# Cloud Log Service

# Product Introduction

# Product Documentation

# Contents

# Product Introduction

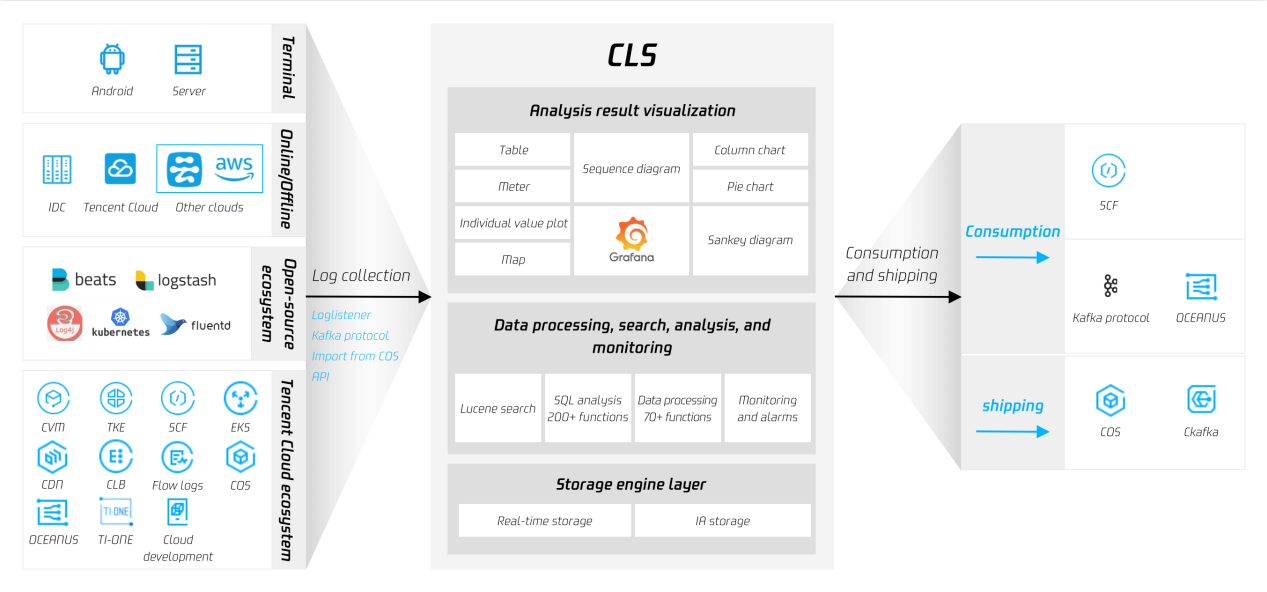
# Overview

Last updated：2024-01-20 16:29:01

Cloud Log Service (CLS) provides a one-stop log data solution. You can quickly and conveniently connect to it in five minutes to enjoy a full range of stable and reliable services from log collection, storage, and processing to search, analysis, consumption, shipping, dashboard generation, and alarming, with no need to care about resource issues such as scaling. It helps you improve the problem locating and metric monitoring efficiency in an all-around manner, making log Ops much easier.

## Overview

CLS has the following features.
Log collection: CLS easily collects logs from different regions, channels, platforms, and data sources (e.g., various Tencent Cloud products) in real time.
Log storage: CLS offers two storage types: real-time storage and STANDARD_IA storage.
Log search and analysis: You can search for logs by keyword to quickly locate exception logs and use SQL statements to collect and analyze log statistics. This helps you get statistical metrics such as log quantity change trend over time and proportion of error logs.
Log data processing: CLS can filter, cleanse, mask, enrich, distribute, and structure logs.
Log shipping and consumption: CLS can ship logs to Tencent Cloud storage and middleware services and consume logs to stream computing services.
Dashboard: CLS can quickly generate custom dashboards for search and analysis results.
Alarming: CLS can trigger alarms for exception logs within seconds and notify you through phone, SMS, email, and custom API callback.




## Log collection

CLS currently supports multiple methods for data collection, such as LogListener, API, Kafka, and COS import.
LogListener real-time collection: Use LogListener to collect logs. This method is easy to install, reliable, and secure. It supports most mainstream Linux operating systems, delivering high performance while occupying few resources.
Collection via API: Call the API to upload logs, without the need to install LogListener. Multiple programming languages are supported.
Collection over Kafka: Upload logs to CLS by using the Kafka Producer SDK.
COS import: Import COS data to CLS.

## Log storage

According to users' different requirements for log search latency and log processing capabilities, CLS provides two storage classes:
Real-time storage: It is suitable for users who require statistical analysis and provides log search within seconds, real-time statistical analysis, real-time monitoring, streaming consumption, and other application capabilities.
STANDARD_IA storage: It is suitable for infrequently accessed logs that do not require statistical analysis, such as archived audit logs. It provides the full-text log search capability, meeting users' requirements for backtracking and archiving historical logs. The overall usage costs of STANDARD_IA storage are 80% lower than those of real-time storage in the long run.

## Log search and analysis

CLS provides real-time log search and analysis to help you quickly locate exception logs and collect system and business metric statistics.
Log search: Use keywords to search for logs by full text or field.

Statistical analysis: Use SQL statements to flexibly collect the system and business metric statistics in logs and display them in charts. This feature is compatible with the SQL-92 standard and supports over 200 SQL functions. It can collect the number of business requests by province and query the request error rate change trend by time.
Superior performance: Query results can be returned within seconds, and the search and analysis of hundreds of millions of logs are supported.

### Log data processing

Real-time processing and distribution of log streams: Log streams are processed, and the processing result is generated in real time and can be distributed to multiple topics in different scenarios.
Log filtering and cleansing: Dirty data is cleansed, logs are filtered by condition, and index is enabled, effectively reducing costs.
Log structuring: Text logs are processed into structured data for future search, analysis, dashboard generation, and alarming.
Log masking: Sensitive information in the log text is masked, such as mobile number and ID number.

### Log shipping and consumption

You can ship specified logs to other Tencent Cloud services, such as COS and CKafka. You can also consume CLS logs to Oceanus.
Log shipping to COS: CLS allows you to ship logs to COS buckets under your account. COS storage is economical and recommended.
Log shipping to CKafka: It is suitable for scenarios where CKafka is used as the source for data analysis and storage.
Real-time log consumption: Logs are consumed directly to big data components such as Flink, Oceanus, and Flume.

### Dashboard

A dashboard is a global view of data analysis, where you can view multiple statistical charts of query and analysis results.
Dashboards: Dashboards save multiple statistical charts of query and analysis results to form a comprehensive scenario-specific view.
Preset dashboards: CLS provides preset dashboards for Tencent Cloud services such as CLB, TKE, and COS to make common monitoring capabilities out-of-the-box.
Template variables: Template variables can be data source variables and quickly filtered to switch between statistical analysis objects and dimensions in the charts of dashboards. They can be added to support more complex business scenarios.

### Alarming

Alarms will be triggered within seconds in case of too many error logs or system and business metrics exceeding the threshold, to identify system and business exceptions.

Notification channels: Notifications can be sent via phone, SMS, email, Weixin, WeCom, and custom API callback (which can be connected to DingTalk and Lark).
Multi-dimensional analysis: When an alarm is triggered, raw logs can be further searched for and analyzed, and the result can be added to the alarm notification to facilitate root cause discovery.

# Features

Last updated：2024-01-20 16:29:43

## Rich Features

LogListener provides one-stop log service, including log collection, storage, search, transfer, and shipping features.
LogListener parses log structures in many ways, including full text in a single line, full text in multi lines, separator, JSON, and regular expressions.
LogListener allows you to select from multiple data access methods based on your business needs. For more information, see Collection Overview.
LogListener provides multiple search syntaxes and allows you to search logs by keyword, fuzzy match, or range.

## Stable and Reliable

CLS has a highly scalable distributed storage architecture. It supports horizontal scaling and automatic service scaling, and easily stores and manages massive log data.
The CLS backend storage manages and stores logs in multiple copies, ensuring data reliability.

## Simple and Efficient

LogListener supports simple, GUI-based configuration. You can quickly access CLS through LogListener.
Data can be consumed immediately after being written to CLS. Query results of hundreds of millions of data records can be returned in seconds.
The service is billed by actual usage. You do not need to build or maintain a log system, preventing resource idling and waste.

