Log Service

User Guide

MORE THAN JUST CLOUD | Alibaba Cloud

User Guide

Project

You can create a project on the Log Service console.

Note: Log Service does not support projects created with APIs or SDKs.

Prerequisites

Before using Log Service, you must activate Log Service and create an Access Key.

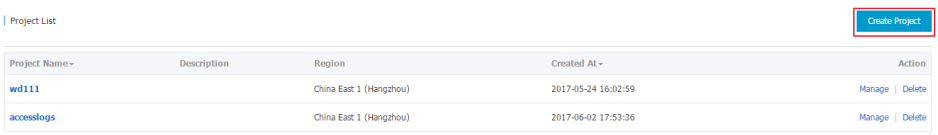
Considerations

- The project name must be globally unique among all Alibaba Cloud regions.
- To create a project, you must specify the Alibaba Cloud region based on the source of the logs to be collected and other conditions. If you want to collect logs from Alibaba Cloud ECS virtual machine, we recommend that you create the project in the same region as the ECS virtual machine to speed up log collection, and make use of the Alibaba Cloud intranet. That is, the project does not occupy the public bandwidth of ECS virtual machine.
- Once a project is created, its corresponding region cannot be changed. As Log Service does not support project migration, carefully select the project region.
- A user can create up to 10 projects in all Alibaba Cloud regions.

Procedure

Log on to Log Service console.

Click **Create Project** in the upper-right corner.

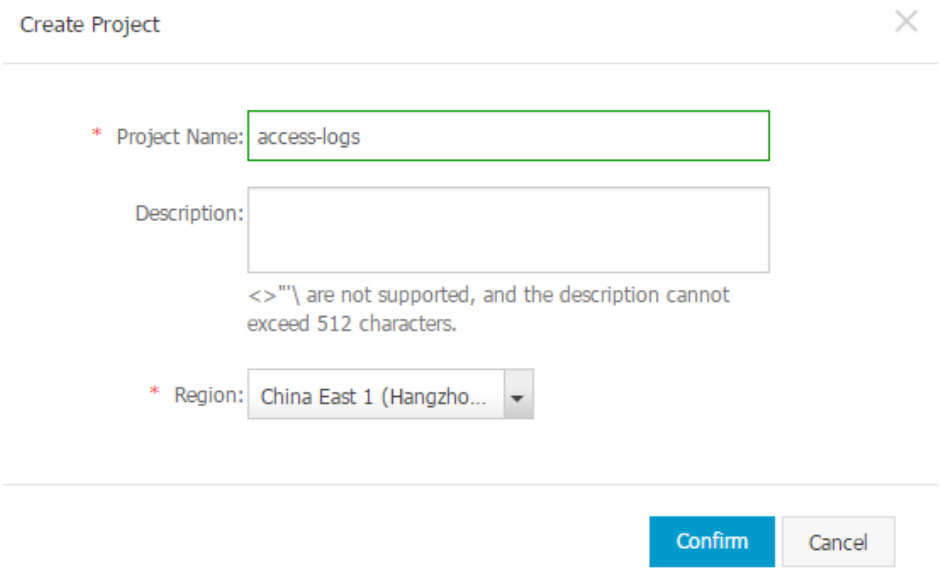


Enter the **Project Name** and **Region**, and click **Confirm**.

Project Name: Specify a project name. The name can be 3 to 63 characters in length and can contain lowercase letters, digits, and hyphens (-). It must begin and end with a lowercase letter or digit.

Note: The project name cannot be modified once the project is created.

Region: You need to specify an Alibaba Cloud region for each project. The region cannot be modified after creation, and the project cannot be migrated between regions.



Additional operations

After a project is created, the system will prompt you to create a Logstore. You can click **Create** to create the Logstore, or click **Cancel** to access the project list page. For information about how to create a Logstore, refer to Create Logstore.

you can also View project list, Manage projects and and Delete a project.

You can view your Log Service project list.

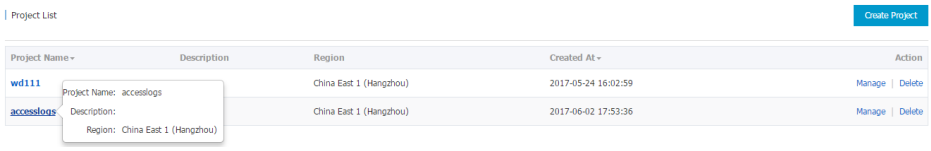
Log on to the Log Service console.

The **Project List** page lists all projects of the current Alibaba Cloud account.

For projects in the project list, you can:

- Manage a project: Click the project name or **Manage** on the right of the desired project list to perform Logstore management and Machine group management.
- Delete a project: Click **Delete** on the right of the desired project list.

You can also hover the cursor over the project name to display more details about the project.



A project is a resource management unit of Log Service. You can manage logstores and machines that you want to collect logs from through projects.

Note: Log Service does not support project management through APIs or SDKs. You can only perform project management through the Log Service console.

The following project management operations are available:

Create or delete the Logstore of a project.

A Logstore is a log storage unit of Log Service, and is used to store a certain category of logs. In actual usage, a project may need to collect various categories of logs, for example, access logs of the frontend Web server, application logs generated by backend applications, and more. Therefore, you need to create separate Logstores for different log categories in the project.

Manage associated machines for log collection.

Logs can be generated from different log sources. The most common log source is a server. In a project, the machines for log collection are managed in machine groups. You can create or delete a machine group in the project, and the machines with the same IP address are allocated into the same machine group. A common practice is to add all log sources that need to be written to the same Logstore (for example, all frontend Web servers) into one machine group for easier management.

In certain cases (for example, if you need to disable Log Service, or to destroy all the logs in a project), you may need to delete an entire project. Log Service supports the deletion of projects through the Log Service console.

Note: When deleting a project, you must first permanently release all the logs in it and its configuration. These procedures are irreversible, and data will be irrecoverable. Therefore, proceed with caution when deleting a project.

Procedure

Log on to the Log Service console.

In the project list, select the project to be deleted.

Click **Delete** on the right.

Click **Confirm** in the pop-up dialog box.

Logstore

You can create a Logstore on the Log Service console or through APIs.

For details about creating a Logstore through APIs, refer to Create LogStore.

Considerations

A Logstore must be created under a specified project.

Up to 100 Logstores can be created for each Log Service project.

The Logstore name must be unique in the project to which it belongs.

The retention period for logs can be modified after a Logstore is created.

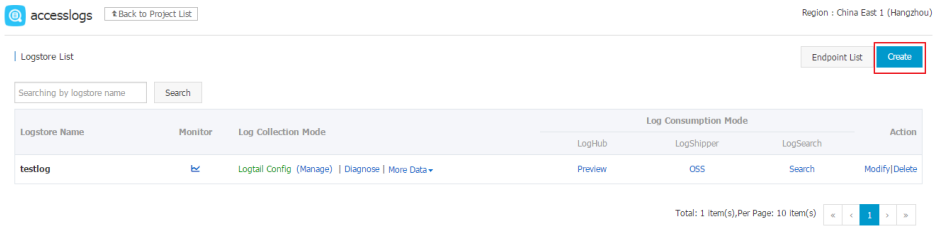
To change the retention period, click **Modify** in the Action column on the Logstore List page, modify **Data Retention Time**, and click **Modify**.

Procedure

Log on to Log Service console.

Select the desired project, click the project name or click **Manage** on the right.

Click **Create**.



Input the configuration information for the Logstore and click **Confirm**.

Logstore name: Logstore name can be 3 to 3 characters in length and can contain lowercase letters, digits, and hyphens (-). The name must begin and end with a lowercase letter or digit.

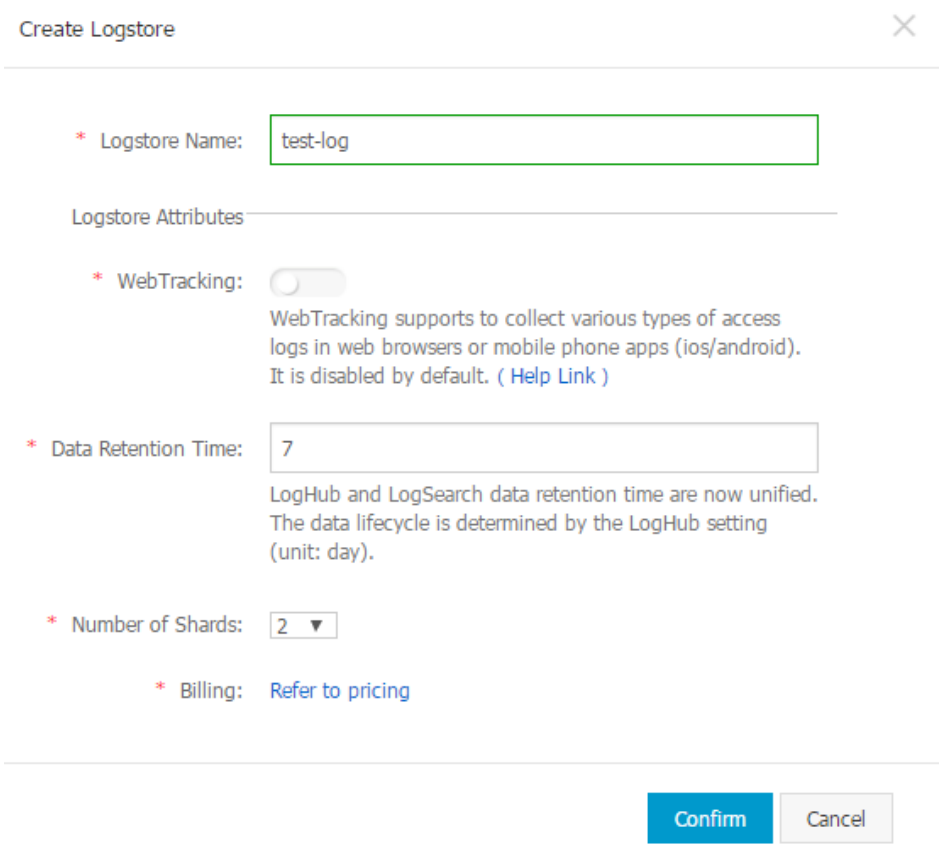
Note:

- The Logstore name must be unique in the project to which it belongs.
- The Logstore name cannot be modified after creation.

WebTracking: Whether to enable the web tracking function. This function supports collecting data from HTML, H5, iOS, or Android platform and customizing dimensions and metrics in the Log Service.

Data retention time: The number of days data should be retained in the Logstore. It can be 1-90 days. Data not set within the time period will be deleted.

Number of shards: The number of shards in the Logstore. The maximum number of shards is 10.



Additional operations

After the Logstore is created, you will be prompted to create Logtail configuration. Click **Create Logtail Config** or click **Cancel** to access the **Logstore List** page.

You can view your Logstore list on the Log Service console.

Operating procedure

Log on to the Log Service console.

Select a project, and click the project name or click **Manage** on the right.

Further operations

You can perform the following operations through the Logstore list.

Click the Monitor icon to view the Logstore monitoring information.

In the Log Collection Mode column, click **Manage** in Logtail Config to create Logtail configuration, view the created Logtail configuration in the current Logstore, or modify the existing Logtail configuration.

Click **Diagnose** to view the log collection errors.

In the Log Collection Mode column, click **More Data**, and then click **Logstash**, **API**, or **SDK** in the drop-down menu to view the Logstash document, API document, or SDK document of Log Service.

In the LogHub column, click **Preview** to view the log information.

Click **Search** under the LogSearch column in the list to access the log search page. For details, refer to Log search.

In the LogShipper column, click **OSS** to enable and set OSS log shipping. For more information, refer to Ship logs to an OSS bucket.

Click **Modify** in the Action column to modify the log data retention time and the number of shards in the Logstore.

Click **Delete** to the right of the Logstore list to delete the corresponding Logstore.

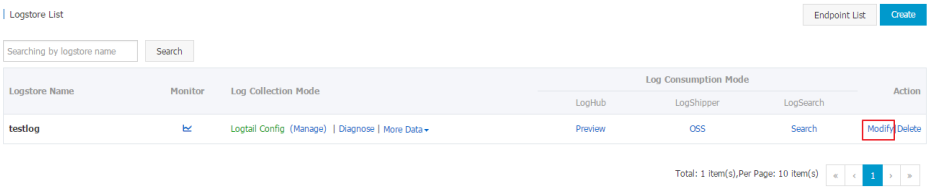
After a Logstore is created, you can modify the Logstore configuration as needed.

Procedure

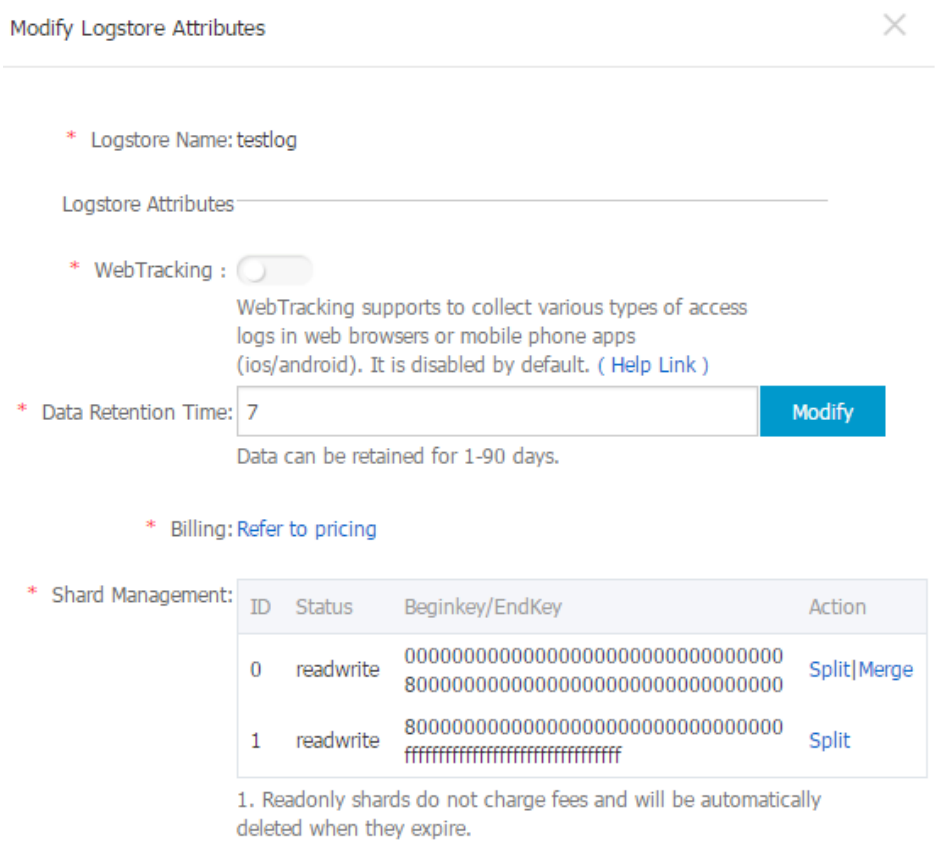
Log on to the Log Service console.

Select the desired project, and click the project name or **Manage** on the right.

On the **Logstore List** page, select the desired Logstore, and click **Modify** in the Action column.



In the displayed dialog box, modify the Logstore configuration, and then close the dialog box.



You can delete Logstores you no longer require through the Log Service console.

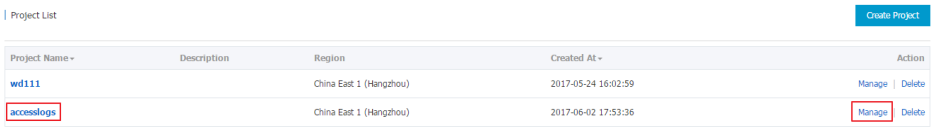
Considerations

- Once a Logstore is deleted, all log data stored in it will be deleted and irrecoverable. Take caution when performing this operation.
- To delete a specified Logstore, all its corresponding Logtail configuration must be deleted first.

