Log Service

Product Introduction

Product Introduction

The Log Service (or Log for short) is an all-in-one service for log-type data that allows you to quickly complete log data collection, consumption, shipping, query, and analysis without the need for development. Log Service can help you increase O&M efficiency, and build processing capabilities, that help you easily handle massive log volumes.

Real-time log collection (LogHub)

Log Service supports more than 30 massive data collection methods.

- Logtail for log collection: stable and reliable, secure, available for all platforms (Linux, Windows, and Docker), high performance, and low resource utilization.
- API/SDK for log collection: flexible and convenient, scalable, available in more than 10 languages and mobile terminals.
- Cloud product log collection: supports logs from Elastic Compute Service (ECS), Container Service, Message Service (MNS), Content Delivery Network (CDN), and other cloud products. One key implementation, convenient and efficient.
- Other methods: Syslog, Unity3D, Logstash, Log4j, Nginx, and more.

Real-time log consumption (LogHub)

Stream computing, collaborative consumption library, multiple-language support.

- Comprehensive functions: compatible with all Kafka functions while offering ordering, elastic scaling, time-frame-based seek, and other functions
- Stable and reliable: any written data can be consumed, 99.9% or higher availability, multiple data copies, elastic scaling within seconds, low cost.
- Easy to use: supports Spark Streaming, Storm, Consumer Library (an automatic load balancing programming mode), SDK subscriptions, and more.

LogShipper

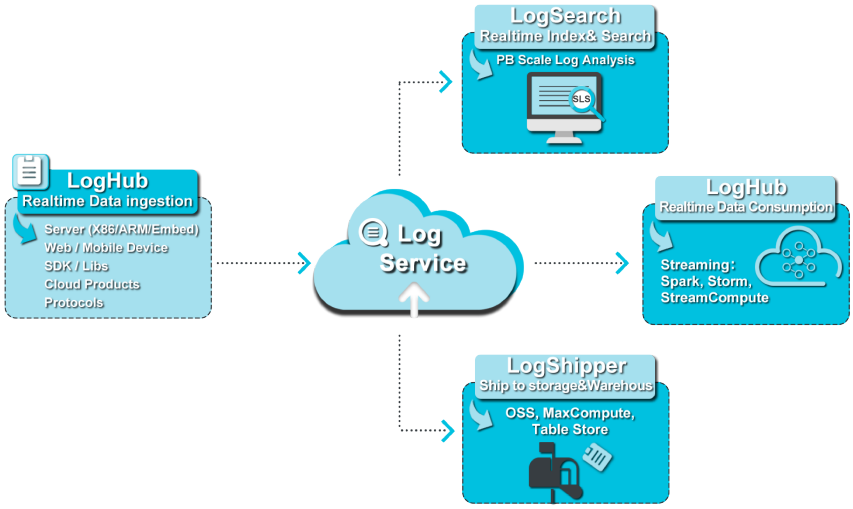
Stable and reliable log shipping. Ship LogHub data to storage services for storage and big data analysis.

- Object Storage Service (OSS): ship logs to OSS and use E-MapReduce for data analysis.
- MaxCompute: ship logs to MaxCompute for analysis (coming soon).