## Ecological Expansion

The logs of certain cloud services have already been integrated into CLS. For more information, see Accessing log sources.
Logs are shipped to Cloud Object Storage (COS), which supports long-term archive storage of logs.
Logs are shipped to CKafka, which supports real-time log consumption and facilitates further log processing and analysis.

# Available Regions

Last updated：2024-01-20 16:30:17

## Overview

You can create logsets and log topics in different regions when using CLS. Regions are independent geographical areas where IDCs are located. Tencent Cloud regions are completely isolated. You can select the nearest region based on different business scenarios and the location of your targeted users to reduce log access latency and improve user experience.

## Available regions

| Region | Code |
| --- | --- |
| Beijing | ap-beijing |
| Guangzhou | ap-guangzhou |
| Shanghai | ap-shanghai |
| Chengdu | ap-chengdu |
| Nanjing | ap-nanjing |
| Chongqing | ap-chongqing |
| Hong Kong (China) | ap-hongkong |
| Silicon Valley | na-siliconvalley |
| Virginia | na-ashburn |
| Singapore | ap-singapore |
| Bangkok | ap-bangkok |
| Mumbai | ap-mumbai |
| Frankfurt | eu-frankfurt |
| Tokyo | ap-tokyo |
| | |

| | |
|---|---|
| Seoul | ap-seoul |
| Moscow | eu-moscow |
| Jakarta | ap-jakarta |
| Toronto | na-toronto |
| São Paulo | sa-saopaulo |

**Note:**
If CLS is integrated into other cloud products, you need to select a logset in the same region as the other cloud products. Cloud products in the same region access each other over a private network, which effectively reduces latency and improves access speed.

# Domain name

CLS has different domain names for each of its modules, as described below:
LogListener
CLS API 3.0
API for log upload
Uploading Logs via Kafka
Kafka Consumption Logs
LogListener is a log collection client provided by CLS and can report local logs to CLS. It uses the following domain names:

| Region | Code | Private Network Domain Name | Public Network Domain Name |
|---|---|---|---|
| Beijing | ap-beijing | ap-beijing.cls.tencentyun.com | ap-beijing.cls.tencentcs.com |
| Guangzhou | ap-guangzhou | ap-guangzhou.cls.tencentyun.com | ap-guangzhou.cls.tencentcs.com |
| Shanghai | ap-shanghai | ap-shanghai.cls.tencentyun.com | ap-shanghai.cls.tencentcs.com |
| Chengdu | ap-chengdu | ap-chengdu.cls.tencentyun.com | ap-chengdu.cls.tencentcs.com |
| Nanjing | ap-nanjing | ap-nanjing.cls.tencentyun.com | ap-nanjing.cls.tencentcs.com |
| Chongqing | ap-chongqing | ap-chongqing.cls.tencentyun.com | ap-chongqing.cls.tencentcs.com |
| Hong Kong | ap- | ap- | ap-hongkong.cls.tencentcs.com |

| | | | |
|---|---|---|---|
| (China) | hongkong | hongkong.cls.tencentyun.com | |
| Silicon Valley | na-siliconvalley | na-siliconvalley.cls.tencentyun.com | na-siliconvalley.cls.tencentcs.com |
| Virginia | na-ashburn | na-ashburn.cls.tencentyun.com | na-ashburn.cls.tencentcs.com |
| Singapore | ap-singapore | ap-singapore.cls.tencentyun.com | ap-singapore.cls.tencentcs.com |
| Bangkok | ap-bangkok | ap-bangkok.cls.tencentyun.com | ap-bangkok.cls.tencentcs.com |
| Mumbai | ap-mumbai | ap-mumbai.cls.tencentyun.com | ap-mumbai.cls.tencentcs.com |
| Frankfurt | eu-frankfurt | eu-frankfurt.cls.tencentyun.com | eu-frankfurt.cls.tencentcs.com |
| Tokyo | ap-tokyo | ap-tokyo.cls.tencentyun.com | ap-tokyo.cls.tencentcs.com |
| Seoul | ap-seoul | ap-seoul.cls.tencentyun.com | ap-seoul.cls.tencentcs.com |
| Moscow | eu-moscow | eu-moscow.cls.tencentyun.com | eu-moscow.cls.tencentcs.com |
| Jakarta | ap-jakarta | ap-jakarta.cls.tencentyun.com | ap-jakarta.cls.tencentcs.com |
| Toronto | na-toronto | na-toronto.cls.tencentyun.com | na-toronto.cls.tencentcs.com |
| São Paulo | sa-saopaulo | sa-saopaulo.cls.tencentyun.com | sa-saopaulo.cls.tencentcs.com |

CLS API 3.0 is the latest version of CLS APIs that complies with the unified API specifications of Tencent Cloud. You can use the APIs to manage resources such as log topics and alarm policies. The APIs use the following domain names:

If you access CLS over the public network, you can also use the unified domain name `cls.tencentcloudapi.com`. Your API request will be automatically resolved to a server nearest to the client. For latency-sensitive businesses, we recommend you specify the domain name in a region.

| Region | Code | Private Network Domain Name | Public Network Domain Name |
|---|---|---|---|
| Beijing | ap-beijing | cls.internal.tencentcloudapi.com | cls.ap-beijing.tencentcloudapi.com |
| Guangzhou | ap-guangzhou | cls.internal.tencentcloudapi.com | cls.ap-guangzhou.tencentcloudapi.com |
| Shanghai | ap-shanghai | cls.internal.tencentcloudapi.com | cls.ap-shanghai.tencentcloudapi.com |

| | | | |
|---|---|---|---|
| Chengdu | ap-chengdu | cls.internal.tencentcloudapi.com | cls.ap-chengdu.tencentcloudapi.com |
| Nanjing | ap-nanjing | cls.internal.tencentcloudapi.com | cls.ap-nanjing.tencentcloudapi.com |
| Chongqing | ap-chongqing | cls.internal.tencentcloudapi.com | cls.ap-chongqing.tencentcloudapi.com |
| Hong Kong (China) | ap-hongkong | cls.internal.tencentcloudapi.com | cls.ap-hongkong.tencentcloudapi.com |
| Silicon Valley | na-siliconvalley | cls.internal.tencentcloudapi.com | cls.na-siliconvalley.tencentcloudapi.com |
| Virginia | na-ashburn | cls.internal.tencentcloudapi.com | cls.na-ashburn.tencentcloudapi.com |
| Singapore | ap-singapore | cls.internal.tencentcloudapi.com | cls.ap-singapore.tencentcloudapi.com |
| Bangkok | ap-bangkok | cls.internal.tencentcloudapi.com | cls.ap-bangkok.tencentcloudapi.com |
| Mumbai | ap-mumbai | cls.internal.tencentcloudapi.com | cls.ap-mumbai.tencentcloudapi.com |
| Frankfurt | eu-frankfurt | cls.internal.tencentcloudapi.com | cls.eu-frankfurt.tencentcloudapi.com |
| Tokyo | ap-tokyo | cls.internal.tencentcloudapi.com | cls.ap-tokyo.tencentcloudapi.com |
| Seoul | ap-seoul | cls.internal.tencentcloudapi.com | cls.ap-seoul.tencentcloudapi.com |
| Moscow | eu-moscow | cls.internal.tencentcloudapi.com | cls.eu-moscow.tencentcloudapi.com |
| Jakarta | ap-jakarta | cls.internal.tencentcloudapi.com | cls.ap-jakarta.tencentcloudapi.com |
| Toronto | na-toronto | cls.internal.tencentcloudapi.com | cls.na-toronto.tencentcloudapi.com |
| São Paulo | sa-saopaulo | cls.internal.tencentcloudapi.com | cls.sa-saopaulo.tencentcloudapi.com |

The following domain names apply to APIs for log upload. Update all other APIs to API 3.0.