Procedure

Log on to the Log Service console.

Select the desired project, and click the project name or **Manage** on the right.



On the **Logstore List** page, select the Logstore to be deleted and click **Delete** on the right.

In the pop-up dialog box, click **Confirm**.

Shard

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 ranges of all intervals equal the entire MD5 value range.

Range

When creating a Logstore, the entire MD5 range will be automatically divided into even sets, based on the specified number of shards. Each shard has a certain range within the range of [00000000000000000000000000000000,ffffffffffffffffffffffffffffffff).

A shard is an MD5 left-closed, right-open interval, and is composed of the following keys.

- BeginKey: The range key is inclusive of the shard range, indicating the start of the shard.
- EndKey: The range key is exclusive of the shard range, indicating the end of the shard.

Shard range is used to support writing logs by specifying hash key, as well as for shard split and merge operations. When reading data from a shard, the corresponding shard must be specified. When writing data, you can use load balancing mode or specify hash key. In load balancing mode, each packet is written to an available shard at random. By specifying hash key, data is written to the shard whose range includes the specified key.

Assume that the MD5 value range of a Logstore is from 00 to ff, and it has a total of 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 5F, this request will go to Shard1. If you specify the MD5 key 8C, this request will go to Shard2.

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 split or merge these shards to increase or decrease the number.

- When writing logs, if the API consistently reports error code 403 or 500, use the Logstore CloudMonitor 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 runs according to your required services, however, the service quality cannot be guaranteed as being optimal.

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 shards
- Merge shards
- Delete a shard

Each shard can write data at a rate of 5 MB/s and read data at a rate of 10 MB/s. You can split a shard when read/write operations exceed a shard’ s service capacity. A shard can also be resized through the split operation.

Guidelines

To perform the split operation, specify a shardId in read/write status and a MD5. Note that the MD5

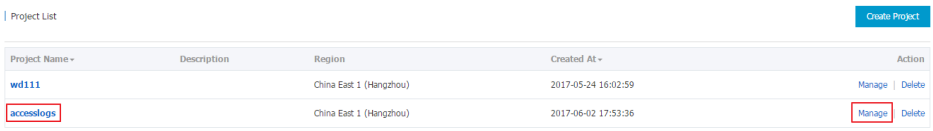
must be greater than the BeginKey of the shard, but less than the EndKey of the shard.

The split operation divides one shard into two additional shards while retaining the original shard. After the split operation is completed, the original shard changes to read-only status. The data in it can still be consumed, but no data can be written into it. The two additional shards are generated in read/write status, are lined after the original shard, and their MD5 range overwrites that of the original shard.

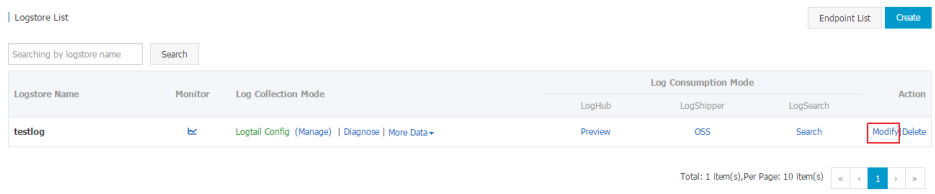
Procedure

Log on to the Log Service console.

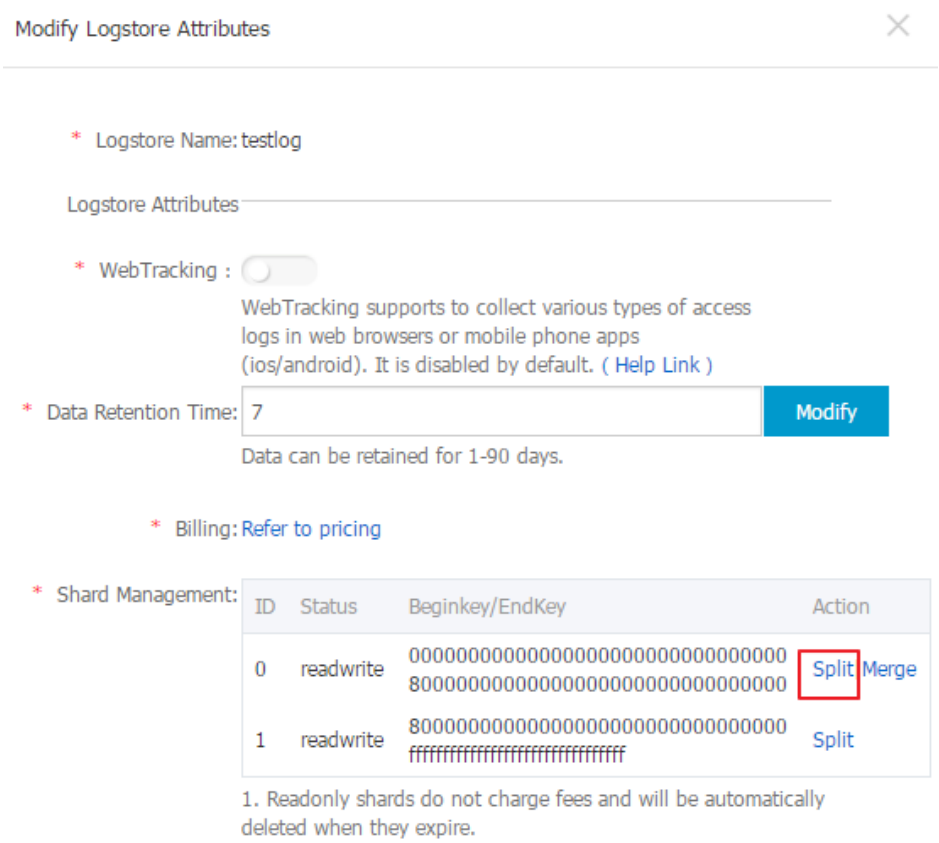
Select the desired project, and click the project name or **Manage** on the right.



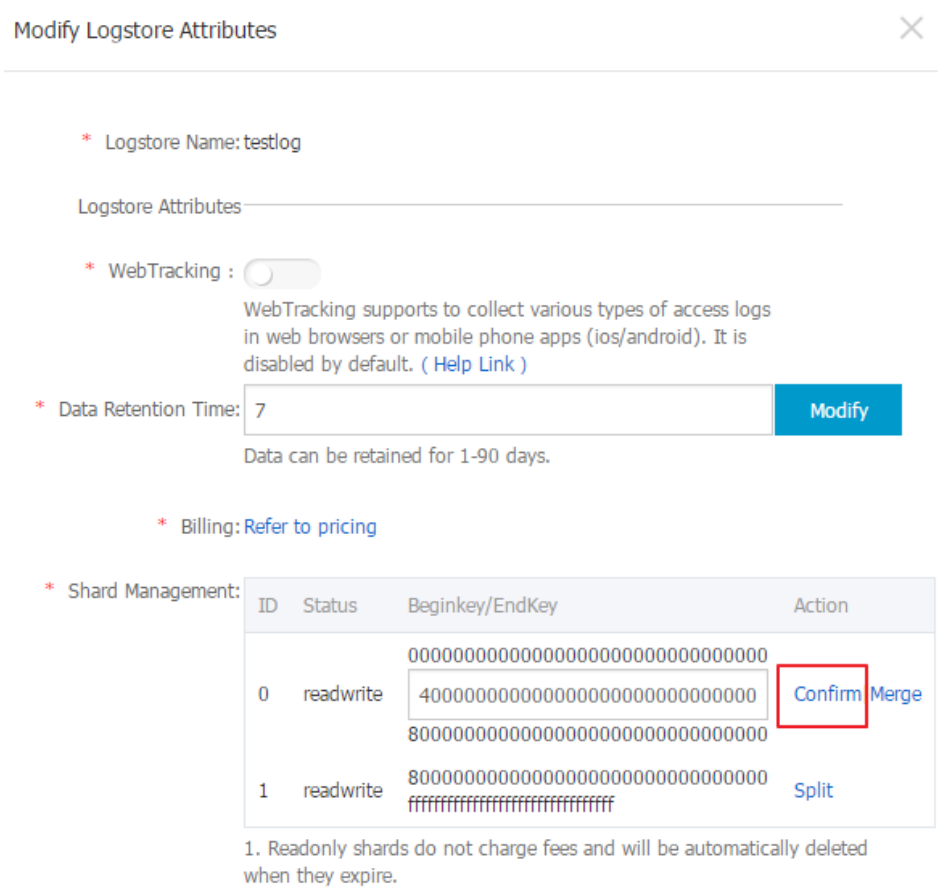
On the **Logstore List** page, select the desired Logstore, and click **Modify** in the Action column.



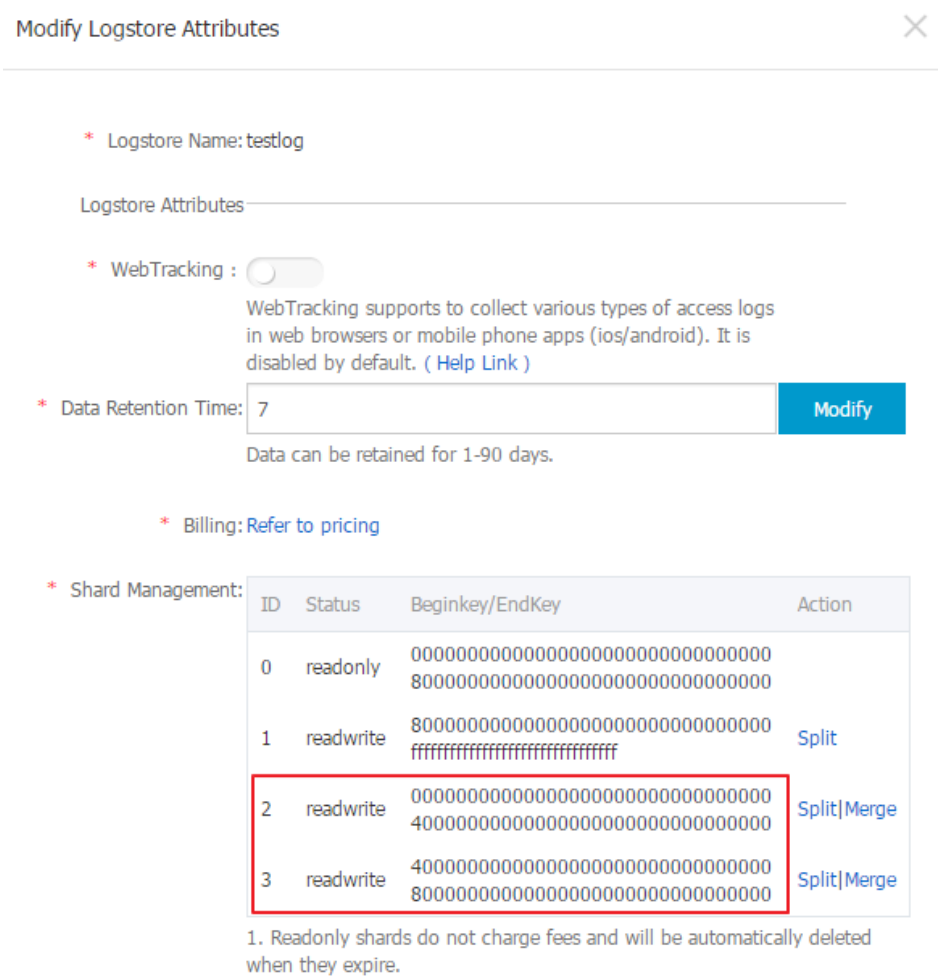
Select the shard to split, and click **Split** on the right.



Click **Confirm** and close the dialog box.



Two shards in read/write status are generated, and their MD5 range overwrites that of the original shard.



You can reduce the number of shards using the merge operation. The merge operation combines the range of the specified shard with that of the adjacent shard to its right. The generated shard’ s MD5 range overwrites that of the original two shards.

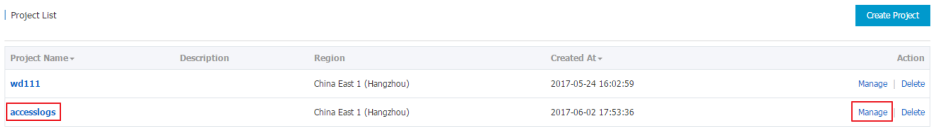
Guidelines

To perform the merge operation, specify a shardId in read/write status. Note that the specified shard cannot be the last shard in read/write status. The server will automatically find the adjacent shard to the right of the specified shard, and combine the range of these two shards. After the merge operation is completed, the specified shard and the adjacent shard on the right turn to read-only status. The data in them can still be consumed, but no data can be written into them. One additional shard is generated in read/write status, and its MD5 range overwrites that of the original two shards.

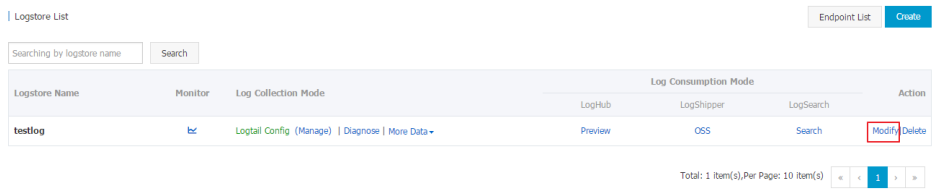
Procedure

Log on to the Log Service console.

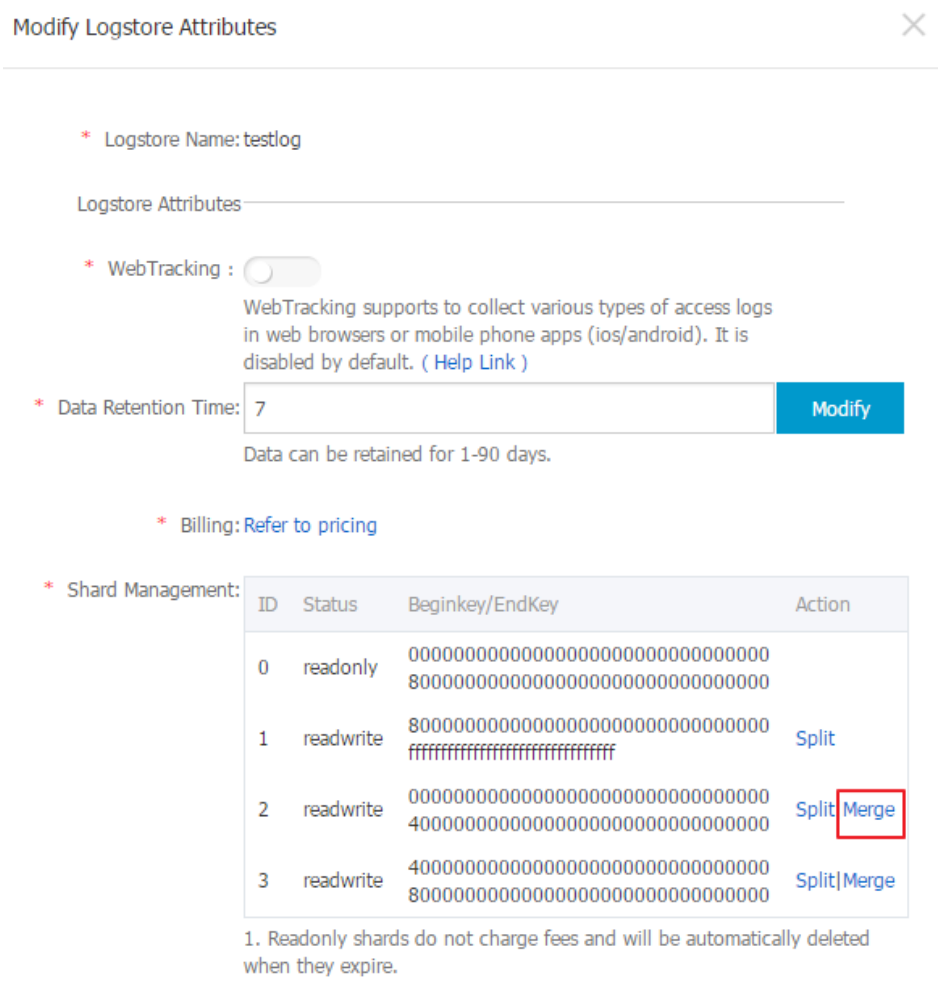
Select the desired project, and click the project name or **Manage** on the right.



