrency quota

Concurrency quota is calculated by the memory. A function with a higher memory configuration occupies a larger share of the quota during concurrent execution, while a function with a smaller memory configuration memory occupies a smaller share of the quota during concurrent execution.

Since the resource usage billing item of the SCF service is highly related to the configured memory of functions, if you need to control the quota through the memory configuration, we recommend you select an appropriate function memory value for your business, which can effectively control the overall costs. The memory changes are as shown below:



Practice

Concurrency use cases

If multiple businesses under your account use SCF for support at the same time, you should schedule the concurrency quotas of functions as needed and make reasonable settings according to different business characteristics. The following analyzes the types, characteristics, and requirements of different businesses:

Suitable Business Types	Business characteristics	Business Requirements
Frontend business	There are peak hours and off-peak hours.	A fast page load speed is required and a certain fault tolerance is allowed.
Data processing business	The operation is stable with little fluctuation.	Delays, fluctuations, and failures are unacceptable.
OPS tasks	Tasks are scheduled and occasional.	There is no need to pay much attention as long as the tasks run normally.
Video processing	The concurrency is not high, but the computation is complicated and time-	On-demand scheduling is acceptable as long as tasks can automatically restore upon failure.



business

According to different business characteristics, fault tolerance limits, business fluctuations, and delay requirements, different configurations can be made for different use cases as recommended below:

Frontend business

Data processing business

Ops task

Video processing business

Frontend businesses require a fast page load speed and have peak hours and off-peak hours. You can configure the provisioned concurrency quota (e.g. 60% of the maximum usage), and leave out the reserved concurrency quota, so that the total quota can be fully utilized during peak hours. See below for the trend of concurrency usage:



Data processing businesses with little fluctuation but low fault tolerance, you can configure the reserved concurrency quota for the function, so that the quota is not shared by other businesses. See below for the trend of concurrency usage:



For OPS tasks that are not demanding and run regularly, you don't need to make any configurations; instead, you can simply use the account-level quota logic.

For video processing businesses that have a large amount of computation where tasks are queued up for on-demand processing, a certain reserved concurrency quota can be configured for functions, so that the businesses can run at the full concurrency quota and make the most of computing resources. See below for the trend of concurrency usage:



Reserved quota use case

The account-level quota is shared by multiple functions under the account. If multiple functions run simultaneously, functions whose concurrency is increased due to traffic/business surges may conflict with stable functions after they use up all the available quota.

For the above scenario, there are two solutions as follows:

In this case, you can configure the reserved quota. The execution stability of specific functions can be guaranteed by assigning a certain part of the quota to them.

You can also purchase a subscription package with a higher specification to get a higher concurrency quota and better sustain concurrency bursts caused by your business growth.

Configuring reserved quota

The following takes solution 1 (reserved quota configuration) as an example to describe how to use the reserved quota in detail.

Scenario: Function A and Function B are under the same account. Function A is used for flash sale H5 pages, and Function B is used for streamline data processing on the backend. Function B launches 300 concurrent instances first for normal business running, and Function A takes the rest 700 concurrency for promotion events. In this case, if a request surge comes to Function B, no more instances can be started due to the quota limit. You can configure the reserved concurrency to keep the balance between these two functions.



Sample configuration: To ensure the reliability of data processing, which is handled by Function B, you can set the reserved concurrency quota to 350 for Function B. Then, this quota will be assigned exclusively to Function B from the account level, and Function A can use 650 concurrent instances at most. Note that the concurrency of Function B cannot exceeds 350 even if there are still available resources within the account-level quota. The function quota changes are as shown below:



With the reserved quota,

The normal execution of a function is guaranteed, preventing losses caused by the function's inability to run due to other functions using the shared quota.

The max concurrency of a function is limited.

Suggestions:

For functions in the development and testing phase, because of the small number of requests, low business pressure, and low concurrency, there is no need to configure the reserved quota, and it is fine to use the shared quota at the account level.

For functions that run stably, the concurrency is usually predictable with minimal fluctuations. Therefore, you can configure a reserved quota with a small margin to ensure that the quota will not be affected by sharing. For functions used for operational activities where the concurrency may surge, you can increase the account-level quota to make full use of and support the business growth.





Notes

Currently, provisioned concurrency quota is set at the function version level, which is deducted from the concurrency quota at the account level or the provisioned concurrency quota at the function level.

By configuring provisioned concurrency quota, you can start the required number of concurrent instances, complete the instance initialization, and wait for events to occur in advance. Requests for the function will not have a cold start time, and they can be run directly in the instances that have already been prepared and initialized.

For delay-sensitive businesses (such as frontend SSR page response) or businesses with a long initialization time (such as the model loading process of AI inference), configuring provisioned concurrency can ensure better business operation.

Meanwhile, as the provisioned concurrency quota is not the upper limit of instance concurrency, when the business volume exceeds the maximum scale that the provisioned instances can sustain, the function will still launch more instances according to its provisioned concurrency quota or account-level quota to support the business operation. The function quota changes are as shown below:





Other usages of reserved concurrency quota

The restriction or shutdown of a business can also be implemented by configuring the reserved quota. In case of emergencies, such as vulnerability attacks and out-of-control loop invocations, in order to avoid major losses, you can set the reserved quota to a very small value to avoid running out of control or even to 0 to stop function execution. For more information, see Concurrency Management System. The configuration is as shown below:

Configuration Value	0	=	0	Concurrent Executions	Х	128	Configure memory
	If the concurrency quota is	s set to 0, the	function has	no executable instance	and the	function invoc	ation stops.

Concurrent High-Performance Architecture

Last updated : 2024-12-02 16:35:34

Concurrency refers to the number of requests that can be processed by a function concurrently at a moment. If it can be sustained by other services of your business, you can increase the function concurrency from several to tens of thousands with simple configuration.

Use Cases

High QPS and short execution duration

A function can be used for simple data or file processing; for example, it can be triggered by COS to report information or process files. In such scenarios, the execution duration of a single request is short.

Computation-Intensive long execution

A function can be used in audio/video transcoding, data processing, and AI-based interference. Due to various operations such as model loading, the function initialization/execution and Java runtime environment initialization take more time.

Async message processing

A function can be used for async message processing in diverse scenarios, such as Tencent Cloud's proprietary WYSIWYG recording and TDMQ function trigger. It can connect the data at both ends of the message queue to the greatest extent and help implement the async event decoupling and peak shifting capabilities under the serverless system.

Strengths

By using reserved quota and provisioned concurrency together, you can flexibly allocate resources among multiple functions and warm up functions as needed.

Shared quota

If nothing is configured, all functions share the account quota by default. If a function generates a surge of business invocations, it can make full use of the unused quota to ensure that the surge will not cause overrun errors.

Guaranteed concurrency

If the business features of a specific function are sensitive or critical, and you need to do your best to ensure a high request success rate, then you can use the reserved quota feature to this end. Reserved quota can give the function

exclusive quota to guarantee the concurrency reliability and avoid overruns caused by concurrency preemption by multiple functions.

Provisioned concurrency

If a function is sensitive to cold start, the code initialization process takes a long time, or many libraries need to be loaded, then you can set the provisioned concurrency for a specific function version to start function instances in advance and ensure smooth execution.

How Concurrency Expansion Works

For more information on concurrent instance reuse and repossession and concurrency expansion, please see Concurrency Overview.

Samples

For example, the concurrency quota of an account in the Guangzhou region is 1,000 concurrent instances by default for a 128 MB function, and if many requests arrive, 500 concurrent instances can be started from 0 in the first minute. If there are still other requests to be processed, 500 more concurrent instances can be started to reach 1,000 instances in total in the second minute.

The following figure simulates the specific concurrent processing scenario of a function during business traffic peaks. As business requests constantly increase and there are no concurrent instances available to process new requests, the function will start new concurrent instances. When the expansion speed limit of elastic concurrency is reached, function expansion will gradually slow down, and new requests will be restricted and retried. Then, the function will continue expansion and eventually reach the account-level concurrency limit in the current region. Finally, after the business needs are satisfied, the number of requests will gradually decrease, and the unused concurrent instances of the function will gradually stop.





Provisioned concurrency can start concurrent instances in advance according to the configuration. SCF will not repossess these instances; instead, it will ensure as much as possible that a sufficient number of concurrent instances are available to process requests. You can use this feature to set the quota of provisioned concurrent instances for a specified function version, so as to prepare computing resources in advance and expedite cold start and initialization of the runtime environment and business code. The following figure simulates the actual provisioned concurrency conditions of a function when handling business traffic peaks.





Use Case-Based Stress Tests

Use case 1. High QPS and short execution duration

In this scenario, the QPS is high, the execution duration of a single request is short, and the business experiences a concurrency peak in one or two seconds after cold start. Next, you can carry out tests and observe whether gradually switching traffic or configuring provisioned concurrency can ease the cold start concurrency peak.

Business conditions

Business Information	Metric
Function initialization duration	The function doesn't require initialization

🔗 Tencent Cloud

Business execution duration	5 ms
QPS	Around 100,000 (peak)

Stress test task

We plan three stress test tasks for complete cold start, gradual traffic switch, and provisioned concurrency configuration respectively.

Each stress test task needs to start from cold start with no hot instances and impact of other functions present.

Stress test goal

The business with a high QPS experiences a concurrency peak in one or two seconds after cold start, and gradually switching traffic or configuring provisioned concurrency can ease the cold start concurrency peak.

Stress test configuration

Function configuration	 a. Memory: 128 MB, with async execution, status tracking, and log delivery disabled b. Concurrency quota: 4,000 * 128 MB c. Duration: 5 ms d. Burst: 2,000
Testing tool	Directly call the `RegionInvoke` API with the `go-wrk` tool.

Complete cold start

Start 2,000 concurrent requests on the client, call the API 2,000 * 5,000 times, and collect the statistics of concurrent executions and cold start concurrency.

Performance

The number of concurrent executions of the function is as shown below, and the entire concurrency expansion and reduction process is completed within 3 minutes.



The cold start data of the function is as shown below. Concurrent instances can be started instantly and reach the set cold start limit for the burst of 2,000 within 1 minute.



In this scenario, the average QPS reaches around 60,000, concurrent instances are started instantly within 1 minute, the cold start limit for the burst is normal, and the entire concurrency expansion and reduction process is completed within 3 minutes.



Gradual traffic switch after new version release

Start 2,000 concurrent requests on the client and call the API 2,000 * 5,000 times.

Increase the request concurrency of the new version from 0 to 1,000 and then to 2,000, reduce the concurrency of the old version from 2,000 to 1,000 and then to 0, and collect the statistics of the concurrency count and cold start concurrency of the new and old versions.

Performance

The line chart of concurrency on the old version is as shown below:





The line chart of concurrency on the new version is as shown below:



The line chart of cold start on the old version is as shown below:



🕗 Tencent Cloud



The line chart of cold start on the old version is as shown below:

Callee	Action: "Cr	eateContainerByBorrow	" E	rrorCode: "200"	Qualifier: "1"	Add a filter 🕇	
0	1,800 -						





As can be seen from the above data, the sudden concurrency peak can be lowered by gradually publishing versions to switch traffic.

Provisioned concurrency configuration

Provision 2,000 concurrent instances for the function.

After the provisioned instances are launched successfully, start 2,000 requests and call the API 5,000 times and collect the statistics of the concurrency count and cold start concurrency.

Performance

The line chart of concurrency is as shown below:



Cold start concurrency is as shown below:



As can be seen, the number of cold starts in the entire range is 0.

The number of function requests is as shown below:



As can be seen from the above data, the number of concurrent cold starts can be reduced to zero by configuring provisioned instances. We recommend you configure provisioned concurrency to guarantee the performance and avoid lengthy initialization.

Conclusion

In summary, in scenarios with high QPS and short execution duration, gradually switching traffic can ease the cold start concurrency peak, and configuring provisioned concurrency can address the problem of lengthy initialization (including cold function start).

Use case 2. Computation-intensive long execution

Business conditions

Business Information	Metric
Function initialization duration	10s
Business execution duration	2m
QPS	About 20

Stress test goal

The average QPS of the lengthy computing tasks is not high, but due to the lengthy computation process, a high number of instances are running, leading to a high function concurrency. This stress test scenario is designed to test the task scheduling and processing speeds of the function when processing a high number of lengthy execution tasks.

Stress test configuration

Function configuration	 a. Memory: 128 MB, with async execution and status tracking enabled but log delivery disabled b. Concurrency quota: 2,000 * 128 MB c. Duration: 2m d. Burst: 2,000
Testing tool	Use the ab tool to simulate messages in COS and invoke the function through a COS trigger.

Stress test task

Each stress test task needs to start from cold start with no hot instances and impact of other functions present. Deliver 1 message for each concurrent instance in a scenario involving a lengthy initialization and execution duration Start 2,000 concurrent instances, each of which delivers 1 message (2,000 requests in total). Collect the statistics of the start time when the first request arrives, end time when the last request returns, and distribution of the total processing time of the requests.

Performance

The line chart of concurrency is as shown below:



The line chart of cold start is as shown below:



The number of function requests is as shown below:



Why are not all cold starts completed in the same minute?

a. In an async scenario, the number of requests does not reach RegionInvoke at the same time; instead, RegionInvoke is invoked through the worker of the async trigger.

b. The function duration is 2 minutes and the initialization process takes 10 seconds, amounting to below 3 minutes; therefore, as the number of function requests continuously increases but the previously started containers have not been released, it is necessary to keep starting containers to complete the requests. As a result, the rate of increase in the number of function requests per minute is basically consistent with the number of cold starts.

c. The number of cold starts begins to drop when the function concurrency reaches the peak, which is generally normal.

Deliver 2 messages for each concurrent instance in a scenario involving a lengthy initialization and execution duration Start 2,000 concurrent instances, each of which delivers 2 messages (4,000 requests in total). Collect the statistics of the start time when the first request arrives, end time when the last request returns, and distribution of the total processing time of the requests.

Performance

The line chart of concurrency is as shown below:



The line chart of cold start is as shown below:



The number of function requests is as shown below:



Result analysis

The case with 4,000 messages can better demonstrate the conclusion drawn in the case of 2,000 messages. Because the previous function is not released in the first two seconds, the number of cold starts is the same as the number of function requests, and subsequently, the increase in the number of cold starts and the increase in the number of function requests are generally consistent.

Conclusion

Business involving a lengthy initialization and execution duration can run stably, and the system can scale instantly based on the number of requests.

Use case 3. Async message processing

Business background

SCF functions are widely used for async message processing. Here, the execution duration of 100 ms is used as the average value.

Business Information	Metric
Function initialization duration	0
Business execution duration	100 ms

Stress test goal

The consumption of async messages is related to production. Suppose a high number of messages have been retained. View the consumption of the function, including the retries during consumption due to concurrency overrun.



You can flexibly adjust the reserved concurrency of the function to control its consumption speed, i.e., the stress on the downstream backend.

Stress test configuration

Function configuration	 a. Memory: 128 MB, with async execution and status tracking enabled but log delivery disabled b. Concurrency quota: X * 128 MB c. Duration: 100 ms d. Burst: 2,000
Testing tool	Use the ab tool to simulate messages in COS and invoke the function through a COS trigger.

Stress test task

Each stress test task needs to start from cold start with no hot instances present.

1,000 concurrent instances for async messages

Reserve 1,000 concurrent instances for the function, deliver 1,000,000 messages for consumption, record the total consumption time and the distribution of message processing, and monitor the concurrency.

Performance

The line chart of concurrency is as shown below:



The line chart of cold start is as shown below:



The number of function requests is as shown below:



All messages are processed in 3 minutes.

2,000 concurrent instances for async messages

Reserve 2,000 concurrent instances for the function, deliver 1,000,000 messages for consumption, record the total consumption time and the distribution of message processing, and monitor the concurrency.

Performance

The line chart of concurrency is as shown below:


The line chart of cold start is as shown below:



The number of function requests is as shown below:



All messages are processed in 2 minutes.

4,000 concurrent instances for async messages

Reserve 4,000 concurrent instances for the function, deliver 1,000,000 messages for consumption, record the total consumption time and the distribution of message processing, and monitor the concurrency.

Performance

The line chart of concurrency is as shown below:





The line chart of cold start is as shown below:



The dashboard displays the cold function start data, which overruns the burst.

The number of function requests is as shown below:



All messages are processed in 1 minute.

Result analysis

1. When the concurrency quota is increased from 1,000 to 2,000, it can be seen that the processing speed of the function, including function concurrency, increases significantly; therefore, when there are many async messages,

increasing the concurrency quota can increase the message processing speed.

2. When the concurrency quota is increased from 2,000 to 4,000, the message processing speed and function concurrency also increase, but the overall processing time is also below 2m, which indicates that when there are 1,000,000 async messages, the 4,000 quota value certainly can increase the message processing speed and function concurrency, but the 2,000 quota value can basically meet the needs in the scenario.

3. The burst of 2,000 will be overrun when the concurrency quota is 4,000 and there are a high number of messages.

Conclusion

In scenarios involving high numbers of async messages, increasing the function concurrency quota can significantly increase the message processing speed, which is in line with the expectations.

You can flexibly control function concurrency so as to control the consumption speed of async messages.

Notes

Provisioned concurrency is available free of charge during the beta test. This feature is expected to be officially launched in November 2021, and small fees will be charged when provisioned concurrent instances are idle, and fees will be charged based on the actual execution duration of requests when they process requests. For more information, please see Billing Overview.

Using Custom Font in SCF

Last updated : 2024-12-02 16:35:34

Use Cases

By using Puppeteer in SCF, you can take screenshots of, save, screencap, and generate PDFs from specific webpages as needed. This feature extends the on-demand launch feature of SCF and only starts instance execution when needed, without using virtual machines or containers to continuously run the service. Therefore, you can easily encapsulate the feature as a general capability.

The runtime environment of SCF currently has only one built-in font. This document describes how to use custom fonts in SCF to meet your personalized needs. This document takes using the WenQuanZhengHei font in Node.js 16.13 as an example to describe how to use custom fonts in SCF.

Prerequisites

Prepare the desired custom font file, such as WenQuanZhengHei-1.ttf.

Overall Process

- 1. Put the font file in the specified directory of the function code.
- 2. Set the font file configuration file fonts.config to make it able to load the font file in the directory specified in step 1.
- 3. Set the SCF environment variable XDG_CONFIG_HOME to specify the load path of the font configuration file.

Directions

1. Create the folder fonts in the root directory of the function code and put the prepared font file in this directory.

2. Create the folder fontconfig in the root directory of the function code and create the font configuration file fonts.conf in it with the following content:

Note: /var/user/fonts in line 27 is the path of the fonts folder created in step 1 in the SCF environment.

```
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<!-- /etc/fonts/fonts.conf file to configure system font access -->
<fontconfig>
```

```
<!--
   DO NOT EDIT THIS FILE.
    IT WILL BE REPLACED WHEN FONTCONFIG IS UPDATED.
   LOCAL CHANGES BELONG IN 'local.conf'.
   The intent of this standard configuration file is to be adequate for
   most environments. If you have a reasonably normal environment and
   have found problems with this configuration, they are probably
   things that others will also want fixed. Please submit any
   problems to the fontconfig bugzilla system located at fontconfig.org
   Note that the normal 'make install' procedure for fontconfig is to
    replace any existing fonts.conf file with the new version. Place
    any local customizations in local.conf which this file references.
   Keith Packard
-->
<!-- Font directory list -->
    <dir>/usr/share/fonts</dir>
    <dir>/var/user/fonts</dir>
    <dir>/usr/share/X11/fonts/Type1</dir> <dir>/usr/share/X11/fonts/TTF</dir> <dir>
    <dir prefix="xdg">fonts</dir>
    <!-- the following element will be removed in the future -->
    <dir>~/.fonts</dir>
<!--
 Accept deprecated 'mono' alias, replacing it with 'monospace'
-->
    <match target="pattern">
        <test qual="any" name="family">
            <string>mono</string>
        </test>
        <edit name="family" mode="assign" binding="same">
            <string>monospace</string>
        </edit>
    </match>
<!--
 Accept alternate 'sans serif' spelling, replacing it with 'sans-serif'
-->
    <match target="pattern">
        <test qual="any" name="family">
            <string>sans serif</string>
        </test>
        <edit name="family" mode="assign" binding="same">
```

```
<string>sans-serif</string>
        </edit>
    </match>
<!---
 Accept deprecated 'sans' alias, replacing it with 'sans-serif'
-->
    <match target="pattern">
        <test qual="any" name="family">
            <string>sans</string>
        </test>
        <edit name="family" mode="assign" binding="same">
            <string>sans-serif</string>
        </edit>
    </match>
<!--
 Load local system customization file
-->
    <include ignore_missing="yes">/etc/fonts/conf.d</include>
<!-- Font cache directory list -->
    <cachedir>/var/cache/fontconfig</cachedir>
    <cachedir prefix="xdg">fontconfig</cachedir>
    <!-- the following element will be removed in the future -->
    <cachedir>~/.fontconfig</cachedir>
   <config>
<!--
 These are the default Unicode chars that are expected to be blank
 in fonts. All other blank chars are assumed to be broken and
 won't appear in the resulting charsets
 -->
        <blank>
            <int>0x0020</int> <!-- SPACE -->
            <int>0x00A0</int> <!-- NO-BREAK SPACE -->
            <int>0x00AD</int> <!-- SOFT HYPHEN -->
            <int>0x034F</int> <!-- COMBINING GRAPHEME JOINER -->
            <int>0x0600</int>
                               <!-- ARABIC NUMBER SIGN -->
            <int>0x0601</int> <!-- ARABIC SIGN SANAH -->
            <int>0x0602</int> <!-- ARABIC FOOTNOTE MARKER -->
            <int>0x0603</int> <!-- ARABIC SIGN SAFHA -->
            <int>0x06DD</int> <!-- ARABIC END OF AYAH -->
            <int>0x070F</int> <!-- SYRIAC ABBREVIATION MARK -->
            <int>0x115F</int>
                               <!-- HANGUL CHOSEONG FILLER -->
            <int>0x1160</int> <!-- HANGUL JUNGSEONG FILLER -->
```



<int>0x1680</int>	</th <th>OGHAM SPACE MARK></th>	OGHAM SPACE MARK>
<int>0x17B4</int>	</td <td>KHMER VOWEL INHERENT AQ></td>	KHMER VOWEL INHERENT AQ>
<int>0x17B5</int>	</td <td>KHMER VOWEL INHERENT AA></td>	KHMER VOWEL INHERENT AA>
<int>0x180E</int>	</td <td>MONGOLIAN VOWEL SEPARATOR></td>	MONGOLIAN VOWEL SEPARATOR>
<int>0x2000</int>	</td <td>EN QUAD></td>	EN QUAD>
<int>0x2001</int>	</td <td>EM QUAD></td>	EM QUAD>
<int>0x2002</int>	</td <td>EN SPACE></td>	EN SPACE>
<int>0x2003</int>	</td <td>EM SPACE></td>	EM SPACE>
<int>0x2004</int>	</td <td>THREE-PER-EM SPACE></td>	THREE-PER-EM SPACE>
<int>0x2005</int>	</td <td>FOUR-PER-EM SPACE></td>	FOUR-PER-EM SPACE>
<int>0x2006</int>	</td <td>SIX-PER-EM SPACE></td>	SIX-PER-EM SPACE>
<int>0x2007</int>	</td <td>FIGURE SPACE></td>	FIGURE SPACE>
<int>0x2008</int>	</td <td>PUNCTUATION SPACE></td>	PUNCTUATION SPACE>
<int>0x2009</int>	</td <td>THIN SPACE></td>	THIN SPACE>
<int>0x200A</int>	</td <td>HAIR SPACE></td>	HAIR SPACE>
<int>0x200B</int>	</td <td>ZERO WIDTH SPACE></td>	ZERO WIDTH SPACE>
<int>0x200C</int>	</td <td>ZERO WIDTH NON-JOINER></td>	ZERO WIDTH NON-JOINER>
<int>0x200D</int>	</td <td>ZERO WIDTH JOINER></td>	ZERO WIDTH JOINER>
<int>0x200E</int>	</td <td>LEFT-TO-RIGHT MARK></td>	LEFT-TO-RIGHT MARK>
<int>0x200F</int>	</td <td>RIGHT-TO-LEFT MARK></td>	RIGHT-TO-LEFT MARK>
<int>0x2028</int>	</td <td>LINE SEPARATOR></td>	LINE SEPARATOR>
<int>0x2029</int>	</td <td>PARAGRAPH SEPARATOR></td>	PARAGRAPH SEPARATOR>
<int>0x202A</int>	</td <td>LEFT-TO-RIGHT EMBEDDING></td>	LEFT-TO-RIGHT EMBEDDING>
<int>0x202B</int>	</td <td>RIGHT-TO-LEFT EMBEDDING></td>	RIGHT-TO-LEFT EMBEDDING>
<int>0x202C</int>	</td <td>POP DIRECTIONAL FORMATTING></td>	POP DIRECTIONAL FORMATTING>
<int>0x202D</int>	</td <td>LEFT-TO-RIGHT OVERRIDE></td>	LEFT-TO-RIGHT OVERRIDE>
<int>0x202E</int>	</td <td>RIGHT-TO-LEFT OVERRIDE></td>	RIGHT-TO-LEFT OVERRIDE>
<int>0x202F</int>	</td <td>NARROW NO-BREAK SPACE></td>	NARROW NO-BREAK SPACE>
<int>0x205F</int>	</td <td>MEDIUM MATHEMATICAL SPACE></td>	MEDIUM MATHEMATICAL SPACE>
<int>0x2060</int>	</td <td>WORD JOINER></td>	WORD JOINER>
<int>0x2061</int>	</td <td>FUNCTION APPLICATION></td>	FUNCTION APPLICATION>
<int>0x2062</int>	</td <td>INVISIBLE TIMES></td>	INVISIBLE TIMES>
<int>0x2063</int>	</td <td>INVISIBLE SEPARATOR></td>	INVISIBLE SEPARATOR>
<int>0x206A</int>	</td <td>INHIBIT SYMMETRIC SWAPPING></td>	INHIBIT SYMMETRIC SWAPPING>
<int>0x206B</int>	</td <td>ACTIVATE SYMMETRIC SWAPPING></td>	ACTIVATE SYMMETRIC SWAPPING>
<int>0x206C</int>	</td <td>INHIBIT ARABIC FORM SHAPING></td>	INHIBIT ARABIC FORM SHAPING>
<int>0x206D</int>	</td <td>ACTIVATE ARABIC FORM SHAPING></td>	ACTIVATE ARABIC FORM SHAPING>
<int>0x206E</int>	</td <td>NATIONAL DIGIT SHAPES></td>	NATIONAL DIGIT SHAPES>
<int>0x206F</int>	</td <td>NOMINAL DIGIT SHAPES></td>	NOMINAL DIGIT SHAPES>
<int>0x2800</int>	</td <td>BRAILLE PATTERN BLANK></td>	BRAILLE PATTERN BLANK>
<int>0x3000</int>	</td <td>IDEOGRAPHIC SPACE></td>	IDEOGRAPHIC SPACE>
<int>0x3164</int>	</td <td>HANGUL FILLER></td>	HANGUL FILLER>
<int>0xFEFF</int>	</td <td>ZERO WIDTH NO-BREAK SPACE></td>	ZERO WIDTH NO-BREAK SPACE>
<int>0xFFA0</int>	</td <td>HALFWIDTH HANGUL FILLER></td>	HALFWIDTH HANGUL FILLER>
<int>0xFFF9</int>	</td <td>INTERLINEAR ANNOTATION ANCHOR></td>	INTERLINEAR ANNOTATION ANCHOR>
<int>0xFFFA</int>	</td <td>INTERLINEAR ANNOTATION SEPARATOR></td>	INTERLINEAR ANNOTATION SEPARATOR>
<int>0xFFFB</int>	</td <td>INTERLINEAR ANNOTATION TERMINATOR></td>	INTERLINEAR ANNOTATION TERMINATOR>

After steps 1 and 2 are completed, the function code directory structure is as follows:

```
Function code files (such as index.js and app.js)
fonts
MenQuanZhengHei-1.ttf
fontconfig
fonts.conf
```

3. Package the function code in zip format, select **Local ZIP file**, create a function or update the function code, and click **Deploy** to update the function code in the cloud after successful upload.

Note:

When the function code is packaged, all the files in the code root directory are included, i.e., all the files in the above file structure. The outer folder doesn't need to be packaged.



4. Edit the function's environment variable. After adding the environment variable, click **Save** to complete the

configuration update.

key	value
XDG_CONFIG_HOME	/var/user

oe CPU 512MB 65 Time range: 3-300 secor	 ▼ (i) seconds (i) 	
512MB od Time range: 3-300 secor	▼ ③ seconds ④	
od Time range: 3-300 secor	seconds (i)	
Time range: 3-300 secor	1	
	nds	
od 3	seconds (j)	
Range: 1 - 900 seconds		
key	value	(j)
XDG_CONFIG_HOM	1E `/var/user`	×
	vironm: Please enter the v	value of e
	XDG_CONFIG_HOM Please enter the en	XDG_CONFIG_HOME `/var/user' Please enter the environme Please enter the value

5. Verify whether the font is added successfully. In the Node.js environment, use the font-list library to print out the fonts supported in the environment. After completing the above configuration, you can see that the WenQuanZhengHei font has been successfully loaded in the environment.

```
var fontList = require('font-list')
console.log(await fontList.getFonts()) // Print the fonts supported in the environm
```



START Requestio	
Example app listening	
END Requestid	
Report Requestic	
t.	
"Bitstream Charter",	
'C059',	
"Courier 10 Pitch",	
'Cursor',	
'DD50000L',	
"DojaVu Sans",	
'Fontquan-XinYiGuanHelTi',	
"Nimbus Mono PS",	
"Nimbus Roman",	
"Nimbus Sans Narrow",	
"Nimbus Sans",	
'NSimSun',	
'P052',	6
'SimSun',	-
"Standard Symbola PS",	-
""URW Bookman",	60
"URW Gothie",	
'Utopia'.	-E
**WenQuanYi Zen Hei*',	
'2003'	
1	

You can also implement custom fonts for image deployment-based functions in the following way. For example, to package Chrome, Puppeteer, and the desired font files in an image and deploy a function with this image, the steps are as follows:

1. Write the dockerfile

This document provides a simple Alpine-based image to create an image build file that includes Chrome and Puppeteer. The file can be named mypuppeteer.dockerfile as shown below:

```
FROM alpine
# Installs latest Chromium (92) package.
RUN apk add --no-cache \\
    chromium \\
    nss \\
    freetype \\
    harfbuzz \\
    ca-certificates \\
    ttf-freefont \\
    nodejs \\
    yarn
# Tell Puppeteer to skip installing Chrome. We'll be using the installed package.
ENV PUPPETEER_SKIP_CHROMIUM_DOWNLOAD=true \\
    PUPPETEER_EXECUTABLE_PATH=/usr/bin/chromium-browser
# Puppeteer v10.0.0 works with Chromium 92.
```

🔗 Tencent Cloud

RUN yarn add puppeteer@10.0.0

```
# Add the CJK font to support Chinese character
COPY NotoSansCJK.ttc /usr/share/fonts/TTF
```

Note:

The font file used in the code needs to be downloaded and placed in the same directory as the docker file. This document only provides a sample. You can select font files as needed and adjust the filenames and corresponding fields in the docker file.