| Region | Code | Private Network Domain Name | Public Network Domain Name |
|---|---|---|---|
| Beijing | ap-beijing | ap-beijing.cls.tencentyun.com | ap-beijing.cls.tencentcs.com |
| Guangzhou | ap-guangzhou | ap-guangzhou.cls.tencentyun.com | ap-guangzhou.cls.tencentcs.com |
| Shanghai | ap-shanghai | ap-shanghai.cls.tencentyun.com | ap-shanghai.cls.tencentcs.com |
| Chengdu | ap-chengdu | ap-chengdu.cls.tencentyun.com | ap-chengdu.cls.tencentcs.com |
| Nanjing | ap-nanjing | ap-nanjing.cls.tencentyun.com | ap-nanjing.cls.tencentcs.com |
| Chongqing | ap-chongqing | ap-chongqing.cls.tencentyun.com | ap-chongqing.cls.tencentcs.com |
| Hong Kong (China) | ap-hongkong | ap-hongkong.cls.tencentyun.com | ap-hongkong.cls.tencentcs.com |
| Silicon Valley | na-siliconvalley | na-siliconvalley.cls.tencentyun.com | na-siliconvalley.cls.tencentcs.com |
| Virginia | na-ashburn | na-ashburn.cls.tencentyun.com | na-ashburn.cls.tencentcs.com |
| Singapore | ap-singapore | ap-singapore.cls.tencentyun.com | ap-singapore.cls.tencentcs.com |
| Bangkok | ap-bangkok | ap-bangkok.cls.tencentyun.com | ap-bangkok.cls.tencentcs.com |
| Mumbai | ap-mumbai | ap-mumbai.cls.tencentyun.com | ap-mumbai.cls.tencentcs.com |
| Frankfurt | eu-frankfurt | eu-frankfurt.cls.tencentyun.com | eu-frankfurt.cls.tencentcs.com |
| Tokyo | ap-tokyo | ap-tokyo.cls.tencentyun.com | ap-tokyo.cls.tencentcs.com |
| Seoul | ap-seoul | ap-seoul.cls.tencentyun.com | ap-seoul.cls.tencentcs.com |
| Moscow | eu-moscow | eu-moscow.cls.tencentyun.com | eu-moscow.cls.tencentcs.com |
| Jakarta | ap-jakarta | ap-jakarta.cls.tencentyun.com | ap-jakarta.cls.tencentcs.com |
| Toronto | na-toronto | na-toronto.cls.tencentyun.com | na-toronto.cls.tencentcs.com |
| São Paulo | sa-saopaulo | sa-saopaulo.cls.tencentyun.com | sa-saopaulo.cls.tencentcs.com |

As described in Uploading Logs via Kafka, you can use Kafka Producer SDKs or other Kafka agents to upload logs to CLS. This feature uses the following domain names:

| Region | Code | Private Network Domain Name | Public Network Domain Name |
|---|---|---|---|
| Beijing | ap-beijing | bj-producer.cls.tencentyun.com | bj-producer.cls.tencentcs.com |
| Guangzhou | ap-guangzhou | gz-producer.cls.tencentyun.com | gz-producer.cls.tencentcs.com |
| Shanghai | ap-shanghai | sh-producer.cls.tencentyun.com | sh-producer.cls.tencentcs.com |
| Chengdu | ap-chengdu | cd-producer.cls.tencentyun.com | cd-producer.cls.tencentcs.com |
| Nanjing | ap-nanjing | nj-producer.cls.tencentyun.com | nj-producer.cls.tencentcs.com |
| Chongqing | ap-chongqing | cq-producer.cls.tencentyun.com | cq-producer.cls.tencentcs.com |
| Hong Kong (China) | ap-hongkong | hk-producer.cls.tencentyun.com | hk-producer.cls.tencentcs.com |
| Silicon Valley | na-siliconvalley | usw-producer.cls.tencentyun.com | usw-producer.cls.tencentcs.com |
| Virginia | na-ashburn | use-producer.cls.tencentyun.com | use-producer.cls.tencentcs.com |
| Singapore | ap-singapore | sg-producer.cls.tencentyun.com | sg-producer.cls.tencentcs.com |
| Bangkok | ap-bangkok | th-producer.cls.tencentyun.com | th-producer.cls.tencentcs.com |
| Mumbai | ap-mumbai | in-producer.cls.tencentyun.com | in-producer.cls.tencentcs.com |
| Frankfurt | eu-frankfurt | de-producer.cls.tencentyun.com | de-producer.cls.tencentcs.com |
| Tokyo | ap-tokyo | jp-producer.cls.tencentyun.com | jp-producer.cls.tencentcs.com |

| | | | |
|---|---|---|---|
| Seoul | ap-seoul | kr-producer.cls.tencentyun.com | kr-producer.cls.tencentcs.com |
| Moscow | eu-moscow | ru-producer.cls.tencentyun.com | ru-producer.cls.tencentcs.com |
| Jakarta | ap-jakarta | jkt-producer.cls.tencentyun.com | jkt-producer.cls.tencentcs.com |
| Toronto | na-toronto | ca-producer.cls.tencentyun.com | ca-producer.cls.tencentcs.com |

As described in Consumption over Kafka, you can use Kafka Consumer SDKs or other big data components to consume the data to data warehouses. This feature uses the following domain names:

| Region | Code | Private Network Domain Name | Public Network Domain Name |
|---|---|---|---|
| Beijing | ap-beijing | kafkaconsumer-ap-beijing.cls.tencentyun.com | kafkaconsumer-ap-beijing.cls.tencentcs.com |
| Guangzhou | ap-guangzhou | kafkaconsumer-ap-guangzhou.cls.tencentyun.com | kafkaconsumer-ap-guangzhou.cls.tencentcs.com |
| Shanghai | ap-shanghai | kafkaconsumer-ap-shanghai.cls.tencentyun.com | kafkaconsumer-ap-shanghai.cls.tencentcs.com |
| Chengdu | ap-chengdu | kafkaconsumer-ap-chengdu.cls.tencentyun.com | kafkaconsumer-ap-chengdu.cls.tencentcs.com |
| Nanjing | ap-nanjing | kafkaconsumer-ap-nanjing.cls.tencentyun.com | kafkaconsumer-ap-nanjing.cls.tencentcs.com |
| Chongqing | ap-chongqing | kafkaconsumer-ap-chongqing.cls.tencentyun.com | kafkaconsumer-ap-chongqing.cls.tencentcs.com |
| Hong Kong (China) | ap-hongkong | kafkaconsumer-ap-hongkong.cls.tencentyun.com | kafkaconsumer-ap-hongkong.cls.tencentcs.com |
| Silicon Valley | na-siliconvalley | kafkaconsumer-na-siliconvalley.cls.tencentyun.com | kafkaconsumer-na-siliconvalley.cls.tencentcs.com |
| Virginia | na-ashburn | kafkaconsumer-na-ashburn.cls.tencentyun.com | kafkaconsumer-na-ashburn.cls.tencentcs.com |
| Singapore | ap-singapore | kafkaconsumer-ap-singapore.cls.tencentyun.com | kafkaconsumer-ap-singapore.cls.tencentcs.com |

| | | | |
|---|---|---|---|
| Bangkok | ap-bangkok | kafkaconsumer-ap-bangkok.cls.tencentyun.com | kafkaconsumer-ap-bangkok.cls.tencentcs.com |
| Mumbai | ap-mumbai | kafkaconsumer-ap-mumbai.cls.tencentyun.com | kafkaconsumer-ap-mumbai.cls.tencentcs.com |
| Frankfurt | eu-frankfurt | kafkaconsumer-eu-frankfurt.cls.tencentyun.com | kafkaconsumer-eu-frankfurt.cls.tencentcs.com |
| Tokyo | ap-tokyo | kafkaconsumer-ap-tokyo.cls.tencentyun.com | kafkaconsumer-ap-tokyo.cls.tencentcs.com |
| Seoul | ap-seoul | kafkaconsumer-ap-seoul.cls.tencentyun.com | kafkaconsumer-ap-seoul.cls.tencentcs.com |
| Moscow | eu-moscow | kafkaconsumer-eu-moscow.cls.tencentyun.com | kafkaconsumer-eu-moscow.cls.tencentcs.com |
| Jakarta | ap-jakarta | kafkaconsumer-ap-jakarta.cls.tencentyun.com | kafkaconsumer-ap-jakarta.cls.tencentcs.com |
| Toronto | na-toronto | kafkaconsumer-na-toronto.cls.tencentyun.com | kafkaconsumer-na-toronto.cls.tencentcs.com |