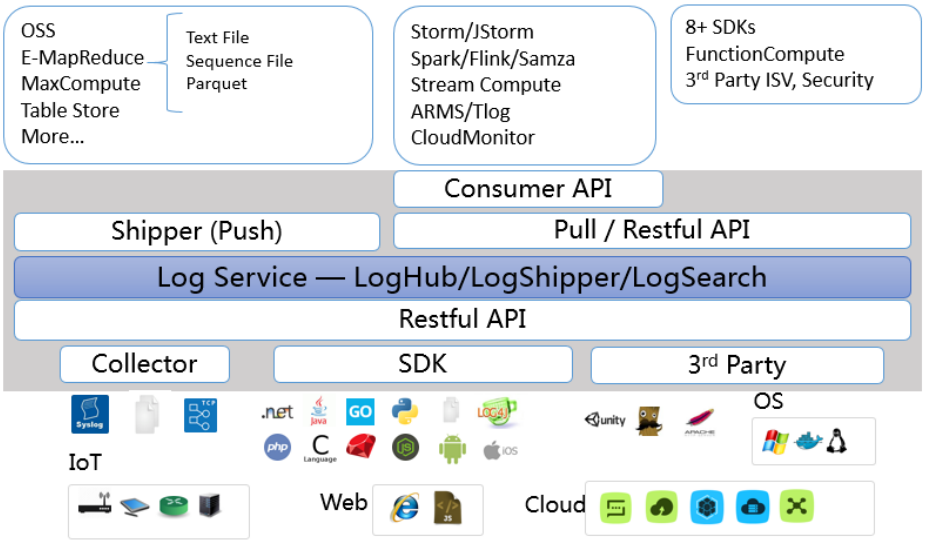
LogSearch

Real-time data indexing and querying. Create indexes for LogHub data with a time and keyword-based search function.

- Large scale: real-time indexing of PB-level data volumes (data can be queried within 1 second of writing), query over a billion log entries per second.
- Low cost: 0.5 CNY/GB indexing cost, massive storage at a low cost, supports lifecycle configuration.
- Flexible queries: supports keyword, fuzzy, cross-topic, and context queries.



The Log Service system architecture is shown in the following diagram.



Logtail

Logtail helps you quickly collect logs through the following features.

- Non-invasive log collection based on log files
  - Read-only files
- Secure and reliable
  - Supports file rotation, so data are not lost
  - Supports local caching
  - Provides network exception retry mechanism
- Convenient management
  - Web client visualization configuration
- Comprehensive self-protection
  - Real-time monitoring of process CPU and memory consumption and restrictions on CPU/memory usage

Frontend servers

Frontend machines are built using LVS+Nginx. Its features are as follows.

- HTTP and REST protocols
- Horizontal scaling
  - When traffic increases, frontend machines can be quickly added to improve processing capabilities
- High throughput, low latency
  - Pure asynchronous processing, a single request exception will not affect other requests
  - Lz4 compression is adopted to increase the processing capabilities of individual machines and reduce network bandwidth consumption

Backend servers

The backend is a distributed process deployed on multiple machines. It provides real-time Logstore data persistence, indexing, query, and shipping to MaxCompute (coming soon). The features of the overall backend service are as follows.

- High data security:
  - Each log you write is saved in triplicate
  - Data are automatically recovered in case of any disk damage or machine downtime
- Stable service:
  - Logstores automatically migrate in case of a process crash or machine downtime
  - Automatic server load balancing ensures that traffic is distributed evenly among different machines
  - Strict quota restrictions that prevent abnormal behavior of a single user from affecting other users
- Horizontal scaling:
  - Horizontal scaling is performed using shards as the basic unit. You can dynamically

add shards as needed to increase throughput

Convenient

- Fully functional management console with complete APIs, allowing you to set up access in 5 minutes.
- More than 30 access methods, seamless connection with cloud products (OSS/E-MapReduce/MaxCompute/Table Store/MNS/ARMS) and open-source software (such as Storm and Spark).

Stable and reliable

- Elastic: meets all processing needs ranging from several MBs to 100 TB per day
- Availability: higher than 99.9% availability, with automatic resumable data transfer and other functions available at the client.
- Security: access control, HTTPS, encryption, and a full range of other security mechanisms.