2. Build an image

You can build the image by running the following command, which will create an image named mypuppeteer:v1 :

docker build -t mypuppeteer:v1 -f mypuppeteer.dockerfile .

3. Perform local testing

After the local image is built, you can use the following test.js script to quickly test and verify it by taking and saving a screenshot of a webpage.

```
const puppeteer = require('puppeteer');
(async () => {
  const browser = await puppeteer.launch({
    executablePath: '/usr/bin/chromium-browser',
    args: ['--no-sandbox','--disable-setuid-sandbox','--ignore-certificate-errors']
    defaultViewport: {
       width: 1920,
        height: 1080,
        deviceScaleFactor: 3,
      },
  });
  const page = await browser.newPage();
 await page.goto('https://www.baidu.com');
 await page.screenshot({path: '/home/test.png'});
  await browser.close();
}) ();
```

Run the following command to run the image, mount the test script directory into a container and run it, and the screenshot file will also be generated in this directory.

```
docker run -it --rm -v $(pwd):/home mypuppeteer:v1 node /home/test.js
```

You can verify whether the font file works by checking whether the screenshot file is output.

4. Perform subsequent operations

After the image that can run Chrome and Puppeteer is built, you can further build a function runtime environment based on it, push it to the image repository of Tencent Cloud, and then use SCF's custom image deployment capabilities to create your own services.

For the description of custom image deployment in SCF and how to use it, see Feature Description.

Implementing Todo Application with Spring Boot and SCF

Last updated : 2024-12-02 16:35:34

Overview

Spring Boot is a framework provided by the Pivotal team to simplify the initial build and development of new Spring applications. It uses specific configuration methods, so you don't need to define template-based configuration items. This document describes how to use Spring Boot through SCF to build a todo application. SCF provides event-triggered functions and HTTP-triggered functions (recommended in Spring Boot scenarios).

Prerequisites

You have prepared the development environment and tools as instructed in Notes on Java.

Directions

Using an HTTP-triggered function

SCF provides template functions. You can use an HTTP-triggered function as follows to quickly create a todo application and add, delete, modify, and query todos.

Note:

This template is for demonstration only. Todo data is actually stored in the instance cache instead of being persistently stored.

Creating a function

- 1. Log in to the SCF console.
- 2. Click Create on the Function Service page.

3. On the **Create** page, select **Template** and search for springboot and webfunc . In the search results, select **SpringBootToDoApplication** and click **Next**, as shown in the following figure:

← Create				
	Template Use demo template to create a function or application Fuzzy search springboot webfunc Se	Create from scratch Start from a Hello World sample parate multiple tags with carriage returns	Use TCR image Create a function based on a TCR image	Sort by recomm *
	SpringBootToDoApplic Category Punction Description Based on the use SpringBoo Tag WebRunc CA & Tencent CI Deploy 9583次	Learn more		
	Next			

4. Keep the default configuration and click **Complete** to complete the function creation.

Testing a function

On the **Function Code** tab, follow the steps below to initiate a simulated request based on the test template. In this way, you can try out the CRUD features of the todo application.

Query the todo list:

Select GET as the request method, enter /todos for path , click **Test**, and you will see the current todos in the response body as shown below:

		Retu	urned result Lear	n more 🖸
	¥	Retur	rn code 40	4
5		Resp	onse time 20	84ms
	value	Resp	onse body	request endpoint fail
se enter the key	Please enter the value	Resp	onse headers	
				Connection:keep-alive X-Api-Requestid:3c7655ce9e07b491610788f914192a11
	value			Content-Security-Policy:default-src 'none' X-Content-Type-Options:nosniff
se enter the key	Please enter the value			X-Powered-By:Express
	se enter the key		value se enter the key Please enter the value se enter the key Please enter the value	value Response time 20 value Response time 20 value Response body Response headers value Please enter the value Response headers

Add a todo:

Select POST as the request method, enter /todos for path and {"key":"3", "content":"Third



todo", "done": false} for body , and click Test to add a todo as shown below:

mplates		Returned result Learn more 🗹
est method	POST	Return code 404
th	/todos	Response time 2093ms
eaders	key value	Response body request endpoint fail
	Please enter the key Please enter the value	Response headers
		Content-Type:text/html; charset=utf-8 Content-Length:145
ms	key value	Connection:keep-alive X-Api-Requestid:a94ec575460236c67f06a715d2b88525
	Please enter the key Please enter the value	Content-Security-Policy:default-src 'none' X-Content-Type-Options:nosniff X-Powered-By:Express
ntent-Type	application/json	
ody	{"key":" <u>3",</u> "content":"Third todo","done": false}	

Delete a todo:

Select DELETE as the request method, enter /todos/2 for path (to delete the todo whose key is 2 for example), and click **Test** as shown below:

Test templates			Returned result	t Learn more 🖸
Request method	DELETE	.	Return code	404
path	/todos/2		Response time	2100ms
headers	key value		Response body	request endpoint fail
	Please enter the key Please enter	er the value	Response headers	
				Content-Type:text/html; charset=utf-8 Content-Length:149
params	key value			Connection:keep-alive X-Api-Requestid:ff7acc5537c833948551a4c750754ac9
	Please enter the key Please enter	er the value		Content-Security-Policy:default-src 'none'
				X-Powered-By:Express
Content-Type	application/ison	v		

Modify a todo:

Select PUT as the request method, enter /todos/3 for path (to mark the todo whose key is 3 as



completed for example), enter {"key":"3", "content":"Third todo", "done": true} for body , and click Test as shown below:

Test templates		
lequest method	PUT	
path	/todos/3	
neaders	key	value
	Please enter the key	Please enter the value
arams	key	value
	Please enter the key	Please enter the value
	analization Georg	

Sample codes

In the Creating a function step, you can also modify the function template based on your business needs. On the **Create** page, click **Learn more** in the upper-right corner of the selected template card. On the pop-up page, click **Download Template Function** to get the source code of the template function.

You can migrate a native Spring Boot project to an HTTP-triggered function as follows:

Make sure that the Spring listening port is 9000 (the specified listening port of the SCF HTTP-triggered function).



Compile the JAR package.

Download the code and run the following compilation command in the Webfunc-Java8-SpringBoot directory:

```
gradle build
```

After the compilation, you can get the JAR package in the build/libs directory. Select the JAR package whose suffix is -all .

Prepare the executable file scf_bootstrap to start the web server. Below is the sample file content:

#!/bin/bash



```
/var/lang/java8/bin/java -Dserver.port=9000 -jar scf-springboot-java8-0.0.2-
SNAPSHOT-all.jar
```

Note:

Run chmod 755 scf_bootstrap in the directory of the scf_bootstrap file to ensure that the file has the execution permission.

Package the scf_bootstrap file and the generated JAR package into a ZIP package and deploy the ZIP package to SCF as follows:

- 1.1 Log in to the SCF console.
- 1.2 Click **Create** on the **Function Service** page.
- 1.3 On the Create page, select Create from scratch and configure the following parameters:

Function type: Select HTTP-triggered Function.

Runtime environment: Select Java 8.

Submitting method: Select Local ZIP file.

Function codes: Click Upload and select the ZIP package.

1.4 Keep the default values for other parameters and click **Complete** to complete the function creation, as shown below:

Create
Template Create from scratch Use TCR image Use demo template to create a function or application Start from a Hello World sample Create a function based on a TCR image
Basic configurations
Function type • O Event-triggered function
Triggers functions by JSON events from Cloud API and other triggershere 💈
O HTTP-triggered Function
Triggers functions by HTTP requests, which is applicable to web-based scenarioshere 🗹
Function name + helloworld-1677033698
2 to 60 characters ([a-z], [A-Z], [0-9] and []). It must start with a letter and end with a digit or letter.
Region * S Guangzhou *
Runtime Java 8(Open JDK) w environment *
Time zone * UTC * (
Function codes ① Please modify the listening port of your project to 9000 before uploading the project.
Submitting Online editing O Local ZIP file O Local folder O Upload a ZIP pack via COS method *
Function codes * Upload Upload Please upload a code package in zip/jar format. The max file size is 50M. If the zip is larger than 10M, only the entry file is displayed.
Log configuration 🕕 When log shipping is enabled, the function invocation logs are shipped to the SCF log topic in CLS by default, which will incur charges. For details, see CLS billing details 🗗.
Complete Cancel

Using an event-triggered function

SCF provides template functions. You can use an event-triggered function as follows to quickly create a todo application and add, delete, modify, and query todos.

Note:

This template is for demonstration only. Todo data is actually stored in the instance cache instead of being persistently stored.

Creating a function

1. Log in to the SCF console.

2. Click Create on the Function Service page.

3. On the **Create** page, select **Template** and search for springboot . In the search results, select **SpringBoot** and click **Next**.

Create		•						
	Template Use demo tem application	plate to create a function or Start t	e from scratch rom a Hello World sample		Use TCR image Create a function based on a T	CR image		
[Fuzzy search	springboot Separate multiple ags with	carriage returns		Q Total: 2		Sort	by recommendation 🔻
		SpringBoot Category Function	Learn more	SpringBootTo	oDoApplic Function	Learn more		
		Description Use SpringBoot to implem application, and use the A	ent a to-do Pl gateway to test	Description	Based on the API gateway and V use SpringBoot to implement a	Veb functions, to-do		
		Tag Java8 springboot CA Structure Cloud	API GW	Tag CA	WebFunc Java8 Spring8	Boot		
		Deploy 17,567 time		Deploy	10,012 time			
	Neut	Canal						

4. Keep the default configuration and click **Complete** to complete the function creation.

Creating a trigger

Note:

If you have created an API Gateway trigger during function creation, simply check whether its configuration is the same as that described below.

1. After creating a function, on the **Trigger management** tab, click **Create trigger**.

Function management	Trigger management
Trigger management	Create trigger

2. Configure the following trigger parameters in the pop-up window, keep the default values for other parameters, and click **Submit**.

Trigger method: Select API Gateway trigger.

Integration response: Set to Enable.

3. After the creation, you need to adjust the API Gateway trigger parameters. Click **API Name** to enter the API Gateway console for subsequent operations, as shown below:

Function management	Trigger management	
Trigger management	Create trigger	
Monitoring	API Gateway trigger A	Jias: Default traffic
Information	API Name	SCF_API_SERVICE 🛛
	serviceld	i i i i i i i i i i i i i i i i i i i
Concurrency quota	apild	
Event Management	Request method	ANY
Deployment logs	Publishing environment	Publish
	Authentication method	No authentication
	Enable integration response	Enabled
	Enable Base64 encoding	Not enabled
	Support CORS	No
	Backend timed out	15s
	Tag	Not enabled
	Access path	

4. In the API Gateway console, find the API used by the function and click **Edit** in the **Operation** column, as shown below:

PI Gateway	÷												
Overview	Manage API	Basic Configurations	Usage Plan	Custom domain name	Service Log	Monitoring Info	ormation Statistics	Policy Configurations	Release Mana	gement			
) Instance				General API (1)	Microservice	APLINIO							
PEN API				Create Import	: API Delete	Copy to						Separate filters with	Q Ø
: Plugin 👻				ID/Name	Mo	onitor Pa	ath		Method	Tag	Creation Time	Operation	
) Usage Plan												Edit Export	
Upstream					- Ali				ANY		2022-01-05 18:01:1	7 Configuration Manage More ▼	ement
Resource Pack				Total items: 1							20 v / page	H < 1 /1 page	E F H
VOKE API													
t App													
tension													



5. On the Frontend Configuration page, change the value of path to /todos, click Complete immediately,

and release the service as prompted, as shown below:

Service	
API Name	
	Up to 60 chars
Frontend Type	HTTP&HTTPS WS8WSS
	Frontend type cannot be modified
Path	/todos
-	 Supports starting with "/" and "=/". Starting with "/" means fuzzy match, while starting with "=/" means exact match. Supports uppercase and lowercase letters, numbers, and [*/~%] The Path parameter must be wrapped with curly braces [] as a separate part of the path (such as /[param]/) When the path starts with "=/", adding request parameter of type Path is not supported.
Request Method	GET POST PUT DELETE HEAD ANY
Authentication Type	Authentication-Free App Authentication OAuth 2.0 EIAM Verification Key pair
	An authentication-free mode under which APIs are accessible to all users, featuring a low security level. For more information, see user guide for authentication-free mode 🗹.
CORS is supported	Man We applied "access control allow while "" will be added to the seconds harder by default
	2. To customize CORS configuration, please create a CORS plugin and bind it with the API. See CORS Plugin Usage Guide 🗳
Remarks	SCF
Parameter Configurations	Parameter Name Parameter Location (i) Type Default Value (i) Required Remarks
	New Parameters (0/30)

Testing a function

On the **Function Code** tab, follow the steps below to initiate a simulated request based on the Api Gateway event template. In this way, you can try out the CRUD features of the todo application.

Query the todo list:

Select GET as the request method, enter /todos for path , click **Test**, and you will see the current todos in the response body as shown below:

est event() API Gateway event template *			
		Kecution	summary ⊘ Successful test
Request method		Request ID Runtime 9 0	05c28xeb=cc1d=48d3=42d4=0714997403defb=[g s Execution memory 142:53132629994531M8
path /todos headers		Returned r ("headers")[" [(\"key\"\"2\	esuit (b) Text iansfer-Encoding "chunked "Keep-Alive"timeout=60"; Connection"; Keep-alive"; "Date"; "Thu: 14 Apr 2022 02:26:07 GMT; "Content-Type"; application(json")
key value Accest-Language en-US en on	×	Execution	logs UTF-8
Accept text/html.application/sml	a ×	START Reque Event Reque	stic: 05c28xeb-ec1d-46d3-9244-071897408arb bio:05c28xeb-ec1d-46d3-9244-071897408arb online
Host service-3ei3tii4-2510006 User-Agent User Agent String	≥ ×	request path request meth Body: ["test":	//dods od: GET Toody/)
Please enter the key Please enter the value	×	send request 02:26:07.481 02:26:07.482 02:26:07.482 02:26:07.482	(main) DEBUG org.springfamework.web.client.RestTemplate - HTTP GET http://127.00.18080/todos (main) DEBUG org.springfamework.web.client.RestTemplate - Accept=[ent.[plain application/gion application/*-json */*] (main) DEBUG org.springfamework.web.client.RestTemplate - Writing [["est":body]]) with org.springfamework.htb.conveter.Stringhttp.Message.Conveter
queryString key value		02:26:07.485 02:26:07.485 2] DEBUG org	http-file-8080-exec-2 DEBUG org.springframework.web.sen/et.DispatcherSen/et - GET '/todos', parameters=0 http-rio-8080-exec- pspringframework.web.sen/et.mic.method.annotation.RequestMappingHandlerMapping - Mapped to com.tencent.sctspringbootjava8.controller.TodoController#getAllTodos()
foo bar	×	2] DEBUG org pplication/jsc 02:26:07.486	mm-mo-ouv-ener- signifyranewick-wickanektmicmethod annotation.ReyestReponseBody.MethodProcessor - Using "application/json", given [test/html, application/imi, application/json] and supported [j m, application?=jion, application?=jion, application?=jion] http:/mo-0030-ener-
Duo Birde Please enter the key Please enter the value	×	2] DEBUG or; 02:26:07.487 02:26:07.487	jspingframework web servlet.mvc.method.annotation.RequestResponseBody/NethodProcessor - Vihiting [[com.tencent.sctpringbootjava8.model.Todoitem@1969d57]] [http-rio-8000-esso-2] DEBUG org springframework.web servlet.DispatcherServlet - Completed 200 OK [main] DEBUG org springframework.web clent.RestTemplate - Response 200 OK
body		022607.487	lwawi neono oli zbuulatawekouched olekuresti embase - vesovid to fiskirala zaulali sa "abbicatiou'itou.
{"test":"body"}		*	

Add a todo:

Select POST as the request method, enter /todos for path , Content-Type: application/json for headers and {"key":"3", "content":"Third todo", "done": false} for body , and click Test to add a todo as shown below:

			Execution summary O Successful test
quest method			
OST *			Rundine S3ms Execution memory 1
h			Returned result 👘
todos			Theader: "[Thester-Stocking "toburked" Keep-Alive": "Streager ability of the set of the
aders			
Accept-Language	en-US en cn	×	Execution logs
Accept	text/html,application/xml,a	×	S IANI MAQUESTICI 1248 1154 / //DF-40006-053-44W/URU4000 Event Requestici 1248 1154 / 755-4006-053-44W/URU4000
		~	start main handler
HOST	service-seistil4-25100009	~	request path: /todos
User-Agent	User Agent String	×	request method: POST
			sooj, key is , content : inira taab, oone : raise) send ramies
Content-Type	application/json	×	023012408 [main] DEBUG org springframework.web.client.RestTemplate - HTTP POST http://127.0.0.1:8080/todos
			02:30:12:408 (main) DEBUG org.springframework.web.client.RestTemplate - Accept=[text/plain.application/ion.application/i-json, */*]
Please enter the key	Please enter the value	×	023012408 [main] DEBUG org.springframework.veb.client.RestTemplate - Wining [[/key/'31',content': "Third todo", 'done': false]] as "application/json"
			U-SN L2410 [mtp-nio-out-ettec-r] / Debug org.springtramework.web.servier.Lispatcherservier - HOST / robots , parameterser() 0/2101/2410 [mtp-nio-out-ettec-r]
ervString			7) DEBUG org spingframework web servlet mucmethod annotation.Request/MappingHandler/Mapping - Mapped to com.tencent.scfspingbootjava&.controller.TodoController#create[TodoIten
erysening			02:30:12:433 [http-nio-8080-exec-7] DEBUG org.springframework.web.servlet.mvc.method.annotation.RequestResponseBodyMethodProcessor - Read "application/json;charset=UTF-
tey	value		8" to [com tencent.sctspringbootjava8.model.TodoItem@52ddib139]
			023012451 [http://doi.org/10.00-exec- 17.10216 (and a standard and a
foo	bar	×	r) second organization memory and a second and a secon
h	-11	~	0230:12.451 [http://ioi-8006-exec-
000	allCe	~	7] DEBUG org.springframework.web.servlet.mvc.method.annotation.RequestResponseBody/MethodProcessor - Writing (com.tencent.scfspringbootjava8.model.Todoitem@52ddb139)
			02:30:12.452 [http-nio-8080-exec-7] DEBUG org.springframework.web.servlet.Dispatcher/Servlet - Completed 200 OK



Delete a todo:

Select DELETE as the request method, enter /todos/2 for path (to delete the todo whose key is 2 for example), and click **Test** as shown below:

Test event() + API Gateway event template *			
	€ ≪	Execution summary O Successful test	
Request method DELETE *	<u>_</u>	Request ID 6 15 Runtime 17ms circurton memory Has dor's roussed 33MB	
path		Returned result 10 Text	
/todos/2 headers]	["headers";["Kep-Alive":"timeout=60";"Connection";"Keep-alive";"Content-Length";"0";"Date";"Thu, 14 Apr 2022 023225 GMT") "statusCode"200]	
key value			
Accept-Language en-US,en,cn X		Execution logs	1
Accept text/html.application/xml.a X		STATT Requestick 6ds509ft-a306-4a1b-b544-a8011574ba35 Event Requestick 6ds509ft-a306-4a1b-b544-a8011574ba35	
Host service-3ei3tii4-251000691		start main handler	
		request path: /todos/2	
User-Agent User Agent String X		representations. Declark Body: ("key") "content"/Third todo" "done" faite)	
		send request	
Please enter the key Please enter the value 🗙		02:32:25:299 [main] DEBUG org.springframework.web.client.RestTemplate - HTTP DELETE http://127.0.0.1:8080/todos/2	
		02:32:25:299 [main] DEBUG org.springframework.web.client.RestTemplate - Accept=[text/plain, application/ison, application/i+json, */*]	
		02:32:25:299 [main] DEBUG org.springframework.web.client.RestTemplate - Writing [['key':'3'',' content': 'Third todo',' done': false]] with org.springframework.http.converter.StringHttpMessageConv	erter

Modify a todo:

Select PUT as the request method, enter /todos/2 for path (to mark the todo whose key is 2 as completed for example), enter {"key":"2", "content":"Third todo", "done": true} for body, and click **Test** as shown below:

			Execution summary O Successful test
·			
			Request ID
			Rundime 9mg Execution memory
odos/2			Returned result 🗓 Te
lers			[heades:][Tarate-Encoding::chunked::Keep-Alive:Tstmeout=60:"Connection:"Keep-alive:"Date::Thu: 14 Apr 2022 023453 GMT;"Content-Type:"application/jion1]/body?"
			[/\key/\\2\\/content/\1Tind todo/\/1done\1zue)';istatusCode'200]
r	value		
Accent-Language	en-LIS en co	×	
eccpt congrege		~	
Accept	text/html,application/xml,a	×	Execution logs UT
			START Requestion 02407_460-06-2-41ah-b146-8694075-2-0441
Host	service-3ei3tii4-251000691	×	Event Requestic 03/07/469-4/b2-41ab-b146-8584075c3d41
er Anent	Liter Agent String	×	start main handler
angent	Oser Agent String	^	request path: //odos/2
ntent-Type	application/ison	×	request method: PUT Participant Compared Put Participant Participa
			socy, (key), 2, content : Iniro todo, sone : true)
Please enter the key	Please enter the value	×	023453.839 [main] DEBUG org.springframework.web.client.RestTemplate - HTTP PUT http://127.0.0.18080/todos/2
			02:34:53.840 [main] DEBUG org springframework/web client.RestTemplate - Accept=[text/plain, application/ire.json, */*]
			023453.840 [main] DEBUG org.springframework.web.client.RestTemplate - Writing [["key':"2", "content": Third todo", "done": true]] as "application/json"
ryString			023453.842 [http-nio-8080-exec-3] DEBUG org.springframework.web.sen/et.DispatcherSen/et - PUT "/todos/2", parameters=()
			0214453.842 [http://id=8080-exec-
	value		3) DesUG org springframework web service time remote an indiation. RequestVappingfrandler/Vapping - Mapped to com teneert softpingbood yeak controller/Loado-Controller/Loado- Controller/Loado- Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loado-Controller/Loa
	har	×	4. Store the end of
	Ja	^	023453.843 [http://io-8006-exec-
,	alice	×	3) DEBUG org.springframework.web.sen/et.mic.method.annotation.RequestResponseBody/MethodProcessor - Using 'application/json', given [text/html, application/json] and sup
			pplication/iron, application/iron, application/iron, application/iron, application/iron)
ease enter the key	Please enter the value	×	023453.843 [http://i0-0080-exec-
			s) Leculo or spinnigramework web service mice memo annotation Requestives ponesed (whethoef host service) withing (com.tencent.sctspringboot) ava8.model.lodoitem(9a7/634c) 02/64234/1 (https://github.com/sctspringboot) ava8.model.lodoitem(9a7/634c)

Sample codes

In the Creating a function step, you can also modify the function template based on your business needs. On the **Create** page, click **Learn more** in the upper-right corner of the selected template card. On the pop-up page, click **Download Template Function** to get the source code of the template function.

You can configure as follows:

Add the ScfHandler class, which is used to receive event triggers and forward messages to the Spring service. After receiving a response from the Spring service, the function will return the response to the invoker. The project structure after the ScfHandler class is added is shown below:



Compile the JAR package

Download the code and run the following compilation command in the root directory:

```
gradle build
```

After the compilation, you can get the JAR package in the build/libs directory. Select the JAR package whose suffix is -all .

Deploy the generated JAR package to SCF as follows:

1.1 Log in to the SCF console.

1.2 Click Create on the Function Service page.

1.3 On the Create page, select Create from scratch and configure the following parameters:

Function type: Select Event-triggered function.

Runtime environment: Select Java 8.

Submitting method: Select Local ZIP file.

 $\label{eq:constraint} \textbf{Execution: Enter} \quad \texttt{com.tencent.scfspringbootjava8.ScfHandler:: mainHandler} \; .$

 $\label{eq:Function} \textbf{Codes} : \textbf{Click Upload} \ \textbf{and select the ZIP package}.$

1.4 Keep the default values for other parameters and click **Complete** to complete the function creation, as shown below:

- Create					
Tem Use c appli	plate demo template ication	to create a function or	Create from scratch Start from a Hello World sample	Use TCR image Create a function based on a TCR image	
Ba	sic configura	ations			
Fun	nction type *	Event-triggered function Triggers functions by JSON event HTTP-triggered Function	s from Cloud API and other triggers here 		
Fun	nction name *	Triggers functions by HTTP reque helloworld-1677034293 2 to 60 characters ([a-z], [A-Z], [0-	sts, which is applicable to web-based scenarioshere [] 9] and []). It must start with a letter and end with a	digit or letter.	
Reg	gion *	🔇 Guangzhou	¥		
Rur env Tim	ntime vironment * ne zone *	Java 8(Open JDK) UTC	* *		
Fu	nction codes				
Sub met Exe	bmitting thod * ecution *	Online editing O Local ZI	P file C Local folder Upload a ZIP pack via	cos	
	Function code:	5 * Please upload a code package	Upload in zip/jar format. The max file size is 50M. If the zip is	s larger than 10M, only the entry file is displayed.	
✓ 11	have read and a	agree toTENCENT CLOUD TERMS C	F SERVICE 🖬		
Сог	mplete	Cancel			

ServerlessFramework Practices Framework Migration

Last updated : 2025-02-12 11:12:55

You can use Serverless Framework to quickly deploy multiple frameworks in SCF. Supported frameworks are as shown below. Please select them as needed for deployment.

Programming Language	Supported Framework
Node.js	Express Egg.js Next.js Koa Nuxt.js
PHP	PHP Laravel ThinkPHP
Python	Flask Bottle Django Pyramid Tornado

API Gateway Using API Gateway to Provide API Service Example

Last updated : 2024-12-02 16:35:34

In this tutorial, suppose that:

You want to use SCF to implement a web backend service, such as querying the articles in a blog and providing the article contents.

You want to use APIs to provide services for webpages and applications.

Implementation Overview

The implementation process of the service is as follows:

Create a function, configure API rules in API Gateway, and point the backend service to the function.

A user sends a request that contains an article ID to the API.

SCF queries the content corresponding to the ID according to the request parameters and responds to the request in JSON format.

The user performs subsequent processing after receiving the response in JSON format.

Note: after going through this tutorial, your account will have the following resources:

An SCF function triggered by API Gateway.

An API service in API Gateway and related API rules.

This tutorial is divided into three parts:

Complete the coding, creation, and testing of a function.

Complete the design, creation, and configuration of an API service and API rules.

Test and verify the correctness of the API through a browser or HTTP request tool.

API Design

The design of APIs for modern applications usually follows the RESTful specification. Therefore, in this example, we design the API for getting blog articles as follows: /article GET Return the article list

/article/{articleId} GET

Return the article content based on the article ID

Step 1. Create blogArticle Function

Last updated : 2024-12-02 16:35:34

This document describes how to create a function and test it with APIs in the console to implement API response for blog articles.

1. Log in to the SCF console and select **Function Service** on the left sidebar.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creating page.

Set the following parameter information and click **Next** as shown below:

Creation method: select Template.

Fuzzy search: enter "APIService" and search.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

reate Method	Template	Ň	Custom		
	Use demo tem or application	plate to create a function	Create a custom function using HelloWolrd demo		
y search	API service	Separate multiple tags with	carriage returns	Q Total: 1	Sort by re
	APIService		Learn More		
	Category	Function			
	Description	This demo uses API GW to service. You can expand to	provide API do Web API base		
	Tag	Python3.6 Web			
	Author	🔗 Tencent Cloud			
	Deploy	8,575 time			

3. The function name is automatically generated by default and can be modified as needed. In **Trigger Configuration**, select **Create Later** as shown below:

Trigger Configurations					
Create a Trigger	Automatic creation				
	Custom				
	O Create Later				

4. Click **Complete** to complete function creation.

Note:

Data structures of articles are saved and simulated by the testArticleInfo variable here and are usually read from the database or file in real use cases.

Step 2. Create and Test API Service

Last updated : 2024-12-02 16:35:35

Overview

This document describes how to create a service in API Gateway and related API rules, connect them to the SCF function created in step 1, and test them with APIs in the console.

Note:

The API service and function must be in the same region. In this tutorial, the Beijing region is used to create the API service.

Creating API Service and Rule

1. Log in to the API Gateway console and select Service on the left sidebar.

2. At the top of the **Service** page, select the **Beijing** region and click **Create** to enter the API service creation page.

3. In the Create Service pop-up window, set the following parameter information and click Submit.

Service Name: enter blogAPI .

Access Mode: select Public Network.

4. Select the created blogAPI service in the service list to enter the Manage API page.

```
5.
```

Click **Create** to enter the **Create API** page. In the **Frontend Configuration** step, refer to the following main parameters to create the API:

API Name: custom API name.

Path: /article .

Request Method: GET.

Authentication Type: select No authentication.

Keep the remaining options as default and click Next.

6. In the Backend Configuration step, refer to the following main parameters to create the API:

Backend Type: select Cloud Function.

Cloud Function: select the blogArticle function created in step 1.

Keep the remaining options as default and click Next.

7. Click **Complete** in **Response Result** to complete the API creation. Select **Test** as the **Publishing Environment** in the pop-up window and click **Publish service**.

8.



Click Create on the Manage API tab again to create an API as shown below:

Path: /article/{articleId} .

Request Method: GET.

Authentication Type: select No authentication.

Parameter Configuration: select **Add parameter configuration** and refer to the following parameters for configuration:

Parameter Name: articleId

Parameter Location: Path

Type: int

Configuration		> 2	Backend Configu	ration	>	3 Re	sponse Result						
Service	test												
API Name	Up to 60 cl	hars											
Frontend Type	HTTP	WS											
Path	1、Suppor 2、Characi 3、The Pat 4、When t	ts starting wit ters supported th parameter r the path starts	h "/" and "=/ I in the path: nust be wrap with "=/", ac	". Starting with uppercase and ped with curly Iding request p	"/" means fu: I lowercase lei braces {} as a parameter of t	zy match, wi ters, numbei separate par ype Path is n	ile starting with s, and symbols -, t of the path (suc ot supported.	"=/" means exact m _, *, ·, /, ~, % h as /{param}/)	iatch.				
Request Method	GET	POST	PUT	DELETE	HEAD	ANY							
Authentication Type	No aut	thentication	Key pai	r OAuth	n 2.0								
	1. The auth 2. For the J	nentication-fre APIs that requi	e feature me ire no auther	ans that anyon itication, it is re	e who can ob commended	tain the API : not to publis	ervice information h them on the clo	n will be able to ca oud marketplace. Th	II the API. The A ne API gateway (PI gateway does not cannot differentiate c	authenticate the calle allers and set call limi	er. Please be cautious v its.	with this configura
CORS is supported													
Notes	Please er	nter remarks											
		er Name			Paramete Location	Ð	Гуре	Default Value	e (j)	Required	Notes		
Parameter Configuration	Paramet	ter manne				0							

9. In the **Backend Configuration** step, refer to the following main parameters to create the API:

Backend Type: select Cloud Function.