# Concepts

# Log

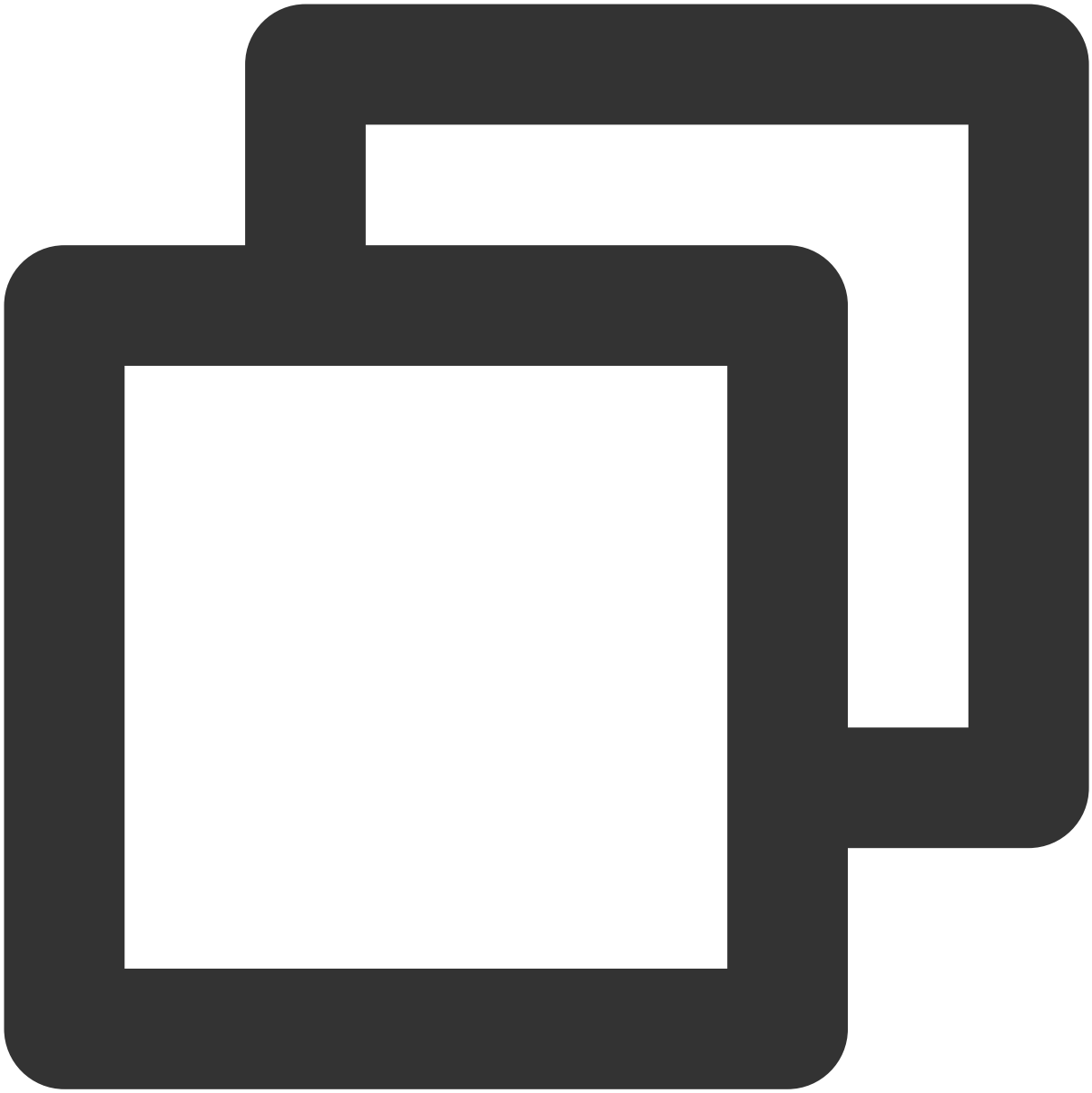
Last updated：2024-01-20 16:31:37

### Log

Logs are record data generated during the running of an application system, such as user operation logs, API access logs, and system error logs. Logs are usually stored in text format on the host where the application system resides. A log corresponding to a system running record may contain one line of text (single-line log) or multiple lines of text (multi-line log).

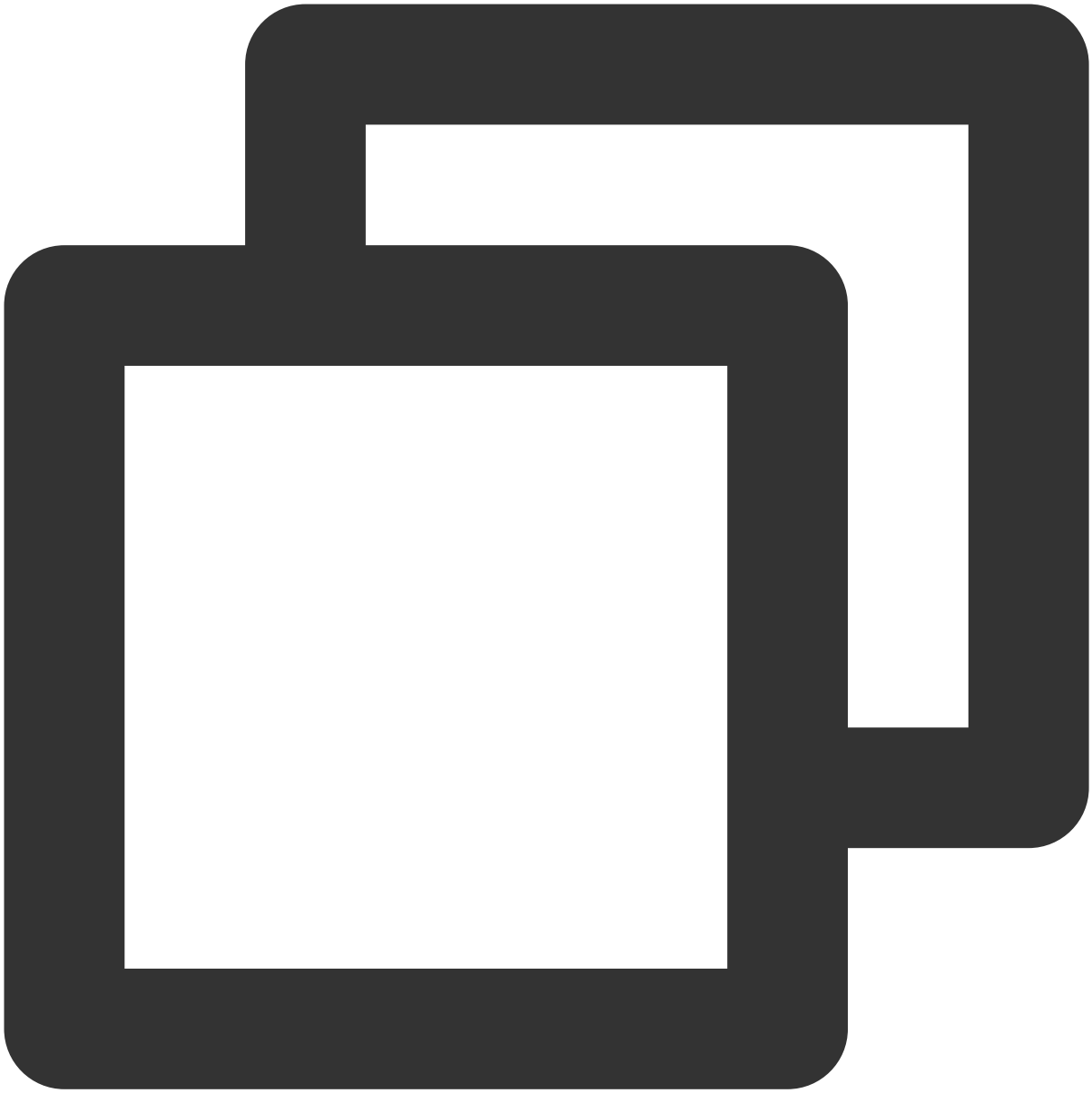
Logs can be uploaded the CLS via LogListener or other methods such as APIs or SDKs.