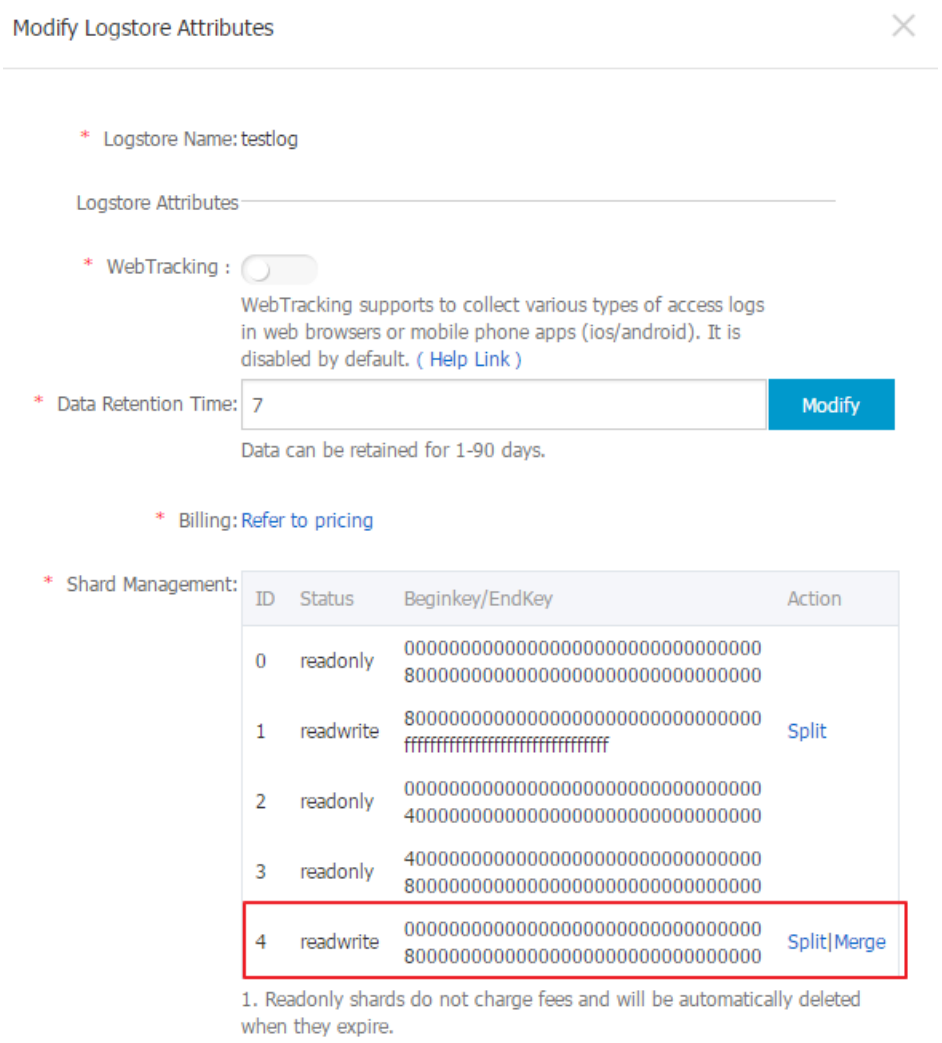
On the **Logstore List** page, select the desired Logstore, and click **Modify** in the Action column.



Select the shard to be merged, and click **Merge** on the right.



After the merge operation is completed, the specified shard and the adjacent shard to the right turn to read-only status, and the MD5 range of the generated shard in read/write status overwrites that of the original two shards.



The lifecycle, that is, the data retention time of the Logstore can be 1-90 day(s). Shards and data in the Logstore will be automatically deleted when the lifecycle expires.

Real-time collection

LogHub supports multiple collection methods, such as client, Web pages, protocol, and SDKs/APIs (mobile devices and games), and all collection methods were based on Restful API. Otherwise, you can develop new collection mehtods through API/SDK.

Through client

- Logtail：LogHub built-in Agent

    - Completly support all Windows/Linux platforms, centralization management, and web control. High-performance/low-consumption.
- Support multiple analytical modes：Simple Mode，Json Mode, Delimiter Mode, Full Mode. Common example of configuration：Nginx, Apache, Log4J, Wordpress, Python, NodeJS, Delimiter (such as CSV, TSV), Logstash, ThinkPHP.
    - Support Rsyslog protocal。

- Logstash

Docker

- Alibaba Cloud Container Service
- other dockers
    - collect by Logtail： refer to Use Logtail to collect logs/Docker section
    - collect by Docker Driver： integrated complete Docker code, refer to Alilog section in the Docker/Deamon/Driver for details, Driver-related Commit
    - FluentBit (provided by clients)

Through SDK/API/mobile terminal

- Java
- LogHub Producer Library (Java): Client writes in high-concurrency situations
- Log4J Appender：Log4J Appender packaged based on the LogHub Producer Library
- Python
- PHP
- C#
- Go
- Native C
- NodeJs
- C++
- Android
- iOS
- JS/Web Tracking：anonymously collect data

Alibaba Cloud products

- Elastic computing service (ECS) log：collect by Logtail
- Alibaba Cloud Container Service
- Message and Notification Service (MNS) logs
- Coming soon：OSS, CDN, API Gateway

Log Service provides access points to three network types: intranet, private network, and the Internet. This allows for easy access and convergence of data in hybrid network environments.

Each region is provided with three access points:

- Intranet (classic network): Can be accessed by services within the same region. This type of access point is recommended for optimal bandwidth link quality.
- Internet (classic network): Can be accessed by any service. The access speed varies with link quality. HTTPS is recommended for transmission security.
- Private network (VPC): Can be accessed through VPC by services in the same region.

Example

Assume that Log Service is created in China East 1 (Hangzhou). The following table shows access to Log Service from different types of networks.

Data source	Network access type	Network solution
China East 1 (Hangzhou)	ECS classic network	Intranet
China East 1 (Hangzhou)	ECS VPC	Private network
China East 2 (Shanghai)	ECS classic network/VPC	Internet (HTTPS)
IDC	Leased line connected to China East 1 (Hangzhou)	Intranet
IDC		Internet (HTTPS)

Access reference

- Logtail: Refer to Install Logtail for Windows and Install Logtail for Linux.
    - To collect logs on a non-ECS server, you need to place a user ID (through the touch command) on the server to indicate the server owner. For details, refer to Configure user IDs for non-Alibaba Cloud ECS.
    - In VPC, if machine groups cannot be identified by IP addresses, use user-defined IDs to manage machines. The IDs are used to group machines as web-server and app-server with support of automatic resizing.
- API/SDK: Configure access points.
- Tracking Pixel/Android/IOS SDK: Access Log Service through the Internet.
- Syslog: Refer to the above Logtail description.

Use Logtail to collect logs

The Logtail access service facilitates quick retrieval of logs from servers.

Logtail configuration

Logtail configuration describes how to collect logs on machines and send the collected logs to the specified Logstore of Log Service. The Logtail configuration format is the same in Windows and Linux, but some configuration items vary between the two platforms. The following table lists the differences in configuration items in Windows and Linux.

Configuration Item	Description
Log path	Indicates the root directory of collected log files. The path must be an absolute path without wildcards.
Log file name	Indicates the name of a collected log file. The name is case-sensitive and may contain wildcards, for example, *.log. The file name wildcards in Linux include *, ?, and […]. In Windows, MS-DOS and Windows wildcards are supported, for example, *.doc and readme.???.
Local storage	Indicates whether to enable the local cache to temporarily store logs that cannot be sent due to short-term network interruptions.
First-line log header	Indicates the starting line of a multiline log by means of a regular expression. Lines cannot be used to separate individual logs in multiline log collection (for example, application logs with stack information). You need to specify a starting line to delimit multiline logs. Because the starting line (for example, timestamp) of each log may be different, you need to specify a starting line match rule. A regular expression is used as a match rule here.
Log parsing expression	Indicates how to extract a piece of log information and convert it into a log format supported by Log Service. You need to specify a regular expression to extract the required log fields and then name each field. For details, refer to Sample.
Log time format	Defines how to parse the time format of the timestamp string in log data. For details, refer to Logtail log time format.

Process

The Log Service console provides easy access to Logtail.

Install Logtail.

Configure user ID for non-Alibaba Cloud ECS.

Create Logtail machine group.

Create Logtail machine group and apply the Logtail configuration to Logtail machine groups.

You can refer to Sample for details about configuring log extraction rules in Logtail configurations.

After the preceding process is completed, logs of a specific type on the ECS servers are collected and sent to the selected Logstore. (Historical logs are not collected. For details, refer to the **Basic functions**.) You can view the collected logs through the Log Service console or SDKs and APIs. To check whether collection is normal or whether an error occurs, log on to the Log Service console to view the Logtail collection status on each ECS server.

For details about how to use the Logtail access service on the Log Service console, refer to Collect logs by Logtail.

Docker

- Alibaba Cloud Container Service
- ECS/IDC self-built Docker (The log directories in containers must be mounted to the host machine.)
    i. Install Logtail for Windows or Install Logtail for Linux.
    ii. Mount the log directories in containers to the directory on the host machine.
        - Method 1: Use the following command (For example, the directory on the host machine is /log/webapp, and the directory in a container is /opt/webapp/log.)

```
docker run -d -P --name web -v /src/webapp:/opt/webapp training/webapp python app.py
```

        - Method 2: Use orchestration template

Features

Non-invasive log collection based on log files.

You do not need to modify any application code, and log collection does not affect the operating logic of your applications.

Exception handling during the log collection process in a stable manner.

Logtail takes data security measures (such as proactive retry and local caching) when the network or Log Service has an exception or when user data temporarily exceeds the reserved write bandwidth limit.

Centralized management capability based on Log Service.

After installing Logtail, you can configure the data sources, collection modes and other parameters on the client for all the servers without the need to log on to the servers and make configurations separately.

Comprehensive management mechanism.

To ensure that the collection agent running on the your machine does not significantly impact the performance of your services, Logtail protects and limits the use of CPU, memory, and network resources.

Basic functions

Currently, the Logtail access service provides the following functions.

Real-time log collection: Logtail dynamically monitors log files and reads and parses incremental logs in real-time. There is a delay of less than 3s between log discovery and transfer of logs to Log Service.

> **Note:** Logtail does not support the collection of historical data. Logs with an interval of more than 5 minutes between the read time and generation time are discarded.

Automatic log rotation processing: Many applications rotate log files according to the file size or date. During the rotation process, the original log file is renamed and a new blank log file is created with data to be written in. (For example, the monitored app.LOG is rotated to generate app.LOG.1 and app.LOG.2.) You can specify the file (for example, app.LOG) to which collected logs are written. Logtail automatically detects the log rotation process and ensures that no log data is lost during this process.

> **Note:** Data may be lost if log files are rotated multiple times within several seconds.

Automatic handling of collection exceptions: Logtail performs proactive retry based on specific scenarios in the case of data transfer failures due to exceptions (such as Log Service errors, network measures, and quota overruns). If retry fails, Logtail writes the data to the

local cache and then resends the data after a time.

> **Note:** The local cache is located in the disk of your server. If the cached data is not received by Log Service within 24 hours, it is discarded and deleted from the cache.

Flexible collection policy configuration: You can perform Logtail configuration to flexibly specify how logs are collected on an ECS server. Specifically, you can select log directories and files (by means of exact match or fuzzy match using wildcards) based on the actual scenario. You can customize an extraction method for log collection and set the names of extracted fields. (Log extraction by regular expression is supported.) Because the log data models of Log Service require that each log have precise timestamp information, Logtail provides custom log time formats, allowing you to extract the required timestamp information from log data of different formats.

Automatic synchronization of collection configurations: After you create or update configurations on the Log Service console, Logtail automatically accepts and applies the configurations within 3 minutes. No collected data is lost during the configuration update process.

Automatic agent upgrade: After you manually install Logtail on a server, Log Service automatically performs agent O&M and upgrade. No log data is lost during the agent upgrade process.

Status monitoring: To prevent the Logtail agent from consuming too many resources and thus affecting your services, Logtail monitors its resource (CPU and memory) consumption in real time. The Logtail agent automatically restarts when the resource usage limit is exceeded to avoid impact on the ongoing operations on the machine. The agent proactively limits network traffic to prevent excessive bandwidth consumption.

> **Note:**
>
> - Log data may be lost when the Logtail agent restarts.
> - If the Logtail agent is exited due to an exception of its processing logic, the corresponding protective mechanism is triggered and the agent is restarted to continue log collection. However, log data may be lost before restart.

Transferred data signature: To prevent data tampering during the transfer process, the Logtail agent proactively obtains your Alibaba Cloud access key to sign all log data packets before they are sent.

> **Note:** The Logtail agent obtains your Alibaba Cloud access key over HTTPS to ensure key security.

Core concepts

Machine group: A machine group contains one or more machines on which logs of a specific type are collected. After a set of Logtail configurations is applied to a machine group, Logtail collects logs of the specified type on all machines in the machine group according to the same Logtail configuration. You can create and delete machine groups as well as add and remove machines in machines groups through the Log Service console.

> **Note:** A single machine group cannot contain both Windows and Linux machines but may have machines of different Windows Server versions or different Linux releases.

Logtail configuration: Logtail configuration describes how to collect a specific type of logs on machines, parse the collected logs, and send the logs to the specified Logstore of Log Service. You can use the console to add Logtail configurations to a Logstore to enable the Logstore to receive logs that are collected based on the configurations.

Logtail agent: Logtail is the agent that runs on your machines to collect logs. After Logtail is installed on ECS servers, add the intranet IP addresses of the servers to a machine group.

- In **Linux**, the agent is installed in the /usr/local/ilogtail directory and launches two independent processes (a collection process and a daemon, whose names start with ilogtail). The program running log is /usr/local/ilogtail/ilogtail.LOG.
- In **Windows**, the agent is installed in the C:\Program Files\Alibaba\Logtail directory (for 32-bit systems) or the C:\Program Files (x86)\Alibaba\Logtail directory (for 64-bit systems). Two Windows services exist in **Windows Management Tools** > **Services**. One service is LogtailWorker for log collection and the other is LogtailDaemon. The program running log is logtail_*.log in the installation directory.

Processing capabilities and constraints

The following table lists the per-server processing capabilities and constraints of the Logtail access service.