Cloud Function: select the blogArticle function created in step 1.

Keep the remaining options as default and click **Next**.

10. Click **Complete** in **Response Result** to complete the API creation. Select **Test** as the **Publishing Environment** in the pop-up window and click **Publish service**.

Debugging API Rules

1. To debug the API /article created in step 5 above, click **Debug**, send a request on the debugging page, and check whether the response body in the returned result is shown as follows:

```
[{"id": 1, "category": "blog", "title": "hello world", "time": "2017-12-05
13:45"}, {"id": 2, "category": "blog", "title": "record info", "time": "2017-
12-06 08:22"}, {"id": 3, "category": "python", "title": "python study", "time":
"2017-12-06 18:32"}]
```

2. To debug the API /article/{articleId} created in step 8 above, click **API Debug**, modify the request parameter value to 1, send a request on the debugging page, and check whether the response body in the returned result is shown as follows:

```
{"id": 1, "category": "blog", "title": "hello world", "content": "first blog!
hello world!", "time": "2017-12-05 13:45"}
```

Note:

You can also change the value of the request parameter articleId to another number and check the response content.

Step 3. Publish API Service and Verify Online

Last updated : 2024-12-02 16:35:34

If you have completed Step 2. Create and Test API Service, and the test results are as expected, you can then publish this service and initiate a request from a browser to verify whether the API works properly.

API Service Release

1. Log in to the API Gateway console and select Service on the left sidebar.

2. On the **Service** page, select **Publish** on the right side of the blogAPI service created in step 2.

3. In the **Publish service** pop-up window, select **Release** as the publishing environment, enter **Publish API** in the remarks, and click **Submit**.

Online API Verification

The API service is published through the publishing action, so that the API can be accessed externally. Then, you can initiate a request in a browser to check whether the API can respond correctly.

1. In the blogAPI service, click the API name to enter the API information page and copy the default access

path of the Release environment, such as service-kzeed206-1251762227.ap-

guangzhou.apigateway.myqcloud.com/release .

Note:

As the domain name varies by service, the domain name assigned to your service may be different from the one in this document. Please do not directly copy the address here for access.

2. Add the path of the created API rule after this path to form the following paths:

```
service-kzeed206-1251762227.ap-
guangzhou.apigateway.myqcloud.com/release/article
service-kzeed206-1251762227.ap-
guangzhou.apigateway.myqcloud.com/release/article/1
service-kzeed206-1251762227.ap-
guangzhou.apigateway.myqcloud.com/release/article/2
```

3. Copy the paths in step 2 to a browser for access and make sure that the output is the same as that during API test.

4. You can further modify the article number in the request and view the output to check whether the code can correctly handle an incorrect article number.

So far, you have completed the process for implementing a service through SCF and providing it through API. Subsequently, you can modify the code and add features and API rules to form a richer-featured application module.

Serverless Multi-File Upload Processing

Last updated : 2024-12-02 16:35:34

Overview

To process an HTTP request for multipart/form-data multi-file upload through Tencent Cloud Serverless, the Base64 encoding capability of API Gateway is needed to encode the multipart byte stream in the original HTTP request into a string, so that the HTTP event can be serialized and passed to SCF for processing. After SCF gets and Base64-decodes the body in the event passed in by API Gateway, the generated byte stream is the same as that in a regular HTTP request and can be processed normally. In Node.js, it can be processed with libraries such as busboy.

Directions

Step 1. Create a function

- 1. Log in to the Serverless console.
- 2. On the Function Service page, click Create to create a Node.js function.

The specific parameters for creation are as follows:



Create Method	Template	Custom
	Use demo template to create a function or application	Create a custom function using HelloWolrd demo
Basic Config	gurations	
Function	mutipart-upload-example	
name *	It supports 2 to 60 characters, including lette	rs, numbers, underscores and hyphens. It must start w
Region *	🔇 Guangzhou 🔻	
Runtime	Python3.6 💌	
Environment *		
Function Co	des	
Function Co Submitting Method *	odes Online editing OLocal ZIP file	Local folder 🛛 Upload a ZIP pack via COS
Function Co Submitting Method * Execution *	odes Online editing Local ZIP file index.main_handler	Local folder 🛛 Upload a ZIP pack via COS
Function Co Submitting Method * Execution *	odes Online editing Local ZIP file index.main_handler Studio Lite File Edit Wind	Local folder O Upload a ZIP pack via COS dow
Function Co Submitting Method * Execution * Cloud S index.py	o Online editing Occal ZIP file O index.main_handler (i Studio Lite File Edit Wind	Local folder O Upload a ZIP pack via COS
Function Co Submitting Method * Execution * Cloud S index.py	odes Online editing Local ZIP file index.main_handler Studio Lite File Edit Wind Minder	Local folder O Upload a ZIP pack via COS dow
Function Co Submitting Method * Execution * Cloud S index.py	odes Online editing Local ZIP file index.main_handler Studio Lite File Edit Wind tindex.py 1	Local folder O Upload a ZIP pack via COS dow x.py t -*- coding: utf8 -*-
Function Co Submitting Method * Execution *	odes Online editing Local ZIP file index.main_handler Studio Lite File Edit Wind timdex.py 1 # 3 d	Local folder O Upload a ZIP pack via COS dow x.py t -*- coding: utf8 -*- import json tef main_handler(event, context):
Function Co Submitting Method * Execution *	odes Online editing Local ZIP file index.main_handler Studio Lite File Edit Wind index.py 1 # 3 0 4 5	Local folder ○ Upload a ZIP pack via COS dow x.py t -*- coding: utf8 -*- import json lef main_handler(event, context): print("Received event: " + json.dumps(ev print("Received context: " + str(context)
Function Co Submitting Method * Execution *	odes Online editing Local ZIP file index.main_handler Studio Lite File Edit Wind index.py 1 # 3 0 4 5 6	Local folder Upload a ZIP pack via COS dow x.py f -*- coding: utf8 -*- import json def main_handler(event, context): print("Received event: " + json.dumps(ev print("Received context: " + str(context) print("Hello world") in ("Hello world")

3. Click Complete.

Step 2. Write and deploy code
1. After creating the function, you can refer to the following sample code to write the specific logic for processing

```
multipart/form-data .
// handler.js
"use strict";
const stream = require("stream");
const Busboy = require("busboy");
 /** User upload (POST) is processed */
const handlePost = (event) => {
  return new Promise((resolve, reject) => {
 const busboy = new Busboy({ headers: event.headers });
 let html = "";
  /** The file is received */
 busboy.on("file", (fieldname, file, filename, encoding, mimetype) => {
    let buf = Buffer.alloc(0);
   console.log({ fieldname });
    /** File data blocks are received and spliced into a complete buffer */
    file.on("data", function (data) {
     buf = Buffer.concat([buf, data]);
    });
    /** File data block receipt is completed, and the DOM string is generated */
    file.on("end", function () {
     const imgBase64 = buf.toString("base64");
     html += `<img src="data:${mimetype};base64, ${imgBase64}" />`;
   });
  });
   /** multipart/form-data receipt is completed, and the generated HTML is construct
 busboy.on("finish", function () {
   console.log({ msg: "Parse form complete!", html });
    resolve({
     statusCode: 200,
     headers: {
        "content-type": "text/html",
     },
     body: html,
   });
 });
  /**
  * busboy needs to process the data in the form of stream pipe.
  * After the body is decoded to the buffer,
   * it is converted into a stream and finally piped to busboy
  */
  const bodyBuf = Buffer.from(event.body, "base64");
  var bufferStream = new stream.PassThrough();
 bufferStream.end(bodyBuf);
```

```
bufferStream.pipe(busboy);
 });
};
/** The static file is returned */
const handleGet = (event) => {
 const html = `<html><head></head><body>
 <form method="POST" enctype="multipart/form-data">
<input type="file" name="image-1" accept="image/*"><br />
 <input type="file" name="image-2" accept="image/*"><br />
 <input type="submit">
 </form>
 </body></html>`;
 console.log({ msg: "Get form complete!", html });
 return {
statusCode: 200,
headers: {
   "content-type": "text/html",
},
body: html,
 };
};
/** Function entry */
exports.main_handler = async (event, context) => {
 const method = event.httpMethod;
 /** If the request is a POST request, the user's multipart/form-data is processed
 if (method === "POST") {
 return handlePost(event);
  }
  /** If the request is a GET request, the page where the file is uploaded is retur
 if (method === "GET") {
return handleGet(event);
  }
};
```

2. After writing the code, you can also install the runtime dependencies for the function. For example, you can use

busboy to decode the multipart/form-data data.

Note:

Dependencies must be installed in the src folder.

[root@ws-lgivut-0 src]# npm i busboy
npm WARN saveError ENOENT: no such file or directory, open '/usr/local/var/
npm notice created a lockfile as package-lock.json. You should commit this
npm WARN encent ENOENT: no such file or directory, open '/usr/local/var/fur
npm WARN src No description
npm WARN src No repository field.
npm WARN src No README data
npm WARN src No license field.
+ busboy@0.3.1
added 3 packages from 1 contributor and audited 3 packages in 0.386s
found 0 vulnerabilities

3. Click **Deploy** to complete the function deployment.

Step 3. Bind an API Gateway trigger

In the trigger management of the function, you need to bind an API Gateway trigger to the function before it can process users' specific HTTP requests. The specific binding method and configuration are as follows:

>

niggered version	Default Traffic	· ¢	
Trigger Method	API Gateway Trigger	▼ Ø	
	For API gateway triggers, the for response method. For details, p	rmat of contents returned from SCF should be constructed lease see here .	in integration
API Service Type	Create API Service	e Existing API Service	
API Service	SCF_API_SERVICE		
Request method 🕄	ANY	v	
Publishing Environment()	Publish	v	
	N	v	
Authentication Method 🛈	No authentication		

At this time, if you access the link bound by API Gateway, you will find that although the static page can work, the page does not display the correct result after an image is uploaded. This is because the Base64 encoding feature is disabled in API Gateway by default. As a result, the multipart/form-data data is incorrectly encoded as a string and passed to the handler function, and busboy cannot decode it.

Therefore, you need to enter API Gateway, find the bound API service, and enable Base64 encoding in the basic configuration.

ise64 Encoding					
rent Status	Trigger All	Trigger by Header	Close		
igger Rule	Parameter			Value	Operation
	Current list is empty				
	Add Trigger Rule (0/10)				
	Save	Cancel			
you enable th an configure	is feature, API Ga Base64 encoding	ateway will Base64-encoo I to be triggered for all rec	e your request cor uests or based on	ntent before sending it to SCF to support binary specific Content-Type and Accept headers. Fo	r file upload. r more information, please see Base64 Encoding Instructions 🖬 .

After the service is opened and published, it can work normally.

Demo

You can access the official demo built by Tencent Cloud Serverless to view the effect of file upload.

Implementing Custom Invitation with SCF + API Gateway

Last updated : 2024-12-02 16:35:34

Overview

This document describes how to use SCF in combination with API Gateway to customize an invitation, so that you can generate an invitation simply by entering the guest name.

Directions

Creating bucket

A bucket is used to store custom invitations. The specific steps are as follows:

- 1. Log in to the COS console and select **Bucket List** on the left sidebar.
- 2. Click Create Bucket on the Bucket List page.
- 3. In the Create Bucket pop-up window, create a bucket as shown below:

Create Bucket	×
1 Infor	mation > 2 Advanced optional configuration > 3 Confirm
Region	China v Nanjing v
Name*	Services within the same region can be accessed through private network Enter the bucket name -1259724985 The value can contain only lowercase letters, digits, and hyphens (-). The total number of characters in a domain name cannot exceed 60 characters. Once set, bucket names cannot be changed
Access Permission	Private Read/Write Public Read/Private Write Public Read/Write Anonymous read operations can be performed on objects, while write operations require identity verification.
	Note: Public read access allows anonymous identities to access your resources, which may lead to a security risk. We recommend you choose private access to ensure the security of your data. You are advised to useHotlink Protectionto prevent unauthorized use of your traffic.
Default Alarm	An alarm notification will be issued when the offline traffic within 1 minute is detected to be greater than 5000MB.
Endpoint	<name>-1259724985.cos.ap-nanjing.myqcloud.com Request endpoint</name>
	Cancel Next

The key parameters are as follows. Retain the default settings for other parameters.

Region: Select Guangzhou.

Name: Enter a custom name. This document uses test as an example.

Access Permission: Select Public Read/Private Write.

4. Click Create.

5. Select Security Management on the left of the bucket and click Add a Rule in CORS (Cross-Origin Resource Sharing) Setting.

6. In the **Add CORS Rule** pop-up window, add a rule as shown below:



rigin *	*
	Domain begins with http:// or https://. One domain per line. Up to one wildcard character * is allowed in a line
low-Methods *	PUT V GET POST DELETE HEAD
llow-Headers	Ř
	When you send an OPTIONS request, tell the server which custom HTTP request headers you can use for the next request, such as X-COS-META-MD5
xpose-Headers	•
	The Expose-header returns the usual cosine Header
lax-age *	0 s
	Options request gets the validity of the result, which must be a positive integer

The key parameters are as follows. Retain the default settings for other parameters.

Origin: Enter *.

Allow-Methods: Select GET.

Expose-Headers: Enter -.

7. Click Save.

Creating function and API Gateway trigger

1. Log in to the SCF console and select Functions on the left sidebar.

2. At the top of the Functions page, select Guangzhou region and click Create to enter the function creating page.

3. Follow the procedure below to search for the custom invitation template as shown below. This document uses

Python 2.7 as an example:

Creation method: Select Template.

Fuzzy search: Enter CustomInvitation and search.

Click **Learn more** in the template to view relevant information in the **Template details** pop-up window, which can be downloaded.



Fuzzy search	invitation Separate multipl	e tags with carriage returns		Q Total: 4				Sort by recommendation 🔻
	CustomInvitation	Learn more	AddTextToP	lictures	Learn more	AddTextToPi	ctures	Learn more
	Category Function		Category	Function		Category	Function	
	Description This example name to cloud	demonstrates from http api post d function, add name to the	Description	This example demonstrates fror name to cloud function, add na	m http api post me to the	Description	This example demonstra name to cloud function,	tes from http api post add name to the
	Tag Nodejs12.16	5 HTTP REST API	Tag	Python3.6 HTTP REST	API	Tag	Nodejs10.15 HTTP	REST API
	FileProcess			FileProcess			FileProcess	
	CA 🔗 Tencent C	loud	CA	🔗 Tencent Cloud		CA	🙆 Tencent Cloud	
	Deploy 14,255次		Deploy	10,220次		Deploy	7,336次	
	AddrextroPictures Category Function Description This example name to cloud Tag Python2.7 FileProcess CA 2 Tencent C Database 12.0137b	demonstrates from http api post d function, add name to the HTTP REST API						
	Deploy 13,011/X							

4. Click Next. The function name is automatically generated by default and can be modified as needed. Follow the prompts to enter the corresponding values of the environment variables required by the template in the Basic Configurations section and keep other parameters as default as shown below:



Function name *	Ac	actuding letters numbers underscore	st and	where it must start with a latter and and with a number or latter
Region *	S Guangzhou	v		genere is meet aan emer a recer and ene mer a nemeer er recer.
Description *	This example demonstrates fr function, add name to the linu upload the processed picture address to the calling termina	rom http api post name to cloud itation letter in cloud function, to COS, and return the download al.		
	Up to 1000 letters, digits, space	s, commas, and periods.		
Environment variable *	key	value		0
	region	the region of target bucke	X	
	target_bucket	target bucket name	×	
	target_path	path of target bucket	Х	
Execution Role *	Enable ① To ensure that the function tem O Configure and use SCF temp	plate can access other Tencent Clouc	l servic	es, please configure and use the SCF template role, or select an existing role that includes QcloudCOSFullAccess preset policies.
	Use the existing role			
Function Code	es Runtime: Python2.7 Execu	ution method: index:main_handler		

Configure environment variables as shown below:

variable *	key	value	
	region	the region of target bucke	×
	target_bucket	target bucket name	×
	target_path	path of target bucket	×

key	value
region	Bucket region, which starts with `ap-` and ends with the region name. This document takes `ap- guangzhou` as an example.

target_bucket	Name of the created bucket.
target_path	Path of the destination bucket. To view the directory path of the bucket for storing invitations, go to Bucket List and enter the corresponding bucket . For example, if the directory is `example`, the destination path here is `/example`. If there is no directory created in the bucket, enter `/` here.

Note:

This example requires SCF to have the operation permissions of COS. **Execution Role** has been enabled by default, and an execution role has been automatically created and associated with the required COS permission policy

QcloudCOSFullAccess . If you need to make adjustments, select Use the existing role or disable Execution

Role.

5. Click **Complete**.

Generating invitation

You can generate an invitation in two ways:

Note:

You can get the access path of the API Gateway trigger on the **Function Details > Trigger management** page as shown below:

Create trigger	
API Gateway trigger Tr	riggered alias: Default traffic
API Name	SCF_APLSERVICE 12
serviceld	
apild	
Request method	ANY
Publishing environment	Publish
Authentication method	No authentication
Enable integration response	Enabled
Enable Base64 encoding	Not enabled
Support CORS	No
Backend timed out	15s
Tag	Not enabled
Access path ①	Public https://service-p8jq2luu-1259724985.gr

Method 1

Run the following command on the command line.



curl API Gateway trigger URL -d 'Name of the invited guest'

The download address of the invitation can be obtained on the device, such as:



Method 2

1. Download the HTML page and change the URL to the URL of the API Gateway trigger as shown below:



2. Open the HTML page, enter the name of the invited guest to generate a poster, and access the URL to download it.

TRTC Practices Using SCF to Input Online Media Streams for SCF +TRTC

Last updated : 2024-12-02 16:35:34

Use Cases

Cases

Online education

In one-to-one or one-to-many small classes, you can record streams in multiple dimensions for different students: For one single student, you can record the student's separate data stream to compose relevant data. In this way, you can record the highlights of each student and push them to their parents.

You can directly record the data in the room and generate videos that students can play back and watch repeatedly for learning.

In order to facilitate repeated watches, redundant data can be removed during recording.

Customer service center

In smart customer service scenarios, you can record the separate data streams of users and combine the data with recognition APIs to implement information verification during card application and smart account opening. You can record the speeches of the customer service representatives and users separately and automatically recognize keywords, so as to evaluate the service quality and iteratively train the smart customer service. You can save and archive the data generated in the course of service.

Social networking

You can record only the desired video streams in the room for data retention, which helps reduce the costs of storage and stream mix.

You can record the specified data in the room, call moderation APIs for content moderation, and set different moderation standards for different users.

You can directly generate segment files based on live streams.

Business process

This document describes how to use API Gateway and SCF to record a single anchor's audio/video stream in a TRTC room and upload the recording file to COS for storage. This solution provides out-of-the-box, flexible, convenient, and

programmable live recording capabilities. SCF offers 512 MB of memory by default for recording file storage. If you need more storage space, you can choose to use the mount capability of CFS. The workflow is as shown below:



The parameters for calling APIs are as follows:

Parameter	Туре	Required	Description
SdkAppld	Int	Yes	Application ID, which is used to distinguish different TRTC applications.
RoomId	Int	Yes	Room ID in integer type, which is used to uniquely identify a room in a TRTC application.
StrRoomId	String	No	Room ID in string type. Either RoomId or StrRoomId must be configured. If both are configured, RoomId will be used.
UserId	String	Yes	Recorded user ID, which is used to uniquely identify a user in a TRTC application.
UserSig	String	Yes	Recorded user signature, which is used to authenticate the user login.
CosConfig	cosConfig	Yes	COS storage configuration for recording file storage.
Callback	String	No	Callback address after recording. The POST method is used for callback.



Mode	String	No	00: single audio stream output in MP3 format (default mode).01: single video stream output in MP4 format.02: single audio/video stream output in MP4 format.
------	--------	----	--

The parameters involved in CosConfig are as follows:

Parameter	Туре	Required	Description
SecretId	String	No	SecretId of Tencent Cloud account. For more information, please see Root Account Access Key Management.
SecretKey	String	No	SecretKey of Tencent Cloud account. For more information, please see Root Account Access Key Management.
Region	String	Yes	COS region, such as ap-guangzhou .
Bucket	String	Yes	Bucket name, such as susu-123456789.
Path	String	Yes	Path in the bucket. For example, for /test , the root directory is / .

Note:

UserId is the specified user ID. Idempotency is not guaranteed for multiple requests to API Gateway.

If SecretId and SecretKey are not configured in CosConfig , the function will use the permission of the execution role SCF_ExecuteRole when accessing COS.

Trigger conditions for stopping recording:

The TRTC room is terminated. When there is no anchor in the TRTC room for more than 300s, the room will be automatically terminated.

The user removal API is actively called to kick the recorded user out of the room.

To stop recording for users with RoomId, you need to call the RemoveUser API.

To stop recording for users with StrRoomId , you need to call the RemoveUserByStrRoomId API.

After recording is stopped, the data returned by the function is as follows:

Parameter	Туре	Required	Description
SdkAppId	String	Yes	Application ID.
RoomId	String	Yes	Room ID in integer type.
UserId	String	Yes	Recorded user ID.
StrRoomId	String	Yes	Room ID in string type.



Files	Array	Yes	[{},{},{},{},{}]]

Note:

If callback is configured, after the stop, SCF will pass the returned data to the callback address in POST method. Each item in the Files array is a JSON object as follows:

Parameter	Туре	Required	Description
UserId	String	Yes	Recorded user ID.
RecordFile	String	Yes	URL of the recording file uploaded to COS.
Status	Int	Yes	0: failure. 1: success.
Message	String	Yes	Execution result of the recording task, such as recording failed, transcoding failed, and write to COS failed.

Directions

Creating function

1. Log in to the SCF console and select **Function Service** on the left sidebar.

2. At the top of the **Function Service** page, select the **Guangzhou** region and click **Create** to enter the function creating page and configure the function as shown below:

÷	Crea	ate		
		Create Method		
			Template Custom Use demo template to create a function Create a custom function using or application HelloWolrd demo	
		Fuzzy search	TRTC Separate multiple tags with carriage returns Q Total: 1	Sort by recomm- 💌
			TRTC Learn More	
			Category Function	
			Description This example demonstrates transcoding video files to TRTC push streaming based on	
			Tag JavaB TRTC ffmpeg video	
			Author 🔗 Tencent Cloud	
			Deploy 9,247 time	
		Next Car	and	
		Car		

Creation method: select Template.

Fuzzy search: enter "TRTC", search, and select the Single audio/video stream recording template.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click **Next** and configure the related information as shown below:



Basic Config	urations
Function	TRTC-1620439282
name *	It supports 2 to 60 characters, including letters, numbers, underscores and hyphens. It must start with a letter and end with a number or letter.
Region *	🕲 Guangzhou 🔻
Description *	This example demonstrates transcoding video files to TRTC push streaming based on ffmpeg.
	Up to 1000 letters, digits, spaces, commas, and periods.
Async	✓ Enable ③
Execution *	When async execution is enabled, the function execution timeout period can be 24 hours at most. You can modify it as needed. Please specify the log publishing topic in log configurations.
Status Trace *	✓ Enable (j)
Execution	43200 s (i)
timeout period *	Range: 1 to 86400 seconds

Function Name: the function name is automatically generated by default.

Async Execution: select the checkbox to enable the async execution. When it's enabled, the function will respond to the event in async execution mode. The event goes to async execution status after being invoked without waiting for the processing result.

Status Trace: select the checkbox to enable the status trace. When it's enabled, it will keep the logs of real-time status of response for async function events. You can query and stop the event and check the related statistics. Data of event status will be retained for three days.

Execution Timeout Period: it can be modified as needed.

Execution Role: **SCF_ExecuteRole** is used as the execution role by default, which grants the access permissions of **QcloudCOSFullAccess** and **QcloudCFSFullAccess**.

4. Configure an API Gateway trigger. An API service is created by default and the integration response is disabled. You can customize the trigger, but please ensure that the integration response is disabled as shown below:

ate a Trigger	• Automatic creation	
	Triggered Version	Default Traffic 🔹 🗘
	Trigger Method	API Gateway Trigger 🔻 🗘
		For API gateway triggers, the format of contents returned from SCF should be constructed in integration response method. For details, please see here .
	API Service Type 🛈	O Create API Service Use Existing API Service
	API Service	SCF_API_SERVICE
	Request method 🛈	ANY
	Publishing Environment	Publish 💌
	Authentication Method ()	No authentication
	Integration Response 🛈	Enable
	Base64 Encoding	Enable

5. Click **Complete**.

6. If you need to use the mount capability of CFS, as CFS can only be accessed in VPC, you should use the VPC of CFS as the VPC of the function as shown below:

Network Co	nfiguration		
Public network	✓ Enable (j)		
Fixed outbound IP	Enable 🚯		
VPC	✓ Enable 🚯		
	Please select the VPC	▼ Please select a subnet	🔻 🗘 Create VPC 🛂



Note:

```
To enable CFS, you need to set the environment variable CFS_PATH to a local directory, such as /mnt/audio/ .
```

Creating TRTC application

1. Log in to the TRTC console and select **Development Assistance** > **Demo Quick Run** on the left sidebar.

2. Enter the demo name and click **Create**. You can choose a template for test run according to your client, such as the demo for desktop browser.

		N (2) H H	
Demo Name Name your demo	2 Download Source Code	Configuration	Compile and Run
Create Reset			

Testing function

1. Create a TRTC application and enter the application.

2. Use Postman to construct an HTTP request, where roomId is the room ID of the created TRTC application, and userId is the ID of another random user (which must be unique). Below is the sample code:

```
{
    "SdkAppId": 140000000,
    "RoomId": 43474,
    "UserId": "user_55952145",
    "Mode": "02",
    "UserSig": "eJwtzNEKgkAUBNB-2efQ3e3eUqG3tMCKJJEIIxxxxxxxxxhvmweLWzGlUxj0mL
    "CosConfig": {
        "Region": "ap-shanghai",
        "Bucket": "test-123456789",
        "Path": "/trtc"
        }
}
```



}

See the figure below:

POST 🗸	https://service-€ 226.gz.apigw.tencentcs.com/re	leas	Params	Send	~	Save
Authorization	Headers Body • Pre-request Script Tests					
form-data	🕽 x-www-form-urlencoded 🔎 raw 🔍 binary 🛛 Text 🗸					
1 { 2 "SdkA 3 "Roon 4 "User 5 "Mode 6 "User 7 "CosC 8 " 9 " 10 " 11 } 12 }	AppId": 1400 nId": 43474, rId": "user_55952145", s":"02", rsig": "eJwtzNEKgkAUBNB-2ef rGFUdUR0UQrAYWDyWGY15cwTwDm Config": {{ region": "ap-shanghai". 'bucket": "test26", 'path": "/trcc	mLs1GXKVf3mgrq hCw1UW*fE4oWusw3dUL	1J7HoSJ2e6d9fM8Y9	8fxUAzWA"	,	
in the J						

3. After the request is sent, an async function response "Async run task submitted" will be received. The RequestId of this function will be returned through x-scf-reqid in the HTTP header as shown below:

POST 🗸	https://service-!	Params	Send					
content-type → application/json; charset=utf-8								
date → Tue, 20 Apr 2021 08:46:21 GMT								
transfer-encoding → chunked								
vary → Accept-Enco	bding							
x-api-appid → 125								
x-api-funcname →	TRTC-161							
x-api-httphost → s	ervice-53 .tencentcs.com							
x-api-id → api-								
x-api-ratelimit-seco	nd → unlimited							
x-api-requestid →	7763a27545027178865993							
x-api-serviceid →	ervice							
x-api-status → 200								
x-api-upstreamstate	is → 200							
x-scf-reqid →	f77763a27545027178865993							
x-scf-status → 200								
x-service-ratelimit-	econd → 4999/5000							

4. On the **Function Service** page in the SCF console, click the name of the function created in the Creating function step above to enter the function details page.

5. Select the **Log Query** tab on the function details page to view the printed recording log information as shown below:

÷	CF Document				
Function Management	Log Query				
Trigger Management	Invocation Logs Advanced Retrie	al			
Monitoring Information	Version: \$LATEST 💌 All Logs	▼ Last 15 minu ▼ 2022-01-05 16:51:15 ~ 2022-01-05 17:06:15 🛅 Refresh	Please enter the req Q		
Log Query	No log information	Request ID: :	statusCode Description and Solution [
Concurrency Quota		Time: Runtime: Execution memory:			
Deployment Logs		Log:	6		

6. Log in to the TRTC console and click the room ID on the **Monitoring Dashboard** page to view all users in the room, one of whom is the recorded user.

7. To stop recording, you can call the RemoveUser or RemoveUserByStrRoomId API to move the user out of the room.

SCF + TRTC for Stream Single Recording

Last updated : 2024-12-02 16:35:34

Use Cases

Cases

Online education

In one-to-one or one-to-many small classes, you can record streams in multiple dimensions for different students: For one single student, you can record the student's separate data stream to compose relevant data. In this way, you can record the highlights of each student and push them to their parents.

You can directly record the data in the room and generate videos that students can play back and watch repeatedly for learning.

In order to facilitate repeated watches, redundant data can be removed during recording.

Customer service center

In smart customer service scenarios, you can record the separate data streams of users and combine the data with recognition APIs to implement information verification during card application and smart account opening. You can record the speeches of the customer service representatives and users separately and automatically recognize keywords, so as to evaluate the service quality and iteratively train the smart customer service. You can save and archive the data generated in the course of service.

Social networking

You can record only the desired video streams in the room for data retention, which helps reduce the costs of storage and stream mix.

You can record the specified data in the room, call moderation APIs for content moderation, and set different moderation standards for different users.

You can directly generate segment files based on live streams.

Business process

This document describes how to use API Gateway and SCF to record a single anchor's audio/video stream in a TRTC room and upload the recording file to COS for storage. This solution provides out-of-the-box, flexible, convenient, and programmable live recording capabilities. SCF offers 512 MB of memory by default for recording file storage. If you need more storage space, you can choose to use the mount capability of CFS. The workflow is as shown below:



The parameters for calling APIs are as follows:

Parameter	Туре	Required	Description
SdkAppld	Int	Yes	Application ID, which is used to distinguish different TRTC applications.
Roomld	Int	Yes	Room ID in integer type, which is used to uniquely identify a room in a TRTC application.
StrRoomId	String	No	Room ID in string type. Either RoomId or StrRoomId must be configured. If both are configured, RoomId will be used.
Userld	String	Yes	Recorded user ID, which is used to uniquely identify a user in a TRTC application.
UserSig	String	Yes	Recorded user signature, which is used to authenticate the user login.
CosConfig	cosConfig	Yes	COS storage configuration for recording file storage.
Callback	String	No	Callback address after recording. The POST method is used for callback.
Mode	String	No	00: single audio stream output in MP3 format (default mode). 01: single video stream output in MP4 format.



02: single audio/video stream output in MP4 format.