Low cost

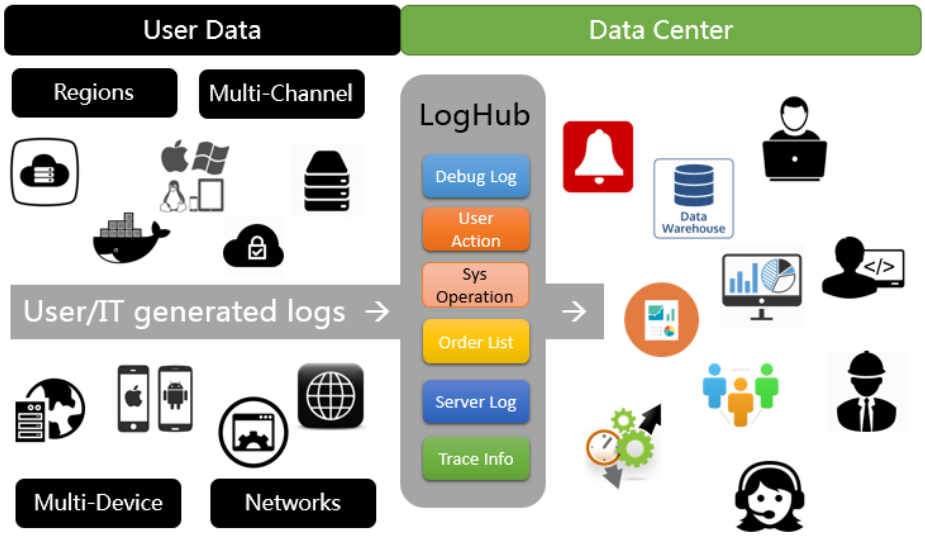
- Pay-As-You-Go billing, reducing costs up to 60% compared to self-built systems.
- Zero O&M and zero deployment, easy upgrading and resizing without impacting your business.
- Client performance is 10 times higher than similar software, while only consuming 10% of the resources.

Typical Log Service application scenarios include data collection, real-time computing, data warehousing and offline analysis, product operation and analysis, and O&M and management.

Data collection and consumption

Log Service offers over 30 data collection measures, supports various networks, environments, devices, and data from different sources. Also, it enables centralized management of distributed logs.

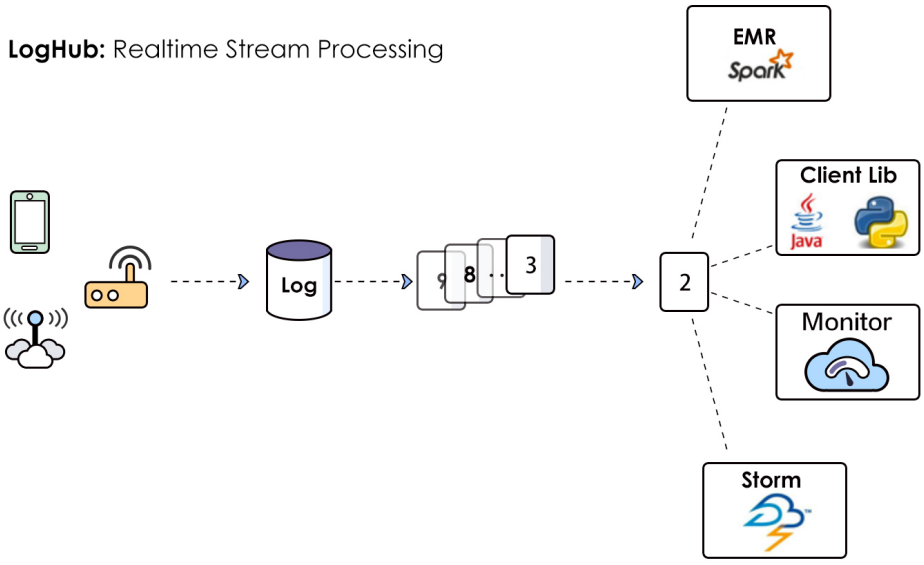
Reference: Log processing example - solution for an ecommerce website.



Interconnection with real-time computing

LogHub ensures log generation and consumption in real-time. It can interconnect with various stream computing tools (Spark, Storm, Stream Compute, and ConsumerLib) and monitoring systems, and perform real-time analysis.