**Example of a single-line log:**



```
59.x.x.x - - [06/Aug/2019:12:12:19 +0800] "GET /nginx-logo.png HTTP/1.1" 200 368 "h
```

**Example of a multi-line log:**



```
java.net.SocketTimeoutException:Receive timed out
  at j.n.PlainDatagramSocketImpl.receive0(Native Method)[na:1.8.0_151]
  at j.n.AbstractPlainDatagramSocketImpl.receive(AbstractPlainDatagramSocketImpl.
  at j.n.DatagramSocket.receive(DatagramSocket.java:812)[^]
  at o.s.n.SntpClient.requestTime(SntpClient.java:213)[classes/]
  at o.s.n.SntpClient$1.call(^:145)[^]
  at ^.call(^:134)[^]
  at o.s.f.SyncRetryExecutor.call(SyncRetryExecutor.java:124)[^]
  at o.s.f.RetryPolicy.call(RetryPolicy.java:105)[^]
  at o.s.f.SyncRetryExecutor.call(SyncRetryExecutor.java:59)[^]
  at o.s.n.SntpClient.requestTimeHA(SntpClient.java:134)[^]
```

```
    at ^.requestTimeHA(^:122)[^]
    at o.s.n.SntpClientTest.test2h(SntpClientTest.java:89)[test-classes/]
    at s.r.NativeMethodAccessorImpl.invoke0(Native Method)[na:1.8.0_151]
```

The main components of a log are as follows:

| Component | Description | Example |
| --- | --- | --- |
| __TIMESTAMP__ | Log time, in the format of a UNIX timestamp in milliseconds | 1640005601188 |
| __FILENAME__ | Log source file | /data/log/nginx/access.log |
| __SOURCE__ | Log source IP | 10.0.1.2 |
| Log body | Log body content, in `key:value` format (`key` is the field name, and `value` is the field value.)<br>If the log extraction mode is full text in a single line or full text in multi lines (the entire raw log is reported without splitting), the entire log is stored in the __CONTENT__ field.<br>If the log extraction mode is any other mode (such as splitting with a separator), each part of the raw log corresponds to a `key:value` pair. | Full text in a single line or full text in multi lines:<br>10.20.20.10;[2018-07-16 13:12:57];GET /online/sample HTTP/1.1;200<br>Other modes:<br>IP: 10.20.20.10<br>request: GET /online/sample HTTP/1.1<br>status: 200<br>time: [2018-07-16 13:12:57] |
| Metadata | Simple description or categorization of logs. For example, the cluster or container of a TKE log is stored in the `key:value` format, where `key` starts with __TAG__. | __TAG__.clusterId:1skzv59c |

## Log group

A log group (LogGroup) is a collection of multiple logs. In the process of log upload, to increase data read/write efficiency, multiple logs can be packaged into one log group and then sent to CLS.

Logs in the same log group have the same basic information (__TIMESTAMP__, __FILENAME__, __SOURCE__, metadata, etc.).

# Log Topic and Logset

Last updated：2024-01-20 16:32:37

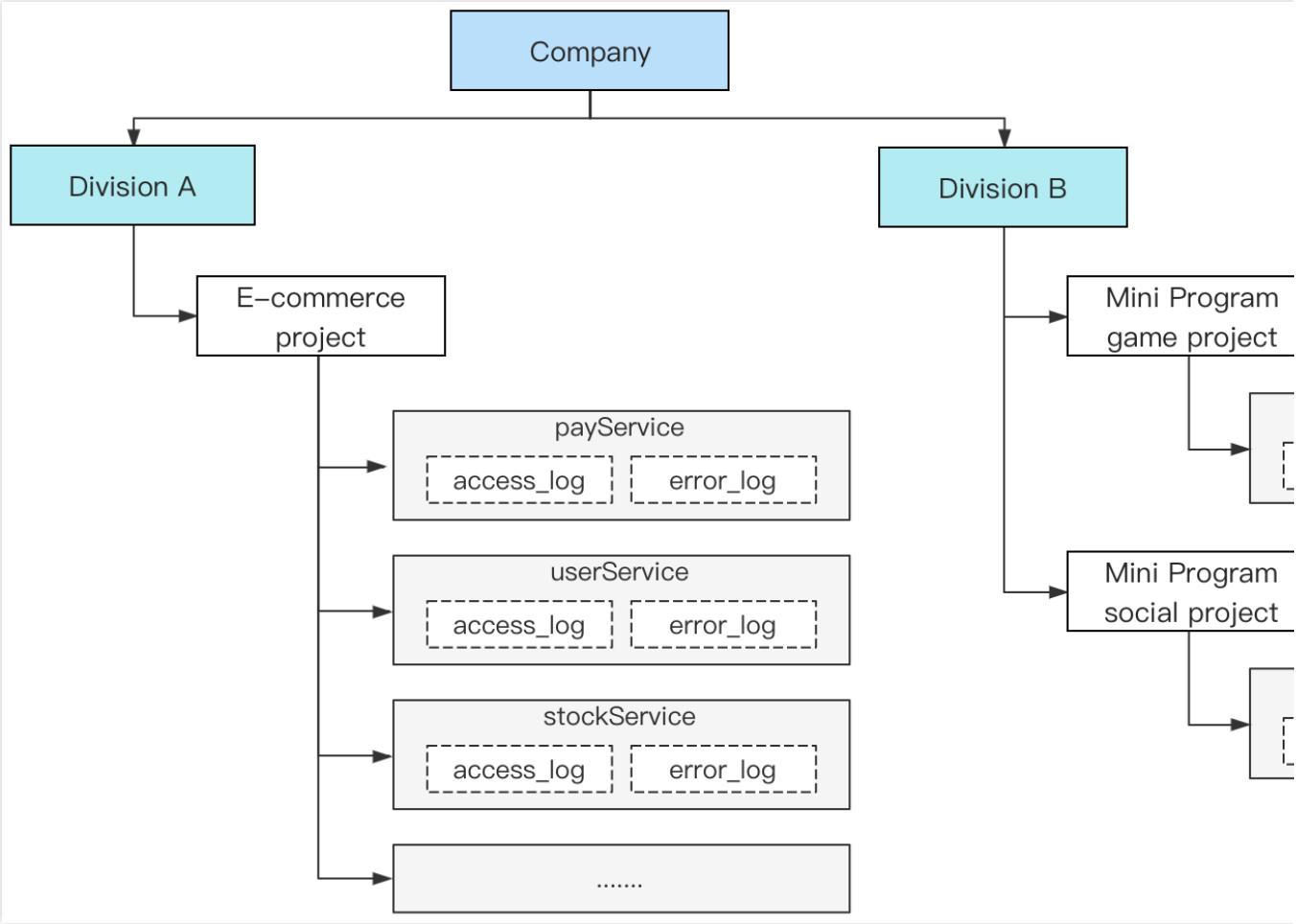
## Log topic

A log topic is a basic unit for log data collection, storage, search and analysis on the CLS platform. The massive amounts of logs collected are managed by log topic. For example, you can configure log collection rules and storage time, search for and analyze logs, and download, consume, and ship logs by log topic.
A log topic usually corresponds to an application/service. It is recommended to collect logs of the same application/service from different servers to the same log topic. For example, a payment service (payService) is deployed on dozens of machines and has two types of logs: access logs (access_log) and error logs (error_log). You can create two log topics, payService_access_log_topic and payService_error_log_topic, to collect the two types of logs on the machines respectively. You can use these two log topics to centrally search for and analyze all logs on the machines.
Log topics and applications/services are not strictly in one-to-one mapping relationships. If the log structures of two services are similar, and the logs of the two services need to be analyzed in a centralized manner, you can report the logs of the two services to the same log topic.

## Logset

A logset is a class of log topics and can contain multiple log topics. A logset itself does not store any log data, but just makes it easier for users to manage log topics.
A logset usually corresponds to a project/business in a company. You are advised to add the log topics of multiple applications/businesses of a project/business to the same logset. For example, if a company's e-commerce project contains multiple services (payService, userService, stockService, etc.), you can create a logset named e_commerce_logset and add the log topics of these services to this logset. In this way, when the company has multiple projects, the project personnel only need to view the log topics in the logset corresponding to the project, without interference from the log topics of other projects.
**Note:**
The logset to which a log topic belongs can be specified when the log topic is created and cannot be changed once specified.

## Example




As shown in the figure above, the company has two departments:

Department A has an e-commerce project that adopts the microservice architecture, and each service has two types of logs: access logs (access_log) and error logs (error_log).

Department B has two projects: Mini Program game project and Mini Program social project. The two projects adopt a simple technical architecture and each has one type of logs: nginx access logs (nginx_log).