Item	Processing capability and constraints
File encoding	UTF-8/GBK-encoded log files are supported. If log files are encoded in other formats, undefined behaviors such as garbled characters and lost data occur. UTF-8 encoding is recommended for improved processing performance.
Log processing throughput capacity	The raw log processing traffic is limited to 1 MB/s by default. Data is sent through the Alibaba Cloud intranet. Logs may be lost when the traffic limit is exceeded. You can set the limit to about 50 MB/s at most in

	accordance with this document.
Network error handling	Local cache storage up to 500 MB is supported. Logtail caches data locally when the network or Log Service has an exception or when user data temporarily exceeds the reserved write bandwidth limit. Logtail then retries sending as soon as possible.
Configuration update	The delay of applying updated configurations is 30s.
Status management detection	Logtail automatically restarts in the case of an exception (abnormal program exit or resource limit overruns).
Monitored directory count	Logtail proactively restricts the formats of directories that can be monitored to prevent user resource overconsumption. When the monitoring upper limit is reached, Logtail stops monitoring more directories and log files. Up to 3,000 directories (including subdirectories) can be monitored.
Soft link support	Monitored directories can be soft links.
Log file size	Unrestricted.
Single log size	The maximum size of a single log is 512 KB. Each line of a multiline log, after the pattern that delineates it, may be up to 512 KB.
Regular expression type	Perl-compatible regular expressions are used.

Supported systems

Logtail supports the following 64-bit Linux releases.

- Aliyun Linux
- Ubuntu
- Debian
- CentOS
- OpenSUSE

Install Logtail

Install Logtail in overwrite mode. By default, Logtail is launched after the installation is complete. (If you have installed Logtail before, the installer uninstalls and deletes the /usr/local/ilogtail directory before installing Logtail.)

Download the installer based on the network environment of your machine and the region of Log Service. Select different parameters for installation.

ECS of classic network

Data is written to Log Service through the Alibaba Cloud intranet without consuming public bandwidth. It is applicable to Alibaba Cloud ECS.

China North 2 (Beijing)

```
wget http://logtail-release-bj.oss-cn-beijing-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_beijing
```

China North 1 (Qingdao)

```
wget http://logtail-release-qd.oss-cn-qingdao-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_qingdao
```

China East 1 (Hangzhou)

```
wget http://logtail-release.oss-cn-hangzhou-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_hangzhou
```

China East 2 (Shanghai)

```
wget http://logtail-release-sh.oss-cn-shanghai-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_shanghai
```

China South 1 (Shenzhen)

```
wget http://logtail-release-sz.oss-cn-shenzhen-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_shenzhen
```

China North 3 (Zhangjiakou)

```
wget http://logtail-release-zjk.oss-cn-zhangjiakou-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn-zhangjiakou
```

Hong Kong

```
wget http://logtail-release-hk.oss-cn-hongkong-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn-hongkong
```

US West 1 (Silicon Valley)

```
wget http://logtail-release-us-west-1.oss-us-west-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install us-west-1
```

Asia Pacific SE 1 (Singapore)

```
wget http://logtail-release-ap-southeast-1.oss-ap-southeast-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-southeast-1
```

Asia Pacific SE 2 (Sydney)

```
wget http://logtail-release-ap-southeast-2.oss-ap-southeast-2-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-southeast-2
```

Asia Pacific NE 1 (Tokyo)

```
wget http://logtail-release-ap-northeast-1.oss-ap-northeast-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-northeast-1
```

Germany 1 (Frankfurt)

```
wget http://logtail-release-eu-central-1.oss-eu-central-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install eu-central-1
```

Middle East 1 (Dubai)

```
wget http://logtail-release-me-east-1.oss-me-east-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install me-east-1
```

ECS of VPC

Data is written to Log Service through Alibaba Cloud VPC. It is applicable to Alibaba Cloud ECS of VPC.

China North 2 (Beijing)

```
wget http://logtail-release-bj.vpc100-oss-cn-beijing.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_beijing_vpc
```

China East 1 (Hangzhou)

```
wget http://logtail-release.vpc100-oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_hangzhou_vpc
```

China East 2 (Shanghai)

```
wget http://logtail-release-sh.vpc100-oss-cn-shanghai.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_shanghai_vpc
```

China South 1 (Shenzhen)

```
wget http://logtail-release-sz.vpc100-oss-cn-shenzhen.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_shenzhen_vpc
```

China North 3 (Zhangjiakou)

```
wget http://logtail-release-zjk.oss-cn-zhangjiakou-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn-zhangjiakou_vpc
```

Hong Kong

```
wget http://logtail-release-hk.vpc100-oss-cn-hongkong.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn-hongkong_vpc
```

US West 1 (Silicon Valley)

```
wget http://logtail-release-us-west-1.vpc100-oss-us-west-1.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install us-west-1_vpc
```

Asia Pacific SE 1 (Singapore)

```
wget http://logtail-release-ap-southeast-1.vpc100-oss-ap-southeast-1.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-southeast-1_vpc
```

Asia Pacific SE 2 (Sydney)

```
wget http://logtail-release-ap-southeast-2.oss-ap-southeast-2-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-southeast-2_vpc
```

Asia Pacific NE 1 (Tokyo)

```
wget http://logtail-release-ap-northeast-1.oss-ap-northeast-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-northeast-1_vpc
```

Germany 1 (Frankfurt)

```
wget http://logtail-release-eu-central-1.oss-eu-central-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install eu-central-1_vpc
```

Middle East 1 (Dubai)

```
wget http://logtail-release-me-east-1.oss-me-east-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install me-east-1_vpc
```

Internet (self-build IDCs or other cloud hosts)

Data is written to Log Service through the Internet, which consumes public bandwidth. It is applicable to non-Alibaba Cloud virtual machines or other IDCs.

> Because Log Service cannot obtain the owner information of non-Alibaba Cloud machines, you need to manually configure a user ID after installing Logtail. For details, refer to Configure user IDs for non-Alibaba Cloud ECS. If you do not configure a user ID, Logtail will have abnormal heartbeats and fail to collect logs.

China North 2 (Beijing)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_beijing_internet
```

China North 1 (Qingdao)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_qingdao_internet
```

China East 1 (Hangzhou)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_hangzhou_internet
```

China East 2 (Shanghai)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_shanghai_internet
```

China South 1 (Shenzhen)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_shenzhen_internet
```

China North 3 (Zhangjiakou)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn-zhangjiakou_internet
```

Hong Kong

```
wget http://logtail-release-hk.oss-cn-hongkong.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn-hongkong_internet
```

US West 1 (Silicon Valley)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install us-west-1_internet
```

Asia Pacific SE 1 (Singapore)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-southeast-1_internet
```

Asia Pacific SE 2 (Sydney)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-southeast-2_internet
```

Asia Pacific NE 1 (Tokyo)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install ap-northeast-1_internet
```

Germany 1 (Frankfurt)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install eu-central-1_internet
```

Middle East 1 (Dubai)

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install me-east-1_internet
```

ECS of AntCloud

China East 1 (Hangzhou)

```
wget http://logtail-release-hz-finance.oss-cn-hzjbp-a-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_hangzhou_finance
```

China South 1 (Shenzhen)

```
wget http://logtail-release-sz-finance.oss-cn-shenzhen-finance-1-internal.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh install cn_shenzhen_finance
```

Uninstall Logtail

Refer to the **Install Logtail** section for details about downloading the **logtail.sh** installer. Run the following command as admin in shell mode.

```
wget http://logtail-release.oss-cn-hangzhou.aliyuncs.com/linux64/logtail.sh; chmod 755 logtail.sh; sh logtail.sh uninstall
```

Supported systems

Logtail supports the following systems.

- Windows 7 (Client) 32bit
- Windows 7 (Client) 64bit
- Windows Server 2003 32bit
- Windows Server 2003 64bit
- Windows Server 2008 32bit
- Windows Server 2008 64bit
- Windows Server 2012 32bit
- Windows Server 2012 64bit

Install Logtail

Download the installation package

You can download the installation package at http://logtail-release.oss-cn-hangzhou.aliyuncs.com/win/logtail_installer.zip.

Install Logtail based on the network environment of your machine and the region of Log Service

Extract logtail.zip to the current directory and start Windows PowerShell or go to the logtail_installer directory in command mode.

Region of Log Service	Network Environment of Your Machine	Installation Command
China North 2 (Beijing)	ECS of classic network	.\logtail_installer.exe install cn_beijing
	ECS of VPC	.\logtail_installer.exe install cn_beijing_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install cn_beijing_internet
China North 1 (Qingdao)	ECS of classic network	.\logtail_installer.exe install cn_qingdao
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install cn_qingdao_internet
China East 1 (Hangzhou)	ECS of classic network	.\logtail_installer.exe install cn_hangzhou
	ECS of VPC	.\logtail_installer.exe install cn_hangzhou_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install cn_hangzhou_internet
	ECS of AntCloud	.\logtail_installer.exe install cn_hangzhou_finance
China East 2 (Shanghai)	ECS of classic network	.\logtail_installer.exe install cn_shanghai

	ECS of VPC	.\logtail_installer.exe install cn_shanghai_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install cn_shanghai_internet
China South 1 (Shenzhen)	ECS of classic network	.\logtail_installer.exe install cn_shenzhen
	ECS of VPC	.\logtail_installer.exe install cn_shenzhen_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install cn_shenzhen_internet
	ECS of AntCloud	.\logtail_installer.exe install cn_shenzhen_finance
China North 3 (Zhangjiakou)	ECS of classic network	.\logtail_installer.exe install cn-zhangjiakou
	ECS of VPC	.\logtail_installer.exe install cn-zhangjiakou_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install cn-zhangjiakou_internet
Hong Kong	ECS of classic network	.\logtail_installer.exe install cn-hongkong
	ECS of VPC	.\logtail_installer.exe install cn-hongkong_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install cn-hongkong_internet
US West 1 (Silicon Valley)	ECS of classic network	.\logtail_installer.exe install us-west-1
	ECS of VPC	.\logtail_installer.exe install us-west-1_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install us-west-1_internet
Asia Pacific SE 1 (Singapore)	ECS of classic network	.\logtail_installer.exe install ap-southeast-1
	ECS of VPC	.\logtail_installer.exe install ap-southeast-1_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install ap-southeast-1_internet
Asia Pacific SE 2 (Sydney)	ECS of classic network	.\logtail_installer.exe install ap-southeast-2
	ECS of VPC	.\logtail_installer.exe install ap-southeast-2_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install ap-southeast-2_internet

Asia Pacific NE 1 (Tokyo)	ECS of classic network	.\logtail_installer.exe install ap-northeast-1
	ECS of VPC	.\logtail_installer.exe install ap-northeast-1_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install ap-northeast-1_internet
Germany 1 (Frankfurt)	ECS of classic network	.\logtail_installer.exe install eu-central-1
	ECS of VPC	.\logtail_installer.exe install eu-central-1_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install eu-central-1_internet
Middle East 1 (Dubai)	ECS of classic network	.\logtail_installer.exe install me-east-1
	ECS of VPC	.\logtail_installer.exe install me-east-1_vpc
	Internet (self-build IDC or other cloud hosts)	.\logtail_installer.exe install me-east-1_internet

Note: Because Log Service cannot obtain owner information of non-Alibaba Cloud machines, after Logtail installation, you need to manually configure a user ID when using Logtail on a self-built IDC or other cloud hosts. For details, refer to Configure user IDs for non-Alibaba Cloud ECS. If you do not configure a user ID, Logtail will have abnormal heartbeats and fail to collect logs.

Uninstall Logtail

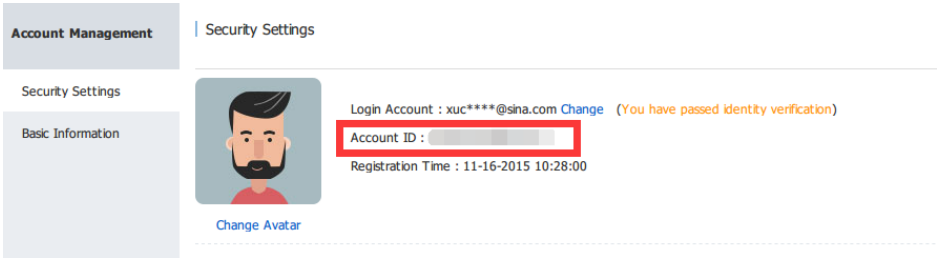
Start Windows PowerShell or go to the logtail_installer directory in command mode and run the following command.

```
.\logtail_installer.exe uninstall
```

Follow the steps below to configure user IDs (account IDs) when you install Logtail on non-Alibaba Cloud ECS; otherwise, Logtail will have abnormal heartbeats and cannot collect data to Log Service.

Step 1 View your Alibaba Cloud account ID

Log on to the Alibaba Cloud account management page to view the account ID of your Log Service project.



Step 2 Log on to machines to configure the account ID file

Create a file named after the account ID in the /etc/ilogtail/users directory. If the directory does not exist, you can create it manually. Additionally, multiple account IDs can be created on a single machine.

Linux

```
touch /etc/ilogtail/users/1559122535028493
touch /etc/ilogtail/users/1329232535020452
```

Delete the user ID when you do not require Logtail to collect data to your Log Service project.

```
rm /etc/ilogtail/users/1559122535028493
```

Windows

The process is the same as that in Linux. Create a file named after the account ID in the C:\LogtailData\users directory. Delete the user ID when you do not require data collection by Logtail.

C:\LogtailData\users\1559122535028493

Note:

- After the account ID is configured on a machine, the cloud account has the permission to collect logs on the machine by using Logtail. Clear any unnecessary account ID files from machines in a timely manner.
- After a user ID is added or deleted, the latest configuration takes effect for Logtail in no more than 1 minute.

Log Service manages all ECS servers where logs need to be collected by Logtail in the form of machine groups.

After configuring Logtail, you can click **Create Machine Group** on the **Apply to machine group** page

to create a machine group. Additionally, in the Log Service project list, you can go to the **Machine Groups** to create a machine group.

A machine group is defined by either of the following two items:

- IP address: Defines the name of the machine group and adds the intranet IP addresses of the machines in the group.
- ID: Indicates membership of the machine group, and is associated with the IDs configured on corresponding machines.

Procedure

After configuring Logtail, click **Create Machine Group** on the **Apply to machine group** page to create a machine group.

Alternatively, on the **Project List** page of Log Service, click the project name. Then, on the left-side navigation bar, click **LogHub - Collect** -> **Logtail Machine Group** to go to the project’ s **Machine Groups** page. There, click **Create Machine Group**.

Enter the **Group Name**.

The name can be 3 to 128 characters in length and can contain lowercase letters, digits, hyphens (-), and underscores (_). It must begin and end with lowercase letters or digits.

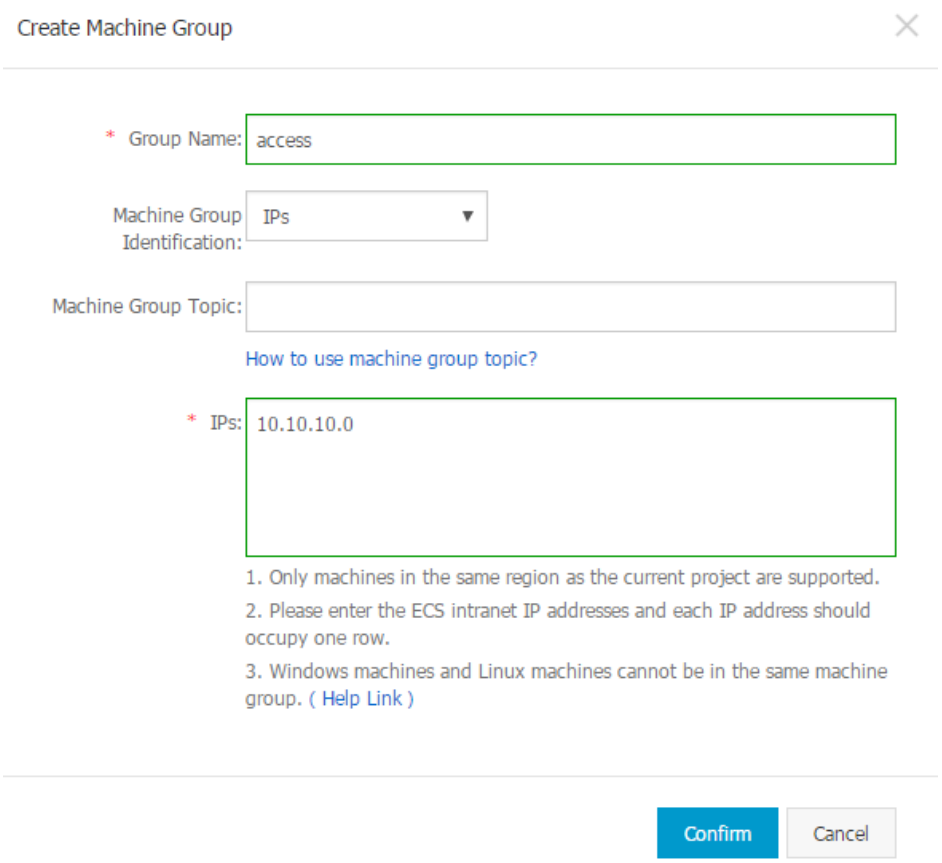
Select the **Machine Group Identification**.