Recommended combination: Log Service + E-MapReduce + Apache Storm

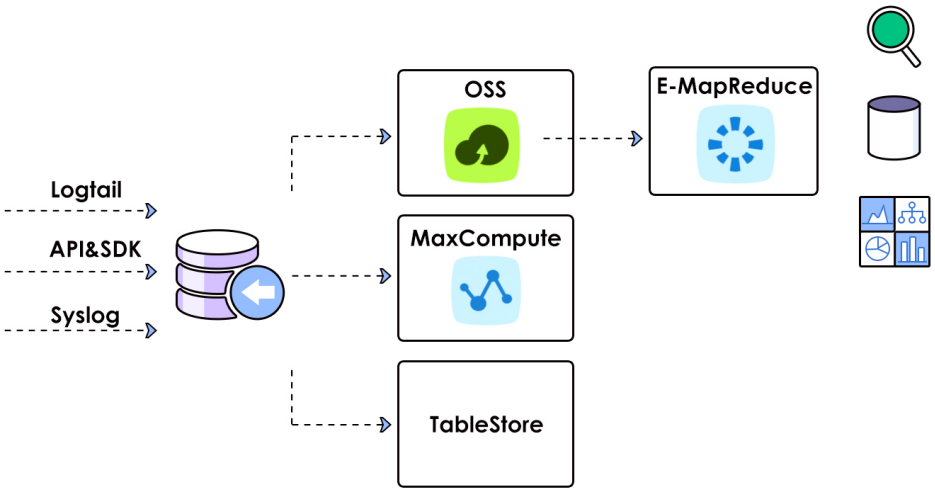


For a detailed solution, refer to Data cleansing and streaming computing (ETL/Stream processing).

Interconnection with the data warehouse

Log Service is seamlessly integrated with OSS, MaxCompute (coming soon), and Table Store, based on which a warehouse with massive data can be built for log analysis.

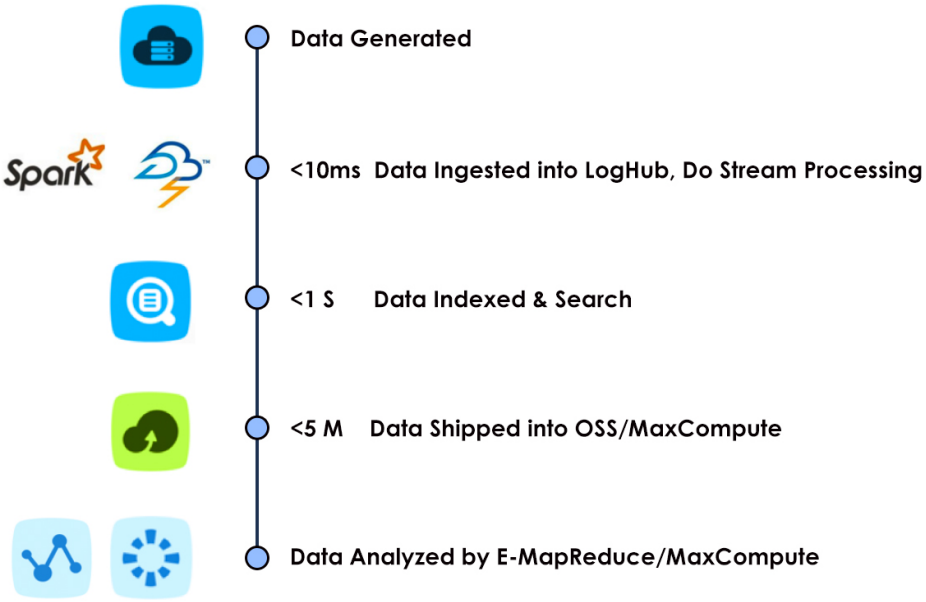
Recommended combination: Log Service + OSS + E-MapReduce + MaxCompute (coming soon)