Parameter	Туре	Required	Description
SecretId	String	No	SecretId of Tencent Cloud account. For more information, please see Root Account Access Key Management.
SecretKey	String	No	SecretKey of Tencent Cloud account. For more information, please see Root Account Access Key Management.
Region	String	Yes	COS region, such as ap-guangzhou .
Bucket	String	Yes	Bucket name, such as susu-123456789.
Path	String	Yes	Path in the bucket. For example, for /test , the root directory is / .

The parameters involved in CosConfig are as follows:

Note:

UserId is the specified user ID. Idempotency is not guaranteed for multiple requests to API Gateway.

If SecretId and SecretKey are not configured in CosConfig , the function will use the permission of the execution role SCF_ExecuteRole when accessing COS.

Trigger conditions for stopping recording:

The TRTC room is terminated. When there is no anchor in the TRTC room for more than 300s, the room will be automatically terminated.

The user removal API is actively called to kick the recorded user out of the room.

To stop recording for users with RoomId , you need to call the RemoveUser API.

To stop recording for users with StrRoomId , you need to call the RemoveUserByStrRoomId API.

After recording is stopped, the data returned by the function is as follows:

Parameter	Туре	Required	Description
SdkAppId	String	Yes	Application ID.
RoomId	String	Yes	Room ID in integer type.
UserId	String	Yes	Recorded user ID.
StrRoomId	String	Yes	Room ID in string type.
Files	Array	Yes	[{},{},{},{})]



Note:

If callback is configured, after the stop, SCF will pass the returned data to the callback address in POST method. Each item in the Files array is a JSON object as follows:

Parameter	Туре	Required	Description
Userld	String	Yes	Recorded user ID.
RecordFile	String	Yes	URL of the recording file uploaded to COS.
Status	Int	Yes	0: failure. 1: success.
Message	String	Yes	Execution result of the recording task, such as recording failed, transcoding failed, and write to COS failed.

Directions

Creating function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Guangzhou** region and click **Create** to enter the function creating page and configure the function as shown below:

Cre	ate				
	Create Method	✓			
		Template Cust	om		
		Use demo template to create a function Creat	e a custom function using		
		or application Hello	Wolrd demo		
	Fuzzy search	TRTC Separate multiple tags with carriage retur		Q Total: 1	Sort by reco
		TRIC	Learn More		
		Category Function			
		Description This example demonstrates transc	oding video		
		files to TRTC push streaming based	lon		
		Tag Java8 TRTC ffmpeg vid	leo		
		Author 🔗 Tencent Cloud			
		Daplay 0.247 time			
		Deploy 9,247 unie			
	Next Ca	cel			
	Cal				

Creation method: select Template.

Fuzzy search: enter "TRTC", search, and select the Single audio/video stream recording template.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click **Next** and configure the related information as shown below:



Basic Config	urations
Function	TRTC-1620439282
name *	It supports 2 to 60 characters, including letters, numbers, underscores and hyphens. It must start with a letter and end with a number or letter.
Region *	🕲 Guangzhou 🔻
Description *	This example demonstrates transcoding video files to TRTC push streaming based on ffmpeg.
	Up to 1000 letters, digits, spaces, commas, and periods.
Async	✓ Enable ③
Execution *	When async execution is enabled, the function execution timeout period can be 24 hours at most. You can modify it as needed. Please specify the log publishing topic in log configurations.
Status Trace *	✓ Enable (j)
Execution	43200 s (i)
timeout period *	Range: 1 to 86400 seconds

Function Name: the function name is automatically generated by default.

Async Execution: select the checkbox to enable the async execution. When it's enabled, the function will respond to the event in async execution mode. The event goes to async execution status after being invoked without waiting for the processing result.

Status Trace: select the checkbox to enable the status trace. When it's enabled, it will keep the logs of real-time status of response for async function events. You can query and stop the event and check the related statistics. Data of event status will be retained for three days.

Execution Timeout Period: it can be modified as needed.

Execution Role: **SCF_ExecuteRole** is used as the execution role by default, which grants the access permissions of **QcloudCOSFullAccess** and **QcloudCFSFullAccess**.

4. Configure an API Gateway trigger. An API service is created by default and the integration response is disabled. You can customize the trigger, but please ensure that the integration response is disabled as shown below:

eate a Trigger	O Automatic creation	
	Triggered Version	Default Traffic 🔻 🗘
	Trigger Method	API Gateway Trigger 🔻 🗘
		For API gateway triggers, the format of contents returned from SCF should be constructed in integration response method. For details, please see here .
	API Service Type 🛈	O Create API Service Use Existing API Service
	API Service	SCF_API_SERVICE
	Request method 🛈	ANY
	Publishing Environment	Publish 🔻
	Authentication Method 🛈	No authentication
	Integration Response()	Enable
	Base64 Encoding	Enable
	Custom	

5. Click **Complete**.

6. If you need to use the mount capability of CFS, as CFS can only be accessed in VPC, you should use the VPC of CFS as the VPC of the function as shown below:

Network Co	nfiguration		
Public network	✔ Enable (j)		
Fixed outbound IP	Enable 🚯		
VPC	🗸 Enable (i)		
	Please select the VPC	▼ Please select a subnet	🔻 🗘 Create VPC 🖬



Note:

```
To enable CFS, you need to set the environment variable CFS_PATH to a local directory, such as /mnt/audio/ .
```

Creating TRTC application

1. Log in to the TRTC console and select **Development Assistance** > **Demo Quick Run** on the left sidebar.

2. Enter the demo name and click **Create**. You can choose a template for test run according to your client, such as the demo for desktop browser.

Demo Name your demo	Demo Name Vour demo Create Reset	Demo Name Na	ame your demo		
	Create				
Create Reset		Create	Reset		

Testing function

1. Create a TRTC application and enter the application.

2. Use Postman to construct an HTTP request, where roomId is the room ID of the created TRTC application, and userId is the ID of another random user (which must be unique). Below is the sample code:

```
{
    "SdkAppId": 140000000,
    "RoomId": 43474,
    "UserId": "user_55952145",
    "Mode": "02",
    "UserSig": "eJwtzNEKgkAUBNB-2efQ3e3eUqG3tMCKJJEIIxxxxxxxxxhvmweLWzGlUxj0mL
    "CosConfig": {
        "Region": "ap-shanghai",
        "Bucket": "test-123456789",
    }
}
```



```
"Path": "/trtc"
}
```

See the figure below:

}

POS	https://service-e 226.gz.apigw.tencentcs.com/releas		Params	Send	~	Save
Authoriza	tion Headers Body • Pre-request Script Tests					
form-	data 🔍 x-www-form-urlencoded 🔎 raw 🔍 binary Text 🗸					
2 3 4 5	"SdkAppId": 1400, "RoomId": 43474, "UserId": "user_55952145", "Mode":"02",					
6	"UserSig": "eJwtzNEKgkAUBNB-2ef *GFUdUR0UQrAYWDyWGY15cwTwDm "CosConfig": {{	mLs1GXKVf3mgrq hCwlUW*fE4oWusw3dULlJ7Ho	SJ2e6d9fM8Y9	8fxUAzWA"	,	
8 9 10	"region": "ap-shanghai". "bucket": " →test- 26", "path": "/true					
11 12 }	N					

3. After the request is sent, an async function response "Async run task submitted" will be received. The RequestId of this function will be returned through x-scf-reqid in the HTTP header as shown below:

POST 🗸	https://service-!	Params	Send
content-type → ap	plication/json; charset=utf-8		
date → Tue, 20 Apr	2021 08:46:21 GMT		
transfer-encoding -	• chunked		
vary → Accept-Enco	bding		
x-api-appid → 125			
x-api-funcname →	TRTC-16 ⁻		
x-api-httphost → s	ervice-53 .tencentcs.com		
x-api-id → api-			
x-api-ratelimit-seco	nd → unlimited		
x-api-requestid →	7763a27545027178865993		
x-api-serviceid →	ervice		
x-api-status → 200			
x-api-upstreamstat	us → 200		
x-scf-reqid →	f77763a27545027178865993		
x-scf-status → 200			
x-service-ratelimit-	second → 4999/5000		

4. On the **Function Service** page in the SCF console, click the name of the function created in the Creating function step above to enter the function details page.

5. Select the **Log Query** tab on the function details page to view the printed recording log information as shown below:

÷			SCF Document
Function Management	Log Query		
Trigger Management	Invocation Logs Advanced Retrieva		
Monitoring Information	Version: \$LATEST 🔻 All Logs	Last 15 minu * 2022-01-05 16:51:15 ~ 2022-01-05 17:06:15 💼 Refresh	Please enter the req Q
Log Query	No log information	Request ID: :	statusCode Description and Solution
Concurrency Quota		Time: Runtime: Execution memory:	
Deployment Logs		Log:	6

6. Log in to the TRTC console and click the room ID on the **Monitoring Dashboard** page to view all users in the room, one of whom is the recorded user.

7. To stop recording, you can call the RemoveUser or RemoveUserByStrRoomId API to move the user out of the room.

SCF + TRTC for Stream Mix Recording

Last updated : 2024-12-02 16:35:34

Use Cases

Cases

Online education

In the online education scenario, this solution can compose and record the audios/videos of the teacher and students during class and add other materials and AI analysis capability to restore the in-person class experience. It can also add other business features to enrich the video playback effect.

Social live streaming

During live streaming, this solution can use the stream mix recording method to mix multiple streams and relay them for audit, thereby reducing the audit costs. It can also store the recording files according to applicable regulations to meet regulatory requirements.

Financial regulation

In scenarios such as online account opening and financial audiovisual recording, this solution can record all transaction processes from a global perspective and archive the recording files to meet subsequent service or regulatory requirements.

Others

In customer service and complaint handling scenarios, this solution can record the service process and complaint content in real time, making it easier to optimize the service quality, check the reasonableness of complaints, and improve the efficiency of complaint handling.

Business process

This document describes how to use API Gateway and SCF to mix record the anchor audio/video streams in a TRTC room and upload the recording file to COS for storage. This solution provides out-of-the-box, flexible, convenient, and programmable live recording capabilities. SCF offers 512 MB of memory by default for recording file storage. If you need more storage space, you can choose to use the mount capability of CFS. The workflow is as shown below:



The recording rules are as follows:

Up to 6 audio/video streams can be recorded.

If there are less than 6 users in the room, streams of subsequent users entering the room will be automatically mixed. If there are more than 6 users in the room, only streams of the first 6 users entering the room will be recorded. If IsReserve is true, after the user exits the room, the user's stream will still be in the stream mix task, and the last frame of video image and mute will be used to complement the stream. If IsReserve is false, after the user exits the room, the user's stream will be deleted from the stream mix task. If a new user enters the room later, the new user's stream will be added to the stream mix.

The parameters for calling APIs are as follows:

Parameter	Туре	Required	Description
SdkAppld	Int	Yes	Application ID, which is used to distinguish different TRTC applications.
RoomId	Int	Yes	Room ID in integer type, which is used to uniquely identify a room in a TRTC application.
StrRoomId	String	No	Room ID in string type. Either RoomId or StrRoomId must be configured. If both are configured, RoomId will be used.
UserId	String	Yes	Recorded user ID, which is used to uniquely identify a user in a TRTC application.
UserSig	String	Yes	Recorded user signature, which is used to authenticate the user login.
-----------	-----------	-----	---
CosConfig	cosConfig	Yes	COS storage configuration for recording file storage.
Callback	String	No	Callback address after recording. The POST method is used for callback.
Mode	String	No	10: mixed audio stream output in MP3 format (default mode).11: mixed video stream output in MP4 format.12: mixed audio/video stream output in MP4 format.
IsReserve	Boolean	No	false: when the anchor exits the room, the stream being mixed will be automatically deleted. true: when the anchor exits the room, the last frame of the video image and mute will be used to complement the stream in the stream mix task. The default value is true .

The parameters involved in CosConfig are as follows:

Parameter	Туре	Required	Description
SecretId	String	No	SecretId of Tencent Cloud account. For more information, please see Root Account Access Key Management.
SecretKey	String	No	SecretKey of Tencent Cloud account. For more information, please see Root Account Access Key Management.
Region	String	Yes	COS region, such as ap-guangzhou .
Bucket	String	Yes	Bucket name, such as susu-123456789.
Path	String	Yes	Path in the bucket. For example, for /test , the root directory is / .

Note:

UserId is the specified user ID. Idempotency is not guaranteed for multiple requests to API Gateway.

If SecretId and SecretKey are not configured in CosConfig , the function will use the permission of the execution role SCF_ExecuteRole when accessing COS.

Trigger conditions for stopping recording:

The TRTC room is terminated. When there is no anchor in the TRTC room for more than 300s, the room will be automatically terminated.

The user removal API is actively called to kick the recorded user out of the room.

To stop recording for users with RoomId , you need to call the RemoveUser API.

To stop recording for users with StrRoomId , you need to call the RemoveUserByStrRoomId API.

After recording is stopped, the data returned by the function is as follows:

Parameter	Туре	Required	Description
SdkAppId	String	Yes	Application ID.
Roomld	String	Yes	Room ID in integer type.
Userld	String	Yes	Recorded user ID.
StrRoomId	String	Yes	Room ID in string type.
Files	Array	Yes	[{},{},{},{}]

Note:

If callback is configured, after the stop, SCF will pass the returned data to the callback address in POST method. Each item in the Files array is a JSON object as follows:

Parameter	Туре	Required	Description
Userld	String	Yes	Recorded user ID.
RecordFile	String	Yes	URL of the recording file uploaded to COS.
Status	Int	Yes	0: failure. 1: success.
Message	String	Yes	Execution result of the recording task, such as recording failed, transcoding failed, and write to COS failed.

Directions

Creating function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Guangzhou** region and click **Create** to enter the function creating page and configure the function as shown below:

Use demo templa application	ite to create a function or	Start from a Hello World sample		Create a function based on a TCR in	nage		
Fuzzy search	TRTC Separate multiple t	ags with carriage returns		Q #3个			Sort by recom
	TRTC	Learn More	SingleReco	rder	Learn More MixRecor	der	Learn M
	Category Function		Category	Function	Category	Function	
	Description This examp files to TRT	e demonstrates transcoding video push streaming based on ffmpeg.	Description	This example demonstrates single-st recording based on TRTC service, and	ream Description	This example demonstrates m recording based on TRTC servi	x-stream ce, and push
	Tag Java8	TRTC ffmpeg video	Tag	Python3.6 TRTC audio CF	S Tag	Python3.6 TRTC audio	CFS
	Author 🔗 Tencent	Cloud		COS ffmpeg video		COS ffmpeg video	
	Deploy 9,532次		Author	🙆 Tencent Cloud	Author	🙆 Tencent Cloud	
			Deploy	9,763次	Deploy	7,985次	

Creation method: select Template.

Fuzzy search: enter "TRTC", search, and select the Mix audio/video stream recording template.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click **Next** and configure the related information as shown below:



Function name *	MixRecorder-1642905408
	It supports 2 to 60 characters, including letters, numbers, underscores and hyphens. It must start with a letter and end with a number o
Region *	🔇 Guangzhou 🔹
Description *	This example demonstrates mix-stream recording based on TRTC service, and push recording files to cos bucket.
	Up to 1000 letters, digits, spaces, commas, and periods.
Execution Role *	✓ Enable (j)
	To ensure that the function template can access other Tencent Cloud services, please configure and use the SCF template role, or select
	O Configure and use SCF template role 🚯
	Use the existing role
Async	✓ Enable (i)
Execution *	When async execution is enabled, the function execution timeout period can be 24 hours at most. You can modify it as needed. Please specify the log publishing topic in log configurations.
Status Trace *	✓ Enable (i)
Function Code	Runtime: Python3.6 Execution Method: index.main_handler

Function Name: the function name is automatically generated by default.

Async Execution: select the checkbox to enable the async execution. When it's enabled, the function will respond to the event in async execution mode. The event goes to async execution status after being invoked without waiting for the processing result.

Status Trace: select the checkbox to enable the status trace. When it's enabled, it will keep the logs of real-time status of response for async function events. You can query and stop the event and check the related statistics. Data

of event status will be retained for three days.

Execution Timeout Period: it can be modified as needed.

Execution Role: **SCF_ExecuteRole** is used as the execution role by default, which grants the access permissions of **QcloudCOSFullAccess** and **QcloudCFSFullAccess**.

4. Configure an API Gateway trigger. An API service is created by default and the integration response is disabled.

You can customize the trigger, but please ensure that the integration response is disabled as shown below:

eate a Trigger	• Automatic creation	
	Triggered Version	Default Traffic 🔹
	Trigger Method	API Gateway Trigger 🔻 🗘
		For API gateway triggers, the format of contents returned from SCF should be constructed in integration response method. For details, please seehere 🗹 .
	API Service Type 🛈	O Create API Service 📃 Use Existing API Service
	API Service	SCF_API_SERVICE
	Request method 🛈	ANY
	Publishing Environment(Publish
	Authentication Method 🛈	No authentication
	Integration Response 🛈	Enable
	Base64 Encoding 🕢	Enable
	Custom	
	Create Later	

5. Click **Complete**.



6. If you need to use the mount capability of CFS, as CFS can only be accessed in VPC, you should use the VPC of CFS as the VPC of the function as shown below:

e			
/PC	🗸 Enable (
		16	▼ .0/18 ▼ Ø Create VPC 🗹
ile System			
File System	🖌 Enable 🛈		
	51.0.1		
	File System ID	Please select the file system ID	▼ Ø Create File System 🗹
	Mount Point ID	Please select the mount point IC	l ▼ Ø
	User ID	10000	
	User Group	10000	
	ID		
	Remote	1	
	Directory		
	Local	/mnt/	
	Directory		

Note:

To enable CFS, you need to set the environment variable CFS_PATH to a local directory, such as

/mnt/audio/ .

Creating TRTC application

1. Log in to the TRTC console and select **Development Assistance** > **Demo Quick Run** on the left sidebar.

2. Enter the demo name and click **Create**. You can choose a template for test run according to your client, such as the demo for desktop browser.

1 Create App	olication > (2 Download Sou Code	irce > 3 Mo Col	odify nfiguration	>	4 Compile and Run
Application Type	O New Existing					
Application Name	Name your demo					
Tag 🛈	Tags allow you to mana + Add	ge resources by category	γ. If existing tags do not meet you	r requirements, you can n	nanage tags	here 🖸 .
Create	Reset					
Quick Guide						

Testing function

1. Refer to Web to make user_00001 and user_00002 enter the TRTC room.

2. Use Postman to construct an HTTP request, where roomId is the room ID of the created TRTC application, and userId is the ID of another random user (which must be unique). Below is the sample code:

```
{
    "SdkAppId": 140000000,
    "RoomId": 43474,
    "UserId": "user_55952145",
    "Mode":"12",
    "UserSig": "eJwtzNEKgkAUBNB-2efQ3e3eUqG3tMCKJJEIIxxxxxxxxhvmweLWzGlUxj0mL
    "CosConfig": {
        "Region": "ap-shanghai",
        "Bucket": "test-123456789",
        "Path": "/trtc"
        },
        "IsReserve":false,
        "Callback":"https:xxxxxx.com/post/xxx"
}
```

See the figure below:



POS	τv	http://service-j4adyq26-1253970226.gz.apigw.tencentcs.com/release/MixRecorder-1623317776		Params	Send	✓ Save
Authoriza	ation H	Headers Body • Pre-request Script Tests				
form-	-data 🔍	x-www-form-urlencoded				
1 { 2 3 4 5 6 7 8 9 10 11 12 }	"SdkAp "RoomI "Wode" "UserS "CosCo "R "B "P }	ppId": 140 Id": 'user_'', ": "12", Sig": "eJwtzMsKglAUheF30eOwfS5bSV SKFVNPZxFlYVRQzmysAlEIucHz0q6Ja946IAgBC onfig": { Region": "ap-shanghai" Bucket": "susu- Path": "/trtc"	Ho 612nqB*1w7	7S-b9AQ5EMtM_	".,	
Body	Cookies	Headers (15) Test Results			Status: 200 OI	< Time: 329
Pretty	Raw	Preview JSON V				Ū
🛛 1 🛛 Un	expected	'A'				

3. After the request is sent, an async function response "Async run task submitted" will be received. The RequestId of this function will be returned through x-scf-reqid in the HTTP header as shown below:

POST 🗸	https://service-!	Params	Send				
content-type → ap	content-type → application/json; charset=utf-8						
date → Tue, 20 Apr	2021 08:46:21 GMT						
transfer-encoding –	transfer-encoding → chunked						
vary → Accept-Encoding							
x-api-appid → 125	x-api-appid → 125						
x-api-funcname →	x-api-funcname → TRTC-16'						
x-api-httphost → s	x-api-httphost → service-53 .tencentcs.com						
x-api-id → api-							
x-api-ratelimit-seco	nd → unlimited						
x-api-requestid →	7763a27545027178865993						
x-api-serviceid →	x-api-service → service						
x-api-status → 200							
x-api-upstreamstate	is → 200						
x-scf-reqid →	f77763a27545027178865993						
x-scf-status → 200							
x-service-ratelimit-	econd → 4999/5000						

4. On the **Function Service** page in the SCF console, click the name of the function created in the Creating function step above to enter the function details page.

5. Select the **Log Query** tab on the function details page to view the printed recording log information as shown below:

Function Management	Log Query
Trigger Management	Invocation Logs Advanced Retrieval
Monitoring Information	Version: \$LATEST • All Logs • Last 15 minu • 2022-01-23 10:51:30 ~ 2022-01-23 11:06:30 🖬 Refresh
Log Query	No log information Request ID: :
Concurrency Quota	Time: Runtime: Execution memory:
Event Management	Log:
Deployment Logs	

6. Log in to the TRTC console and click the room ID on the **Monitoring Dashboard** page to view all users in the room, one of whom is the recorded user.

7. To stop recording, you can call the RemoveUser or RemoveUserByStrRoomId API to move the user out of the room.

COS Practices Audio/Video Transcoding

Last updated : 2024-12-02 16:35:34

Overview

In use cases such as video and social networking, users upload high numbers of images and audio/video files frequently, which has high requirements for the real-timeness and concurrency capabilities of the processing system. Multiple functions can be used to process uploaded videos with different definitions in order to meet the needs of users in diversified scenarios and adapt to small-bandwidth and unstable mobile networks.

How It Works

SCF, FFmpeg, and COS can be used in combination as shown below to transcode audio and video files:



In SCF, you can write custom business logic in different programming languages (Python, Node, PHP, Java, and Go). Taking transcoding as an example, the steps are as follows:

1. Create a function and deploy the FFmpeg resource package and transcoding logic in it.

2. Configure a COS bucket trigger to process the source video in real time and generate logs, monitoring data, and alarms in a relayed manner.

3. Return the transcoded video to COS and trigger automatic prefetch.

Comparison with TKE

Compared with TKE, the combination of SCF and FFmpeg has the following advantages and disadvantages in audio/video transcoding:

🔗 Tencent Cloud

Item	TKE-based Implementation	SCF-based Implementation	Analysis
Implementation method	FFmpeg is run in a Docker container. FFmpeg is customized, and different versions have different binary dependencies, all of which are packaged as images. Docker image tags can be autonomously controlled. Different services load different Docker images, which contain different FFmpeg versions.	Different FFmpeg versions can be compiled into different executable files, deployed to "layers" of SCF functions, and decoupled from the business logic. Different functions can be written and bound to different FFmpeg versions at the "layers" to provide different transcoding features. A scheduling function can be written to segment videos and schedule different transcoding services.	The implementation methods do not differ greatly.
Development/Testing/Deployment experience	After local development and testing, the CI/CD process will be triggered, and an image will be burnt to complete the deployment. The grayscale release system needs to be maintained, and TKE cluster updates are slow.	After local development and testing, the CI/CD process will be triggered to complete the deployment. SCF offers beta test/release/rollback features, which allow you to integrate APIs directly to fully automate the deployment process. Sub-accounts can use the CLI tool to perform development and debugging in real time in the test environment, thus improving the efficiency.	SCF makes the development process easier and more efficient.
Logging/Monitoring/Alarming	It is necessary to launch the agent in the TKE cluster	SCF offers logging/monitoring/alarming capabilities and APIs for	SCF offers a wider range of capabilities, but

	and connect it to the logging platform, monitoring center, and alarming platform.	connection to third-party logging/monitoring platforms. It supports launching the agent process in the runtime and reporting data synchronously.	for connection to self-created platforms, it is more complex to launch the agent in SCF than in TKE.
OPS	TKE clusters need to be maintained on your own and have low auto- scaling efficiency.	Maintenance is not required, as SCF guarantees cluster availability, load balancing, and auto scaling.	SCF is easier to use.
Fees	It is necessary to reserve container resources based on the peak values, which may cause a waste of resources.	Resources are auto-scalable and pay-as-you-go, saving more than 30% of costs.	SCF has obvious advantages.

Through comparison with TKE in multiple dimensions above, the following conclusions can be drawn:

Advantages:

SCF provides a standard runtime environment, ensures the high availability and auto scalability of resources, and requires no dedicated maintenance.

SCF is billed by actual usage, eliminating the waste of resources.

Development and debugging in SCF are more efficient, and dependencies are decoupled from the business and can be hot updated separately in real time.

Runtime environments are isolated, so the failure of one single request will not affect the normal execution of other requests.

Disadvantages:

The introduction of SCF needs to be integrated with the existing CI/CD process, so there may be some changes in the development method.

The existing business code requires modifications mainly in terms of FFmpeg encoding/decoding. SCF can provide encoding/decoding tools and development assistance to facilitate the modifications.

Prerequisites

This document uses the Guangzhou region as an example:

Go to the COS console, create a bucket, and set the access permission of the bucket to **public read/private write**. (Optional) If the video file is above 500 MB in size, you need to go to the CFS console to activate the CFS service so as to expand the local storage space of SCF. For more information, please see Mounting CFS File System. Go to the CAM console, create an execution role for SCF, and grant the role COS and CFS read/write permissions to authorize SCF to access the corresponding services. For more information, please see Role and Authorization.

Directions

Creating function

1. Log in to the SCF console and click **Function Service** on the left sidebar.

2. Select the region where to create a function at the top of the **Function Service** page and click **Create** to enter the function creation process.

3. Select a function template as follows on the **Create Function** page as shown below:

	Template Use demo template to create a function or application	Custom Create a custom function HelloWolrd demo	using			
Fuzzy search	Transcode Separate multiple tags wit	h carriage returns		Q Total: 5		Sort by rea
	VideoTranscode	Learn More	Transcode	Learn More	Transcode	Learn
	Category Function		Category	Function	Category	Function
	Description This function will transc api, when you upload vi	ode your video by call deo to cos bucket	Description	This example demonstrates triggering a cloud function from COS or a gateway, obtaining	Description	This example demonstrates triggering a clo function from COS or a gateway, obtaining.
	Tag Nodejs8.9 Ckafka	Transcode	Tag	Python3.6 COS ffmpeg CFS	Tag	Python2.7 COS ffmpeg CFS
	Author 🔗 Tencent Cloud		Author	🙆 Tencent Cloud	Author	🕗 Tencent Cloud
	Deploy 7,659 time		Deploy	13,543 time	Deploy	10,247 time
	Transcode	Learn More	Transcode	Learn More		
	Category Function		Category	Function		
	Description This example demonstra function from COS or a	ates triggering a cloud gateway, obtaining	Description	This example demonstrates triggering a cloud function from COS or a gateway, obtaining		
	Tag Nodejs12.16 COS	ffmpeg CFS	Tag	Nodejs10.15 COS ffmpeg CFS		
	Author 🔗 Tencent Cloud		Author	🙆 Tencent Cloud		
	Deploy 7,931 time		Deploy	14,797 time		

Creation method: select Template.

Fuzzy search: enter **VideoTranscode** and search. This document uses the Python 3.6 runtime environment as an example.

Click **Learn More** in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

4. Click **Next**. The function name is automatically generated by default and can be modified as needed. Follow the prompts to configure the environment variables and execution role as shown below:

← Create		
	Basic Configu	urations v
	Function name *	Transcode-1611306806 It supports 2 to 60 characters, including letters, numbers, underscores and hyphens. It must start with a letter and end with a number or letter.
	Region *	🔇 Guangzhou 🔻
	Description *	This example demonstrates triggering a cloud function from COS or a gateway, obtaining the source file download address, and transcoding based on ffmpeg using HTTP read stream.
		Up to 1000 letters, digits, spaces, commas, and periods.
	Environment Variable *	key value
		region the region of target bucket
		target_bucket target bucket name X
		target_path path of target bucket X
	Execution Role •	To ensure that the function template can access other Tencent Cloud services, please configure and use the SCF template role, or select an existing role that includes QcloudCOSFullAccess, QcloudCFSFullAccess preset policies. C Configure and use SCF template role Use the existing role
	Complete	Back

Environment Variable: enter as shown below:

key	value	
region	ap-guangzhou	×
target_bucket	serverless-	×
target_path	/transcode	×



key	value
region	COS bucket region.
target_bucket	The created COS bucket to which the output video will be uploaded after being transcoded.
target_path	The specified directory of the bucket to which the output video will be uploaded after being transcoded.

Execution Role: check **Enable**, select **Configure and use SCF template role**, and the system will automatically create and select an SCF template execution role associated with full access permissions of COS and CFS. You can also check **Use the existing role** and select the existing role created in Prerequisites in the drop-down list. This document takes **Configure and use SCF template role** as an example.

During execution, the function will use the execution role to get a temporary key to read and write resources in the COS bucket.

5. In the **Trigger Configurations** section, configure the COS trigger as shown below:

eate a Trigger	Custom			
	Triggered Version	Default Traffic	- ¢	
	Trigger Method	COS trigger	v	
		SCF publishes events to SCF fun	tion, and uses the received logs as the paran	neters to trigger the function. Learn More
	COS Bucket(j)		¢	Create COS Bucket
	Event Type 🛈	All Creation Events	v	
	Prefix Filtering (j)	demo/		
	Suffix Filter()			
	Enable Now	✓ Enable		

The main parameter information is as follows. Keep the remaining options as default:



Trigger Method: select COS trigger.

COS Bucket: select a bucket. If you want to store the source video and the output video in the same bucket, you need to configure the trigger with a prefix filtering rule /, such as demo/.

6. Click **Complete**.

(Optional) Configuring CFS mounting

Note:

If you have activated the CFS service, please enable both the VPC and file system mounting capabilities in the function by following the steps below.

1. On the **Function configuration** page of the function, click **Edit** in the top-right corner and configure the function.

VPC: check **Enable** and select a VPC and subnet.

File System: check Enable and select the file system created in Prerequisites.

2. Click the **Function code** tab to enter the function code editing page and modify the file upload path as shown below:

75	logger.info('key: '+ key)
76	<pre># upload_path = '/tmp/new-'+ key.split('/')[-1]</pre>
77	upload_path = '/mnt/new-'+ key.split('/')[-1]
78	<pre>logger.info('upload_path: '+ upload_path)</pre>

Comment on line 76 of the code and add the following code to line 77.

```
upload_path = '/mnt/new-'+ key.split('/')[-1]
```

Testing Features

Log in to the COS console, enter the corresponding bucket directory, upload a video file, and check whether a compressed video file is generated in the corresponding transcoding directory as shown below: **Note:**

The time it takes to compress a video varies by video size. If a video is large, it will take longer to compress the video and display the output video.