IPs

After the option is selected, you need to enter the ECS instance’ s intranet IP address in the **IPs** field.

Note:

- Confirm the entered ECS instance belongs to the current Alibaba Cloud account.
- Confirm the entered ECS instance is in the same Alibaba Cloud region as the current Log Service project.
- Use the ECS instance’ s intranet IP address (not Internet IP address) and use line breaks to separate multiple IP addresses.
- Do not add Windows and Linux ECS instances to the same machine group.



User-defined Identity

After this option is selected, you must enter your custom ID in the **User-defined Identity** field.

For more information on custom IDs, refer to Use custom IDs and Create a machine group using IDs.

Enter the **Machine Group Topic**.

Click **Confirm**.

You can now view the created machine group on the machine group list.

More operations

After creating a machine group, you can view the machine group list, modify the machine groups, view statuses, manage configurations, and delete machine groups. For more information, refer to Logtail machine groups.

After a launch, Logtail reports machine IDs to Log Service. Logtail only operates correctly when the IDs reported by the agent are the same as the IDs within the machine group.

Application scenarios

- By default, Logtail identifies machines by their IP addresses. The IP addresses of different machines may conflict in custom network environments (for example, VPC). This can cause Logtail management failure in Log Service.
- Multiple machines use the same user-defined ID for automatic log collection during resizing.

Enable user-defined ID

Linux Logtail

Set the user-defined ID in the /etc/ilogtail/user_defined_id file.

For example, set a user-defined machine ID as follows:

```
#cat /etc/ilogtail/user_defined_id
aliyun-ecs-rs1e16355
```

Windows Logtail

Set the user-defined ID in the C:\LogtailData\user_defined_id file.

For example, set a user-defined machine ID as follows:

```
C:\LogtailData>more user_defined_id
aliyun-ecs-rs1e16355
```

Add aliyun-ecs-rs1e16355 to a machine group. The configuration takes effect in 1 minute.

Note: If the /etc/ilogtail/, C:\LogtailData directory, /etc/ilogtail/user_defined_id, or C:\LogtailData\user_defined_id file does not exist, create it manually.

Disable user-defined ID

If you want to reuse IP addresses to identify machines, delete the user_defined_id file.

Linux Logtail

```
rm -f /etc/ilogtail/user_defined_id
```

Windows Logtail

```
del C:\LogtailData\user_defined_id
```

Effective time

After you add, delete, or modify the user_defined_id file, the latest configuration takes effect for Logtail in 1 minute or less.

If you want to apply the configuration instantly, restart Logtail.

Linux Logtail

```
/etc/init.d/ilogtaild stop
/etc/init.d/ilogtaild start
```

Windows Logtail

Restart LogtailWorker windows service.

The Logtail client can help Log Service users collect logs from ECS instances through the console.

After creating a Logstore, the system will prompt you to create Logtail configuration. In the pop-up box, click **Create Logtail Config**. In addition, you can also create Logtail configuration on the **Logstore List** page.

Prerequisites

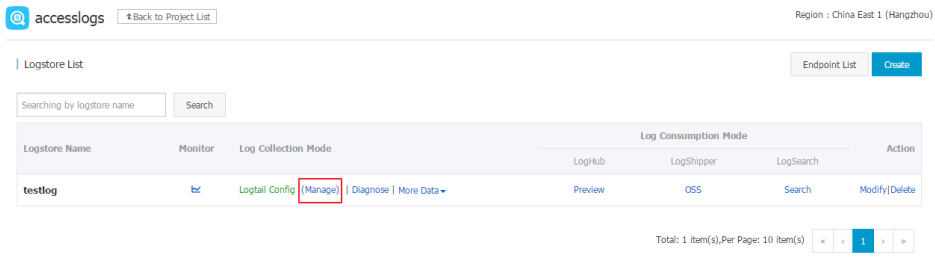
You must install Logtail before using it to collect logs. Logtail supports Windows and Linux operating systems. For installation methods, refer to Install Logtail on Windows and Install Logtail on Linux.

Constraints

- A single file can only be collected using one configuration.
- Fot details of operating system versions, refer to Using Logtail to write logs.

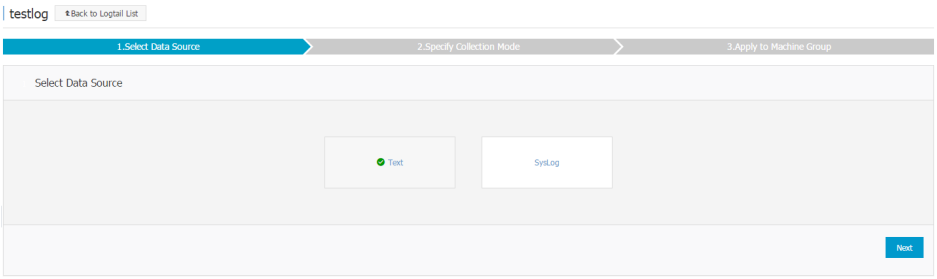
Procedure

Log on to the Log Service console. Select the desired project, and click **Manage**.



Click **Create** in the upper-right corner.

Select **text** as the data source type and click **Next**.



Specify the **Configuration Name**.

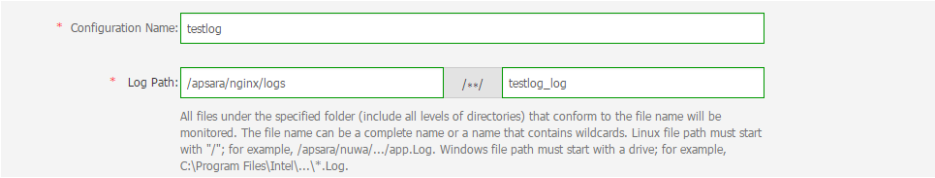
The configuration name can be 3 to 63 characters in length, and can contain lowercase letters, digits, hyphens (-), and underscores (_). It must begin and end with lowercase letters or digits.

Note: Once the configuration name is set, it cannot be modified.

Specify the log directory structure.

All files under the specified folder (including all levels of the directories) conforming to the file name will be monitored. The log file name must be a complete file name or a name that contains wildcards. Linux file paths must start with “/ “; for example, /apsara/nuwa/.../app.Log. Windows file paths must start with a drive; for example, C:\Program Files\Intel\...*.Log.

Note: A single file can only be collected by one configuration.



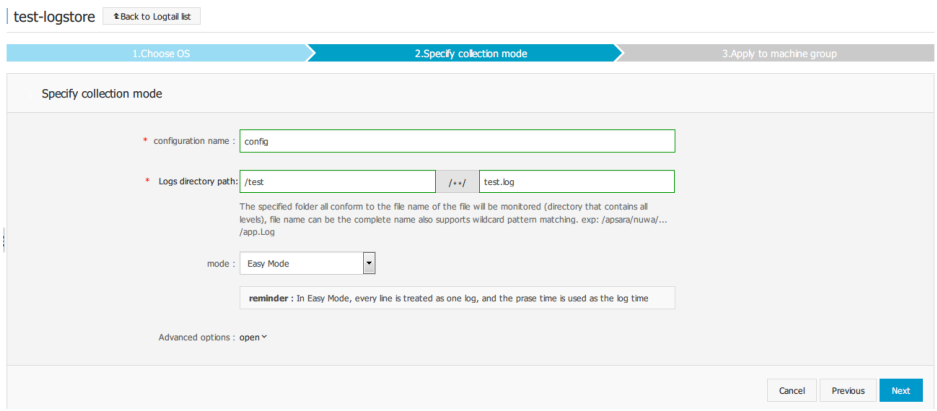
Set the collection mode.

Logtail supports simple mode, delimiter mode, JSON mode, regular expression mode, and other log collection methods. For details, refer to Collection modes. In this example, simple mode and regular expression mode are used to introduce the collection mode settings.

Simple mode

In simple mode, namely single line mode, one line of data is treated as a log by default. Two logs are separated by a line break. In single line mode, the system does not extract fields (that is, the regular expression is (.*) by default), and uses the system time of the current server as the log generation time. If, you need to use more detailed settings, you can change the configuration to regular expression mode and configure all the settings. For details about how to modify the Logtail configuration, refer to Logtail configuration.

In simple mode, you only need to specify the file directory and file name, Logtail will collect a log line by line. Logtail will not extract fields from the log content. In addition, the log time is set to the time the log was captured.



Full mode

If you need to configure more personalized extraction settings for log contents (for example, cross-row logs or field extraction), select **Full mode**. For details on the specific meanings and setting methods for these parameters, refer to Logtail log collection parameters .

Enter the **Log Sample**.

The purpose of providing a log sample is to facilitate the Log Service console in automatically extracting the regular expression matching mode in logs. Be sure to use a log from the actual environment.

Set **Singleline**.

Singleline mode is the default option. This means that the log is partitioned line by line. If you need to collect cross-line logs (such as Java program logs), you must disable **Singleline** and set **Regular Expression**.

Set **Regular Expression**.

This option provides two functions: automatic generation and manual input. After entering the log sample, click **Auto Generate** to automatically generate a regular expression. If it fails, you can switch to manual mode and input a regular expression for verification.

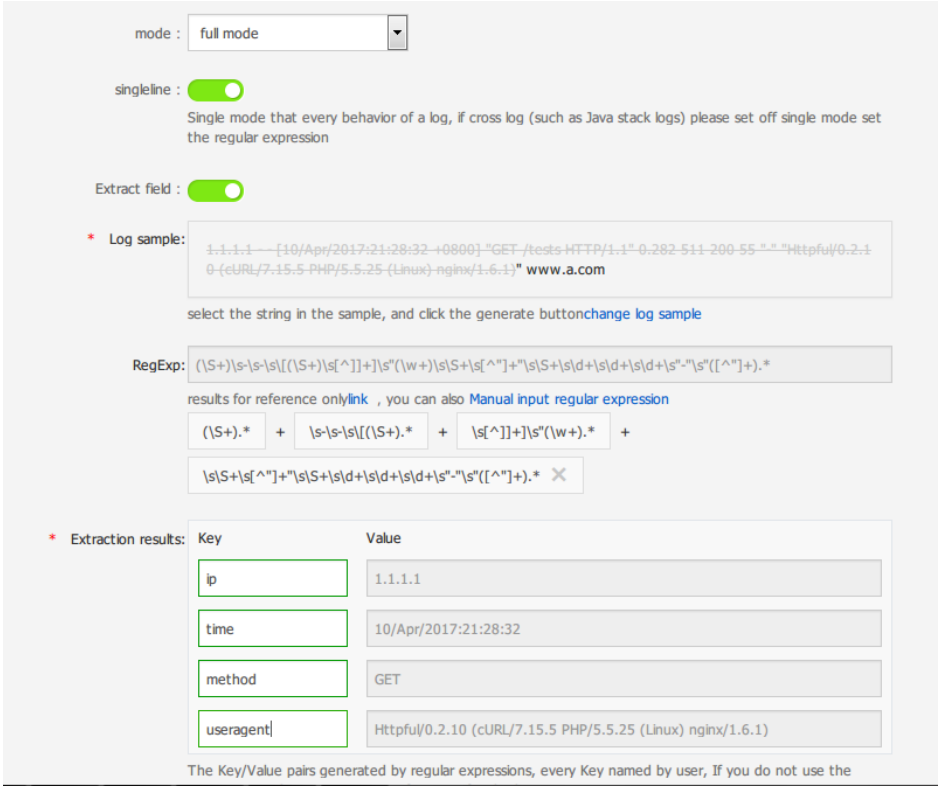
Set **Extract Field**.

If you need to analyze and process fields one by one in the log content, use the **Extract Field** function to convert the specified field into a key-value pair before sending it to the server. Therefore, you need to specify a method for parsing the log content (specifically, a regular expression).

The Log Service console allows you to specify a regular expression for parsing in two ways. The first option is to automatically generate a regular expression through simple interactions. You can use the "Drag Select" method on the log sample to indicate the fields to be extracted, and then click **Generate RegEx**, the Log Service console will automatically generate a regular expression.

Additionally, you can manually input regular expressions. You can click **Manually Input Regular Expression** to switch to manual input mode. After the expression is inputted, click **Validate** on the right to verify whether the expression can parse and extract the sample log.

Regardless of whether the regular expression for log parsing is automatically generated or manually inputted, you need to name each extracted field (that is, set the key for the field).



Set **Use System Time**.

Use System Time is set by default. If it is disabled, you must specify a certain field (value) to be used as the time field during field extraction and name this field **time**. After selecting a time field, you can click **Auto Generate** in **Time Format** to generate a method to parse this field. For more information on log time formats, refer to Logtail date format.

After configuring the settings, click **Next**.

Select the desired machine group and click **Apply to Machine Group** to apply the configuration.

If you have not created a machine group, you must create one first. For information about creating machine groups, refer to Create a machine group.

Note:

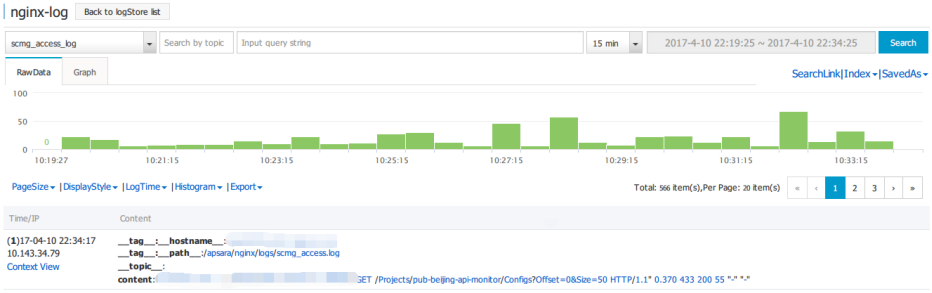
- It may take up to 3 minutes for the Logtail configuration to come into effect after being pushed.
- If you need to collect IIS access logs, you must first refer to the IIS log collection best practices to configure IIS.
- After creating Logtail configuration, view the Logtail configuration list where you can modify

or delete Logtail configurations. For details, refer to Logtail configuration.