Log auditing

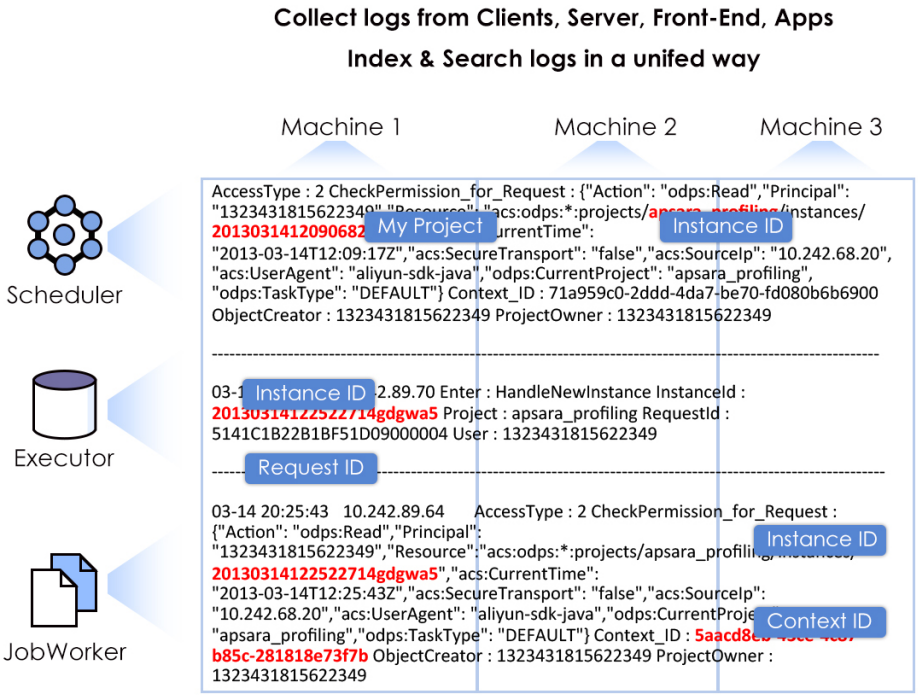
The log auditing function collects and centralizes storage logs. Based on Log Service, you can monitor logs in real-time, query recent hot data in real-time, and back up cold data.

Recommended combination: Log Service + OSS + E-MapReduce + MaxCompute (coming soon)



Log management/query

The log collection and query function can centralize data from applications, systems, and machines. It facilitates rapid investigation of user, application, or system problems to improve operation efficiency.



For a detailed solution, refer to Log management.

Basic concepts

Log

Log is an abstraction of system changes during the runing process. The content is the time-ordered collection of some operations and operation results of specified objects. LogFile, Event, BinLog and Metric data are different carriers of logs. In LogFile, every log file is composed of one or more logs, and every log describes a single system event. A log is the minimum data unit processed in Log Service.

Log group

A log group is a collection of logs, and it is the basic unit for writing and reading.

Log topic

Logs in the same Logstore can be grouped by log topics. You can specify the topic when writing a log, and specify the log topic when querying logs.

Project

A project is the Log Service’ s resource management unit that isolates and controls resources. You can use a project to manage all the logs and related log sources of one application. Projects manage Logstores and machine configurations of log collection. They also serve as the portal through which you can access Log Service resources.

Logstore

A Logstore is the unit used in Log Service for log data collection, storage, and query. Each Logstore belongs to one project, and multiple Logstores can be created for a single project.