Scalability

You can add automated CDN purge/prefetch capabilities to the demo in this document. For example, when the output video is returned to the COS bucket, a new function can be triggered to execute the CDN purge/prefetch features. For more information, please see CDN Cache Purging.

You can also leverage the high concurrence of SCF to implement fast transcoding or segmenting. For example, function A can schedule tasks while function B can perform actual transcoding/segmenting tasks. With the aid of the CFS mounting capabilities, you can also implement cross-function file sharing with ease.

Acquire Image on COS and Create a Thumbnail Example illustration

Last updated : 2024-12-02 16:35:34

Implementation Scenario

Note:

1. Two COS buckets must be used. If you use the same bucket as both source and target buckets, each thumbnail uploaded to the source bucket may trigger another object creation event that will invoke the function again and thus cause an infinite loop.

2. Make sure that the function and the COS bucket are in the same region.

In this tutorial, suppose that:

A user is going to upload a photo to a specific COS bucket.

You need to create a thumbnail for every image uploaded by the user.

The created thumbnails need to be stored in another COS bucket.

Implementation Overview

The implementation process of this function is as follows:

You create the event source mapping between the function and the COS buckets.

The user uploads an object to the source COS bucket (object creation event).

The COS bucket detects the object creation event.

COS invokes the function and passes the event data as parameters to the function, thus publishing the

 $cos:ObjectCreated: \ \ event to the function.$

The SCF platform receives the invocation request and executes the function.

The function gets the bucket name and filename from the received event data, gets the file from the source bucket,

uses the graphic database to create a thumbnail, and saves the thumbnail in the target bucket.

Note: after going through this tutorial, your account will have the following resources:

A SCF function used to create thumbnails.

Two COS buckets: the source bucket for storing uploaded original images and the target bucket for storing cropped images.

Step 1. Prepare COS Bucket

Last updated : 2024-12-02 16:35:34

Note:

1. The source and target buckets must be in the same region as the function. In this tutorial, the Guangzhou region is used.

2. Two COS buckets must be used. If you use the same bucket as the source and target buckets, each thumbnail uploaded to the source bucket will trigger the function again and thus generate unnecessary recursion.

1. Log in to the COS console.

2. Click **Create Bucket** on the **Bucket List** tab to create a source COS bucket. For more information, see Console.

Set the the following parameters:

Name: Set the value to mybucket1 .

Region: Select Guangzhou .

Access permissions: Select Private Read/Write .

3. Perform the same steps to create the target bucket mybucket-resized1 .

Step 2. Create and Test Thumbnail Function

Last updated : 2024-12-02 16:35:34

Creating Thumbnail Function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Guangzhou** region and click **Create** to enter the function creating page and configure the function as shown below:

	Use demo ten	nplate to create a function	Create a custom function	n using				
	or application		HelloWolrd demo					
Euzzy search	thumbnail 🔇	Separate multiple tags w	ith carriage returns		🕲 Q Total: 3			Sort by reco
	Thumbnail		Learn More	Thumbnail	Learn More	Thumbnail		Learn
	Category	Function		Category	Function	Category	Function	
	Description	This demo need to config object has been upload to	COS trigger. When COS bucket, th	Description	This demo need to config COS trigger. When object has been upload to COS bucket, th	Description	This demo need to config object has been upload to	COS trigger. When COS bucket, th
	Tag	Python2.7 COS th	humbhail	Tag	Nodejs8.9 COS thumbnail	Tag	Nodejs6.10 COS	thumbnail
	Author	Content Cloud		Author	Tencent Cloud	Author	S Tencent Cloud	
	Deploy	6,857 time		Deploy	7,618 time	Deploy	7,423 time	
	Deploy	6,857 time		Deploy	7,618 time	Deploy	7,423 time	

Creation method: select Template.

Template search: enter **compression** and search. This document uses the Python 2.7 runtime environment as an example.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click **Next**. The function name is automatically generated by default and can be modified as needed. Follow the prompts to configure the execution role:

Function	Thumbnail-1628508907	
name -	It supports 2 to 60 characters, ir	including letters, numbers, underscores and hyphens. It must start with a le
Region •	🔇 Guangzhou	•
Description *	This demo need to config COS upload to COS bucket, the SO compress it.	S trigger. When object has been CF will download it to /tmp and
	Up to 1000 letters, digits, space	es, commas, and periods.
Execution	✓ Enable (i)	
Role *	To ensure that the function temp	plate can access other Tencent Cloud services, please configure and use th
	Configure and use SCF temp	plate role (j)
	 Use the existing role 	
函数代码	Runtime: Python2.7 Execution N	Method: index.main_handler
Cloud	Studio Lite File Ec	dit Window
index.py		
E 0 4	^	🔷 index.py 🔹 🕀
	I PIL	17 appid = XXXXXX # Please replace with your APP
> in	I Pillow-5.2.0.dist-info	18 secret_id = u' # Please replac
	index.py	20 region = u'ap-guangzhou' # Please repla
		21 token = ''
		22 resized_bucket = 'mybukect-resizeed1 ' compressed pictures.请替换为您用于存放压缩后图片的

Execution Role: check Enable. Configure and use SCF template role is selected here as an example as detailed below:

Configure and use SCF template role: select this option and the system will automatically create and select an SCF template execution role associated with full access permissions of COS.

Use the existing role: you need to select an existing role that contains the above permissions in the drop-down list. Note:



During execution, the function will use the execution role to get a temporary key to manipulate relevant Tencent Cloud resources.

4. When using this template function, you need to modify the configuration information in the function code as prompted.

```
Click to expand the Function Code block and replace <code>appid</code> , <code>secret_id</code> , <code>secret_key</code> , <code>region</code> , and
```

resized_bucket in the function code with your APPID, SecretId, SecretKey, region, and resized bucket

Note:

You can get the APPID on the Account Info page in the console.

You can get SecretId and SecretKey on the API Key Management page in the console.

Configuring COS Trigger

1. In the Trigger Configurations section, select Custom and configure as prompted as shown below:

Create a Trigger	Custom	
	Triggered Version	Default Traffic 🔹
	Trigger Method	COS trigger SCF publishes events to SCF function, and uses the received logs as the parameters to trigger the function. Learn More
	COS Bucket	mybucket1 .cos.ap-guangzhou.myqcloud.com Create COS Bucket
	Event Type	All Creation Events
	Prefix Filtering	
	Suffix Filter(i)	
	Enable Now	✓ Enable
	Create Later	

Main parameters are as follows:



Trigger Method: select COS trigger. COS Bucket: select the bucket mybucket created in Step 1. Event Type: select All Creation Events. 2. Click Complete.

Testing Function

1. Switch to the COS console, select the created bucket mybucket1 , click **Upload File**, and select a random .jpg or .png image for upload.

2. Switch to another COS bucket mybucket-resized1, check whether there is a file with the same name generated, download it, and compare the size of the two images.

3. Go to the SCF console to view the execution result. You can see the printed log information in **Execution Logs**.

Forming Data Lake with SCF + COS

Last updated : 2024-12-02 16:35:34

Overview

The data lake formation scheme of the serverless architecture is to capture and record the information of a batch of data after starting a data call through a function trigger. This implements various capabilities such as structure conversion, data format conversion, and data compression in the function. In this document, SCF and COS are used to back up TDMQ messages.

Directions

Creating COS bucket

1. Log in to the COS console.

2. Select Bucket List on the left sidebar and click Create Bucket to create the source COS bucket.

3. Set the **Name** to **scf-data-lake**, the **Region** to **Guangzhou**, and the **Access Permission** to **Private Read/Write** (default), and click **OK**.

Creating TDMQ resources

1. Create resources such as cluster and topic in the TDMQ console. For more information, see Resource Creation and Preparation.

2. Connect the TDMQ cluster to a VPC. For more information, see VPC Access.

Configuring function through COS app integration

1. Log in to the COS console.

2. Select **App Integration** on the left sidebar and select **TDMQ Message Backup** in **Data Migration and Backup** as shown below:

Disabled Vew Documentation You can deliver MyGQL backup files to a specified bucket in COS for penistent storage to avoid data loss or corruption. Configure Backup Rule	Disabled View Documentation Viou can deliver MongoDB backup files to a specified bucket in COS for pensistent storage to avoid data loss or comption. Configure Eactup Rule	SQL Server Backup Disabled View Documentation COS allows you to back up SQL Server files to a specified bucket for pensistent storage to avoid data loss or corruption. Contigure Backup Rule
Disabled Vew Documentation COS allows you to back up Redis files to a specified bucket for persistent storage to avoid data loss or corruption. Configure Backup Rule	CKafka Message Backup Disabled View Documentation OOS supports dumping messages in CKafka instances to a specified bucket for data analysis, download, and more. Configure Backup Rule	Disabled View Documentation Dumps TDMQ by Copic messages to a specified bucket for analysis, download, and more. Configure Backup Rule
Disabled Vew Documentation Durps CDN Log Backup Durps CDN access logs to a specified budet to analyce popular resources/active users, and calculate metrics such as the average response time and average download speed. Configure Backup Rule	Disabiled View Documentation You can dump CLS-collected logs to a specified bucket in COS for log analysis. Configure Backup Rule	Bill Delivery Vew Documentation Regularly saves bills to a COS bucket in the format of files. Start Delivery

3. On the **TDMQ Message Backup** page, select the region of the created COS bucket and click **Add Function**.

4. In the Create TDMQ Message Backup Function pop-up window, enter the Function Name such as data-

lake , associate the created scf-data-lake bucket, select Authorize SCF Service, and then click Next.

5. Click **Next** to configure TDMQ as follows:

Cluster: A TDMQ cluster as the message source. Only TDMQ clusters in the same region are supported.

Namespace: A namespace in the cluster

Topic: A topic used as the message source

Subscription: Select a subscription. If existing subscriptions cannot meet your needs, you can create a new one in Consumption Management in the TDMQ console.

Start Point: Start point of historical messages

Role: Select the TDMQ role (a TDMQ-specific concept, which is different from that mentioned in Tencent Cloud), which is the minimum unit for permission division within TDMQ. You can create multiple roles and grant them different message production/consumption permissions in different namespaces.

Role Key: Select the TDMQ role key, which is an authentication tool. You can add a key in the client to access TDMQ to produce/consume messages. Each role has a unique key corresponding to itself.

Address: It must be a VPC access address.

Note:

There must be an available IP in the VPC subnet, and DHCP must be supported.

6. Click **Next** to configure delivery. The configuration item is as follows:

Delivery Path: It is the delivery path prefix of the backups. If it is not specified, backups will be stored in the root directory of the bucket. The specified prefix must end with a slash (/).

7. Click **Confirm**.

You can perform the following operations on the created function:

Click View Log to view the historical running status of TDMQ message backup. If an error is reported, you can click

View Log to quickly redirect to the SCF console to view the error log details.

Click **Edit** to modify a TDMQ message backup rule.

Click **Delete** to delete an unwanted TDMQ message backup rule.

Testing function

- 1. Log in to the TDMQ console.
- 2. Click **Topic Management** on the left sidebar and select the target region, cluster, and namespace.
- 3. On the **Topic Management** list page, click **Send Message** on the right of the associated topic.
- 4. Enter the message content of up to 64 KB in the pop-up window.
- 5. Click Submit.

6. Go to the details page of the associated COS bucket. For example, check whether a file has been created in the **Delivery Path** defined in scf-data-lake, and if so, the TDMQ message backup is successful.

Implementing Custom Calculation of File Hash Value with SCF + COS

Last updated : 2024-12-02 16:35:34

Overview

Errors may occur when data is being transmitted between the client and the server. Cloud Object Storage (COS) combined with Serverless Cloud Function (SCF) can ensure the integrity of uploaded data through data verification, for example, MD5 verification. When users upload files to COS, SCF will help verify the objects uploaded by the users to ensure the integrity and correctness of the uploaded data.

Background

None of the existing public cloud object storage services in the industry provides MD5 verification, and after a user uploads a file, the following situations may occur: Duplicated files and rising costs File errors and lower business efficiency File missing

Solution Strengths

Visual operation: One-click configuration simplifies the development process without coding, greatly improving R&D efficiency

Multiple options: MD5, SHA1, SHA256, and CRC64 are available, meeting user requirements in various scenarios Automatic execution: Once a file is uploaded to COS, the workflow for calculating the verification code is triggered

Directions

1. Log in to the COS console.

2. Create a workflow, customize the format filter rule, and create a custom function node. For detailed directions, see Configuring Workflow.



Workflow Name *	Enter workflow name Only a combination of letters, numbers, Chinese characters, underscores (_) and hyphens (-) with a length no greater than 128 characters is supported
Input Bucket Name	test-1300858599
Input Path	If not filled in, it is valid for all paths under the bucket Select
Format(Mainstream video/audio files 🛈 💫 Image 🛈 🔘 Custom rule 🛈 📄 All files 🛈
	Video(i)
	Audio()
	Image()
	Custom Extension C Enter a custom extension such as jpg, jpeg, or png.
	Content-Type ()
Queue 🛈 *	Media processing queue (queue-1) $ = \diamondsuit$
Callback	O Use queue callback O Custom
Queue Callback URL 🕃	Empty Edit
Configure Workflow	Input Add a node by dicking "+" to open the workflow

3. In the function node pop-up window, click **Add Function**.

4. On the SCF creation page, select the template for **COSHashCalculate**.

Fuzzy search	workflow Separate multiple tags with carriage returns	Q Total: 3	Sort by recommendation
	COSWorkflowDemo Learn r Category Function	COSHashCalculate Learn more Category Function	e COSWorkflowFFmpegT Learn more Category Function
	Description This is a demo for cos workflow Tag Nodejis12.16 COS cos workflow demo CA ♂ Tencent Cloud Deploy 7.150次	Description This demo uses cos trigger and SCP to calculate hash of cos object, and add result t Tag Nodejs12.16 Hash cos workflow CA Image: Constraint of the cost object of the cost object of the cost object o	Description This is a demo for cos workflow to transcode media file with ffmpeg Tag Nodejs12.16 COS ffmpeg cos workflow cos workflow ffmpeg transcode CA Incent Cloud Deploy 7,623/2

5. Based on the user's file size, configure the execution timeout duration in the basic settings and configure sufficient memory in the advanced settings.

6. Configure the function code. This function template supports the following two environment variables:



hashTypeList: Indicates the list of calculation algorithms. This variable is optional. The default value is ["crc64",

"md5", "sha1", "sha256"].

caseType: Indicates the hash case. This variable is optional. The default value is lowercase . You can also pass

in uppercase .

7. Enable permission configuration and bind a role that has the read/write permission of the current bucket. To create an execution role, see Role and Authorization.

8. Click **Complete**.

9. Go back to the previous workflow page, select the custom transcoding function created just now, and save the workflow.

1 Create a workflow	2 Enable the workflow	3 Automatic execution	4 View the result
You can create a workflow and configure the actions to perform. Currently, audio/video transcoding and frame-capturing are supported. For detailed configurations, please see Workflow Guide	Once created, enable the workflow and upload files to this specific path via the File List page or via COS's APIs/SDK.	An enabled workflow auto- processes files uploaded to the configured path and saves the result to the specified path.	Each object executed in a workflow will generate an execution instance. You can view the execution instance for the object processing information.
Create Workflow			Workflow Name v Please enter search conte
Vorkflow ID/Name	Input Path	Creation Time	Upload-Triggered Ex Operation
01be9395376b4ccabd5af6f1a89887bc	1	2022-04-29 16:12:52	Execute Workflow Result Mor

10. Upload the file. After the workflow processing is successful, you can see that multiple hash headers are successfully added to the uploaded file.

Content-Type	video/mp4
x-cos-meta-hash-crc64	750816529385890857
x-cos-meta-hash-md5	612223edcc37527251cc3e07f825390c
x-cos-meta-hash-sha1	77a747e7379b61fc25616dbbfadad4c9749f27b1
x-cos-meta-hash-sha256	4f34dc6a1813ac591c4f439a344ce054350098b1396b15fa6a1
x-cos-meta-scf-cos-hash-calculate	true
x-cos-meta-source	cos-data-process

Implementing Custom Transcoding with SCF + COS

Last updated : 2024-12-02 16:35:34

Overview

As the largest part of traffic in information dissemination, audio and video are very important in all industries, and the processing logic of audio and video in different business scenarios may be specific to the industry. Although public cloud provides a large number of video processing services for users to choose from, it still cannot fully cover users' requirements for special processes and customization. Using the workflow processing of Cloud Object Storage (COS) combined with the customization logic of Serverless Cloud Function (SCF) is an excellent choice to help users quickly create a variety of audio and video processing businesses to meet their needs.



Applications

Fast access to users' self-built transcoding clusters, compatible with their existing businesses

Formats and processing logic for special industries are supported, including movie and media.

Supports custom processing logic, meeting process customization requirements of various scenarios

Workflow for template-based batch processing, meeting common audio and video processing requirements of video websites, education, and social networking industries

Solution Strengths

Accelerated development: Eliminates the need to focus on resource Ops and component overhead, greatly reducing the complexity for constructing service structures.

Reduced overhead: When the platform is idle, no resource is executed. When a function is executed, the service fee is determined by the number of requests and the running time of the computing resource. The price advantage is obvious.

High availability and scalability: The platform automatically adjusts service resources in parallel based on requests, thus implementing near-infinite scalability and eliminating the risk of service interruption inherent in single-availability zone operations.

Directions

1. Log in to the COS console, create a workflow, customize the filter rule, and create a custom function node. For detailed directions, see Configuring Workflow.

Create Workflow		
	Workflow Name *	Enter workflow name Only a combination of letters, numbers, Chinese characters, underscores () and hyphens (-) with a length no greater than 128 characters is supported
	Input Bucket Name	sis-cloudfunction-ap-beijing-code-1300858589
	Input Path 🚯	If not filled in, it is valid for all paths under the bucket Select
	Format	Mainstream videolaudio files ①
	Queue (i)*	Media processing queue (queue-1) 🍸 🗘
	Callback	O Use queue caliback O Custom
	Queue Callback URL 🚯	Empty Edit
	Configure Workflow	Input C C End Add a node by clicking *** to open the workflow

2. In the function node pop-up window, click **Add Function**.

3. On the function creation page, select the template for **COSWorkflowFFmpegT...**.

Fuzzy search	workflow 🔇 Separate multiple tags with carriage returns	8	Q Total: 3		Sort by recommendatio
	COSWorkflowDemo Learn	more COSHashCalculate	Learn more	COSWorkflowFFmpegT.	Learn mo
	Category Function	Category Function		Category Function	
	Description This is a demo for cos workflow Tao Nodeis12.16 COS	Description This demo uses calculate hash	s cos trigger and SCF to of cos object, and add result t	Description This is a den media file w	no for cos workflow to transcode ith ffmpeg
	cos workflow demo	Tag Nodejs12.16 cos hash calc	Hash cos workflow	Tag Nodejs12. cos workfl	16 COS ffmpeg ow
	CA 🔗 Tencent Cloud	CA 🔗 Tencent Clo	oud	cos workfi	ow ffmpeg transcode
	Deploy 7,120次	Deploy 10,992次		CA 🔗 Tencent	Cloud
				Deploy 10,019次	

4. Based on the user's file size, configure the execution timeout duration in the basic settings and configure sufficient memory in the advanced settings.

5. Configure environment variables in **Advanced configuration** > **Environment configuration**. The function template supports the following environment variables:

targetBucket: Target bucket. Required.

targetRegion: Region of the target bucket. Required.

targetKeyTemplate: Target path template. Optional. Defaults to

\${InputPath}\${InputName}_transcode.\${ext} .

ffmpegTemplate: Transcoding command template. Required. Example: \${ffmpeg} -loglevel error -i

\${source} -r 10 -b:a 32k \${target} .

localTmpPath: Temporary save path. When CFS is bound, you can change the temporary path. Optional. Defaults to /tmp .



MEM	1024MB	* (j)	
Initialization	65	seconds (i)	
timeout period	Time range: 3-300 seconds		
Environment variable	key	value	0
	targetBucket	Please enter the value of e	
	Please enter the envire Please	e enter the value of environment variable	
Permission Cor	nfiguration		
Execution Role	✓ Enable (i)		

6. Enable permission configuration and bind a role that has the read/write permission of the current bucket. To create an execution role, see Role and Authorization.

Tencent Cloud Advisor (advisor)	Aegis (aegis)	Shenbi Low-Code Application Platform as a Service (shenbi)	API Gateway (apigw)	Auto Scaling (as)
Application Service Workflow (asw)	TBaaS (tbaas)	Billing (billing)	BlueKing (blueking)	Cloud Physical Machine (bm)
BPaaS (bpaas)	Cloud Access Security Broker (casb)	Tencent Kubernetes Engine (tke)	Cloud Database (cdb)	Cloud Data Coffer Service (cdcs)
CDN (cdn)	Cloud File Storage (cfs)	Cloud Firewall (cfw)	Customer Growth Expert (cge)	Cloud Infinite (ci)
CKafka (ckafka)	Cloud Loader Balance (clb)	Cloud Audit (cloudaudit)	Cloud Studio (cloudstudio)	Cloud Log Service (cls)
cloud/Waf (waf)	Cloud Monitor (monitor)	Cloud Media Engine (cme)	CODING DevOps (coding)	COS (cos)
Cloud Storage Gateway (csg)	Cloud Training Platform (ctp)	TencentDB for CTSDB (ctsdb)	Cloud Virtual Machine (cvm)	Cloud Workload Protection (cvp)
Tencent Cloud Developer-TDP (devops)	DI (di)	Data Lake Compute (dic)	Data Security Center (dsgc)	Data Tranmission Service (dts)
EventBridge (eb)	Elasticsearch MapReduce (emr)	faceid (faceid)	Game Sever Elastic-scaling (gse)	Image AI (facerecognition)
IDaaS (idaas)	lotHub (lothub)	IoT Suite (lotsuite)	Internet of Things Video (lotvideo)	Intelligent Surveillance Storage (iss)
Developer Laboratory (labs)	Cloud Streaming Services (live)	CDB for MariaDB (TDSQL) (mariadb)	StreamLive (mdl)	StreamPackage (mdp)
StreamPackage (mdp)	Message Center (message)	Mobile Game Online Battle Engine (mgobe)	Cloud MangoDB (mangadb)	Media Processing Service (mps)
Migration Service Platform (msp)	Media Transcoding Service (mts)	Network Assets Risk Monitor System (narms)	Publicly Accessible Instance-PAI (pai)	Serverless Cloud Function (scf)
Stream Compute Service (scs)	Serverless Framework (sls)	Security Situation Awareness (ssa)	Secrets Manager (ssm)	Tencent Cloud Base (tcb)
Tencent Cloud Display (tcd)	Tencent Cloud Mesh (tcm)	Tencent Container Registry (tcr)	Tencent Container Security Service (tcss)	Tencent Database Middleware (tdm)
444444 (cams)	TI (ti)	TI Accelerator (tia)	Tencent Cloud infrastructure as Code (tic)	TencentCloud TI Platform TI-EMS (tiems)
TencentCloud TI Platform TI-ONE (tione)	TI Self-Learning (tis)	Tencent Interactive Whiteboard (tiw)	Tencent Cloud Service Engine (tse)	Tencent Service Framework (tsf)
Tencent Support System (tss)	Cloud Shield - Data Data Access Security Broker (dasb)	Video Moderation System (vm)	VOD (vod)	Vulnerability Scan Service (vss)
WeData Data Development Platform (wedata)	WeMall (wemail)	workorder (workorder)	YouMall (youmall)	Cloud Operations Console (zhiyun)

7. Click Complete.

8. Go back to the previous workflow page, select the custom transcoding function created just now, and save the workflow. Start the workflow on the workflow list page.

1 Create a workflow	2 Enable the workflow	3 Automatic execution	4 View the result
You can create a workflow and configure the actions to perform. Currently, audio/video transcoding and frame-capturing are supported. For detailed configurations, please see Workflow Guide	Once created, enable the workflow and upload files to this specific path via the File List page or via COS's APIs/SDK.	An enabled workflow auto- processes files uploaded to the configured path and saves the result to the specified path.	Each object executed in a workflow will generate an execution instance. You can view the execution instance for the object processing information.
Create Workflow			Workflow Name
Vorkflow ID/Name	Input Path	Creation Time	Upload-Triggered Ex Operation
v01be9395376b4ccabd5af6f1a89887bc	I	2022-04-29 16:12:52	Execute Workflow Result Mo

9. Upload the file. After the workflow processing is successful, you can see that the uploaded video is successfully transcoded and is saved as a new file.

3.8.0.mp4	3.54GB	ł
5G.mp4	5.25GB	ł
testMp4.mp4	3.54MB	ð
testMp4_transcode.avi	1.12MB	Ą

MapReduce Method that Uses WordCount as an Example Example

Last updated : 2024-12-02 16:35:34

In this tutorial, suppose that: You will upload some text files (such as logs) to a specific COS bucket from time to time. You need to count the words in these text files.

Implementation Overview

The implementation process of this function is as follows:

You create functions and COS buckets.

You upload an object to the source COS bucket (object creation event).

The COS bucket detects the object creation event.

COS invokes the function and passes the event data as parameters to the function, thus publishing the

 $cos:ObjectCreated: \ \ event to the function.$

The SCF platform receives the invocation request and executes the function.

The function gets the bucket name and filename from the received event data, gets the file from the source bucket, uses the wordcount implemented in the code to count the words, and saves the word count in the target bucket.

Note:

Three COS buckets are used in this example. If you use the same bucket as both source and target buckets, each file uploaded to the source bucket may trigger another object creation event that will invoke the function again and thus cause an infinite loop.

Make sure that the function and the COS buckets are in the same region.

Note: after going through this tutorial, your account will have the following resources:

Two SCF functions: Mapper and Reducer.

Three COS buckets: srcmr, middlestagebucket and destmr.

The Mapper function will be bound to srcmr for triggering, the Reducer function will be bound to

middlestagebucket for triggering, and destmr will be used to receive the final statistics.
Step 1: Prepare a COS Bucket

Last updated : 2024-12-02 16:35:34

Make sure that you have obtained the permission to use SCF before implementing this sample.

1. Log in to the Tencent Cloud console and select **Cloud Product** > **Cloud Object Storage**.

2. Select **Bucket List** on the left sidebar to enter the "Bucket List" page.

3. Click Create Bucket on the "Bucket List" page.

4. In the **Create Bucket** pop-up window, create the source COS bucket as follows:

Set the name of the COS bucket as srcmr, select Chinese Mainland > Beijing as the region, set the access permission to the default value of private read/write, and click Next.

5. Keep the default values for other options and click **Next** to create a COS Bucket.

 $6. Repeat steps 1-4 to create the intermediate bucket \verb"middlestagebucket" and target bucket "destmr".$

Step 2. Create Mapper and Reducer Functions

Last updated : 2024-12-02 16:35:34

Creating Mapper Function

Creating in console through template function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creating page and configure the function as shown below:

	Template	Custom		
	or application	HelloWolrd demo		
Fuzzy search	map_function Separate multiple tags with the second	th carriage returns	Q Total: 2	
	MapFunction	Learn More ReduceFu	inction	Learn Mor
	Category Function	Category	Function	
	Description This is one of the MapRedu is map_function.	uce function which Description	This is one of the MapReduce is map_function	function which
	Tag Python2.7 MapReduc map_function	e Tag	Python2.7 MapReduce reduce_function	
	Author 🔗 Tencent Cloud	Author	🔗 Tencent Cloud	
	Deploy 9,407 time	Deploy	12,363 time	

Creation method: select Template.

Fuzzy search: enter "map_function", search, and select the "map_function" template.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click Next. The function name is automatically generated by default and can be modified as needed. Follow the

prompts to configure the execution role:

Execution Role: check **Enable**. **Configure and use SCF template role** is selected here as an example as shown below:

Basic Configu	irations						
Function	MapFunction-123456						
name *	It supports 2 to 60 characters, including letters, numbers, underscores and hyphens. It must start with a letter and end with a number or letter.						
Region *	S Beijing 👻						
Description *	This is one of the MapReduce function which is map_function.						
	Up to 1000 letters, digits, spaces, commas, and periods.						
Execution	✓ Enable ③						
Role *	To ensure that the function template can access other Tencent Cloud services, please configure and use the SCF template role, or select an existing role that includes QcloudCOSFullAccess preset pr						
	O Configure and use SCF template role 🛈						
	O Use the existing role						

Configure and use SCF template role: select this option and the system will automatically create and select an SCF template execution role associated with full access permissions of COS.

Use the existing role: you need to select an existing role that contains the above permissions in the drop-down list. Note:

During execution, the function will use the execution role to get a temporary key to manipulate relevant Tencent Cloud resources.

4. Click to expand the **Function Code** tab, where you can browse the code information. If you want to modify the parameter values such as variables, you can modify and save them online.

Configuring COS trigger

1. In the **Trigger Configurations** section, select **Custom** and enter relevant information according to the displayed parameters as shown below:

eate a Trigger	Custom						
	Triggered Version	Default Traffic	₹ Ø				
	Trigger Method	COS trigger	v				
		SCF publishes events to SCF function	n, and uses the	received logs as	the parameters t	o trigger the function. Le	earn More
	COS Bucket(Please select a COS bucket	¢		ap-beijing 🔻	.myqcloud.com Create C	OS Bucket
	Event Type 🛈	All Creation Events	Ŧ				
	Prefix Filtering (
	Suffix Filter(j)						
	Enable Now	✓ Enable					

Main parameters are as follows:

Trigger Method: select COS trigger. COS Bucket: select srcmr.

Event Type: select All Creation Events.

2. Click Complete.

Creating Reducer Function

Creating in console through template function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creating page and configure the function as shown below:

	Use demo template to create a funct	tion Create a custom function using HelloWolrd demo	
zzy search	reduce_function 🛛 Separate mul	tiple tags with carriage returns	🙁 🔍 Total: 1
	ReduceFunction	Learn More	
	Category Function		
	Description This is one of the Ma is map_function	pReduce function which	
	Tag Python2.7 Map reduce_function	Reduce	
	Author 🔗 Tencent Cloud		
	Deploy 12,363 time		

Creation method: select Template.

Fuzzy search: enter "reduce_function", search, and select the "reduce_function" template.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click **Next**. The function name is automatically generated by default and can be modified as needed. Follow the prompts to configure the execution role:

Execution Role: check **Enable**. **Configure and use SCF template role** is selected here as an example as shown below:

Basic Config	urations
Function	ReduceFunction-123456
name *	It supports 2 to 60 characters, including letters, numbers, underscores and hyphens. It must start with a letter and end with a number or letter.
Region *	S Beijing 🔹
Description *	This is one of the MapReduce function which is map_function
	Up to 1000 letters, digits, spaces, commas, and periods.
Execution	✓ Enable ③
Role *	To ensure that the function template can access other Tencent Cloud services, please configure and use the SCF template role, or select an existing role that includes QcloudCOSFullAccess preset pr
	O Configure and use SCF template role (1)
	O Use the existing role

Configure and use SCF template role: select this option and the system will automatically create and select an SCF template execution role associated with full access permissions of COS.