To use CLS to monitor the logs of the applications of the company, you can create the following logsets and log topics:

| Logset | Log Topic | Label |
| --- | --- | --- |
| e_commerce_logset | payService_access_log_topic | Department: Department A |
| e_commerce_logset | payService_error_log_topic | Department: Department A |
| e_commerce_logset | userService_access_log_topic | Department: Department A |
| e_commerce_logset | userService_error_log_topic | Department: Department A |
| e_commerce_logset | stockService_access_log_topic | Department: Department A |
| | | |

| | | |
|---|---|---|
| e_commerce_logset | stockService_error_log_topic | Department: Department A |
| e_commerce_logset | ..... | Department: Department A |
| gameApplet_logset | gameApplet_nginx_log_topic | Department: Department B |
| socialApplet_logset | socialApplet_nginx_log_topic | Department: Department B |

The labels are used to identify the departments to which the logsets and log topics belong. Combined with permission policies, the personnel in each department can view the data of their own department only.
For department A, the e_commerce_logset logset covers the log topics of all e-commerce services. If other projects are added later, you only need to create a logset.
For department B, although the current technical architecture is relatively simple, with only two types of logs in total, two logsets are created, each with only one log topic, for the following purposes:
Support for subsequent architecture expansion: If the service scale increases and the technical architecture changes to the microservice architecture, you can continue to use the current logset and add log topics to the current logset. In this way, projects do not affect each other.
Flexible response to project adjustment: If the entire department is used as a logset, and a project needs to be changed to an independent department or moved to another department, log adjustment will be difficult because the logset of a log topic cannot be changed directly, and you may need to collect logs again. If each project corresponds to a logset, this situation does not exist, and you only need to adjust the labels corresponding to the logset and log topic for department adjustment.

# Machine Group

Last updated：2024-01-20 16:33:07

## Machine group

A machine group is a list of machines for which logs need to be collected. In machine groups, CLS manages all servers for which logs are collected using LogListener.

A machine group can contain multiple machines. When an application/service is deployed on multiple machines with the same log file path, the machines can be grouped into a machine group. In this way, you only need to configure the log data collection rule once in the console, and the rule takes effect on all the machines in the machine group in batches.

A machine group can be associated with a log topic. That is, all the logs in the machine group are reported to the same log topic. You can also associate a machine group with multiple log topics. That is, logs of different paths in a machine group are reported to different log topics.

Machine groups can be defined in two ways:

IP address: add an IP address list to a machine group. Then the machines corresponding to the IPs will be automatically added to the machine group.

Label: add labels to machines when installing LogListener. Then the machines with the labels will be automatically added to the machine group.

## Examples

An e-commerce project has 6 machines and 3 services (payService, userService, and stockService). The deployment method is as follows:

payService is deployed on 2 machines, and there are 2 log file paths. userService and stockService are deployed on 4 machines, and there are 4 log file paths. Each of the log types (access_log and error_log) of each of the services needs to be reported to an independent log topic.

| Machine | Service Deployed | Log File Path |
| --- | --- | --- |
| 192.168.1.1 | payService | /data/log/payService/access_log.log<br>/data/log/payService/error_log.log |
| 192.168.1.2 | payService | /data/log/payService/access_log.log<br>/data/log/payService/error_log.log |
| 192.168.1.3 | userService,stockService | /data/log/userService/access_log.log<br>/data/log/userService/error_log.log<br>/data/log/stockService/access_log.log<br>/data/log/stockService/error_log.log |
| 192.168.1.4 | userService,stockService | /data/log/userService/access_log.log |

| | | |
|---|---|---|
| | | /data/log/userService/error_log.log<br>/data/log/stockService/access_log.log<br>/data/log/stockService/error_log.log |
| 192.168.1.5 | userService,stockService | /data/log/userService/access_log.log<br>/data/log/userService/error_log.log<br>/data/log/stockService/access_log.log<br>/data/log/stockService/error_log.log |
| 192.168.1.6 | userService,stockService | /data/log/userService/access_log.log<br>/data/log/userService/error_log.log<br>/data/log/stockService/access_log.log<br>/data/log/stockService/error_log.log |

When deploying LogListener on servers, you can add a label for each machine according to the services running on the machine. See the table below.

| Machine | Service Deployed | Label |
|---|---|---|
| 192.168.1.1 | payService | payService |
| 192.168.1.2 | payService | payService |
| 192.168.1.3 | userService,stockService | userService,stockService |
| 192.168.1.4 | userService,stockService | userService,stockService |
| 192.168.1.5 | userService,stockService | userService,stockService |
| 192.168.1.6 | userService,stockService | userService,stockService |

Then create 3 machine groups in the console and define them with labels payService, userService, and stockService. Then, associate the machine groups with log topics and add the corresponding collection configuration. By now, log reporting configuration is completed.

| Log Topic | Associated Machine Group | Collection Path |
|---|---|---|
| payService_access_log_topic | payService | /data/log/payService/access_log.log |
| payService_error_log_topic | payService | /data/log/payService/error_log.log |
| userService_access_log_topic | userService | /data/log/userService/access_log.log |
| userService_error_log_topic | userService | /data/log/userService/error_log.log |
| stockService_access_log_topic | stockService | /data/log/stockService/access_log.log |
| | | |

| | | |
|---|---|---|
| stockService_error_log_topic | stockService | /data/log/stockService/error_log.log |

If a service needs to be expanded in the future, you only need to add the service label to the machines to be added so that the new machines can be automatically added to the corresponding machine group for log collection, significantly improving Ops and deployment efficiency.