Shard

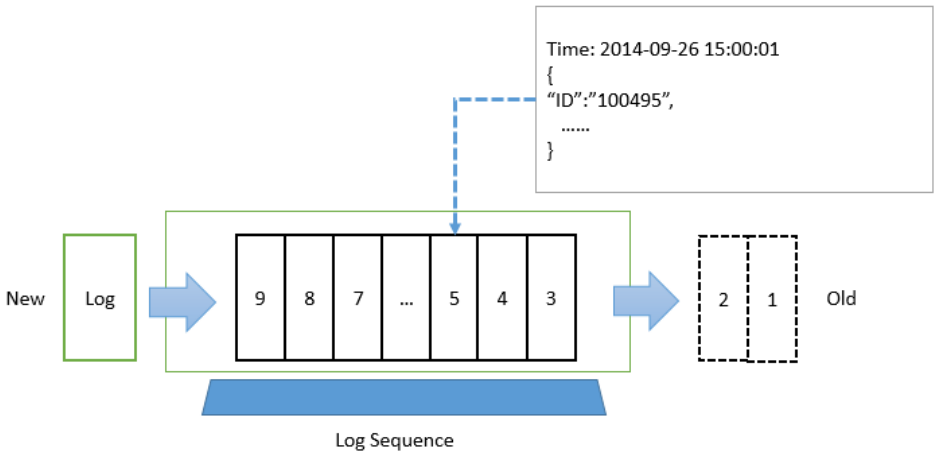
Each Logstore is divided into several shards and each shard is composed of an MD5 left-closed, right-open interval. These intervals do not overlap and the range of all intervals is the entire MD5 value range.

Historically, the term “log” was associated with a thick notebook written by a ship captain or operator. Now with the advent of technology, logs are produced and consumed from all technological areas: servers, routers, sensors, GPS devices, orders, and various IoT devices generate and use logs.

What is a log?

Consider the example of a ship captain’ s log. In addition to a recorded timestamp, a log may contain all sorts of information, including a text record, an image, weather conditions, or sailing course. Presently, the concept of a “captain’ s log” has been expanded to include orders, payment records, user accesses, database operations, and many other fields that relate to today’ s technology.

The reason why logs are an enduring concept is that they are the simplest storage abstraction. A log is a series of records, that is, data arranged by chronological time order. A log can be therefore viewed as follows.



Each record has a unique log record number that fits into a definite sequence.

The log sequence is determined by time. From the preceding image, we can see that this log’ s time sequence runs from right to left. However, a log must always record the time an event happened.

Logs in Log Service

A log is an abstraction of system changes during the runing process. The content is the time-ordered collection of some operations and operation results of specified objects. LogFile, Event, BinLog and Metric data are different carriers of logs. In LogFile, every log file is composed of one or more logs, and every log describes a single system event. A log is the minimum data unit processed in Log Service.

Log Service uses a semi-structured data model to define a log. This model is composed of four data fields: Topic, Time, Content and Source.

Furthermore, Log Service has different format requirements for different fields, as described in the following table.

Data Field	Meaning	Format
Topic	A custom field to mark a batch of logs. For example, access logs can be marked according to sites.	Any string up to 128 bytes, including null strings. By default, this field is a null string.
Time	This is a reserved field in the log, used to indicate the generation time of the log. It is typically generated directly based on the time in the log.	It should be an Integer in standard UNIX time format. The unit is in seconds. This field indicates the number of seconds from 1970-1-1 00:00:00 UTC.
Content	This field is used to record the specific content of the log. Content is composed of one or more content items, and each content item is a Key-Value pair.	Key is a UTF-8 encoded string up to 128 bytes, and can contain letters, underscores, and numbers. It cannot start with a number. The following keywords cannot be used in the key:

		__time__, __source__, __topic__, __partition_time__, _extract_others_, and __extract_others__. The value can be any string up to 1024*1024 bytes.
Source	The source of the log, for example, the IP address of the machine generating the log.	Any string up to 128 bytes. By default, this field is null.