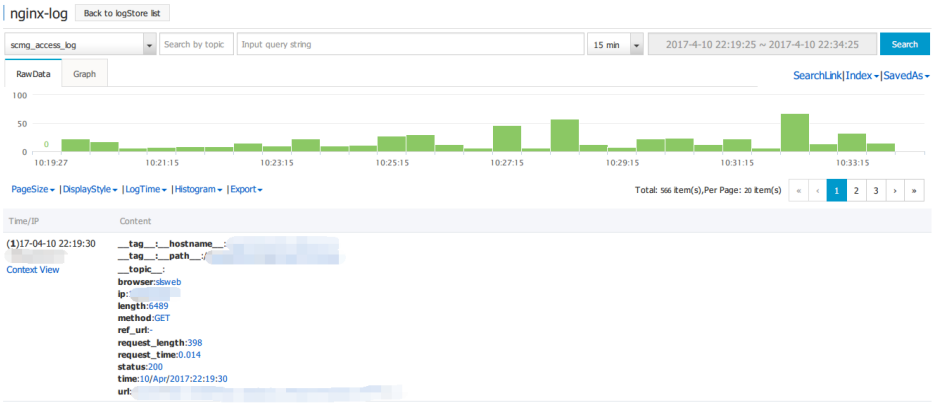
Additional operations

After the configuration is complete, Log Service can collect logs. You can view the collected logs. For information about log queries, refer to Query logs.

An example of a log collected on the server in simple mode is shown below. All log contents are displayed under the key named **content**.



An example of log content collected on the server in regular expression mode is shown below. The log contents are collected on the server according to the set key-value.



More information

Logtail configuration

Configuration Item	Description
Log path	Indicates the root directory of collected log files. The patha can be a complete file name or a name that contains wildcards.
Log file name	Indicates the name of a collected log file. The name is case-sensitive and may contain wildcards, for example, *.log. The file name wildcards in Linux include *, ?, and [...]. In Windows, MS-DOS and Windows wildcards are supported, for example, *.doc and readme.???.

Local storage	Indicates whether to enable the local cache to temporarily store logs that cannot be sent due to short-term network interruptions.
First-line log header	Indicates the starting line of a multiline log by means of a regular expression. Lines cannot be used to separate individual logs in multiline log collection (for example, application logs with stack information). You need to specify a starting line to delimit multiline logs. Because the starting line (for example, timestamp) of each log may be different, you need to specify a starting line match rule. A regular expression is used as a match rule here.
Log parsing expression	Indicates how to extract a piece of log information and convert it into a log format supported by Log Service. You need to specify a regular expression to extract the required log fields and then name each field. For details, refer to Sample.
Log time format	Defines how to parse the time format of the timestamp string in log data. For details, refer to Logtail log time format.

Other collection mode

In addition to using Logtail to collect logs, Log Service also provides APIs and SDKs help you write logs.

Use API to write logs

Log Service provides RESTful APIs to help you write logs. You can use the PostLogStoreLogs interface to write data. For a complete API reference, refer to API Reference.

Use SDKs to write logs

In addition to APIs, Log Service also provides SDKs in multiple languages (Java, .NET, PHP, and Python) that allow you to easily write logs. For a complete SDK reference, refer to SDK Reference.

Process of text-file log collection

Logtail collects text-file logs based on the following process.

Specify the file path > specify the procedure for separating log lines > extract log fields > specify the log time

Log line separation method

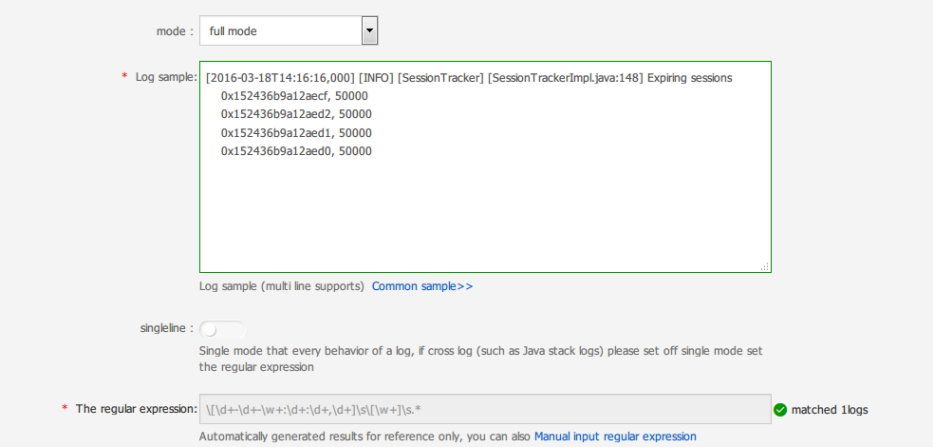
An access log (for example, Nginx access log) occupies a line. Individual logs are separated by linefeeds. Two access logs are shown as follows.

```
10.1.1.1 - - [13/Mar/2016:10:00:10 +0800] "GET / HTTP/1.1" 0.011 180 404 570 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; 360se)"
10.1.1.1 - - [13/Mar/2016:10:00:11 +0800] "GET / HTTP/1.1" 0.011 180 404 570 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; 360se)"
```

For Java applications, a program log spans several lines. The beginning of a log is used to distinguish the beginning of the line. A Java program log is shown as follows.

```
[2016-03-18T14:16:16,000] [INFO] [SessionTracker] [SessionTrackerImpl.java:148] Expiring sessions
0x152436b9a12aecf, 50000
0x152436b9a12aed2, 50000
0x152436b9a12aed1, 50000
0x152436b9a12aed0, 50000
```

The beginning of the Java log is a fixed time format. The regular expression is \[\d+-\d+-\w+:\d+:\d+,\d+]\s.*.



Log field extraction

According to the Log Service data models, a log contains one or more key–value pairs. To extract specified fields for analysis, you need to set a regular expression. If log content is not processed, the log can be considered as a key–value pair. For the preceding access log:

When fields are extracted

```
  Regular expression:(\S+)\s-\s-\s\[(\S+)\s[^]]+]\s"(\w+).*
Extracted content:1 ) 10.1.1.1 ; 2 ) 13/Mar/2016:10:00  ; 3 ) GET
```

When fields are not extracted

```
Regular expression:(.*)
Extracted content:1 ) 10.1.1.1 - - [13/Mar/2016:10:00:10 +0800] "GET / HTTP/1.1" 0.011 180 404 570 "-"
"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; 360se)"
```

Log time

According to the Log Service data models, each log must have a time field in UNIX timestamp format. The log time can be set to the system time at which Logtail captures the log or time in the log content. For the preceding access log:

Time in the log content

```
Time:13/Mar/2016:10:00:10
Time expression:%d/%b/%Y:%H:%M:%S
```

Log capture time

```
Time: Timestamp when the log is captured
```

Through Logtail, you can configure TCP ports locally to receive syslog data transferred by syslog agents using TCP protocol. Logtail parses the received data and forwards it to LogHub.

Prerequisites

You must install Logtail before using it to collect logs. Logtail supports Windows and Linux operating systems. For the installation methods, refer to Install Logtail on Winddows and Install Logtail on Linux .

Procedure

To collect syslog data, you need to configure the following settings.

Step 1: Create a Logtail syslog configuration on the Log Service console

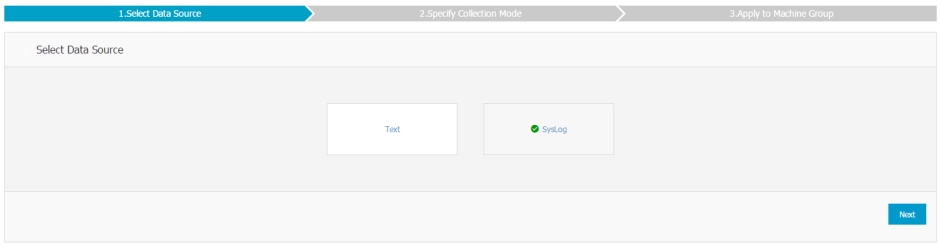
Log on to the Log Service console.

Select the desired project, and click the project name or click **Manage** on the right.

Select the desired Logstore and click Logtail Config **Manage**.

Click **Create** in the upper-right corner.

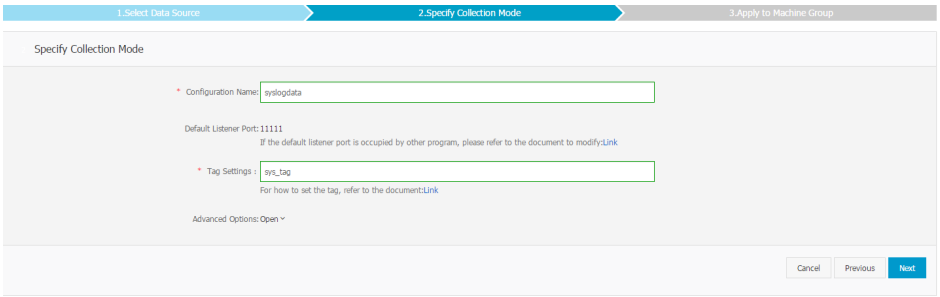
Select **Syslog** as the data source and then click **Next**.



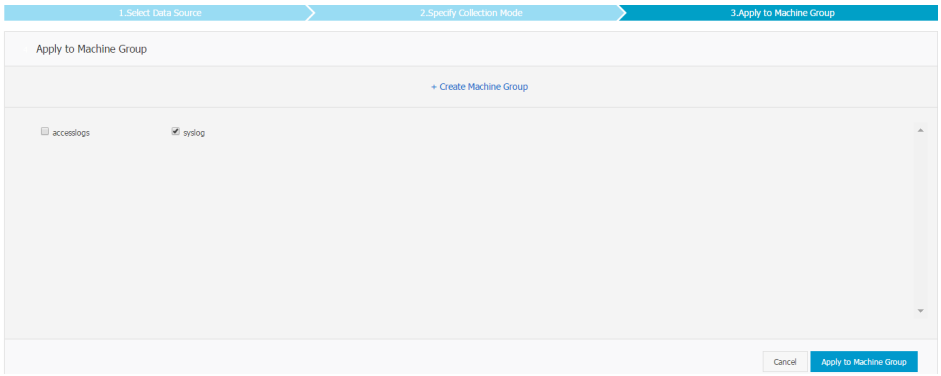
Specify the collection mode and click **Next**.

The configuration name can be 3 to 63 characters in length and lowercase letters, digits, hyphens (-), and underscores (_). It must start and end with lowercase letters or digits.

Note: After the configuration name is set, it cannot be modified.



Select the desired machine group and click **Apply to Machine Group** to apply the configuration.



Step 2: Configure the Logtail protocol

In the machine’ s Logtail installation directory, find ilogtail_config.json. Typically, it is in the /usr/local/ilogtail/ directory. Modify the syslog configuration as needed.

Check whether the syslog function is enabled.

False indicates disabled, **true** indicates enabled.

```
"streamlog_open" : true,
```

Specify the memory pool size used by syslog to receive logs.

When the program is launched, it initiates a one-time application for the memory size. Set the memory size in MBs based on the machine’ s memory and your actual needs.

```
"streamlog_pool_size_in_mb" : 50,
```

Specify the buffer size used each time Logtail calls the socket io rcv interface. The unit is in bytes.

```
"streamlog_rcv_size_each_call" : 1024,
```

Specify the syslog log format.

```
"streamlog_formats":[],
```

Specify the TCP port Logtail uses to receive syslog logs. The default value is 11111.

```
"streamlog_tcp_port" : 11111
```

After modifying the configuration, restart Logtail. To restart Logtail, first stop it and then start it again.

```
sudo /etc/init.d/ilogtaild stop
sudo /etc/init.d/ilogtaild start
```

Step 3: Install rsyslog and modify its configuration

Install rsyslog.

If rsyslog has already been installed on the machine, skip this step.

For the installation method, refer to:

- Ubuntu installation method
- Debian installation method
- RHEL/CENTOS installation method

In /etc/rsyslog.conf, modify the configuration as needed.

An example is shown as follows.

```
$WorkDirectory /var/spool/rsyslog # where to place spool files
$ActionQueueFileName fwdRule1 # unique name prefix for spool files
$ActionQueueMaxDiskSpace 1g # 1gb space limit (use as much as possible)
$ActionQueueSaveOnShutdown on # save messages to disk on shutdown
$ActionQueueType LinkedList # run asynchronously
$ActionResumeRetryCount -1 # infinite retries if host is down
# Defines the fields of log data
$template ALI_LOG_FMT,"0.1 sys_tag %timegenerated:::date-unixtimestamp% %fromhost-ip%
%hostname% %pri-text% %protocol-version% %app-name% %procid% %msgid% %msg:::drop-last-
lf%\n"
*.* @@10.101.166.173:11111;ALI_LOG_FMT
```

Note: In the template **ALI_LOG_FMT**, the value of the second field is **sys_tag**. This value must be consistent with the configuration created in step 1. This configuration indicates that all the (*.*) syslog logs received by this machine will be formatted according to the **ALI_LOG_FMT** template, and use TCP protocol to forward it to 10.101.166.173:11111. The machine 10.101.166.173 must be in the machine group selected in step 1 and configured according to step 2.

Start rsyslog (sudo/etc/init.d/rsyslog restart).

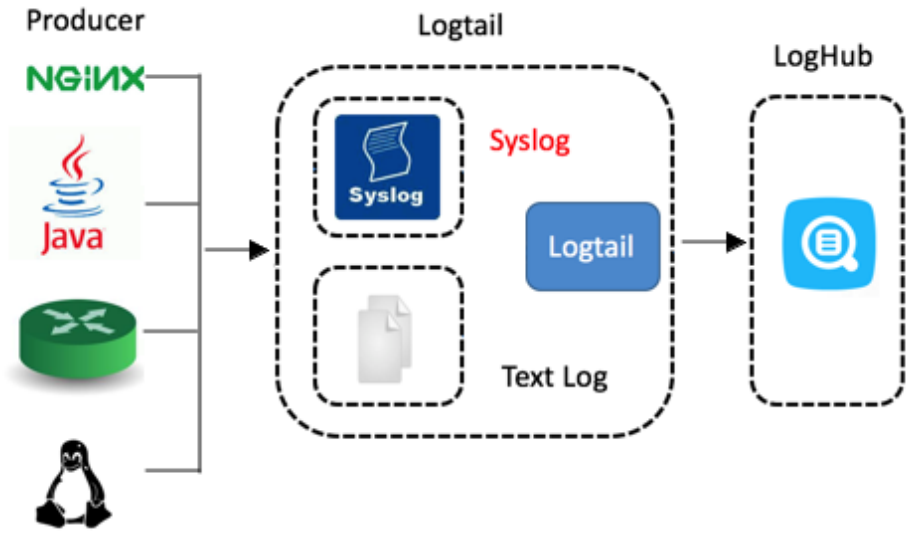
Before launching rsyslog, check whether another syslog agent is installed on the machine, such as syslogd, sysklogd, or syslog-ng. If there is one, stop rsyslog.

After completing the preceding three steps, you can collect syslog data on the machine to Log Service.

Further information

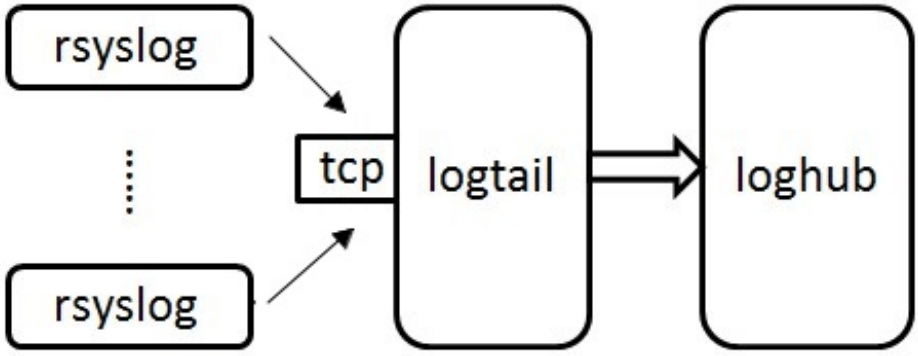
For more information about syslog log collection, as well as how to format syslog data, refer to Syslog data collection reference.