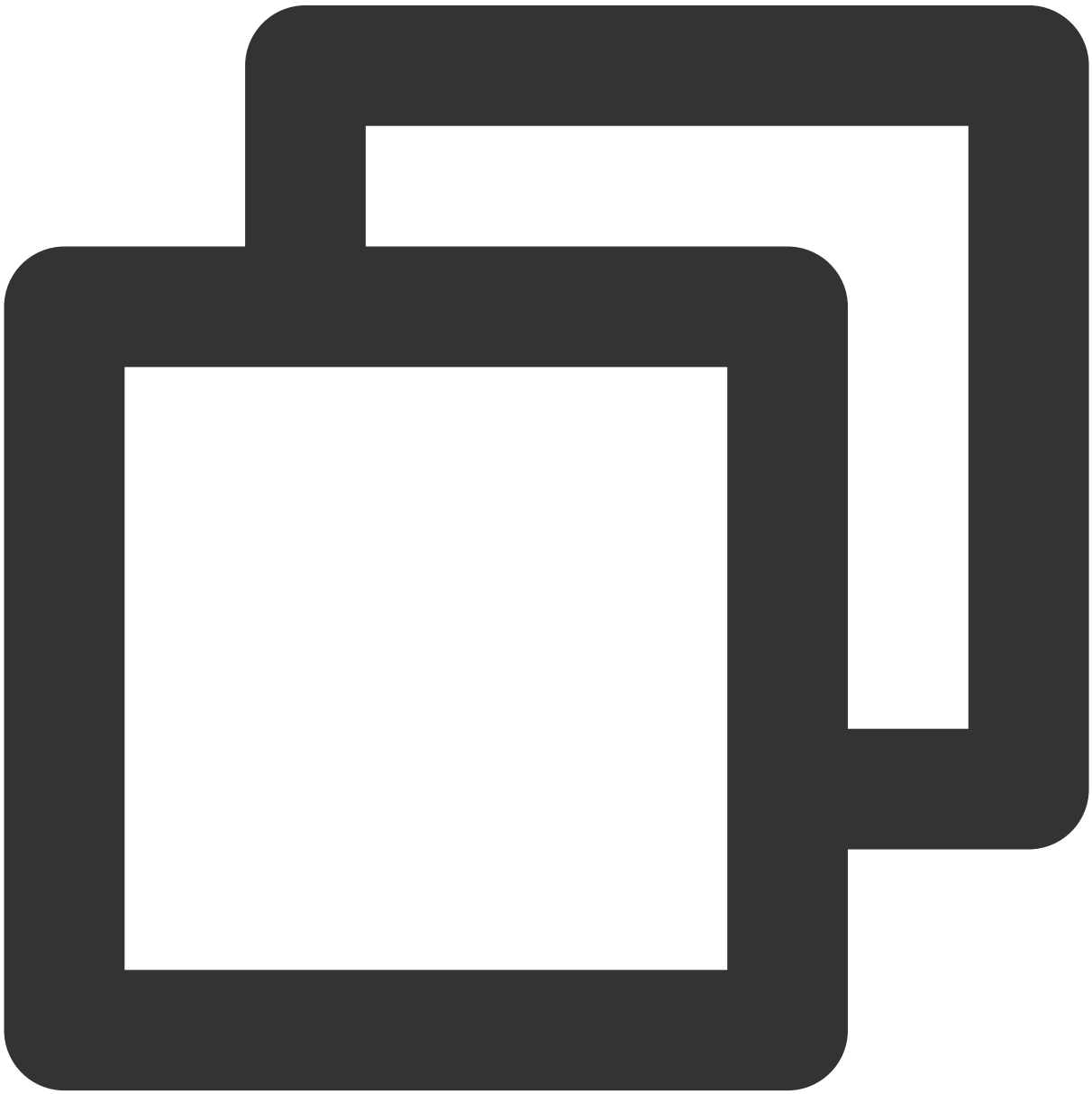
# Segment and Index

Last updated：2024-01-20 16:33:38

## Segmentation

### Segmentation explanation

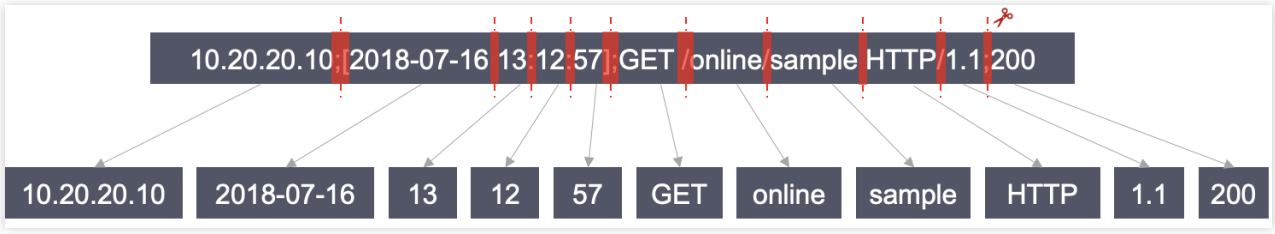
To search for a long log, you usually only need to search for part of the log content. Assume that you need to search for the following complete log that contains `sample` :



```
10.20.20.10;[2018-07-16 13:12:57];GET /online/sample HTTP/1.1;200
```

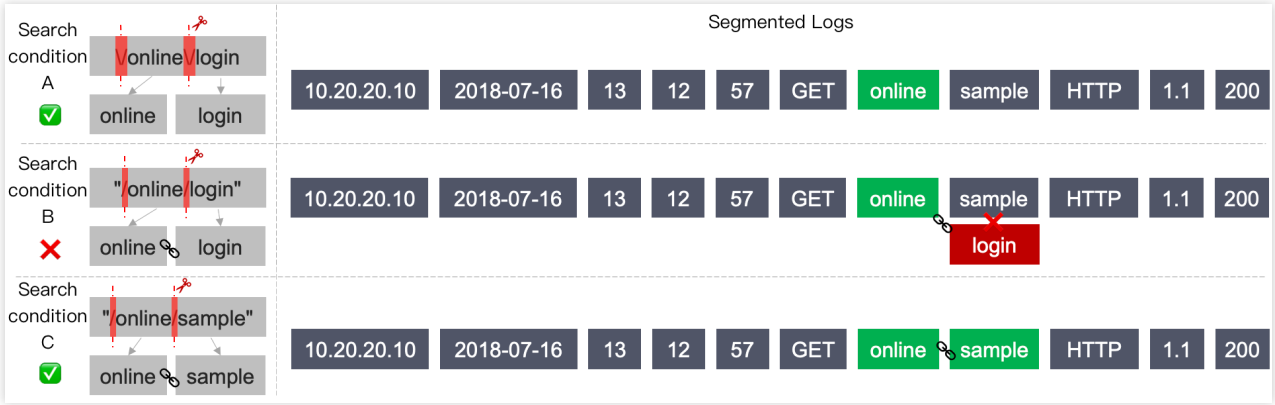
The full text of the log is long, and the log contains many other content in addition to `sample`. The complete log and the search condition are not identical. Therefore, `sample` cannot be used as the search condition to search for the log. To meet this search requirement, the full text of the log needs to be divided into multiple fragments, and each fragment is called a word, and this process is called "segmentation".

For example, the above sample log can be divided based on symbols `@&()='",;:<>[]{}/ \\n\\t\\r`, that is, wherever the symbols occur, segmentation is executed. The segmentation result is as follows:




If the search condition is `sample`, the preceding log after segmentation contains `sample` and therefore is considered to meet the search condition.

The search condition itself is also segmented, such as the 3 types of search conditions in the figure below:




Search condition A: `\\/online\\/login`

`\\` is used to escape the `/` symbol (this symbol is a reserved symbol of the search syntax and therefore needs to be escaped).

The escaped `/` symbol is a delimiter, so the actual search condition is `online OR login`. A log containing `online` **or** `login` is considered to meet the search condition.

The example log provided above **meets** the search condition.

Search condition B: `"/online/login"`

Being enclosed by double quotation marks, the `/` symbol does not need to be escaped.

The content in the double quotation marks is also divided into two words. However, the double quotation marks indicate that only a log that **contains both the two words in the exact order** as that in the search condition is considered to meet the search condition.

The example log provided above does not contain `login` and therefore **does not meet** the search condition.

Search condition C: `"/online/sample"`

The example log provided above contains both `online` and `sample` in the exact order as that in the search condition and therefore is considered to **meet** the search condition.

In most scenarios, if the search condition contains delimiters, you are advised to use double quotation marks to avoid adding escape characters while accurately searching for desired logs. For more information about the search syntax, see Syntax and Rules.

**Segmentation configuration**

The basis of log segmentation falls into two categories. For configuration details, please see Configuring Indexes.

Delimiter: you can customize the symbols for log segmentation, including English symbols and \\n\\t\\r. For example, in the example above, `@&()='",;:<>[]{}/ \\n\\t\\r` are delimiters.

Allow Chinese Characters: Chinese is special, and Chinese symbols cannot be used as delimiters, and segmentation according only to symbols often fails to achieve the desired effect. For example, if a log in Chinese is


and you want the log to be searched for by


, segmentation according to symbols cannot be used to meet the search requirement. In this case, you can set the log to **Allow Chinese Characters** by referring to Configuring Indexes so that CLS will automatically segment every Chinese character into a separate word.

## Index

To enable quick log searches, CLS performs a lot of preprocessing, including segmentation, on the logs uploaded to the platform, and this process is called "indexing". Indexes determine the conditions under which logs can be searched and analyzed. Therefore, before uploading log data, you need to set a reasonable index rule for the corresponding log topic to facilitate subsequent search and analysis. For more information, please see Configuring Indexes.

Index rules take effect only for newly uploaded log data, not historical data. Index creation involves index traffic and index storage, resulting in certain usage costs.

# Topic Partition

Last updated：2024-01-20 16:34:10

## Overview

Partition is the smallest read/write unit of CLS. A log topic can be divided into several partitions and must have at least one partition. CLS uses the value range of MD5 as the valid range and controls the overall throughput performance by merging or splitting partitions. A log topic supports up to 50 partitions. We recommend that you use and work with topic partitions rationally to prevent waste of resources.
Basic attributes of a partition:
**Partition number**: every partition has a unique number under the same log topic, which is assigned by the system after partition creation or operation.
**Partition range**: every partition has a left-closed and right-open interval.
**Partition status**:
Read-write: the current partition allows read and write.
Read-only: the current partition is read-only and no data can be written to it.
**Note:**
Topic partitioning is a complex concept. You are advised to use the Auto Split function in actual practice. CLS automatically adjusts topic partitioning based on the volume of log data.

## Partition Range

With a partition range, logs can be written in the HashKey mode. The valid range of a log topic is the value range of MD5, which is `[00000000000000000000000000000000,ffffffffffffffffffffffffffffffff)`. All read-write partitions segment the entire value range of a log topic and each occupies a left-closed and right-open interval to ensure that every log record collected is written to the corresponding partition.
CLS provides two write modes: load-balancing mode and HashKey mode.
**Load-balancing mode**: every data packet is written to a random log topic partition.
**HashKey mode**: every data packet is written to the topic partition that contains the current Key value.
For example, a log topic has three read-write partitions and their ranges are as follows:

| Partition No. | Status | Partition Range |
| --- | --- | --- |
| 1 | Read-write | `[00000000000000000000000000000000,7fffffffffffffffffffffffffffffff` |
| | | |

| | | |
|---|---|---|
| 2 | Read-write | `[7fffffffffffffffffffffffffffffff,a0000000000000000000000000000000` |
| 3 | Read-write | `[a0000000000000000000000000000000,ffffffffffffffffffffffffffffffff` |

If the write mode is HashKey, log data with the Key value of `2fffffffffffffffffffffffffffffff` is written to partition 1 and log data with the Key value of `9f000000000000000000000000000000` is written to partition 2.