Use the existing role: you need to select an existing role that contains the above permissions in the drop-down list. 4. Click to expand the **Function Code** tab, where you can browse the code information. If you want to modify the parameter values such as variables, you can modify and save them online.

Configuring COS trigger

1. In the **Trigger Configurations** section, select **Custom** and enter relevant information according to the displayed parameters as shown below:

te a Trigger	Custom				
	Triggered Version	Default Traffic	ΨØ		
	Trigger Method	COS trigger	v		
		SCF publishes events to SCF functio	on, and uses the r	eceived logs as	s the parameters to trigger the function. Learn More
	COS Bucket(j)	Please select a COS bucket	φ		ap-beijing 🔻 .myqcloud.com Create COS Bucket
		Ľ			
	Event Type 🛈	All Creation Events	Ψ.		
	Prefix Filtering 🛈				
	Suffix Filter(i)				
	Enable Now	✓ Enable			

Main parameters include:

Trigger Method: select COS trigger.

COS Bucket: select middlestagebucket.

Event Type: select All Creation Events.

2. Click **Complete**.

Step 3. Test Function

Last updated : 2024-12-02 16:35:34

- 1. Download the text file in the test sample and extract test.txt .
- 2. Switch to the COS Console, select the created bucket srcmr , and click Upload File.
- 3. In the pop-up "Upload File" window, select the downloaded test.txt file and click **Upload** as shown below:



4. Switch to the SCF Console to view the execution result. You can see the printed log information in Logs as shown below:



unction managemen	ıt			
Function configuration	Function codes	Layer management	Monitoring information	Log Query
Invocation logs	Advanced retrieval			
All logs v	Last 15 minutes 🔍 👻	2022-07-15 11:10:26 ~ 2022-07	7-15 11:25:26 💼 Refresh	
No log information		Request ID: :		
		Time : Runtime: Executi	ion memory:	
		Log:		

5. Switch to the COS Console, select the created bucket destmr, and view the generated file as shown below:

Upload Files Create Folder Incomplete Multip	art Upload Clear Buckets Mo	re Actions 🔻		Online
Enter a prefix for searching, Only search for objects in the \cdot ${\bf Q}$	Refresh Total 1 objects		1	00 objects per page
Object Name \$	Size ‡	Storage Class T	Modification Time \$	Operation
test.jpg	92.27KB	STANDARD	2021-04-26 11:48:43	Details Preview More 🔻

CKafka Practice Dumping CKafka Data to ES

Last updated : 2024-12-02 16:35:34

Overview

With the prosperity of the Kafka community, more and more users have started to use Kafka for activities such as log collection, big data analysis, and streaming data processing. CKafka has made the following optimizations on open-source Kafka:

It features distributed architecture, high scalability, and high throughput based on Apache Kafka.

It is 100% compatible with Apache Kafka APIs (0.9 and 0.10).

It offers all features of Kafka with no need for deployment.

It encapsulates all cluster details, eliminating the need for OPS on your side.

Its message engine is optimized to deliver a performance 50% higher than that of open-source Kafka.

SCF has been deeply integrated with CKafka, and a lot of practical features have been launched. With the help of SCF and CKafka trigger, it is easy to dump CKafka messages to COS, ES, and TencentDB. This document describes how to use SCF in place of Logstash to dump CKafka messages to ES as shown below:



How It Works

SCF can consume messages in CKafka in real time in various scenarios such as data storage, log cleansing, and real-time consumption, and the data dump feature has been integrated in the CKafka console and can be enabled quickly, making it easier to use as shown below:



Scheme Advantages

Compared to a CVM-based self-created CKafka consumer, SCF has the following advantages:

You can enable the CKafka trigger quickly in the SCF console to automatically create a consumer, and SCF will maintain the high availability of components.

The CKafka trigger itself supports many practical configurations, such as the offset position, 1–10,000 messages to be aggregated, and 1–10,000 retry attempts.

Business logic developed based on SCF naturally supports auto scaling, eliminating the need to build and maintain server clusters.

Compared to CVM-based self-created Logstash service, SCF has the following advantages:

SCF comes with a consumer component that allows for aggregation.

The scalable function template of SCF provides message aggregation and partial cleansing capabilities.

SCF clusters are highly available and support log monitoring, enabling you to launch your business more quickly.

SCF is pay-as-you-go and more cost effective than self-created clusters, saving 50% of costs.

Prerequisites

This document uses the Guangzhou region as an example:

You need to activate Elasticsearch Service.

You need to activate the CKafka service.

Directions

Creating function and CKafka trigger

1. Log in to the Serverless console and click Function Service on the left sidebar.

2. Select the region where to create a function at the top of the **Function Service** page and click **Create** to enter the function creation process.

3. Select a function template as follows on the **Create Function** page and click **Next** as shown below:

← Create				
	Create Method	Template Use demo template to create a function or application Create a custom fu HelloWorld demo	function using o	
	Fuzzy search	CkafkaToElastics Separate multiple tags with carriage return	ums Q Total: 4	Sort by recomm. *
		CkafkaToElasticsearch Learn Mc	Nore CkafkaToElasticsearch Learn More	CkafkaToElasticsearch Learn More
		Category Function	Category Function	Category Function
		Description This demo will connect Ckafca and consume message automatically.	Description This demo will connect Ckafca and consume message automatically.	Description This demo will connect Ckafca and consume message automatically.
		Tag Nodejs12.16 Ckafka ES Logstash	Tag Nodejs10.15 Ckafka ES Logstash	Tag Python3.6 Ckafka ES Logstash Author ⅆ Tencent Cloud
		Author 🔗 Tencent Cloud	Author 🔗 Tencent Cloud	Deploy 9,015 time
		Deploy 6,610 time	Deploy 8,393 time	
		CkafkaToElasticsearch Learn Mc	/lore	,
		Category Function		
		Description This demo will connect Ckafca and consume message automatically.		
		Tag Python2.7 Ckafka ES Logstash		
		Author 🙆 Tencent Cloud		
		Deploy 2,301 time		
[Next			
l	Can			

Creation method: select Template.

Fuzzy search: enter **CkafkaToElasticsearch** and search. This document uses the Python 3.6 runtime environment as an example.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

4. In the **Basic Configurations** section, the function name has been automatically generated and can be modified as needed. Follow the prompts to configure environment variables, execution role, and VPC as shown below:

← Create						
	Basic Config	urations				
	Function	CkafkaToElasticsearch-161130625				
		It supports 2 to 60 characters, includ	ding letters, numbers, undersco	res and I	hyphens. It must start with a letter and end with	a number or letter.
	Region *	🔇 Guangzhou	r			
	Description *	This demo will connect Ckafca and automatically.	d consume message			
		Up to 1000 letters, digits, spaces, co	mmas, and periods.			
	Environment Variable *	key	value		0	
		ES_Address	ES address			
		ES_User	ES user name	×		
		ES_Password		×		
		ES_Index_KeyWord	ES prefix keywords for inde	×		
	Execution Role *	✓ Enable (i)				
		To ensure that the function template preset policies.	e can access other Tencent Clo	ud servio	es, please configure and use the SCF template	ole, or select an existing role that includes QcloudElasticsearchServiceFullAccess,QcloudCKafkaFullAccess
		O Configure and use SCF template	role (i)			
		Use the existing role				
	VPC *	✓ Enable (i)				
		Please select the VPC	۳	lease sel	ect a subnet 🔹	🗘 Create VPC 🗹
	Complete	Back				

Environment Variable: add the following environment variables and configure them as shown below:

key	value	
ES_Address	http://172.xx.xx.xx.xxxxxxx	×
ES_User	elastic	×
ES_Password	X000000X	×
ES_Index_KeyWord	Log	×

key	value	Required
ES_Address	ES address.	Yes

ES_User	ES username, which is `elastic` by default.	Yes
ES_Password	ES password.	Yes
ES_Index_KeyWord	ES keyword index.	Yes
ES_Log_IgnoreWord	Keyword to be deleted. If this parameter is left empty, all keywords will be written. For example, you can enter `name` or `password`.	No
ES_Index_TimeFormat	Index by day or hour. If this parameter is left empty, the index will be by day. For example, you can enter `hour`.	No

Execution Role: check **Enable**, select **Configure and use SCF template role**, and the system will automatically create and select an SCF template execution role associated with full access permissions of ES and CKafka. You can also check **Use the existing role** and select an existing role that has the above permissions in the drop-down list. This document takes **Configure and use SCF template role** as an example.

VPC: check Enable and select the VPC of ES.

5. In the Trigger Configurations section, select Custom and enter relevant

information according to the displayed parameters as shown below:

Triggered Version	Default Traffic	- ¢
Trigger Method	CKafka trigger	v
	SCF can consume messages in CKafka. Le	arn More 🛂
CKafka instance	Please select a CKafka instance	👻 🗘 - Create Ckafka 🛂
Topic	Please select a CKafka topic	• (1)
Maximum messages	- 50 +	
Start Point 🛈	O Latest	
	Earliest Specific time	
Retry Attempts 🛈	- 10000 +	
Enable Now	✓ Enable	

The main parameter information is as follows. Keep the remaining parameters as default:

Trigger Method: select CKafka trigger.

CKafka Instance and Topic: select the corresponding topic as needed.

Start Point: select Earliest.

6. Click **Complete**.

Viewing ES and function execution logs

Note:

If you have not ingested actual data into CKafka, you can use the client tool to simulate message production.

Select Log Query on the sidebar of the function to view the function execution log.

View Kibana. For more information, please see Accessing Clusters from Kibana.



Scalability

If you want to implement advanced log cleansing logic, you can modify the logic in the code location.



SCF + CKafka for Message Dump to TencentDB for MySQL

Last updated : 2024-12-02 16:35:34

Overview

CKafka allows you to dump messages to TencentDB for MySQL to persistently store filtered data.

Prerequisites

Currently, this feature relies on the SCF and TencentDB for MySQL services, which should be activated first before you can use this feature.

Directions

Message dump to TencentDB for MySQL will be performed with the CKafka trigger in SCF.

1. Log in to the CKafka console.

2. Click **Instance List** on the left sidebar and click the **ID/Name** of the target instance to enter the instance details page.

3. On the instance details page, select the **Topic Management** tab and click **Message Dump** in the **Operation** column.

4. Click Add Message Dump and select General template as the dump type.

Dump Type: select General template.

Starting Position: the topic offset of historical messages when dumping.

SCF Authorization: indicate your consent to activating SCF and create a function. Then, you need to go to the function settings to set more advanced configuration items and view monitoring information.

5. After the creation is completed, click the Function Management link to enter the SCF console for the next step.

6. Upload the code of the CKafkaToMysql template (available on GitHub) to the SCF console.



bmitting Method () • Online editing • Execution	⑦ * index.main_handler Runtime Environment Nodejs8.9	Development Tutorial of Node.js8.9 🗳	Download
Cloud Studio	ninal Help	🗐 Test Template:Hello World event template 🛛 🛱 Test	🚯 Deploy
EXPLORER Local ZIP file	JS index.js ×		Π.
 > OPEN EDIT Local folder > APIGWWEI Upload a ZIP pack 57 DEPLOY > src > demo.html J5 index.js 	<pre>src > JS indexjs > 1 const fs = require('fs') 2 const path = require('path') 3 4 exports.main_handler = async (event, context, callback) => { 5 let html = fs.readFileSync(path.resolve(_dirname, './demo.ht 6 encoding: 'utf-8' 7 }) 8 return { 9 lisBase64Encoded: false, 10 statusCode: 200, 11 headers: { 'Content-Type': 'text/html; charset=utf-8' }, 12 body: html 13 }</pre>	tml'), {	STATE Martin Contraction Martin

7. Add the following environment variables in **Function Configuration** of the function.

key	value	(i)
dbhost	172.16.0.59	×
dbuser	tabor	×
dbpwd	1233123323	×
dbtable	123321	×
dbdatabase	canmengtest	×
Please enter the environme	Please enter the value of e	×
	key dbhost dbuser dbpwd dbtable dbdatabase Please enter the environme	keyvaluedbhost172.16.0.59dbusertabordbpwd1233123323dbtable123321dbdatabasecanmengtestPlease enter the environm+Please enter the value of e

dbhost=172.16.0.59 // Databases VPC host address
dbuser=tabor // Database username
dbpwd=1237018 // Database password
dbdatabase=canmengtest // Database name
dbtable=123321 // Table name

8. Modify the VPC in **Function Configuration** to make the VPC of the function the same as that of TencentDB.

Network Conf	figuration		
Public network	✓ Enable (i)		
Fixed outbound	Enable (j		
VPC	✓ Enable (j)	•	▼ 🗘 Create VPC 🗹

9. In the TencentDB for MySQL DMC console, add the relevant databases, tables, and table structures. Create a database with the same name as in the environment variables:

Create a table with the same name as in the environment variables:

Create a table structure with the same insertion structure as in the function code. offset and Megs will be inserted to by default. You can modify the insertion structure on line 33 in the index.py file.

You can also run a TencentDB for MySQL command to directly create tables and data structures:

CREATE TABLE `test_table` (`offset` VARCHAR(255) NOT NULL , `Megs` LONGTEXT NOT NU

10. Enable the CKafka trigger on the trigger management page in the SCF console.

Restrictions and Billing

The dump speed is subject to the limit of the peak bandwidth of the CKafka instance. If the consumption is too slow, check the peak bandwidth settings.

The CKafkaToMySQL scheme uses the CKafka trigger. For more information on related settings such as retry policy and maximum number of messages, see CKafka Trigger.

When the dump to MySQL feature is used, the dumped messages are the offset and msgBody data of the CKafka trigger by default. If you want to process the logic by yourself, see Event Message Structure for CKafka Trigger.

This feature is provided based on the SCF service that provides a free tier. For more information on the fees for excessive usage, see the billing rules of SCF.

SCF + CKafka for Message Dump to CKafka

Last updated : 2024-12-02 16:35:34

Overview

CKafka allows you to dump messages to another CKafka cluster to sync data between CKafka clusters.

Prerequisites

Currently, this feature relies on the SCF and CKafka services, which should be activated first before you can use this feature.

Directions

Dumping message

Message dump to CKafka will be performed with the CKafka trigger in SCF to sync messages to another CKafka cluster.

1. Log in to the CKafka console.

2. Click **Instance List** on the left sidebar and click the **ID/Name** of the target instance to enter the instance details page.

3. On the instance details page, select the **Topic Management** tab and click **Message Dump** in the **Operation** column.

4. Click Add Message Dump and select CKafka as the dump type.

Dump Type: select CKafka.

Dump Instance: pull the list of CKafka instances in the current region. If you want to dump data to an instance in another region or self-built Kafka, see Custom dump settings.

Dump Topic: pull the CKafka topic information of the selected instance.

Starting Position: the topic offset of historical messages when dumping.

Role Authorization: to use the SCF service, you need to grant a third-party role to perform access to related products for you.

SCF Authorization: indicate your consent to activating SCF and create a function. Then, you need to go to the function settings to set more advanced configuration items and view monitoring information.

5. After configuring the dump task, click **Submit**. After it is successfully created, dump will not start immediately; instead, you need to manually enable dump in the console.

Custom dump settings

In the general creation process, dump cannot be performed directly across regions or in self-built Kafka. For such dump, you need to configure the network or delivery information for the function. The cross-region dump process is as follows:

1. Create a CKafka dump template, go to the SCF console, and configure the delivery instance and topic as needed.

2. Modify Environment Variable and Network settings in the function configuration.

Environment variable configuration description:

kafka_address: Kafka IP address.

kafka_topic_name: Kafka topic name.

Note:

To dump CKafka data across regions, simply modify the relevant environment variables. You need to configure a **peering connection** for VPC.

For a CVM-based self-built Kafka instance, you need to change them to the VPC and Kafka topic of the instance. For other self-built Kafka instances, you need to change the IP and topic in the environment variable to the information of the instance. If there is no Direct Connect, data needs to be transferred over the SCF public network.

3. Save the configurations and enable the dump feature.

Notes

Access methods

SCF supports two CKafka access methods: PLAINTEXT and SASL_PLAINTEXT , which can be modified in the SCF code.

PLAINTEXT access method:

kafka_to_kafka = KafkaToKafka(kafka_address)

SASL_PLAINTEXT access method:

```
kafka_to_kafka= KafkaToKafka(
  kafka_address
  security_protocol = "SASL_PLAINTEXT",
  sasl_mechanism="PLAIN",
  sasl_plain_username="ckafka-80o10xxx#Taborxx",
  sasl_plain_password="Taborxxxx",
  api_version=(0, 10, 2)
)
```

Note:

sasl_plain_username consists of Instance ID and username concatenated with #.

Viewing dump log and troubleshooting

CKafka dump capabilities are implemented based on SCF, and you can find relevant dump information and status in the SCF log.

Restrictions and Billing

The dump speed is subject to the limit of the peak bandwidth of the CKafka instance. If the consumption is too slow, check the peak bandwidth settings.

The CKafkaToCKafka scheme uses the CKafka trigger. For more information on related settings such as retry policy and maximum number of messages, see CKafka Trigger.

When the dump to CKafka feature is used, the dumped messages are the offset and msgBody data of the CKafka trigger by default. If you want to process the logic by yourself, see Event Message Structure for CKafka Trigger.

This feature is provided based on the SCF service that provides a free tier. For more information on the fees for excessive usage, see the billing rules of SCF.

CLS CLS Function Processing Overview

Last updated : 2024-12-02 16:35:34

Through the function processing service, you can quickly complete complex log processing tasks such as execution log collection, ETL (extraction-transformation-loading), and message dumping for various Tencent Cloud services like CVM. Function processing is an async process. All data collected into CLS can be delivered to SCF for consumption and processing through configurations. You only need to complete simple configurations in the CLS console to connect CLS to SCF.

The source log data can be submitted to SCF through a CLS trigger, and then data processing and analysis, event triggering, and auto scaling can be implemented through function computing of Serverless Framework. The entire workflow requires no OPS and is pay-as-you-go as shown below:



Advantages of Function Processing

Data can be collected, stored, processed, analyzed, and displayed in a one-stop manner.

Log processing tasks are fully managed and can be triggered regularly and retried automatically.

More and more built-in function templates are added to reduce the development costs for log processing in popular use cases.

Data processing logic and custom code logic are provided based on SCF.

Computing power is provided based on SCF, with features such as auto scaling, OPS-free usage, and pay-as-you-go billing.

Multi-scenario Function Processing Practices

CLS can send data in a log topic to SCF for processing through a CLS trigger to satisfy the needs of various use cases such as log processing and log cleansing, as detailed below:

Function Processing Scenario	Description
Log ETL	Log data is cleansed, processed, or transformed through SCF
CLS data dump to CKafka	Log data is cleansed and delivered to CKafka through SCF
CLS data dump to COS	Log data is cleansed and delivered to COS through SCF
CLS data dump to ES	Log data is delivered to ES through SCF

Note:

Data is delivered to SCF, which incurs corresponding computation fees. For billing details, please see Billing Overview.

Log ETL

Last updated : 2024-12-02 16:35:34

Overview

This document describes how to use SCF to process CLS logs. CLS is mainly used for log collection, while SCF mainly provides node computing capabilities for data processing. The data flow is as follows:



Directions

Creating logset and topic

1. Log in to the CLS console and click Log Topic on the left sidebar.

2. On the logset management page, select the region of the logset at the top.

3. Click Create Log Topic and enter relevant information in the Create Logset pop-up window:

Log Topic Name: enter project_test for example.

Logset Name: enter nginx for example.

4. Click OK.

5. The log topic is successfully added, and you will be redirected to the log topic management **Note:**

As both the source and destination of data ETL are CLS, you need to create at least two topics.

Creating SCF function

1. Log in to the SCF console and select **Function Service** on the left sidebar.

2. At the top of the Function Service page, select the Beijing region and click Create to enter the function creation

page and configure the following parameters:

Function name: enter "CLSdemo".

Runtime environment: select Python 2.7.

Creation method: select Function Template.

Fuzzy search: enter "CLSLogETL" and search.

3. Click **Learn More** in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

4. After configuring the basic information, click **Next** to enter the function configuration page.

5. Keep the default function configuration and click **Complete** to complete the function creation.

Configuring CLS trigger

1. Log in to the Serverless console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the region and namespace where the function for which to configure a CLS trigger resides.

3. Click the function name to enter the function details page.

4. On the function details page, select **Trigger Management** on the left to enter the trigger browsing and operation page. Click **Create a Trigger** to create a trigger.

5. Add the created function in the Create a Trigger pop-up window

6. After completing the trigger configuration, click **Submit** to create the trigger.

Testing function

1. Download the log file in the test sample, extract demo-scfl.txt , and import it to the source CLS service.

2. Switch to the Serverless console to view the execution result.

Select the Log Query tab on the function details page to view the printed log information

3. Switch to the target CLS service to view the data processing result.

Note:

You can write specific data processing methods as needed.

SCF +CLS Dump to CKafka

Last updated : 2024-12-02 16:35:34

Overview

This document describes how to use SCF to dump CLS logs to CKafka. CLS is mainly used for log collection, SCF mainly provides node computing capabilities for data processing, and CKafka mainly provides terminal dumping capabilities for data streams. For the data processing flowchart, please see Function Processing Overview.

Directions

Creating logset and topic

1. Log in to the CLS console and click Log Topic on the left sidebar.

2. On the logset management page, select the region of the logset at the top.

3. Click Create Log Topic and enter relevant information in the Create Logset pop-up window:

Log Topic Name: enter project_test for example.

Logset Name: enter nginx for example.

Create Log Topic	×	
Log Topic Name	project_test	
Storage Class	Real-time	
Log Retention Period	- 30 + days	
	Value range: 1-366	
Logset Operation	Select an existing logset. Create Logset	
Logset	nginx 💌	
Advanced Settings		
	OK Cancel	

🔗 Tencent Cloud

4. Click OK.

5. The log topic is successfully added, and you will be redirected to the log topic management page as shown below:

← proje	ect_test								
Basic Info	Co	lection Configuration	Index Configuration	Ship to COS	Ship to CKafka	Function Processing			
Basic Inf	0								
Log Topic	Name	project_test							
Log Topic	D	aab0870c-0362-4ee4-97ea	a-0888a26991a3 🕞						
Logset		nginx							
Logset ID		a9d53dcc-0161-4171-835	8-8a6b4d9c6c0a 🛅						
Region		Guangzhou							
Partition A	uto-Split	Enabled							
Storage Cl	ass	Real-time							
Maximum	Partitions	50							
Retention ⁻	lime	30 days							
Tag		<i>l</i> [*]							
_									
Topic Pa	rtition Ma	inagement							What is log topic partition
Partition	ID \$		Status		Partition Ran	ge	Last Modified	Operation	
1			Read & Write		Start: 000 End: fffff	00000000000000000000000000000000000000	-	Edit	

Creating SCF function

1. Log in to the Serverless console and enter the function service page.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creation page and configure the following parameters:

Function name: enter "CLSdemo".

Runtime environment: select Python 2.7.

Creation method: select Function Template.

Fuzzy search: enter "CLSToCkafka" and search.

3. Click **Learn More** in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

4. After configuring the basic information, click **Next** to enter the function configuration page.

5. Keep the default function configuration and click **Complete** to complete the function creation.

Note:

You should select the same VPC and subnet of CKafka for the created function on the "Function Configuration" page as shown below:



VPC	✓ Enable (i)					
	Please select the VPC	•	Please select a subnet	▼	¢	Create VPC

Configuring CLS trigger

1. Log in to the CLS console and click Log Topic on the left sidebar.

2. Find the created logset (such as "project_test") and click **View** in the **Operation** column on the right to enter the logset details page.

3. On the log topic details page, select **Function Processing** and click **Create** in the top-right corner. Add the created function in the pop-up "Function Processing" window as shown below:

Namespace	Please sel	ect	•	Φ			
Function Name	Please sel	ect	•	φ	Create Function 🛂		
Version/Alias	Please sel	ect	•	φ			
Batch Window(i)	- 60	+ s					
	Value range:	3-300					

The main parameter information is as follows. Keep the remaining configuration items as default:

Namespace: select the function namespace.

Function name: select the function created in the Creating SCF function step.

Alias: select a function alias.

Maximum waiting time: configure the longest waiting time for a single event pull. Default value: 60s.

Testing function

- 1. Download the log file in the test sample, extract demo-scfl.txt , and import it to the source CLS service.
- 2. Switch to the Serverless console to view the execution result.

Select the Log Query tab on the function details page to view the printed log information as shown below:

All Logs v Last 15 min. v	2021-11-03 16:53:54 ~ 2021-11-03 17:08:54 📩 Refresh	Please enter the re Q
021-11-03 17:08:42 Invoked successfully	Request ID: : 7a42ea38-2c24-459e-bfb3-ecda90f022f4	statusCode Description and Solution
	Time: 2021-11-03 17:08:42 Runtime:4ms Execution memory:45.6328125MB	
	Log: START Requestid: d6aa8c7c-70e4-44cc Event Requestid: d6aa8c7c-70e4-44cc- start main handler ('Start Request {}', '2020-07-07 02:30:47') Key is demo-scf1.txt Get from [loganalysis-] to download file [demo-scf1.txt] get object, url=https://loganalysis- ?.cos.ap-beijing.myqcloud.com/demo-scf1.txt ,headers=:{}, params=:{} Download file [demo-scf1.txt] Get from [loganalysis-] to download file [demo-scf1.txt] get object, url=https://loganalysis- ?.cos.ap-beijing.myqcloud.com/demo-scf1.txt ,headers=:{}, params=:{} Download file [demo-scf1.txt] ('Start analyzing data {}', '2020-07-07 02:30:47') ('Analyzing Successfully, Start writing to database {}, '2020-07-07 02:30:47') /var/user/pymysql/cursors.py:329: Warning: (1051, u*Unknown table 'mason_demo.url") self_do_get_result() /var/user/pymysql/cursors.py:329: Warning: (1051, u*Unknown table 'mason_demo.state") self_do_get_result() /var/user/pymysql/cursors.py:329: Warning: (1051, u*Unknown table 'mason_demo.terminal") self_do_get_result() /var/user/pymysql/cursors.py:329: Warning: (1051, u*Unknown table 'mason_demo.terminal") self_do_get_result() /var/user/pymysq	6

3. Log in to the CKafka console to view the data dumping and processing result.

Note:

You can write specific data processing methods as needed.

SCF +CLS Dump to COS

Last updated : 2024-12-02 16:35:34

Overview

This document describes how to use SCF to dump CLS logs to COS. CLS is mainly used for log collection, SCF mainly provides node computing capabilities for data processing, and COS mainly provides persistent data storage capabilities. For the data processing flowchart, please see Function Processing Overview.

Directions

Creating logset and topic

- 1. Log in to the CLS console and click Log Topic on the left sidebar.
- 2. On the logset management page, select the region of the logset at the top.
- 3. Click Create Log Topic and enter relevant information in the Create Logset pop-up window:

Log Topic Name: enter project_test for example.

Logset Name: enter nginx for example.

Create Log Topic		×
Log Topic Name	project_test	
Storage Class	Real-time	
Log Retention Period	- 30 + days	
	Value range: 1-366	
Logset Operation	O Select an existing logset. Create Logset	
Logset	nginx 👻	
Advanced Settings		
	OK Cancel	

4. Click OK.

5. The log topic is successfully added, and you will be redirected to the log topic management page as shown below:

← project_test											
Basic Info	Collection Configuration	Index Configuration	Ship to COS	Ship to CKafka	Function Processing						
Basic Info											
Log Topic Name	project_test										
Log Topic ID	aab0870c-0362-4ee4-97e	a-0888a26991a3 🕞									
Logset	nginx										
Logset ID	a9d53dcc-0161-4171-835	a9d53dcc-0161-4171-8358-8a6b4d9c6c0a 🖺									
Region	Guangzhou										
Partition Auto-S	olit Enabled										
Storage Class	Real-time										
Maximum Partiti	ons 50										
Retention Time	30 days	30 days									
Tag	i										
Topic Partition Management What is log topic partition											
Partition ID		Status		Partition Ra	nge	Last Modified		Operation			
1		Read & Write		Start: 000 End: ffff	00000000000000000000000000000000000000	-		Edit			

Creating SCF function

1. Log in to the Serverless console and enter the function service page.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creation page and configure the following parameters:

Creation method: select Template.

Fuzzy search: enter "CLSSCFCOS" and search.

3. Click **Learn More** in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

Tencent Cloud	Overvie	w Produ	icts v Serverles	s Cloud Function	Cloud Access Manage	ment Cloud Object Stor	ige Cloud Kafka	API Gateway	Cloud Message Queu	e ••• +	Ĕ	🛛 🌄 Ticket 🔻	Billing Center v	English Y
Serverless Cloud Function	÷	Create												
B Overview														
Function Service			Create Method	Template		Custom								
🛇 Layer				Use demo ten or application	nplate to create a function	Create a custom function HelloWolrd demo	using							
			Fuzzy search	CLSSCFCOS	Separate multiple ta	gs with carriage returns		© Q					Sort by recomm v	
				CLSSCFCC)S	Learn More								
				Category	Function									
				Description	This demo will connect C message automatically. T	LS and consume he SCF will write								
				Tag	Python2.7 CLS Consumer ETL	cos								
				Author	🔗 Tencent Cloud									
				Deploy	1.1									
			Next Car	ncel										

4. After configuring the basic information, click **Next** to enter the function configuration page.

Function name: enter "CLSdemo".

Select the **Beijing** region.

5. Keep the default function configuration and click **Complete** to complete the function creation.

Configuring CLS trigger

1. Log in to the CLS console and click Logset on the left sidebar.

2. Find the created logset and click View in the Operation column on the right to enter the logset details page.

3. On the log topic details page, select Function Processing and click Create. Add the created function in the pop-

up "Function Processing" window as shown below:

Function Proce	ssing		×
Namespace	Please select	▼ Ø	
Function Name	Please select	✓ Create Function	
Version/Alias	Please select	▼ ¢	
Batch Window(i)	- 60 + s Value range: 3-300		
		OK Cancel	

The main parameter information is as follows. Keep the remaining configuration items as default:

Namespace: select the function namespace.

Function name: select the function created in the Creating SCF function step.

Alias: select a function alias.

Maximum waiting time: configure the longest waiting time for a single event pull. Default value: 60s.

Testing function

- 1. Download the log file in the test sample, extract demo-scfl.txt , and import it to the source CLS service.
- 2. Switch to the Serverless console to view the execution result.

Select the Log Query tab on the function details page to view the printed log information as shown below:


3. Log in to the COS console to view the data dumping and processing result.

Note:

You can write specific data processing methods as needed.

Dumping Logs to ES with SCF + CLS

Last updated : 2024-12-02 16:35:34

Scenario

This document describes how to use SCF to dump CLS logs to ES. CLS is mainly used for log collection, and SCF mainly provides node computing capabilities for data processing. For the data processing flowchart, see Function Processing Overview.