Logtail supports the collection of syslogs and text files.



Basic concept

Syslog collection support is based on TCP. You need to configure the TCP port locally so that Logtail can receive the logs forwarded by syslog agents. The following figure shows the relationship between Logtail, syslogs, and LogHub. Logtail, with the TCP port enabled, receives the syslogs forwarded by rsyslog or other syslog agents over TCP, parses the received logs, and forwards the logs to LogHub.



The following sections describes the syslog formatting method and syslog collection principle. If you do not require syslog formatting, you can skip the following content.

Syslog log format

Logtail receives data as streams through the TCP port. If you want to parse individual logs from the data streams, ensure that the log format meets the following requirements.

- Logs are separated by linefeeds (\n), but do not contain linefeeds.
- Only the message body of a log can contain spaces; other fields cannot contain spaces.

The log format is defined as follows.

```
$version $tag $unixtimestamp $ip [$user-defined-field-1 $user-defined-field-2 $user-defined-field-n] $msg\n"
```

Where:

- version: The version of the log format. Logtail uses the version to parse user-defined-field.
- tag: The data tag used to find the desired project or Logstore. It cannot contain spaces or linefeeds.
- unixtimestamp: The timestamp of the log.
- ip: The IP address of the machine corresponding to the log. If the IP address is 127.0.0.1, it is replaced with the peer address of the TCP socket when the log is sent to Log Service.
- user-defined-field: Zero or multiple user-defined-fields can be set. The fields cannot contain spaces or linefeeds. Brackets indicate that the fields are optional.
- msg: The message body of the log. The \n symbol appended to the message is a linefeed, but the message cannot contain linefeeds.

Sample log in the preceding format:

```
2.1 streamlog_tag 1455776661 10.101.166.127 ERROR com.alibaba.streamlog.App.main(App.java:17) connection refused, retry
```

Logtail supports the collection of syslogs and other log types that meet the following requirements:

- Logs can be formatted as previously described.
- Logs can be appended to the remote end over TCP.

Syslog parsing

Logtail needs additional configuration for parsing logs in the preceding format.

Configuration example:

```
"streamlog_formats":
[
{"version": "2.1", "fields": ["level", "method"]},
{"version": "2.2", "fields": []},
{"version": "2.3", "fields": ["pri-text", "app-name", "syslogtag"]}
]
```

Logtail identifies the corresponding user-defined-field format in streamlog_formats based on the version field.

The preceding sample log, with the version field set to 2.1, contains two user-defined-fields: level and method. The sample log is parsed in the following format.

```
{
"source": "10.101.166.127",
"time": 1455776661,
"level": "ERROR",
"method": "com.alibaba.streamlog.App.main(App.java:17)",
"msg": "connection refused, retry"
}
```

The version field is used to parse user-defined-fields, and the tag field is used to find the project or Logstore where the data is sent. These two fields are not included in the logs sent to Log Service.

Furthermore, Logtail provides predefined log formats where the version field starts with “0.” or “1.”, such as 0.1 and 1.1. Therefore, do not define any version field starting with “0.” or “1.”.

Syslog collection by Logtail

log4j

Introduce the Log4j library.

```
<dependency>
<groupId>org.apache.logging.log4j</groupId>
<artifactId>log4j-api</artifactId>
<version>2.5</version>
</dependency>
<dependency>
<groupId>org.apache.logging.log4j</groupId>
<artifactId>log4j-core</artifactId>
<version>2.5</version>
</dependency>
```

Introduce the Log4j configuration file log4j_aliyun.xml to programs.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration status="OFF">
<appenders>
<Socket name="StreamLog" protocol="TCP" host="10.101.166.173" port="11111">
<PatternLayout pattern="%X{version} %X{tag} %d{UNIX} %X{ip} %-5p %l %enc{%m}%n" />
</Socket>
</appenders>
<loggers>
<root level="trace">
<appender-ref ref="StreamLog" />
</root>
</loggers>
</configuration>
```

10.101.166.173:11111 is the address of the machine where Logtail is located.

Set ThreadContext in programs.

```
 package com.alibaba.streamlog;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.apache.logging.log4j.ThreadContext;

public class App
{
private static Logger logger = LogManager.getLogger(App.class);
public static void main( String[] args ) throws InterruptedException
{
ThreadContext.put("version", "2.1");
ThreadContext.put("tag", "streamlog_tag");
ThreadContext.put("ip", "127.0.0.1");
while(true)
{
logger.error("hello world");
Thread.sleep(1000);
}
//ThreadContext.clearAll();
}
}
```

Tengine

Tengine can collect syslogs by ilogtail.

Tengine uses the ngx_http_log_module for logging to the local syslog agent, which forwards the logs to rsyslog.

Example:

Send INFO-level access logs of the user type to Unix dgram(/dev/log) of the local machine and set the application tag to nginx.

```
access_log syslog:user:info:/var/log/nginx.sock:nginx
```

rsyslog configuration:

```
module(load="imuxsock") # needs to be done just once
input(type="imuxsock" Socket="/var/log/nginx.sock" CreatePath="on")
$template ALI_LOG_FMT,"2.3 streamlog_tag %timegenerated:::date-unixtimestamp% %fromhost-ip% %pri-text%
%app-name% %syslogtag% %msg:::drop-last-lf%\n"
if $syslogtag == 'nginx' then @@10.101.166.173:11111;ALI_LOG_FMT
```

Nginx

The collection of Nginx access logs is used as an example.

Access log configuration:

```
access_log syslog:server=unix:/var/log/nginx.sock,nohostname,tag=nginx;
```

rsyslog configuration:

```
module(load="imuxsock") # needs to be done just once
input(type="imuxsock" Socket="/var/log/nginx.sock" CreatePath="on")
$template ALI_LOG_FMT,"2.3 streamlog_tag %timegenerated:::date-unixtimestamp% %fromhost-ip% %pri-text%
%app-name% %syslogtag% %msg:::drop-last-lf%\n"
if $syslogtag == 'nginx' then @@10.101.166.173:11111;ALI_LOG_FMT
```

For more information, refer to nginx.

Python Syslog

Example:

```
import logging
import logging.handlers

logger = logging.getLogger('myLogger')
logger.setLevel(logging.INFO)

#Add handler to the logger using unix domain socket '/dev/log'
handler = logging.handlers.SysLogHandler('/dev/log')

#Add formatter to the handler
formatter = logging.Formatter('Python: { "loggerName":"%(name)s", "asciTime":"%(asctime)s",
"pathName":"%(pathname)s", "logRecordCreationTime":"%(created)f", "functionName":"%(funcName)s",
"levelNo":"%(levelno)s", "lineNo":"%(lineno)d", "time":"%(msecs)d", "levelName":"%(levelname)s",
"message":"%(message)s"}')

handler.formatter = formatter
logger.addHandler(handler)

logger.info("Test Message")
```

Log Service manages ECS servers whose logs need to be collected by Logtail in the form of machine groups. You can go to the **Machine Groups** page by selecting a project from the project list of Log Service. Log Service allows you to create, modify, and delete machine groups, view the machine group list and the status of machine groups, manage configurations, and apply machine group IDs.

Create a machine group

A machine group is defined by either of the two items:

- IP address: Defines the name of the machine group and adds the intranet IP addresses of the machines in the group.
- ID: Defines the ID of the machine group and configures the IDs of the machines in the group for association.

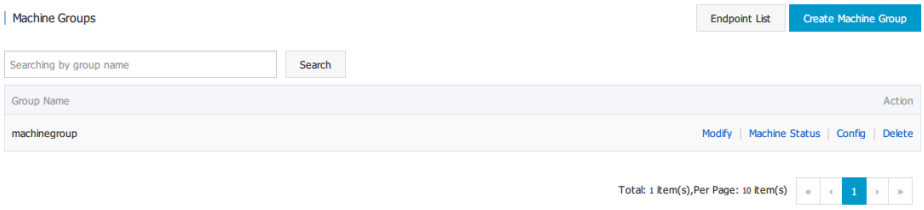
For details about creating a machine group, refer to Create a machine group.

View the machine group list

Log on to the Log Service console.

Click **LogHub - Collect** > **Logtail Machine Group** in the left navigation bar to go to the project' s **Machine Groups** page.

All machine groups under the project are displayed.



Modify a machine group

After a machine group is created, you can adjust the ECS servers in the group according to your needs.

Note: The name of a machine group cannot be changed once the machine group is created.

Log on to the Log Service console.

Click **LogHub - Collect** > **Logtail Machine Group** in the left navigation bar to go to the project' s **Machine Groups** page.

Select the machine group to be modified and click **Modify**.

Modify the configuration of the machine group and then click **Confirm**.

View the machine group status

To verify that the Logtail agent is successfully installed on all ECS servers in a machine group, you can view the heartbeat information of the Logtail agent.

Log on to the Log Service console.

Click **LogHub - Collect** > **Logtail Machine Group** in the left navigation bar to go to the project' s **Machine Groups** page.

Select the desired machine group and click **Machine Status**.

If the heartbeat status is **OK**, the Logtail agent is successfully installed on all ECS servers. If the heartbeat status is **FAIL**, perform a self-check by following on-screen instructions. If the problem persists, submit a ticket to Alibaba Cloud.

Note:

- The **OK** heartbeat status indicates that Logtail is properly connected to Log Service. After a server is added to a machine group, there may be a delay of several minutes before the **OK** heartbeat status is displayed.
- If the heartbeat status of an ECS server repeatedly displays **FAIL**, perform troubleshooting in accordance with Install Logtail for Windows and Install Logtail for Linux.

Manage configurations

One important management item is the configuration for collection of the Logtail agent. For details, refer to Collect text files using Logtail) and Collect syslogs using Logtail. You can apply or delete Logtail configurations of a machine group to decide what logs are collected, how the logs are parsed, and to which Logstore the logs are sent by the Logtail agent on each ECS server.

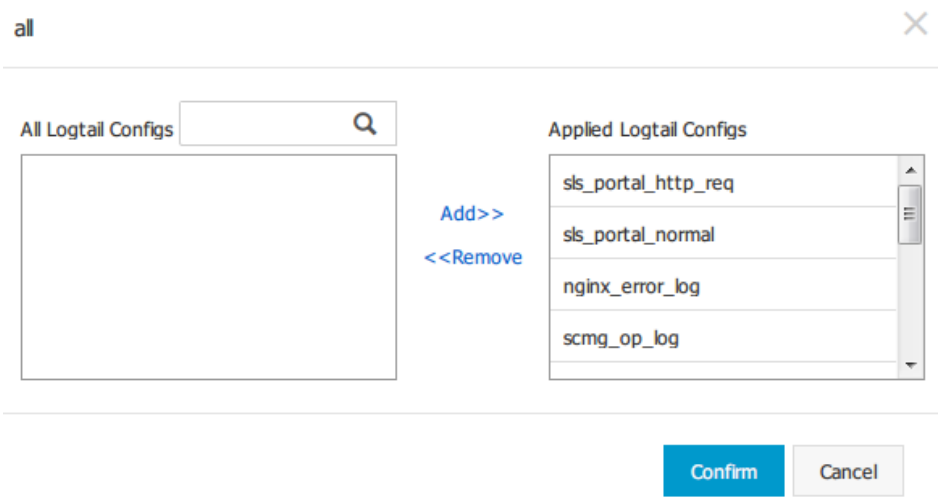
Log on to the Log Service console.

Click **LogHub - Collect** > **Logtail Machine Group** in the left navigation bar to go to the project' s **Machine Groups** page.

Select the desired machine group and click **Config**.

Select the desired Logtail configuration and click **Add** or **Remove** to add/delete the configuration to/from the machine group.

After Logtail configuration is added, it is assigned to the Logtail agent on each ECS server in the machine group. After Logtail configuration is deleted, it is removed from the Logtail agent.



Delete a machine group

Log on to the Log Service console.

Click **LogHub - Collect** > **Logtail Machine Group** in the left navigation bar to go to the project' s **Machine Groups** page.

Select the machine group to be deleted and click **Delete**.

Click **Confirm** in the confirmation dialog box.

Logtail configuration

The Logtail agent provides an easy way to collect logs from ECS servers through the Log Service console. After Logtail installation, you must configure log collection settings for the Logtail agent.

For details about Logtail installation, refer to Install Logtail for Windows and Install Logtail for Linux.

You can create and modify the Logtail configuration of Logstores in the Logstore list.

> If you need to collect IIS access logs, refer to IIS log collection best practices for configuration.

Create Logtail configuration

For details about creating Logtail configuration on the Log Service console, refer to Use Logtail to

collect text files and Use Logtail to collect syslogs.

View the Logtail configuration list

Log on to the Log Service console.

Select the desired project, and click the project name or click **Manage** on the right.

On the **LogStore List** page, click Logtail Config **Manage** to go to the **Logtail Configuration List** page.

The page lists all configurations of the specified Logstore in three columns: configuration name, data sources, and configuration details. When the data sources column shows **text**, the configuration details column shows the file path and file name.

Note: A file is collected through only one configuration.

Modify Logtail configuration

Log on to the Log Service console.

Select the desired project, and click the project name or click **Manage** on the right.

On the **LogStore List** page, click Logtail Config **Manage** to go to the **Logtail Configuration List** page.

Click the name of the Logtail configuration to be modified.

You can modify the log collection mode and specify the machine group where the modified mode is applied. The configuration modification process is the same as the configuration creation process.

Delete Logtail configuration

Log on to the Log Service console.

Select the desired project, and click the project name or click **Manage** on the right.

On the **LogStore List** page, click Logtail Config **Manage** to go to the **Logtail Configuration List** page.

Select the Logtail configuration to be deleted and click **Delete** on the right.

After the configuration is deleted successfully, it is unbound from the machine group and Logtail no longer collects the log files matching the deleted configuration.

Note: Before you delete a Logstore, delete all its Logtail configuration.

Application scenarios

In the following scenarios, you need to configure the Logtail startup parameters.

- Memory usage is high due to a large number of log files to be collected. The meta information of each file must be maintained in memory, including the file signature, collection location, and file name.
- CPU usage is high due to heavy log data traffic.
- Traffic sent to Log Service is heavy due to large data volume.
- Syslogs and TCP data streams are collected.

Startup configuration

File path

/usr/local/ilogtail/ilogtail_config.json

File format

JSON

File sample (which only shows partial configuration items)

```
{
...
"cpu_usage_limit" : 0.4,
"mem_usage_limit" : 100,
"max_bytes_per_sec" : 2097152,
"process_thread_count" : 1,
"send_request_concurrency" : 4,
"streamlog_open" : false,
"streamlog_pool_size_in_mb" : 50,
"streamlog_rcv_size_each_call" : 1024,
```