# Read and Write Capability of Partitions

Each partition has a certain level of read and write capability. We recommend that you plan partition count based on the actual log traffic of your business. Split partitions when the traffic is higher than the read and write capability of the log topic, and merge partitions when the traffic is lower than the read and write capability of the log topic to save resources.

| Feature | Item | Description |
|---|---|---|
| Frequency control | Write requests | Every partition supports a maximum of 500 QPS of write operations. It will reject requests and return the status code 429 with an error message of "SpeedQuotaExceed" when the limit is exceeded. |
| | Read requests | Every partition supports a maximum of 200 QPS of read operations. It will reject requests and return the status code 429 with an error message of "SpeedQuotaExceed" when the limit is exceeded. |
| Traffic throttling | Write traffic | Every partition supports the write traffic of up to 5 MB/sec. It will truncate data and return the status code 429 with an error message of "SpeedQuotaExceed" when the limit is exceeded. |

# Partition Status

A partition can be in **read-write** or **read-only** mode. Only read-write partitions support data writing. Read-only partitions do not allow data writing but can be consumed within their lifetime. All partitions are readable and writable when they are created, but merging and splitting operations will change the status to read-only.

### Merging partitions

Two adjacent read-write partitions can be merged into one partition. After two partitions are merged, the two original partitions become read-only, which only allow data consumption, but not data writing. The new partition is readable and writable and covers the range of the two original partitions.

### Splitting a partition

A read-write partition can be split into two partitions with smaller ranges. When splitting a partition, you need to specify the MD5 value of a split point, which must be larger than the value of the start point and smaller than the value of the end point. After a partition is split, the original partition becomes read-only, which only allows data consumption, but not data writing. The new partition is readable and writable and covers the range of the original partition.

# Limits
# Log Collection
# LogListener Limits

Last updated：2024-01-20 16:34:39

This document describes LogListener's capabilities and limits of file collection, configuration, resources, performance, and error handling related to data collection.

## File Collection Capabilities and Limits

| Item | Capability and Limit |
| --- | --- |
| File encoding | Supports UTF8 and GBK codec.<br>**Note:** only LogListener v2.6.2 and above support GBK codec. |
| Soft link | Supported. |
| Size of a single log | The size of a single-line log is limited to 512 KB. If the size of a single-line log exceeds 512 KB, the log will be truncated and only the first 512 KB is retained. For a multi-line log, after it is divided according to the first-line regular expression, the maximum limit for a single log is 1 MB. |
| Regular expression | Perl-compatible regular expressions are supported. |
| First log collection behavior | LogListener supports full/incremental collection policies:<br>Full collection: after LogListener is started for the first time after installation, all logs that meet requirements are collected, including files that have not been written.<br>Incremental collection: after LogListener is started for the first time after installation, existing files will be collected from the latest position.<br>**Note:** Full/Incremental collection policies are available starting from LogListener v2.6.2. |
| Log file rotation | Supported. (It is recommended that the file name after rotation not be matched by the wildcard collection path.) |
| Collection behavior when log parsing fails | We recommend you enable the feature of uploading parsing-failure logs. If the feature is enabled, parsing-failure logs will be uploaded to the preset index in the format of full text in a single line. Otherwise, the logs will be discarded. |
| File opening | LogListener opens files when reading files for log collection and closes the files when the reading is completed. |

## Checkpoint Management Capabilities and Limits

| Item | Capability and Limit |
| --- | --- |
| Checkpoint storage location | The default save path is the `data` directory in the LogListener installation directory. You can customize a save path in the `loglistener.conf` file. |
| Checkpoint storage policy | LogListener stores two copies of checkpoint metadata:<br>One copy records the information only of checkpoints where upload is completed, and the information is persisted to a disk in real time.<br>The other copy records the information of checkpoints where upload is completed and checkpoints where upload is not completed, and the information is periodically persisted to a disk. Persistence takes precedence when the program exits. |

## Resource and Performance Capabilities and Limits

| Item | Capability and Limit |
| --- | --- |
| Default CPU resource limits | LogListener does not limit CPU resources by default. You can configure CPU resource limits in the configuration file.<br>Without CPU resource limits, the current implementation architecture of LogListener can achieve a maximum CPU usage of 110% (up to 100% for a single business thread and about 10% for a control thread). |
| Default memory resource limits | LogListener's default memory threshold is 2 GB. You can change the threshold to 300 MB or higher as needed. |
| Default bandwidth resource limits | LogListener does not limit bandwidth resources by default. You can configure bandwidth resource limits in the `loglistener.conf` file. |
| Resource overrun handling policy | If LogListener overruns CPU and memory resources for more than 5 minutes, LogListener will be forced to restart automatically. |
| Log compression | Collected logs are shipped as compact logs by default. You can modify log compression configuration in the `loglistener.conf` file. |
| Number of monitored directories | The recommended maximum number of monitored directories is 5,000. If this limit is exceeded, log collection may fail. |
| | |

| | |
|---|---|
| Number of monitored files | The recommended maximum number of monitored files is 10,000. If this limit is exceeded, log collection may fail. |

## Error Handling

| Item | Capability and Limit |
|---|---|
| Network error handling | Except exceptions requiring special handling (such as log topic deletion), all other errors (such as network exceptions, timeout, frequency control, and overdue payment) will be retried. |
| Maximum retry timeout duration | If data fails to be sent after retrying for more than 1 hour, the system discards the data. The default behavior is to retry at an interval, and the retry interval becomes longer and longer until the maximum retry timeout duration is exceeded. |
| Maximum number of retries | The maximum number of retries can be set in the `loglistener.conf` configuration file and is not set by default. By default, the system retries until the maximum retry duration is exceeded, and then discards the data.<br>If the maximum number of retries is set, the system retries until the maximum number of retries is exceeded, and then discards the data. |

## File Collection Rules

| Item | Capability and Limit |
|---|---|
| Log upload policy | LogListener automatically aggregates and uploads logs of the same file when any of the following aggregation conditions is met: 10,000 logs, the total logset size reaching 1 MB, or the log collection time exceeding 3 seconds. |
| File collection handling policy | A single target file (a file that can be matched by the collection path) can be uploaded to only one log topic. The same file cannot be matched by the collection paths of multiple topics.<br>If you need to upload a file to multiple topics, use soft links. You can create different soft links for the same target file so that it can be collected to different topics via different soft links. |
| Log collection delay | In the case of real-time collection, data collection, transmission, and storage to disks will be completed within 1 minute, and the logs are retrievable in the console.<br>If the log volume is large, or LogListener can use only a limited amount of resources, there will be some collection delay. |

## Machine Group Related Rules

| Item | Capability and Limit |
| --- | --- |
| Machine group logic | At present, machine groups are divided into the following two categories, and their usage methods are independent of and incompatible with each other. If the two usage methods are used together, LogListener will not be able to pull the correct collection configuration, resulting in no collection.<br>IP machine group: a machine IP must be manually added to the machine group in the console, and the `group_label` field in `loglistener.conf` on the corresponding machine must be empty.<br>Label machine group: the machine group label is set in the console, and the `group_label` field in `loglistener.conf` on the corresponding machine must be set to the same label. |
| Relationships between machine groups and log topics | A single log topic can be bound to multiple machine groups.<br>A single machine group can be bound to multiple log topics. |
| Relationships between machine groups and machines | A single machine can be added to multiple machine groups.<br>For an IP machine group, there is no limit to the number of machine groups that a machine can join.<br>For a label machine group, the maximum number of machine groups that a machine can join is 20. |
| Limits on the labels of label machine groups | Currently, the label of a label machine group can contain up to 32 characters.<br>Up to 20 labels can be configured for a single label machine group. |

## Collection Path/Collection Blocklist Usage

| Item | Capability and Limit |
| --- | --- |
| Collection blocklist | This feature is used to specify the content to ignore in the collection path. Currently, collection blocklists support two modes:<br>FILE mode: specify the files to ignore in the collection path.<br>PATH mode: specify the subdirectories to ignore in the collection path. Wildcard filtering is supported.<br>**Notes:**<br>The FILE and PATH modes can be used together.<br>A collection blocklist is to exclude content from the collection path, and for both the FILE and PATH modes, the specified path to exclude must be a subset of the collection path. |