Various log formats are used in actual application scenarios. For ease of understanding, the following describes how to map an original Nginx access log to the Log Service log data model. Assume that the IP address of the user’ s Nginx server is 10.249.201.117, and the following is the original log.

```
10.1.168.193 - - [01/Mar/2012:16:12:07 +0800] "GET /Send?AccessKeyId=8225105404 HTTP/1.1" 200 5 "-"
"Mozilla/5.0 (X11; Linux i686 on x86_64; rv:10.0.2) Gecko/20100101 Firefox/10.0.2"
```

Map the original log to the Log Service log data model as follows.

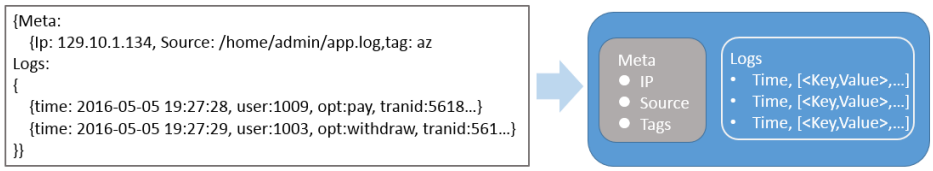
Data Field	Content	Description
Topic	“”	Use the default value (null string).
Time	1330589527	Precise generation time of the log, indicating the number of seconds from 1970-1-1 00:00:00 UTC. This time is converted from the time stamp in the original log.
Content	Key-Value pair	Content of the log.
Source	“10.249.201.117”	Use the IP address of the server as the log source.

You can decide how to extract the original content of the log and combine it into Key-Value pairs. The following table shows an example.

Key	Value
ip	“10.1.168.193”
method	“GET”
status	“200”
length	“5”
ref_url	“- ”
browser	“Mozilla/5.0 (X11; Linux i686 on x86_64; rv:10.0.2) Gecko/20100101 Firefox/10.0.2”

A log group is a collection of logs, and it is the basic unit for writing and reading.

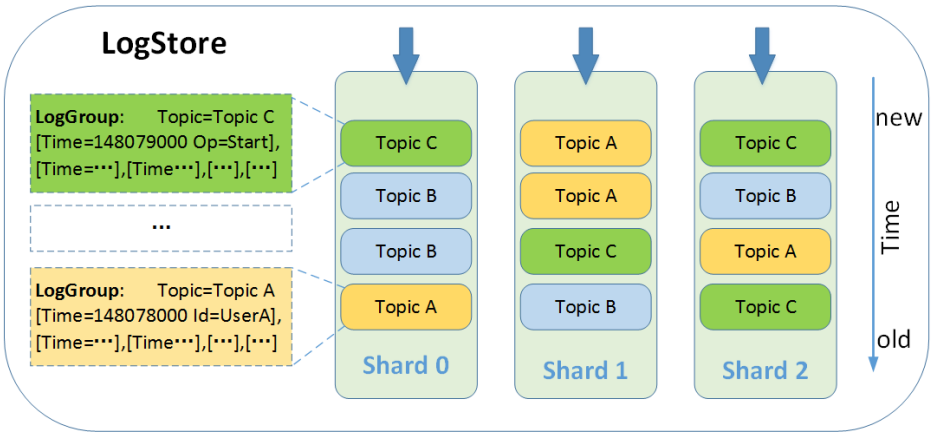
The maximum capacity of a log group is up to 4096 logs or 10 MB.



Logs in the same Logstore can be grouped by log topics. You can specify the topic when writing a log, and specify the log topic when querying logs. For example, platform users can use the user IDs as the log topics and write them into the logs. In this way, users can select to only view their own logs based on log topics. If there is no need to group the logs in one Logstore, the same log topic can be used for all logs.

> **Note:** A null string is a valid log topic, and it is the default log topic for writing and querying logs. Therefore, if there is no need to use a log topic, the easiest method is to use a null string, when writing and querying logs.

The following diagram describes the relationship between Logstore, log topic, and log.



A project is the Log Service' s resource management unit. Projects isolate and control resources.