Directions

Creating logset and topic

- 1. Log in to the CLS console, and click Log Topic on the left sidebar.
- 2. On the logset management page, select the region of the logset at the top.
- 3. Click Create Log Topic and enter relevant information in the Create Logset pop-up window:

Log Topic Name: Enter project_test for example.

Logset Name: Enter nginx for example.

2	project_test	
og retention policy *	Log access *	STANDARD storage
	~	All capabilities are available for the STANDARD storage data. IA is less expensive but does not support SQL, chart analysis, and alarms. For details, see Storage Class Overview
	Retention Time *	Terminable retention 💌 🦳 30 🕂 days
		Logs can be stored for 1 to 3600 days or persistently. Transitioning can be enabled when the log retention period exceeds 7 days.
	Log transition	
		After it is enabled, the data will be transitioned from STANDARD storage to IA when data access meets the set period. The storage costs will be reduced after the data is transitioned. For details, see Log transition 2.
ogset Operation *	Select an existin	ig logset. O Create Logset
ogset Name *	nginx	
ogset Tag 🚯	Tag key	▼ Tag value ▼ X
	+ Add	
og Topic Tag 🚯	Tag key	▼ Tag value ▼ X
	+ Add	
	Optional. Up to 2	55 characters.
og topic description		
og topic description		
og topic description		

4. Click OK.

5. The log topic is successfully added, and you will be redirected to the log topic management page.

project_	test							
Basic Info	Collection Configuration	Index Configuration	Ship to COS	Ship to CKafka	Function Processing			
Basic Info								
Log Topic Name	project_test							
Log Topic ID	aab0870c-0362-4ee4-97e	ea-0888a26991a3 🕞						
Logset	nginx							
Logset ID	a9d53dcc-0161-4171-835	58-8a6b4d9c6c0a 🖺						
Region	Guangzhou							
Partition Auto-S	plit Enabled							
Storage Class	Real-time							
Maximum Partit	ions 50							
Retention Time	30 days							
Tag	i							
Topic Partitic	on Management							What is log topic partition
Partition ID	ŧ	Status		Partition Ran	ge	Last Modified	Operation	
1		Read & Write		Start: 000 End: fffff	00000000000000000000000000000000000000	-	Edit	

Creating an SCF function

1. Log in to the SCF console and enter the **Functions** page.

2. At the top of the **Functions** page, select the **Beijing** region and click **Create** to go to the function creating page and configure the following parameters:

Creation method: Select Template.

Fuzzy search: Enter CLSToElasticsearch and search.

3. You can click **Learn more** in a template to view its details or download the template. Click **Next** to enter the **Function configuration** page.

÷	Create						
		Create Method	Template Use demo tem or application	nplate to create a function	Custom Create a custom functio HelloWolrd demo	n using	
		Fuzzy search	CLSToElastics	search 🛞 Separate multip	le tags with carriage return	าร	©Q
			CLSToElast Category Description Tag Author Deploy	icsearch Function This demo will connect CL message automatically. Python2.7 CLS ES Tencent Cloud	Learn More S and consume		
		Next Can	cel				

4. In the **Basic configurations** section, the function name has been automatically generated and can be modified as needed. Follow the prompts to configure environment variables and execution role.

Environment variable: Add the following environment variables and configure them.

Environment variable *	key	value	
	ES_Address	ES_Address	×
	ES_User	ES_User	×
	ES_Password	ES_Password	×

key	value	Required/Optional
ES_Address	ES service address	Yes



ES_User	ES username, which is elastic by default.	Yes
ES_Password	ES password.	Yes

Execution Role: Select Enable and click Configure and use SCF template role.

5. In **Network configuration**, select **Enable** for **VPC**, and select the same VPC and subnet as Elasticsearch.

VPC	✓ Enable (i)					
	Please select the VPC	*	Please select a subnet	*	¢	Create VPC

6. Keep the default configuration and click **Complete** to complete the function creation.

Configuring CLS triggers

1. Log in to the CLS console and click Logset on the left sidebar.

2. Locate the existing logset you want, and click **View** under **Operation** on the right to go to the logset details page.

3. On the log topic details page, select **Function Processing** and click **Create**. Add the created function in the **Function Processing** pop-up window.

Function Proce	ssing		
Namespace	Please select	τØ	
Function Name	Please select	ΨØ	Create Function 🗹
Version/Alias	Please select	▼ Ø	
Batch Window 🕄	- 60 + s		
	Value range: 3-300		
		ОК	Cancel

The main parameter information is as follows. Use the default values for the remaining configuration items.

Namespace: Select the function namespace.

Function name: Select the function created in the Creating SCF function step.

Alias: Select a function alias.



Max waiting time: Configure the longest waiting time for a single event pull. Default value: 60s.

Testing the function

- 1. Download the log file in the test sample, extract demo-scfl.txt , and import it to the source CLS service.
- 2. Go to the SCF console to view the execution result.

Select the Log Query tab on the function details page to view the printed log information.

D21-11-03 17:08:42 Invoked successful Request ID: : 7a42ea38-2c24-4599-bfb3-ecda9002214 Imme: 2021-11-03 17:08:42 Runtime:4ms Execution memory:45.6328125MB Imme: 2021-11-03 17:08:42 Runtime:4ms Execution memory:45.6328125MB Imme: 2021-11-03 17:08:42 Runtime:4ms Execution memory:45.6328125MB Imme: 2021-11-03 17:08:42 Runtime:4ms Execution memory:45.6328125MB Imme: 2021-11-03 17:08:42 Runtime:4ms R	All Logs 🛛 🔻	Last 15 minu v 2	021-11-03 16:53:54 ~ 2021-11-03 17:08:54 💼 Refresh	Please enter the re C
Time: 2021-11-03 17:08:42 Runtime:4ms Execution memory:45:6328125MB Log: START Requestid: d6aa8c7c-70e4-44cc Execution memory:45:6328125MB StaRT Requestid: d6aa8c7c-70e4-44cc- start main handler ('Start Request)(', '2020-07-07 02:30:47') Key is demo-scf1.txt Get from [loganalysis-] to download file [demo-scf1.txt] get object. urf=.https://oganalysis- ?cos.ap-beijing.myqcloud.com/demo-scf1.txt ,headers=:{}, params=:{} Download file [demo-scf1.txt] get odject. urf=.https://oganalysis- ?cos.ap-beijing.myqcloud.com/demo-scf1.txt ,headers=:{}, params=:{} Download file [demo-scf1.txt] get odject. urf=.https://oganalysis- ?cos.ap-beijing.myqcloud.com/demo-scf1.txt ,headers=:{}, params=:{} Download file [demo-scf1.txt] get odject.urf=.https://oganalysis- ?cos.ap-beijing.myqcloud.com/demo-scf1.txt ,headers=:{}, params=:{} Download file [demo-scf1.txt] get get_result] Yutivescripymsgl/cursors py:329. Warning: (1051, u*Unknown table 'mason_demo.urf") self_do_get_result() Yurivser/pymsgl/cursors py:329. Warning: (1051, u*Unknown table 'mason_demo.teminal") self_do_get_result() Yurivser/pymsgl/cursors py:329. Warning: (1051, u*Unknown table 'mason_demo.teminal") self_do_get_result() Yurivser/pymsgl/cursors py:329. Warning: (1051, u*Unknown table 'mason_demo.teminal") self_do_get_result() Yurivser/pymsgl/cursors py:329. Warning: (1051, u*Unknown table 'mason	21-11-03 17:08:42	Invoked successfully	Request ID: : 7a42ea38-2c24-459e-bfb3-ecda90f022f4	statusCode Description and Solution
Log: START Requestid: d6aa8c7c-70e4-44cc Event Requestid: d6aa8c7c-70e4-44cc- start main handler ('Start Request (), '2020-07-07 02:30:47') Key is demo-scf1.bt Get from [loganalysis]] to download file [demo-scf1.bt] get object, url=:https://loganalysis- [Obwnload file [demo-scf1.bt] get object, url=:https://loganalysis- ('Start analyzing Successtilly, Start writing to database ()', '2020-07-07 02:30:47') ('Analyzing Successtilly, Start writing to database ()', '2020-07-07 02:30:47') Nar/user/pymysql/cursors py:329. Warning: (1051, u*Unknown table 'mason_demo.url*') self_do_get_result() Nar/user/pymysql/cursors py:329. Warning: (1051, u*Unknown table 'mason_demo.terminal*') self_do_get_result() Nar/user/pymysql/cursors py:329. Warning: (1051, u*Unknown table 'mason_demo.terminal*') self_do_get_result() ('Write to database successfully ()', '2020-07-07 02:30:48')			Time: 2021-11-03 17:08:42 Runtime:4ms Execution memory:45.6328125MB	
END Requestid: d6aa8c7c-70e4-44cc-b2c5			Log: START Requestid: d6aa8c7c-70e4-44cc Event Requestid: d6aa8c7c-70e4-44cc- start main handler ('Start Request)(', '2020-07-07 02:30:47') Key is demo-scf1.txt Get from [loganalysis-] to download file [demo-scf1.txt] get object, url=.https://loganalysis- // cos.ap-beijing.myqcloud.com/demo-scf1.bt ,headers=:{}, params=:{} Download file [demo-scf1.txt] get object, url=.https://loganalysis- // cos.ap-beijing.myqcloud.com/demo-scf1.bt ,headers=:{}, params=:{} Download file [demo-scf1.txt] get object, url=.https://loganalysis- // 2020-07-07 02:30:47') ('Analyzing Successfully, Start writing to database {}, '2020-07-07 02:30:47') ('Analyzing Successfully, Start writing to database {}, '2020-07-07 02:30:47') /var/user/pymysql/cursors.py:329: Warning: (1051, u*Unknown table 'mason_demo.url*') selfdo_get_result() /var/user/pymysql/cursors.py:329: Warning: (1051, u*Unknown table 'mason_demo.terminal*') selfdo_get_result() /var/user/pymysql/cursors.py:329: Warning: (1051, u*Unknown table 'mason_demo.terminal*') selfdo_get_result() /Write to database successfully ()', '2020-07-07 02:30:48') END Requestid: d6aa8c?rc-70e4-44cc-b2c5 Enonet Beaverdid: d6ca8c?rc-70e4-44cc-b2c5	ι Γ

3. Log in to the ES console to view the data dumping and processing result.

Note:

You can write specific data processing methods as needed.

CLB Practice Using SCF and CLB to Quickly Deploy Web Service

Last updated : 2024-12-02 16:35:34

Overview

This document describes how to use CLB as the access entry of a serverless service and work with SCF to quickly deploy a web service. In this way, you can enjoy the strengths of Serverless services, such as low cost and OPS-free usage, and smoothly migrate your application to the cloud.

Directions

Creating VPC

Log in to the VPC console to create a VPC and subnet as instructed in Building Up an IPv4 VPC.

Note:

The VPC should be deployed in the same region as the CLB instance and SCF webpage. This document uses **Shanghai** as an example.

Creating CLB instance

1. Log in to the CLB console to create a CLB instance as instructed in Getting Started with CLB.

This document uses **Shanghai** as an example. Select the VPC created in the previous step.

2. After successful creation, find the target CLB instance on the **Instance Management** page and configure a listener for it as instructed in Configuring an HTTP Listener.

This document uses the listener clb-scf-web and listening protocol port number 81 as an example.

Creating SCF function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Shanghai** region and click **Create** to enter the function creating page.

Set the following parameter information and click **Next** as shown below:

Creation Method: select Template.



Fuzzy Search: enter "Web static page hosting" and "Python3.6" and search.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. In **Basic Configuration**, enter the **function name** and select the **function region**.

Function Name: enter clb-scf-web for example.

Region: select the region of the CLB instance, such as **Shanghai**.

4. In Trigger Configuration, select Custom Creation and use the trigger to bind CLB to SCF.

Triggered Version: select **Default Traffic**.

Trigger Method: select **CLB Trigger**.

Instance ID: select the CLB instance created in the previous step, such as clb_serverless_web .

Listener: select the configured listener. In this example, the listener listens on port 81.

Domain Name/Host: select Create Rule.

Add Domain Name: enter the VIP of the CLB instance in the added domain name.

Note:

The VIP is the IP address through which CLB provides service to clients.

URL Path: enter the URL path of the website starting with /, such as /demo .

5. Click **Complete** to redirect to the **Deployment Log** page and view the function and trigger creation progress.

Testing CLB entry

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the Function Service page, click the created function clb-scf-web .

3. On the function details page, select **Trigger Management**.

4. On the **Trigger Management** page, get the trigger access path and view the webpage.

MPS MPS Function Processing Overview

Last updated : 2024-12-02 16:35:34

Through the function processing service, you can quickly process and operate on the callback events generated by MPS. Events are pushed to SCF through an MPS trigger, and then callback event processing and response are implemented through function computation of Serverless Framework.

The overall data processing flow is as shown below:



Function Processing Practices

CLS can send data in a log topic to SCF for processing through an MPS trigger to satisfy the needs of various use cases, such as video event notification, status monitoring, and alarm processing as detailed below:

Function Processing Scenario	Description



Video task callback backup to COS	Backs up task callbacks generated by MPS to COS through SCF promptly
Video task callback notification tool	Receives MPS data messages in real time and pushes them through WeCom, email, etc.

Note:

Data is delivered to SCF, which incurs corresponding computation fees. For billing details, please see SCF Billing Overview.

Video Task Callback Backup to COS

Last updated : 2024-12-02 16:35:34

Overview

This document describes how to use SCF to back up task callbacks generated by MPS to COS. SCF is mainly used to process callback information, MPS is mainly used for video processing tasks, and COS mainly provides persistent storage capabilities.

Directions

Creating function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creating page and configure the function.

Creation method: select Template.

Fuzzy search: enter "MPS_SCF_COS" and search.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click **Next**. The function name is automatically generated by default and can be modified as needed.

Configuring MPS trigger

1. In the **Trigger Configurations** section, select **Custom** and enter relevant information according to the displayed parameters.

Triggered Version: select Default Traffic.

Trigger Method: select MPS Trigger.

Event Type: select Workflow Task.

Note:

When creating an MPS trigger for the first time, you need to click **SCF_QcsRole** and **MPS_QcsRole** to authorize the relevant service role.

Event type: an MPS trigger pushes events in the account-level event type. Currently, two event types are supported: workflow task (WorkflowTask) and video editing task (EditMediaTask).

Event processing: an MPS trigger uses events generated at the service level as the event source, regardless of attributes such as region and resources. Only one single function can be bound to each event type for each account. If you need multiple functions to handle a task, please see SDK for Node.js.

2. Click **Complete**.

Testing function

1. Log in to the MPS console and execute a video processing workflow.

2. On the Function Service page in the SCF console, click the name of the function created in the Creating function

step above to enter the function details page.

3. Select the **Log Query** tab on the function details page to view the printed log information.

4. Log in to the COS console to view the data dumping and processing result.

Note:

You can write specific data processing methods as needed.

Video Task Callback Notification Tool

Last updated : 2024-12-02 16:35:34

Overview

This document describes how to use SCF to push MPS callback information. SCF is mainly used to process callback information, while MPS is mainly used for video processing tasks.

Directions

Creating function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creating page and configure the function.

Creation method: select Template.

Fuzzy search: enter "MPSWebhookDemo" and search.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

3. Click **Next**. The function name is automatically generated by default and can be modified as needed.

Configuring MPS trigger

1. In the **Trigger Configurations** section, select **Custom** and enter relevant information according to the displayed parameters.

Triggered Version: select Default Traffic.

Trigger Method: select MPS Trigger.

Event Type: select Workflow Task.

Note:

When creating an MPS trigger for the first time, you need to click **SCF_QcsRole** and **MPS_QcsRole** to authorize the relevant service role.

Event type: an MPS trigger pushes events in the account-level event type. Currently, two event types are supported: workflow task (WorkflowTask) and video editing task (EditMediaTask).

Event processing: an MPS trigger uses events generated at the service level as the event source, regardless of attributes such as region and resources. Only one single function can be bound to each event type for each account. If you need multiple functions to handle a task, please see SDK for Node.js.

2. Click Complete.

Testing function

1. Log in to the MPS console and execute a video processing workflow.

2. On the **Function Service** page in the SCF console, click the name of the function created in the Creating function step above to enter the function details page.

- 3. Select the **Log Query** tab on the function details page to view the printed log information.
- 4. Switch to **WeCom** to view the callback notification result.

Note:

You can write specific data processing methods as needed.

CDN SCF + CDN for Scheduled Prefetch/Purge

Last updated : 2024-12-02 17:40:42

Scheduled purge/prefetch tasks can be triggered through SCF. Such tasks are included in the daily purge/prefetch quota and may fail to be executed if the quota is exceeded.

Configurations

Log in to the CDN console, select **Plugin Center** on the sidebar, click the scheduled purge/prefetch plugin feature block, and enable scheduled purge/prefetch to enter the task configuration page. After this feature is enabled, you can also click **Basic Configuration** at the bottom of the block to enter the scheduled purge/prefetch task list page for configuration.

On the **Create Scheduled Task** page, select the desired task type, set the cron scheduling expression (see the example below), enter the corresponding purge/prefetch URLs, and perform SCF authorization. Then, the system can automatically generate the corresponding SCF function and trigger the task as scheduled.

Note:

Do not delete this function in the SCF console.

On the task status page, you can view the last execution of the scheduled task.

Cron Expression

A cron expression contains seven values separated by spaces.

Value	Field	Value Range	Wildcard
1	Second	An integer between 0 and 59	, -*/
2	Minute	An integer between 0 and 59	, -*/
3	Hour	An integer between 0 and 23	, -*/
4	Day	An integer between 1 and 31 (the number of days in the month needs to be considered)	, -*/
5	Month	An integer between 1 and 12 or JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, or DEC	, -*/
6	Day of	An integer between 0 and 6 or MON, TUE, WED, THU, FRI, SAT, or SUN,	, -*/

殓 Tencent Cloud

	week	where 0 means Monday, 1 means Tuesday, and so on	
7	Year	An integer between 1970–2099	, -*/

Wildcards have the following meanings:

Wildcard	Description
, (comma)	It represents the union of characters separated by commas; for example, 1, 2, 3 in the "Hour" field means 1:00, 2:00 and 3:00
- (hyphen)	It contains all values in the specified range; for example, in the "Day" field, 1-15 contains the 1st to the 15th day of the specified month
* (asterisk)	It means all values; for example, in the "Hour" field, * means every o'clock
/ (forward slash)	It specifies the increment; for example, in the "Minute" field, you can enter 1/10 to specify repeating every ten minutes from the first minute on (e.g., at the 11th minute, the 21st minute, the 31st minute, and so on)

Note:

When both the "Day" and "Week" fields in a cron expression are specified, they are in an "or" relationship, i.e., the conditions of both are effective separately.

Example

One-Time task

33 22 11 6 7 * 2021 means triggering the task at 11:22:33 on July 6, 2021.

00 00 20 25 10 * 2021 means triggering the task at 20:00:00 on October 25, 2021.

Periodic task

*/5 * * * * * means triggering the task once every 5 seconds.
0 2 1 * * means triggering the task once at 2 AM on the 1st day of every month.
0 15 10 * MON-FRI means triggering the task once every day at 10:15 AM Monday through Friday.
0 0 10,14,16 * * * means triggering the task once every day at 10 AM, 2 PM, and 4 PM.
0 */30 9-17 * * * means triggering the task once every half hour from 9 AM to 5 PM every day.
0 0 12 * * WED * means triggering the task once at 12:00 noon every Wednesday.

Billing Description

The scheduled purge/prefetch feature is free of charge. However, it will call SCF to create scheduled tasks, which may incur SCF fees if the free tier of SCF is exceeded. For more information, see Free Tier.

CDWPG SCF + CDWPG for CKafka Data Import

Last updated : 2024-12-02 17:40:42

Overview

This document introduces a free-of-maintenance approach to import Kafka data to Cloud Data Warehouse PostgreSQL instances by using SCF.

Cloud Data Warehouse PostgreSQL (CDWPG) can sync messages from the messaging middleware for analysis.

Limits

Only Tencent Cloud CKafka is supported as the data source. External Kafka services are not supported. One function can only import data to one table in CDWPG. To write data into multiple tables, you need to create one function for each table.

Directions

Step 1. Create a function

In the SCF console, select Functions > Create. In the Create page, enter ckafka and CDW in the Fuzzy search field, complete the settings and click Next.

Ckafka CDW Separate multiple tags with carriage returns Q Total: 2 CkafkaLoadToCDWByT Learn more Category Function Description This example consumes CKafka data by CKafka trigger, then write the data to CDW. Tag Python3.6 Ckafka_CDW ETL CA Tencent Cloud Deploy 14,460 time	Template Use demo templ application	ate to create a function or	Create from scratch Start from a Hello World sample		Use TCR image Create a function based on a TCR image		
CkafkaLoadToCDWByT Learn more CkafkaConnectorSinkT Learn more Category Function Category Function Description This example consumes CKafka data by CKafka Category Function Tag Python3.6 CKafka CDW EL CA Tag Python3.6 CKafka CDW EL Description Tag Python3.6 CKafka CDW EL CA Category Sink CDW EL EL Deploy 14,460 time CA Tage 9,759 time	Fuzzy search	ckafka CDW Separate mu	tiple tags with carriage returns		Q Total: 2	s	ort by recommendatio
Description This example consumes CKafka data by CKafka trigger, then write the data to CDW. Description This example consumes CKafka data by CKafka Connector Sink, then write the data to CDW. Tag Python3.6 CKafka CDW Tag Python3.6 CKafka Sink CDW ETL CA Connector Sink CDW ETL ETL ETL Deploy 14,460 time CA Cancent Cloud Deploy 9,759 time		CkafkaLoadToCDWByT Category Function	Learn more	CkafkaCon	rectorSinkT Learn more		
Tag Python3.6 CKafka CDW ETL Tag Python3.6 CKafka Connector CA \textcircled{O} Tencent Cloud Sink CDW ETL Sink CDW ETL Deploy 14,460 time CA \textcircled{O} Tencent Cloud CA \textcircled{O} Tencent Cloud		Description This example co trigger, then wri	nsumes CKafka data by CKafka ie the data to CDW.	Description	This example consumes CKafka data by CKafka Connector Sink, then write the data to CDW.		
Deploy 14,460 time CA CA OF Tencent Cloud Deploy 9,759 time		Tag Python3.6 (CA So Tencent Clou	CKafka CDW ETL	Tag	Python3.6 CKafka Connector Sink CDW ETL		
		Deploy 14,460 time		CA Deploy	Tencent Cloud 9,759 time		

On the Function configuration page, complete the settings in Environment configuration and Network configuration in Advanced configuration as follows:

Environment configuration

Memory: Set the memory based on the actual running status, which is 128 MB by default. If it is insufficient during data import, you should increase it.

Environment variable:

Parameter	Required	Description
DB_DATABASE	Supported	Database name
DB_HOST	Supported	If the function is deployed in a VPC and in the same subnet as CDWPG, you can enter the private IP of CDWPG; otherwise, enter the public IP and configure an allowlist.
DB_USER	Supported	Username
DB_PASSWORD	Supported	User password
DB_SCHEMA	Supported	Schema name. If it is not specified during table creation, it will be `public` in general.
DB_TABLE	Supported	Table name
DB_PORT	No	CDWPG port, which is 5436 by default.
MSG_SEPARATOR_ASCII	No	ASCII code of the data delimiter in CKafka, which is 39 (comma) by default. As commas usually show up in the business data, we



		recommend you set this parameter to 11 (vertical bar).
MSG_NULL	No	NULL value of CKafka consumption. The default value is `\\N`
REPLACE_0X00	No	Whether to replace "0x00" in strings. The default value is 0 (1 indicates to replace).
ENABLE_DEBUG	No	Whether to print error records. The default value is 0 (1 indicates to print).
ENABLE_COS	No	Whether to dump unwritten records to COS. The default value is 0 (1 indicates to dump).
COS_SECRET_ID	No	`secret_id` for COS access. If `ENABLE_COS` is 1, this field is required.
COS_SECRET_KEY	No	`secret_key` for COS access. If `ENABLE_COS` is 1, this field is required.
COS_BUCKET	No	COS bucket name. If `ENABLE_COS` is 1, this field is required.
STATMENT_TIMEOUT	No	Query timeout period, which is 50 seconds by default.

Network configuration

VPC: Activate VPC and set the same VPC and subnet values as those of the CDWPG instance.



The corresponding values in CDWPG are as shown below:

vpc-2ta0n6h3 (subnet-pel1n8u4)	1
	2

Public Network Access: Enable

Step 2. Configure a trigger

In the **Functions** list in the SCF console, click the name of the newly created function to enter the function details page and click **Trigger management** > **Create trigger** on the left to create a trigger. Here, set **CKafka trigger** for **Trigger method**.

Custom						
Triggered Version	Version: \$LATEST	v				
Trigger Method	CKafka trigger	v				
	SCF can consume messages in CKafk	a. Learn More 🗹				
CKafka instance 🛈	Please select a CKafka instance	👻 🗘 Create Ckafka 🛂				
Торіс	Please select a CKafka topic	•				
Maximum messages()	- 100 +					
Consumption start point	 Latest Earliest Specific time 					
Retry Attempts 🛈	- 10000 +					
Max Waiting Time 🕄	- +					
Enable Now	Enable					

For details of trigger settings, see CKafka Trigger Description.

VOD SCF + VOD for Event Notification Receipt

Last updated : 2024-12-02 17:40:42

Overview

Demo features

This document uses the video upload and transcoding process as an example to describe how to use the event notification mechanism of VOD.

Architecture and process

An HTTP service is built based on SCF in the demo to receive event notification requests from VOD. It initiates video transcoding and gets the transcoding result by processing NewFileUpload (video upload completion) and ProcedureStateChanged (task flow status change) event notifications.

The system mainly involves four components: console, API Gateway, SCF, and VOD. Here, API Gateway and SCF are the deployment objects of this demo.

The specific business process is as follows:

1. Upload a video to VOD in the console.

2. The VOD backend initiates the NewFileUpload event notification request to the demo.

3. The demo parses the event notification content and calls the ProcessMedia API of VOD to initiate transcoding for the uploaded video with the 100010 and 100020 preset transcoding templates.

4. After completing the transcoding task, VOD initiates the ProcedureStateChanged event notification request to the demo.

5. The demo parses the event notification content and prints the URL of the transcoding output file into SCF logs. **Note:**

The SCF code in the demo is developed based on Python 3.6. SCF also supports other programming languages such as Python 2.7, Node.js, Go, PHP, and Java for your choice as needed. For more information, please see Development Guide.

Fees

The VOD event notification receipt service demo provided in this document is open-source and free of charge, but it may incur the following fees during service building and use:

Fees for purchasing a Tencent Cloud CVM instance to run the service deployment script. For more information, please see Instance Billing Modes.



Fees for using signature distribution service provided by SCF. For more information, please see Billing Mode and Free Tier.

Fees for using Tencent Cloud API Gateway to provide public network APIs for SCF. For more information, please see Billing Overview.

Fees for VOD storage of uploaded videos. For more information, please see Pay-as-You-Go (Postpaid Daily Billing Cycle).

Fees for VOD video transcoding. For more information, please see Pay-as-You-Go (Postpaid Daily Billing Cycle).

Avoiding affecting production environment

The business logic of the event notification receipt service demo uses the VOD event notification mechanism; therefore, you need to set the event notification address during deployment. If there is already a VOD-based production environment under your account, changing the event notification address may cause business exceptions. Before doing so, please confirm that this will not affect the production environment. If you are not sure, please use a new account to deploy the demo.

Quickly Deploying Event Notification Receipt Service

Step 1. Prepare a CVM instance

The deployment script needs to be executed on a CVM instance meeting the following requirements: Region: not limited.

Model: the minimum official configuration (1 CPU core and 1 GB memory) is sufficient.

Public network: a public IP is required, and the bandwidth should be at least 1 Mbps.

Operating system: official public image Ubuntu Server 16.04.1 LTS 64-bit or Ubuntu Server 18.04.1 LTS 64-bit .

For detailed directions on how to purchase a CVM instance and reinstall the system, please see Operation Guide -Creating Instances via CVM Purchase Page and Operation Guide - Reinstalling System, respectively.

Note:

The event notification receipt service demo itself does not depend on CVM but only uses CVM to run the deployment script.

If you do not have a CVM instance satisfying the above conditions, you can also run the script on another Linux (such as CentOS or Debian) or macOS server with public network access, but you need to modify certain commands in the deployment script based on the operating system. Please search for the specific modification method by yourself.

Step 2. Activate VOD

Please activate the VOD service as instructed in Getting Started - Step 1.

Step 3. Get the API key and APPID

Your API key (i.e., SecretId and SecretKey) and APPID are required for deploying and running the event notification receipt service demo.

If you have not created an API key yet, please generate one as instructed in Root Account Access Key. If you have already created a key, please get it as instructed in the same document.

You can view the APPID on the Account Information page in the console as shown below:

Account Center	Account Information
2 Account Information	Basic Information
🔁 Security Settings	Account Email
Project Management	Account Name
🖽 Identity	Account ID
Verification	APPID
G Message Subscription ☑	Registered On 2019-07-26 16:10:05

Step 4. Deploy the event notification receipt service

Log in to the CVM instance prepared in step 1 as instructed in Logging into Linux Instance in Standard Login Method and enter and run the following command on the remote terminal:

Note:

Please assign the corresponding values obtained in step 3 to SECRET_ID, SECRET_KEY, and APPID in the command.

This command will download the demo source code from GitHub and automatically run the installation script. The installation process will take several minutes (subject to the CVM network conditions), during which the remote device will print the following information:

```
[2020-06-05 17:16:08] Start installing pip3.
[2020-06-05 17:16:12] pip3 is successfully installed.
[2020-06-05 17:16:12] Start installing Tencent Cloud SCF.
```



```
[2020-06-05 17:16:13] SCF is successfully installed.
[2020-06-05 17:16:13] Start configuring SCF.
[2020-06-05 17:16:14] SCF configuration is completed.
[2020-06-05 17:16:14] Start deploying the event notification receipt service of
VOD.
[2020-06-05 17:16:24] The event notification receipt service of VOD is
deployed.
[2020-06-05 17:16:26] Service address: https://service-xxxxxxx-
125xxxxxx.gz.apigw.tencentcs.com/release/callback
```

Copy the address of the event notification receipt service in the output log (which is https://service-

xxxxxxx-125xxxxxx.gz.apigw.tencentcs.com/release/callback in this example).

Note:

If the following warning is displayed in the output log, it is generally because the CVM instance cannot immediately parse the service domain name deployed just now. You can ignore this warning.

[2020-04-25 17:18:44] Warning: the event notification receipt service failed the test.

Step 5. Configure the event notification address

As described in Avoiding affecting production environment, please confirm that your business in the production environment does not depend on VOD event notifications before performing the following operations: Log in to the VOD console, click **Settings**, select **Normal Callback** as the callback mode, enter the event notification receipt service address obtained in step 4 as the callback URL, check all callback events, and click **OK** as shown below:



Note:

If two callback URL configuration items (v2.0 format and v3.0 format) are displayed at the same time in the console, please configure the v3.0 one.

Step 6. Test the demo

Upload a test video to VOD as instructed in Uploading Video - Local Upload. Select the default option "No Processing After Upload" during the upload. After the upload is completed, you can see that the video status is "Processing" on the "Uploaded" tab, which indicates that the demo has received the NewFileUpload event notification and initiated a transcoding request.

FileId	File Name	Video Name	Video Size	Video Category	Process Video	Upload Start Time 🗘	Status T	Operation
387702294991205128	38b98423b54969167a	38b98423b54969167a	15.25MB			2022-01-23 15:36:09	Uploaded successfully	Delete Records
Total items: 1						10	🔻 / page 🔣 🖣 1	/1 page 🕨 🕨

Wait for the video processing to complete (indicated by the "Normal" status), click **Quick View**, and you can see that there are two output videos for the uploaded video on the right of the page as shown below:

Upload Video Batch Delete Process Vide	eo Quick Edit Mo	re 🔻		Video Name	38b98423b54969167a900241456a088e.m	np4 🎤	
Separate key words with a vertical line " " and separate filter	conditions by pressing Enter	Q		Category	Other 🎤		
Video Name/ID	Video Status	Video Category T	Uploa	Label	1		
38b98423b54969167a900241456a088	e.mp4	Other	2022.0	Size	15.25MB		
Quick View	Worman	ould	2022	Duration	00:02:30		
testmp4		011-11		Expiration Time 🛈	Permanent 🎤		
00:08:02	Vormal Normal	Other	2020-	File Description	i		
Total items: 2				Video URL			
				Copy All URLs	Batch Delete		
				Specification	Format	Resolution	Opera
				Original Video	MP4	448 x 960	Сору Ц

Log in to the SCF console and enter the log page to view the SCF logs. In the latest log, you can see that the URLs of the two output files have been printed out. In actual use cases, you can use SCF to record the URLs in your database or publish them to viewers through other channels.

{"issource_incoded": false, "statusCode": 200, "headers": {"Content- Type": "text/plain; charset=utf-8", "Access-Control-Allow-Origin": "*", "Access-Control-Allow- Methods": "POST,OPTIONS"}, "body": null}
Event RequestId: e
Transcode succeeded. Fileld: 5285890804224192945, Definition: 100010, Url: http://140006272
0.vod2.myqcloud.com/cf72b740vodtranscq1400062720/49b71b2c5285890804224192945/v.f1
00010.mp4.
Transcode succeeded. Fileld: 5285890804224192945, Definition: 100020, Url: http://140006272
0.vod2.myqcloud.com/cf72b740vodtranscq1400062720/49b71b2c5285890804224192945/v.f1
00020.mp4.

Note:

SCF log generation may have a certain delay. If no logs are displayed on the page, please wait for one or two minutes and click **Reset** to refresh.

System Design Description

API protocol

The event notification receipt function uses API Gateway to provide APIs. For the specific API protocols, please see Video Upload Completion and Task Flow Status Change.

Event notification receipt service code interpretation

1. main_handler() is the entry function.

2. Call parse_conf_file() and read the configuration information from the config.json file. The configuration items are as described below:

Field	Data Type	Description
secret_id	String	API key
secret_key	String	API key
region	String	TencentCloud API request region, which can be any region for VOD
definitions	Integer Array	Transcoding template
subappid	Integer	Whether the event notification is from a VOD subapplication

3. For NewFileUpload event notifications, call deal_new_file_event() to parse the request and get the FileId of the newly uploaded video from it.

```
if event_type == "NewFileUpload":
    fileid = deal_new_file_event(body)
    if fileid is None:
        return ERR_RETURN
```

4. Call trans_media() to initiate transcoding, output the TencentCloud API's response packets to SCF logs, and send the response packets to the event notification service of VOD.

```
rsp = trans_media(configuration, fileid)
if rsp is None:
    return ERR_RETURN
print(rsp)
```

5. In trans_media() , call the TencentCloud API SDK to initiate the ProcessMedia request:

```
cred = credential.Credential(conf["secret_id"], conf["secret_key"])
client = vod_client.VodClient(cred, conf["region"])
method = getattr(models, API_NAME + "Request")
req = method()
req.from_json_string(json.dumps(params))
```



```
method = getattr(client, API_NAME)
rsp = method(req)
return rsp
```

6. For ProcedureStateChanged event notifications, call deal_procedure_event() to parse the request to get the transcoding output video URL and print it to SCF logs:

```
elif event_type == "ProcedureStateChanged":
    rsp = deal_procedure_event(body)
    if rsp is None:
        return ERR_RETURN
```

SMS SCF + SMS for SMS Verification Code

Last updated : 2024-12-02 17:40:42

Sending verification codes through SMS is the most popular and securest way to verify user identities. Currently, SMS verification codes are widely used in various application scenarios such as user registration, password reset, login protection, identity verification, random password generation, and transaction confirmation. This document uses developing a verification code-enabled login and signup service based on SCF as an example to describe how to implement the SMS verification code feature. In addition to SCF, you can also use the SendSms API for this purpose.

Preparations

You have signed up for a Tencent Cloud account and verified your organizational identity.

You have purchased an SMS package.

Prepare SMS signature owner qualification certificates. For detailed file list and specifications, please see the signature review standards.

This document takes a business license as a qualification certificate for example. Understand the SMS body content review standards.

Get the SDKAppID of the SMS application.

Reference

Demo source code Other products' documentation VPC documentation TencentDB for Redis documentation SCF documentation

Step 1. Configure SMS content

After an SMS signature or body template is submitted, it will be reviewed within two hours generally. You can configure alarm contacts and set template/signature review notifications to receive review result notifications.

Step 1.1. Create a signature

- 1. Log in to the SMS console.
- 2. Select Chinese Mainland SMS > Signature Management on the left sidebar and click Create Signature.
- 3. Set the following parameters as needed and according to the signature review standards:

Parameter	Sample Value			
Signature purpose	For self-use (the signative verified under the curre	ture is a company name, we ent account)	bsite, product name	, or something else
Signature type	Арр			
Signature content	Test demo			
Certificate type	Screenshot of WeChat Mini Program settings page			
Certificate upload	● 館名公众平台 1 小程序 ● 館方 ● 館方 ● 館子 ● 館子 ● 結子 ● 続子 ● 読む ●	支まなど ション ション <td></td> <td></td>		

4. Click OK.

Wait for signature review. The SMS signature will be available only after its status changes to **approved**.

Step 1.2. Create a body template

- 1. Log in to the SMS console.
- 2. Select Chinese Mainland SMS > Body Templates on the left sidebar and click Create Body Template.
- 3. Set the following parameters as needed and according to the body template review standards:

Parameter	Sample Value
Template	Verification code SMS

Name	
SMS type	Regular SMS
SMS content	Your signup verification code is {1}. Please enter it within {2} minutes. If the signup was not initiated by you, please ignore this message.

4. Click OK.

Wait for body template review. The body template will be available only after its status changes to **approved**. Please note down the template ID.

Step 2. Set the SMS delivery rate limit (optional)

Note:

Individual users have no permission to modify the rate limit. To use this feature, change "Individual Identity" to "Organizational Identity". For detailed directions, see Identity Verification Change Guide.

To ensure business and channel security and minimize potential financial losses caused by malicious calls of SMS APIs, we recommend you set the SMS delivery rate limit.

This document uses the default SMS delivery rate limit policy as an example.

For SMS messages with the same content, a maximum of one such message can be sent to the same mobile number within 30 seconds.

A maximum of 10 messages can be sent to the same mobile number on a calender day.

Step 3. Configure the VPC and subnet

By default, SCF is deployed in the public network and can access public network only. If you need to access Tencent Cloud resources such as TencentDB instances, you need to build a VPC to ensure data and connection security.

1. Plan the network design as needed.

2. Create a VPC. For detailed directions, please see Creating VPC.

Note:

The CIDRs of the VPC and subnet cannot be modified after creation.

Parameter	Sample Value
Region	South China (Guangzhou)
Name	Demo VPC
IPv4 CIDR Block	10.0.0/16

Subnet name	Demo subnet
IPv4 CIDR Block	10.0.0/16
Availability Zone	Guangzhou Zone 3

Step 4. Configure a TencentDB for Redis instance

The region and subnet AZ of the TencentDB for Redis instance must be the same as those of the VPC configured in step 3.

1. Purchase a TencentDB for Redis instance. For detailed directions, please see Creating TencentDB for Redis Instance.

Parameter	Sample Value
Billing Mode	Pay-as-you-go
Region	Guangzhou
Database version	Redis 4.0
Architecture	Standard architecture
Network	Demo VPC and demo subnet
Instance name	Demo database
Quantity	1

Step 5. Create a function

SCF currently supports development in Python, Node.js, PHP, Java, and Go. This document uses Node.js as an example.

1. Create a function in the region of the VPC created in step 3. For detailed directions, please see Writing Function.

Parameter	Sample Value
Function name	Demo
Runtime environment	Node.js 8.9
Creation method	Template function: helloworld

2. Deploy the function and set **API Gateway Trigger** as the trigger. For detailed directions, please see Deploying Function.

Step 6. Enable public network access (optional)

Functions deployed in a VPC before April 29, 2020 are isolated from the public network by default. If you want them to have access to both private network and public network, you can do so by enabling public network access. Log in to the SCF console, select **Function Service**, click the name of the target function in the function list to enter the function configuration page. Click **Edit**, check **Public Network Access**, and click **Save** to save the configuration. Functions deployed on or after April 29, 2020 have public network access enabled by default, and no additional operations are required.

Step 7. Deploy the SMS demo

1. Go to the SCF console and select the SMS demo to deploy it.



emplate Ise demo temp pplication	late to create a function or	Create from scratch Start from a Hello World sample	Use TCR image Create a function based on a TCR image	
Fuzzy search	sms Separate multiple tags	with carriage returns	Q 共1个	
	SmsVerificationCode Category Function Description This example overification co	Learn More uses Tencent cloud SMS to send de, and verify the verification		
	Tag Nodejs8.9 Author 🔗 Tencent Cl Deploy 8,228次	Sms		
Nevt	Cancel			

2. Set the environment variables of the demo in Advanced Configuration.




Field	Description
REDIS_HOST	Redis database address.
REDIS_PASSWORD	Redis database password.
SMS_TEMPLATE_ID	Template ID. You must enter the ID of an approved template, which can be viewed in the SMS console.
SMS_SIGN	Content of the SMS signature, which should be encoded in UTF-8. You must enter an approved signature, which can be viewed in the SMS console. Note: this parameter is required for Chinese Mainland SMS.
SMS_SDKAPPID	SMS SdkAppid actually generated after an application is added in the SMS console, such as 1400006666.

3. Set the same VPC environment as the Redis database in **Advanced Configuration**.



Network Conf	iguration		
Public network	✓ Enable (j)		
Fixed outbound	Enable (i)		
VPC	✔ Enable (i)		
		·	👻 🗘 Create VPC 🗹
_			

4. Set the permissions of SCF execution role in Advanced Configuration.

Execution Role *	✓ Enable 🛈
	To ensure that the function template can access other Tencent Cloud services, please configure and use the SCF t
	○ Configure and use SCF template role (i)
	O Use the existing role
	SCF_QcsRole Create Execution Role

You need to associate the <code>QcloudSMSFullAccess</code> policy with the <code>SCF_QcsRole</code> role in the CAM console.

ect Policies (1 Total)			1 selected		
upport search by policy name/description/remarks	Q		Policy Name	Policy type	
Policy Name	Policy type 🔻		OcloudSMSFullAccess		
QcloudSMSFullAccess	Dracet Dolicy		Full read-write access to SMS	Preset Policy	(
Full read-write access to SMS	reserroncy				
		÷			

In this way, the ``TENCENTCLOUD_SECRETID , TENCENTCLOUD_SECRETKEY , and

TENCENTCLOUD_SESSIONTOKEN` environment variables can be obtained in the code, which will be used by the SMS SDK.

5. Click **Complete** to deploy the function.

6. Create an SCF **API Gateway trigger** and request the trigger address to use SMS capabilities.

Trigger Method	API Gateway Trigger	- ¢
	For API gateway triggers, the format of or response method. For details, please see	contents returned from SCF should be constructed in integration the constructed in integration the constructed in the construct
API Service Type	O Create API Service Use Existing	API Service
API Service	SCF_API_SERVICE	
Request method 🕄	ANY	▼
Publishing Environment(i)	Publish	▼
Authentication Method 🛈	No authentication	v

Step 8. Use the features

Verification codes have a high requirement for timeliness. You can store verification codes in the memory or TencentDB for Redis and use the mobile number as a key to store information such as sending time, verification code, number of verification attempts, and verification result.

Features

Sending SMS verification code

Request parameters:

Field	Туре	Description
method	string	Request method, whose value is getSms
phone	string	Mobile number in the format of area code + mobile number, such as 86185662466**

Verifying verification code (login)

Request parameters:

Field	Туре	Description
method	string	Request method, whose value is login
phone	string	Mobile number in the format of area code + mobile number, such as 86185662466**



code string 6-digit verification code

Error codes

Field	Description
InValidParam	Missing parameter
MissingCode	Missing verification code parameter
CodeHasExpired	The verification code has expired
CodeHasValid	The verification code is invalid
CodelsError	Please check whether the mobile number and verification code are correct

If you have any questions, contact SMS Helper for assistance.

ES SCF + ES for Quick Search Service

Last updated : 2024-12-02 17:40:41

Search Service

Search services are everywhere, such as Google search in daily life, wiki search in work, and product search in shopping. Data in such scenarios is generally large in size and structured and has more reads than writes. However, due to their transaction characteristics, traditional databases cannot well utilize space in search scenarios and are slow in full-text searches (such as LIKE statement). Elasticsearch thus came into being. Elasticsearch is an open-source search engine widely used in full-text search. It can quickly index, search, and analyze a massive amount of text data. Tencent Cloud ES is a highly available and scalable cloud-hosted Elasticsearch service. It well supports both structured and non-structured data and provides easy-to-use RESTful

APIs and clients in multiple languages to help you quickly build stable search services.

This document describes how to use ES and SCF to quickly build a search service by referring to the official Tencent Cloud ES documentation.

Resource Preparations

Purchase an ES cluster. The cluster scale is subject to the search service QPS and data size of the stored documents. For more information, see Evaluation of Cluster Specification and Capacity Configuration.

Deploying Search Service

Use SCF to deploy the frontend page and backend service of the search service.

1. In the top-left corner of the Functions page in the SCF console, select the region of the purchased ES cluster.

Serverless Cloud Function	Functions	🕓 Guang	zhou(29) 🔻 I	lamesp	ace:
Overview	Create	Delete	0 functions se	elected.	Up to
Ø Function Service				-	
🛇 Layer	Function	Name *	Status	,	torin g

2. Click **Create** to create a function service as instructed in Create a Function.

3. After the function is created, click **Function name** to enter the function details page.

4. On the **Function configuration** page, click **Edit** in the top-right corner, enable **Private network access**, select the VPC selected during ES cluster creation, and click **Save**.

Network Config	guration		
Public network	✓ Enable (i)		
Fixed outbound	Enable (i)		
VPC	✓ Enable ①	Y	▼ 🗘 Create VPC 🛂

5. Click Code ZIP package to download the sample file.

6. In **Submitting method** on the **Function codes** page, select **Upload local ZIP package**, select the ZIP package just downloaded, and click **Deploy**.

Function configuration Function Codes Layer N Submitting Method() • Online editing • Execution()	anagement Monitoring Information Log Query	Development Tutorial of Nodejs8.9 [2	Download
Cloud Studio Online editing Go Termin EXPLORER Local ZIP File > OPEN EDIT Local folder > OPEN EDIT > OPEN EDIT > OPEN EDIT > OPEN EDIT > OPEN EDIT	<pre>il Help // index.js x src > J5 index.js > 1 const fs = require('fs') 2 const path = require('path') 3 4 exports.main_handler = async (event, context, callback) => { 5 let html = fs.readFileSync(path.resolve(dirname, './demo.html') 6 lencoding: 'utf-8' 7 }) 8 return { 9 isBase64Encoded: false, 10 statusCode: 200, 11 headers: { 'Content-Type': 'text/html; charset=utf-8' }, 12 body: html 13 } </pre>	Test Template:Hello World event template	

7. Modify the code on the Function Code page. You need to modify the index.py and index.html files: Change es_endpoint in index.py to the private network address of your ES cluster, such as

http://10.0.3.14:9200 .

Change es_password in index.py to the password of ES Platinum Edition; if your ES is not Platinum Edition, leave it unchanged.

Change server_name in index.html to the name of the created SCF function, which is myserver by default.

Note:

es_corpus_0126 is used as the index name by default in the sample. Make sure that the index is not used by other businesses. To modify it, modify the es_index variable in index.py .

8. Click **Add trigger method** on the **Trigger method** page, add an API Gateway trigger as shown below, enable integration response, and click **Save**.

Trigger Method	API Gateway Trigger 🔹	ϕ
	For API gateway triggers, the format of conte response method. For details, please see here	nts returned from SCF should be constructed in integration 🖸 .
API Service Type	• Create API Service Use Existing API S	ervice
API Service	SCF_API_SERVICE	
Request method	ANY -	
Publishing Environment(i)	Publish 💌	
Authentication Method (i)	No authentication 💌	
Integration Response	✓ Enable	
-		

9. View the function **access path** in **Trigger method** and access the page by clicking the path.

API Gateway Trigger 🛛	Alias: Default Traffic
API Name	SCF_API_SERVICE 🖬
serviceld	service-a47z4tim
apild	api-cjo3653i
Request method	ANY
Publishing Environment	Publish
Authentication Method	No authentication
Enable integration response	On
Enable Base64 encoding	Disabled
Support CORS	No
Backend timed out	15s
Access path	https

Upload the sample data of Elasticsearch Service. Click the text above the search box to automatically import data.
 At this point, you have deployed a simple ES-based Q&A search service backend.

Learn More

Importing stopword and custom dictionary

Stopwords will not be searched by ES, and words in the custom dictionary will be retained during segmentation. In the previous sample, the default stopword dictionary and custom dictionary are imported. You can also import your own stopword and custom dictionaries in **Plugin List** > **Update Dictionary** on the ES cluster details page.

Synonym configuration

To configure synonyms, you need to specify them when creating an index. Synonyms in Solr and WordNet formats are supported. For more information on the format, see Solr synonyms.

Scheduled Task Scheduled Task Overview

Last updated : 2024-12-02 17:40:41

With SCF scheduled triggers, you can quickly create scheduled tasks without having to purchase computing resources in advance. Such triggers leverage Serverless' powerful elastic scalability to provide stable and fast capabilities of scheduled task processing.

Unlike other event-based triggers, scheduled triggers can execute tasks by using scheduled time-driven functions. Specifically, you can configure Cron expressions to quickly use such triggers without relying on external invocations. They have natural advantages in typical scheduled task scenarios such as automated monitoring, testing, and alarming, automated task execution, and data archiving, cleansing, and backup. The relevant workflow is as shown below:



Advantages of Function Processing

The full lifecycle of scheduled tasks can be managed to help you quickly build scheduled triggers. Tasks are fully managed and can be triggered regularly and retried automatically by standard Cron expressions. More and more built-in function templates are added to reduce the development costs for scheduled tasks in popular use cases

Computing power is provided based on SCF, with features such as auto scaling, OPS-free usage, and pay-as-you-go billing.

Multi-scenario Function Processing Practices

The following table lists existing templates for typical use cases and their specific descriptions:

Function Processing Scenario	Description
High-Availability scheduled testing	Use SCF to implement high-availability scheduled testing.
Scheduled task execution	Use SCF and Puppeteer to perform scheduled tasks on webpage content such as data collection and storage.
Scheduled data archiving and backup	Use SCF to regularly back up databases to COS.

Note:

SCF offers a free tier. You will be charged if you exceed the free tier limit. For billing details, please see SCF Billing Overview.

Performing Scheduled Task

Last updated : 2024-12-02 17:40:41

Overview

This document uses an SCF function and Puppeteer to perform scheduled tasks on webpage content such as data collection and storage. You can also perform complicated scheduled web tasks like data crawling, scheduled sign-in, and webpage inspection.

Directions

Creating function

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select the **Beijing** region and click **Create** to enter the function creating page and configure the function as shown below:

	Template Create a func template	tion using the demo	Custom Create a custom function using HelloWolrd demo	
uzzy search	task Separ	ate multiple tags with carria	ge returns	
	TimerTask	Le	arn More	
	Category	Function		
	Description	This is timer task functior	h.	
	Tag	Nodejs8.9 定时任务		
	Author	🔗 Tencent Cloud		
	Deploy	690 time		
L				

Creation method: select Template.

Fuzzy search: enter TimerTask and search.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded.

Click Next, and the function name will be automatically generated by default. If you need to modify the function code, click to expand the Function Code block and make changes as instructed in Modifying function template.
 In the Trigger Configurations section, select Automatic creation, and a scheduled trigger that will run on the hour will be created by default as shown below:

Create a Trigger	O Automatic creation	
	Triggered Version	Default Traffic 🔹
	Trigger Method	Scheduled triggering
		The scheduled trigger will trigger the SCF function automatically by the specified period. Learn More 🗹
	Scheduled task name(i)	timer
	Trigger Period	Every hour (Execute once at the 0th mi 💌
	Remarks	No
	Enable Now	Enable
		If it's checked, the scheduled trigger will be activated and executed at the start point of next period.
	Custom	

Note:

If you need to adjust the trigger configuration according to your needs, please select **Custom**.

To create a scheduled trigger after the test is successful, please select **Create Later**.

5. Click Complete.

Testing function

1. At the bottom of the function code page, click Test to view the execution log of the function.

2. After the test is passed, you can configure the scheduled trigger in the **Trigger Method** tab according to the actual situation and check the related Base64 value.

Relevant Operations

Modifying function template

The current template function references Puppeteer to screencapture the webpage content and convert it to a Base64 value for printout in the function log. You can modify the template according to your own scheduled task needs. For example, run the following command to get the page title:

```
// Sample page title
const title = await page.title();
console.log(title);
```

Add the following code to set the page click attribute.

```
// Sample page click attribute
await page.click('a');
```

For more information on how to use Puppeteer, please see here. With this tool, you can access page content at scheduled times and perform task operations on the page, such as data crawling and sign-in.

Video Processing Implementing Batch Automated Video Editing with SCF and FFmpeg

Last updated : 2024-12-02 17:40:42

Overview

Common video editing tools such as Premiere and AE can implement complex editing effects and are widely used in promotional and advertising video production. However, in the following scenarios, traditional editing tools and template-based video processing software programs cannot meet the requirements for **batch**, **automated**, **and custom** video editing:

A school expects to output the highlight videos of all students during online class with the school logo and slogan added immediately after the students finish the classes and allow them to quickly share such videos. Suppose there are 10,000 students and one unique video should be produced for each student, then the school needs to complete the batch editing of 10,000 different videos automatically.

Profile photo-based videos need to be generated for users to attract them to a marketing campaign. As each user has a different profile photo, the generated videos are also different. There may be tens of thousands of users; therefore, automatic editing is desired.

An MCN company wants to generate unified promotional videos for 100 hosts. It may be acceptable to designate a person to edit 100 videos. However, it will be very troublesome if 100 videos need to be edited every week. Therefore, automated, batch, and custom editing is desired.

In these video editing scenarios, resource usage is concentrated within certain hours, and the volume of computed data is high. In this case, dedicated high-end servers may have a low utilization, while low-end servers' computing power cannot meet the requirements. By contrast, SCF perfectly satisfies such requirements with its pay-as-you-go billing mode and high-performance computing power. It not only achieves a 100% utilization but also provides high computing power on demand. In addition, it delivers a higher flexibility by supporting multiple programming environments for developers with different development habits.

Prerequisites

All video editing features mentioned in this document are implemented by FFmpeg.

How to Use FFmpeg

FFmpeg is an open-source video processing tool. It supports video editing, transcoding, splicing, audio processing, text adding, and live stream push/pull. You can use different FFmpeg commands to program and implement different video editing features. You can also combine and orchestrate them to execute various batch automated tasks. Below are common video editing scenarios:

- 1. Video transcoding
- 2. Video clipping
- Text adding to video
- 4. Image adding to video
- 5. Video splicing
- 6. Audio adding to video
- 7. Video transition
- 8. Video special effect
- 9. Video playback speed adjusting

FFmpeg commands

Some FFmpeg commands are as described below. You can install FFmpeg locally first and then perform tests.

ffmpeg -i input.mov output.mp4
// Change the frame rate of the original video to 24 FPS
ffmpeg -i input.mp4 -r 24 -an output.mp4

// Convert an .mov video to an .mp4 video

// Convert an .mp4 video to a video stream suitable for live streaming
ffmpeg -i input.mp4 -codec: copy -bsf:v h264_mp4toannexb -start_number 0 -hls_time

// Change the video resolution to 480x360 and change the bitrate to 400 Kbps
ffmpeg -i input.mp4 -vf scale=480:360,pad=480:360:240:240:black -c:v libx264 -x264-

// Add text such as subtitles and title to the video // `fontfile` is the path of the font to be used, and `text` is the text to be adde // `fontcolor` is the font color, `fontsize` is the font size, and `box` is the tex // `box=1` indicates to enable the text box, while `0` indicates to disable it. `bo // `x` and `y` indicate the text position. They can be numbers or expressions. For ffmpeg -i input.mp4 -vf "drawtext=fontfile=/path/to/font.ttf:text='your text':fontc

// Add an image such as logo, profile photo, and sticker to the video. `filter_comp
ffmpeg -i input.mp4 -i avatar.JPG -filter_complex "[0:v][1:v] overlay=25:25:enable=

// Splices videos. `list.txt` lists the paths of all video files to be spliced in s



```
// Note: If the videos have different resolutions, splicing will fail.
ffmpeg -f concat -safe 0 -i list.txt -c copy -movflags +faststart output.mp4
// `list.txt` is in the following format:
file 'xx.mp4'
file 'yy.mp4'
// Add audio to the video. `stream_loop` indicates whether to loop the audio (-1: i
ffmpeg -y -i input.mp4 -stream_loop -1 -i audio.mp3 -map 0:v -map 1:a -c:v copy -sh
```

Note:

FFmpeg also supports editing audio separately. For detailed directions, visit the FFmpeg website.

Running FFmpeg commands

This document uses Python as an example. You can refer to the following sample code to run FFmpeg commands:

Using FFmpeg in SCF

1. Log in to the SCF console and select Function Service on the left sidebar.

2. At the top of the **Function Service** page, select a region and click **Create** to enter the function creation page.

3. Set the following parameter information and click Next as shown below:

Creation method: Select Template.

Fuzzy search: Enter video editing and search.

Click Learn More in the template to view relevant information in the **Template Details** pop-up window, which can be downloaded. You can click the GitHub address to get the template source code, where you can view the API call parameters and use methods.

Jse demo templa opplication	ite to create a fu	nction or Start from a Hello Wo	id sample	Create a function based on a TCR imag	e	
Fuzzy search		Separate multiple tags with carriage returns		Q 共2个		Sort by recomm v
	VideoCom	position Lea	m more VideoSplice	: L	earn more	
	Category	Function	Category	Function		
	Description	This demo uses API GW trigger and SCF to build video composition service.	Description	This demo uses API GW trigger and SCF build video splice service.	to	
	Tag	Python3.7 APIGW Video FFmpeg VOD	Tag	Python3.7 APIGW Video FFmpeg VOD		
	CA	🔗 Tencent Cloud	CA	🔗 Tencent Cloud		
	Deploy	8,026次	Deploy	8,632次		
L						

4. In **Basic Configurations**, **Function Name** is generated by default, which can be modified as needed. You can configure async execution and execution role as follows:

Async Execution: Select **Enable**.

Execution Role: Select **Enable**, select **Configure and use SCF template role**, and the system will automatically create and select an SCF template execution role associated with full access permissions of VOD. You can also check **Use the existing role** and select an existing role that has the above permissions in the drop-down list. This document takes **Configure and use SCF template role** as an example, as shown below:

← Create		
	Basic Configura	ations v
	Function name *	VideoComposition-1649838678 It supports 2 to 60 characters, including letters, numbers, underscores and hyphens. It must start with a letter and end with a number or letter.
	Region *	S Beijing v
	Description *	This demo uses API GW trigger and SCF to build video composition senice.
		Up to 1000 letters, digits, spaces, commas, and periods.
	Async execution *	Enable O When async execution is enabled, the function execution timeout period can be 24 hours at most. You can modify it as needed. Resse specify the log publishing topic in log configurations.
	Status trace *	Enable ()
	Execution Role *	Z Enable ①
		To ensure that the function template can access other Tencent Cloud services, please configure and use the SCF template role, or select an existing role that includes Qcloud/ODFullAccess preset policies.
	Eunction Code	 Bandan Dahard T. Suranjan anthrop ladar mala kandlar.

Note:

This example requires SCF to have the operation permissions of VOD. **Execution Role** has been enabled by default, and an execution role has been automatically created and associated with the required VOD permission policy

QcloudVODFullAccess . If you need to make adjustments, select **Use the existing role** or disable **Execution Role**.

5. In Trigger Configurations, select Automatic creation as shown below:

Note:

To use an existing API service to create an API Gateway trigger or modify the trigger configuration, select **Custom**.

Triggered version	Default traffic 🔹
Trigger method	API Gateway trigger 🔹 🗘
	For API gateway triggers, the format of contents returned from SCF should be constructed in integration response method. For details, please seehere 🙆.
API service type	Create API service Use an existing API service
API Service	SCF_APL_SERVICE
Request method (ANY
Publishing environment 🛈	Publish 👻
Authentication method	No authentication
Integration response (j)	Enable
Base64 Encoding	Enable
Tag 🚯	Defaults to the tag that used for function creation
Custom	
Create later	
	API service type() API Service Request method() Publishing environment() Authentication method() Integration response() Base64 Encoding() Tag() Custom Create later

6. Click **Complete** to complete function and trigger creation and get the HTTP triggering domain of the function.

Success Story

The customer is an online education company. It needs to create 30-second videos as students' study results from online class recordings every time students finish classes. This case has the following key information:

- 1. Generally, a class has 200 students, so 200 videos need to be produced at the same time.
- 2. A one-hour class recording needs to be clipped into a 30-second video.
- 3. As the screen of each student is different, the recorded videos are also different.
- 4. Student names and profile photos need to be added to the final output videos.
- 5. The time points when students finish a class are concentrated; therefore, there may be a short time period of high concurrency during video production.

6. Videos need to be produced only after the class is over; therefore, video production start times are fixed and concentrated.

Based on the above characteristics, SCF has the following strengths for video editing:

- 1. SCF can sustain 200 concurrent video editing tasks, so the customer doesn't need to set up additional servers.
- 2. SCF incurs no fees during idle hours.
- 3. SCF has a high computing power to quickly produce videos.

The reference architecture for this case is as shown below:





Summary

You can orchestrate, combine, and reuse the above audio/video editing capabilities to meet the requirements for various scenarios. You can also convert the parameters used to control various effects in video editing to the parameters passed in during service call in order to implement various custom effects.