You can use a project to manage logs and related log sources of one application. A project manages the Logstores of a user and machine configurations of a log collection. It also serves as the portal for users access the Log Service resources.

Projects provide the following functions.

> Projects help you organize and manage different Logstores. In actual use, you may use Log Service to collect and store different project, product, or environment logs in a centralized

manner. You can classify different logs for management in different projects to facilitate subsequent log consumption, exporting, or indexing. In addition, projects also carry the log access permission management functions.

Projects provide portals to access Log Service resources. Log Service allocates a unique access portal to each created project. This access portal supports log writing, reading, and management via the network.

You can use the Log Service console to perform the following project operations.

- Create a project
- View project list
- Manage a project
- Delete a project

Logstores are the units used in Log Service for log data collection, storage, and query. Each Logstore belongs to one project, and multiple Logstores can be created for a single project.

You can create multiple Logstores for one project according to your needs. Typically, an independent Logstore is created for each type of log in one application. For example, assume that you have a game application “big-game”, and there are three types of logs on the server: operation_log, application_log, and access_log. You can first create a project named “big-game”, and then create three Logstores for the three types of logs under this project.

Whether writing or querying logs, you must specify a Logstore for the operation. If you want to ship the log data to MaxCompute (coming soon) for offline analysis, the data will be shipped in Logstore units for data synchronization (that is, the data in a single Logstore are shipped to a single MaxCompute table).

Logstores provide the following functions.

- Log collection, supports real-time logging
- Log storage, supports real-time consumption
- Index creation, supports real-time log query
- A data tunnel that ships logs to MaxCompute

You can use the Log Service console to perform the following Logstore operations.

- Create a Logstore
- View Logstore list
- Modify Logstore configurations
- Delete a Logstore

Logstore read/write logs must be saved in a certain shard. Each Logstore is divided into several

shards and each shard is composed of MD5 left-closed, right-open intervals. These intervals do not overlap and the interval ranges add up to the entire MD5 value range.

Range

When creating a Logstore, the entire MD5 range is automatically divided evenly based on the specified number of shards. Each shard has a certain range within the following value range: [00000000000000000000000000000000,ffffffffffffffffffffffffffffffff).

Each shard is composed of the following keys.

- BeginKey: indicates the start of the shard. This key is included in the shard range.
- EndKey: indicates the end of the shard. This key is excluded from the shard range.

With the shard range, you can write logs by specifying Hash Key, as well as split or merge shards. The corresponding shard must be specified when reading data from it. You can use load balancing mode or specified hash key mode when writing data. In load balancing mode, a data packet is written to an available shard at random. In specified Hash Key mode, data is written to the shard whose range includes the specified key.

In the following example, assume that the MD5 value range of the Logstore is [00,ff), and the Logstore contains 4 shards with the following ranges.

Shard No.	Range
Shard0	[00,40)
Shard1	[40,80)
Shard2	[80,C0)
Shard3	[C0,FF)

If you write a log and specify the MD5 key as 5F, the log data will be written into shard1 that contains the MD5 key 5F. If you specify the MD5 Key as 8C, the log data will be written into shard2 that contains the MD5 key 8C.

Shard read/write capacities

Each shard has certain service capacities.

- Write: 5 MB/s, 2000 times/s
- Read: 10 MB/s, 100 times/s

You can calculate the number of shards needed based on the traffic. If you have already set up several shards, you can also split or merge shards.

Note:

- When writing logs, if the API consistently reports 403 or 500 error, refer to Logstore CloudMonitor metrics to view the traffic and status code and determine whether you need to increase the number of shards.
- For read/write operations that exceed a shard’ s service capacities, the system will attempt to provide the needed services, but the service quality cannot be ensured.

Shard status

- readwrite: capable of reading and writing
- readonly: read-only data

Shard operations

On the Log Service console, you can perform the following shard operations.

- Split a shard
- Merge shards
- Delete a shard