```
"streamlog_formats":[],
"streamlog_tcp_port" : 11111,
"buffer_file_num" : 25,
"buffer_file_size" : 20971520,
"buffer_file_path" : "",
...
}
```

Common configuration parameters

Parameter	Value	Description
cpu_usage_limit	CPU usage threshold, double type and calculated per core.	For example, the value 0.4 indicates the CPU usage of Logtail is limited to 40% of single-core capacity. Logtail restarts automatically when the threshold is exceeded. In many cases, the single-core processing capability is about 24 MB/s in simple mode and about 12 MB/s in full mode.
mem_usage_limit	In-memory usage threshold, int type and measured in MBs.	For example, the value 100 indicates the memory usage of Logtail is limited to 100 MBs. Logtail restarts automatically when the threshold is exceeded. If you need to collect more than 1,000 distinct files, properl increase the threshold value.
max_bytes_per_sec	Traffic limit on the raw data sent by Logtail, int type and measured in bytes per second.	For example, the value 2,097,152 indicates the data transfer rate of Logtail is limited to 2 MB/s.
process_thread_count	Number of threads Logtail uses to write data to log files.	The default value is 1, which supports a write speed of 24 MB/s in simple mode and 12 MB/s in full mode. Adjust the threshold only when necessary.
send_request_concurrency	By default, Logtail sends data packets asynchronously. You can set a larger asynchronous concurrency value if the write TPS is large.	By default, four asynchronous concurrencies are available. You can calculate the concurrency quantity based on the condition that one concurrency supports 0.5–1 MB/s network throughout. The actual concurrency quantity varies with network

		delay.
streamlog_open	Syslog reception switch, bool type.	false indicates that syslog reception is disabled; true indicates that syslog reception is enabled.
streamlog_pool_size_in_mb	Size of the cache storing received syslogs, measured in MBs.	This parameter indicates the size of the memory pool storing received syslogs. Logtail requests memory of the specified size at one time when launched. Set the pool size according to the machine’ s memory size and your needs.
streamlog_rcv_size_each_call	Size of the cache Logtail uses when calling the Linux socket rcv interface, measured in bytes.	You can increase the value in the case of heavy syslog traffic. The recommended value range is 1,024 to 8,192 bytes.
streamlog_formats	Method of parsing received syslogs.	
streamlog_tcp_port	TCP port through which Logtail receives syslogs.	By default, port 11111 is used.
buffer_file_num	When a network exception occurs or the write quota is exceeded, Logtail writes the logs that are parsed in real time to a local file (located in the installation directory) and then tries to resend the logs to Log Service after recovery. This parameter indicates the maximum number of cached files.	The default value is 25 for public cloud users.
buffer_file_size	Maximum number of bytes per cached file. The (buffer_file_num * buffer_file_size) indicates the maximum disk space available for storing cached files.	The default value is 20,971,520 (20 MBs).
buffer_file_path	Directory that stores cached files. After you modify this parameter, manually transfer the files named in the format of logtail_buffer_file_* in the old cache directory to the new directory so that Logtail can read the cached files and delete them after sending.	The default value is null, indicating the cached files are stored in the Logtail installation directory (/usr/local/ilogtail).

Note:

- The preceding table only lists the common startup parameters. If ilogtail_config.json has parameters not listed above, the default values are applied.
- Add or modify the values of configuration parameters according to your need. Any unused configuration parameters (for example, parameters related to the collection of syslog data streams) do not need to be added to ilogtail_config.json.

Modify configuration

Configure ilogtail_config.json according to your needs. Confirm the modified configuration is JSON compatible.

Check that the modified configurations are JSON compatible.

Restart Logtail to apply the modified configuration.

```
/etc/init.d/ilogtaild stop
/etc/init.d/ilogtaild start
/etc/init.d/ilogtaild status
```

Each log in Log Service has a timestamp. When Logtail collects log data from files, it extracts the timestamp string of each log and parses it into a timestamp. Therefore, you need to specify a timestamp format for Logtail.

In Linux, Logtail supports all time formats provided by the strftime function. Logtail only parses and uses the timestamp strings that can be expressed in the log formats defined by the strftime function.

The timestamp strings of logs have multiple formats. To make configuration easier, Logtail supports the following common log time formats.

Format	Meaning	Example
%a	locale’ s abbreviated weekday name	Example: Fri
%A	locale’ s full weekday name	Example: Friday
%b	locale’ s abbreviated month name	Example: Jan
%B	locale’ s full month name	Example: January
%d	day of the month as a decimal number [1,31]	Example:7, 31
%h	same as %b	Example: Jan
%H	hour (24-hour clock) as a	Example: 22

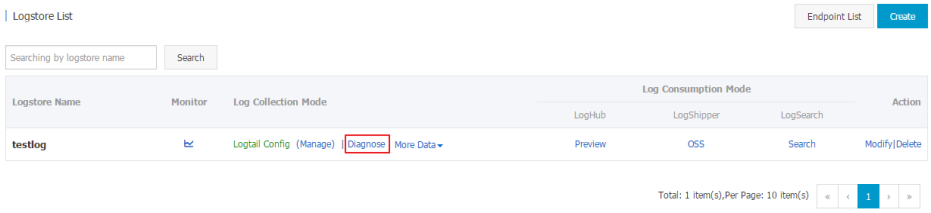
	decimal number	
%I	hour (12-hour clock) as a decimal number	Example: 11
%m	month as a decimal number	Example: 08
%M	minute as a decimal number [00,59]	Example: 59
%n	newline character	Linefeed
%p	locale’ s equivalent of either a.m. or p.m	Example: AM/PM
%r	time in a.m. and p.m. notation (%I:%M:%S %p)	Example: 11:59:59 AM
%R	time in 24 hour notation (%H:%M)	Example: 23:59
%S	second as a decimal number[00,59]	Example: 59
%t	tab character	Tab character
%y	year without century as a decimal number [00,99]	Example: 04,98
%Y	year with century as a decimal number	Example: 2004,1998
%z	timezone name or abbreviation, or by no bytes if no timezone information exists	Example:-07:00, +0800
%C	century number as a decimal number [00-99]	Example: 16
%e	the day of the month as a decimal number [1,31]; a single digit is preceded by a space{color}	
%j	day of the year as a decimal number [001,366]	Example: 365
%u	weekday as a decimal number [1,7], with 1 representing Monday	Example: 2
%U	week number of the year (Sunday as the first day of the week) as a decimal number [00,53]	Example: 23
%V	week number of the year (Monday as the first day of the week) as a decimal number [01,53]	Example: 24
%w	weekday as a decimal	Example: 5

	number [0,6], with 0 representing Sunday{	
%W	week number of the year (Monday as the first day of the week) as a decimal number [00,53]	Example: 23
%c	locale’ s appropriate date and time representation	More information, such as long date and short date, must be specified. The preceding supported formats can be used for more precise expression.
%x	locale’ s appropriate date representation	More information, such as long date and short date, must be specified. The preceding supported formats can be used for more precise expression.
%X	locale’ s appropriate time representation	More information, such as long date and short date, must be specified. The preceding supported formats can be used for more precise expression.
%s	unix timestamp	Example: 1476187251

Errors may occur during log collection by Logtail, such as regular expression parsing failures, incorrect file paths, and traffic exceeding the shard service capability. Log Service provides the query function for debugging log collection errors.

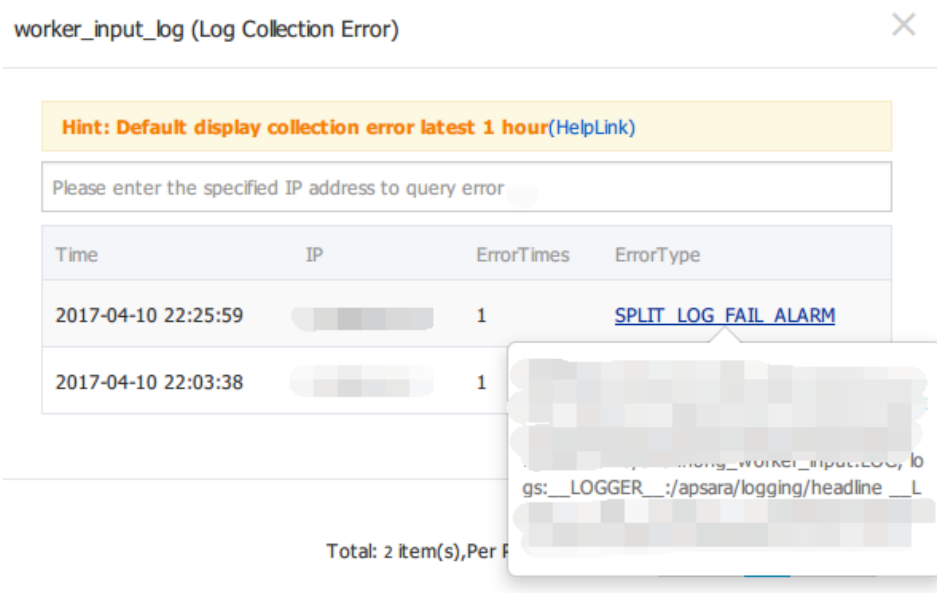
Error query portal

Select a specific project to go to the Logstore list and click **Diagnose** in the Log Collection Mode column.



Log collection error display

The query page lists the Logtail collection errors of the specified Logstore.



Log collection error query by machine

To query all the occurred log collection errors on a specific machine, enter the IP address of the machine in the search box on the query page. Logtail reports errors every 5 minutes.

Logtail error list and handling methods

Error type	Description	Handling method
LOGFILE_PERMINSSION_ALARM	Logtail has no permission to read the specified file.	Check the Logtail launch user on the machine. It is recommended that Logtail be launched by a root user.
SPLIT_LOG_FAIL_ALARM	The beginning of the line cannot be parsed by the regular expression, and thus the log cannot be split into lines.	Check whether the regular expression at the beginning of the line is correct.
MULTI_CONFIG_MATCH_ALARM	A file can only be collected based on one Logtail configuration.	Check whether the same file is collected based on multiple Logtail configurations and delete redundant configuration, if any.
REGEX_MATCH_ALARM	The log content and the regular expression do not match.	Copy the sample log in the error content and retry matching.
SENDER_BUFFER_FULL_ALARM	The Logtail parsing capability lags behind the log generation speed.	Submit a ticket to Alibaba Cloud.
LOGTAIL_CRASH_ALARM	Logtail has crashed due to	Please submit a ticket to

	the machine resource usage limit being exceeded.	Alibaba Cloud.
INOTIFY_DIR_NUM_LIMIT_ALARM	The system has more than 8,000 inotify watches, but Logtail can use 3,000 watches at most. The number of Logtail monitored directories exceeds 3,000.	Check whether the directories in log collection have multiple subdirectories.
DIR_IS_LINK_ALARM	Logtail detects that the monitored directory is a soft link.	The same directory can be configured as a soft link or raw link, but cannot be configured as both in different configuration.

If the collection runs into exception when you use Logtail to collect logs,go through the following steps for troubleshooting:

1 Check that the Logtail heartbeat is normal in the machine group

Log on to the console and click the machine group to view its status. For details, refer to Logtail machine group. If the heartbeat status shows OK, go to the next step; if the heartbeat status shows FAIL, continue with the troubleshooting.

The causes of Logtail heartbeat failure include:

Logtail is not installed on machines.

Check the client status in Linux:

```
sudo /etc/init.d/ilogtaild status
```

Check the client status in Windows:

```
Go to Control Panel > Management Tools > Services.
View the running status of two Windows services: LogtailDaemon and LogtailWorker.
```

If the Logtail client is not installed, install it based on the network type and the region where your Log Service project is located. For details, refer to Install Logtail.

Parameters are selected incorrectly during installation.

As the Log Service operates by region, you must specify the correct endpoint when installing

the Logtail client. Check configuration of your installed clients in the following paths:

- Linux：/usr/local/ilogtail/ilogtail_config.json
- Windows x64 : C:\Program Files (x86)\Alibaba\Logtail\ilogtail_config.json
- Windows x32 : C:\Program Files\Alibaba\Logtail\ilogtail_config.json

Check the following settings:

- The network ingress connected to the client is located in the same region as your Log Service project. For a list of network ingresses, refer to Endpoints.
- The correct domain is selected based on the network environment of your machines. If an internal domain is selected for a VPC environment, client connection will fail. You can use Telnet to test the domain configured in ilogtail_config.json, for example, telnet logtail.cn-hangzhou-intranet.log.aliyuncs.com 80.

The IP addresses or user IDs configured at the server end are incorrect.

Generally, Logtail obtains the IP address of a machine in one of the following ways:

- If the machine is configured with host name binding in /etc/hosts, confirm the bound IP address. Run the hostname command to view the host name.
- If host name binding is not configured, Logtail obtains the IP address of the machine’ s first network adapter.

View the IP addresses on the server in the following paths:

- Linux：/usr/local/ilogtail/app_info.json
- Windows x64 : C:\Program Files (x86)\Alibaba\Logtail\app_info.json
- Windows x32 : C:\Program Files\Alibaba\Logtail\app_info.json

If the IP addresses of the machine group at the server end are inconsistent with the IP addresses obtained by the Logtail client, make the following changes as needed:

- If the IP addresses of the machine group at the server end are incorrect, change them to the correct IP addresses and save the change. Then, wait for 1 minute and check the IP addresses again.
- If the network configuration (for example, /etc/hosts) of the machines is modified, restart Logtail to obtain the machine IP addresses again.

Run the following command to restart Logtail if necessary:

- Linux：sudo /etc/init.d/ilogtaild stop; sudo /etc/init.d/ilogtaild start
- For Windows, go to **Control Panel** > **Management Tools** > **Services** > **Restart LogtailWorker**.

2 Check that collection configuration is created and

applied to the machine group.

After you confirm that the Logtail client status is normal, check the following configuration:

2.1 Check that Logtail configuration is created

For details, refer to Logtail configuration.Check that the log monitoring directory and log file name match those on machines. The directory must be set to an absolute path, because fuzzy match is not supported, while the log file name supports fuzzy match.

2.2 Check that Logtail configuration is applied to the machine group

View the Logtail machine group and select **Manage Configuration** to check that the target configuration is applied to the machine group.

3 Check for collection errors

If Logtail is properly configured, check that new data is generated in the log file in real time. As Logtail collects incremental data only, it does not read inventory files if the files are not updated. If the log file is updated but the updates cannot be queried in the Log Service, diagnose the problem following the steps below:

Collection error diagnosis

Refer to Query Logtail collection errors to run queries and continue with the troubleshooting in accordance with the errors returned by Logtail.

Logtail log checking

Client logs include key information and all warning and error logs. To query complete and real-time errors, view client logs in the following paths:

- Linux : /usr/local/ilogtail/ilogtail.LOG
- Windows x64 : C:\Program Files (x86)\Alibaba\Logtail\logtail_*.log
- Windows x32 : C:\Program Files\Alibaba\Logtail\logtail_*.log

Quota exceeded

To collect large volumes of logs, files or data, you may need to modify the Logtail startup parameters for higher log collection throughput. For details, refer to Logtail startup configuration parameters.

If the problem persists, open a ticket to consult Log Service engineers and attach the key

information collected during troubleshooting to the ticket.

Use Logstash to collect logs

Installation package

Log Service provides an installation package based on LogStash 2.2.2, which integrates JRE 1.8, Log Service write plug-in, and NSSM 2.24. The deployment process using this package is simpler than custom installation. You can refer to the custom installation process according to your need.

Install Logstash

Download the installation package and extract it to the C: drive.

Confirm the Logstash launch program is C:\logstash-2.2.2-win\bin\logstash.bat.

You can install Logstash using quick installation or custom installation methods.

Step 1: Install Java

Download the installation package.

Go to the Java website to download JDK for installation.

Set environment variables.

Add or modify environment variables in advanced system settings.

- PATH: C:\Program Files\Java\jdk1.8.0_73\bin
- CLASSPATH: C:\Program Files\Java\jdk1.8.0_73\lib;C:\Program Files\Java\jdk1.8.0_73\lib\tools.jar
- JAVA_HOME: C:\Program Files\Java\jdk1.8.0_73

Perform verification.

Run PowerShell or cmd.exe for verification.

```
PS C:\Users\Administrator> java -version
```

```
java version "1.8.0_73"
Java(TM) SE Runtime Environment (build 1.8.0_73-b02)
Java HotSpot(TM) 64-Bit Server VM (build 25.73-b02, mixed mode)
PS C:\Users\Administrator> javac -version
javac 1.8.0_73
```

Step 2: Install Logstash

Download the installation package from the Logstash website.

Select Version 2.2 or later on the Logstash homepage.

Install Logstash.

Extract logstash-2.2.2.zip to the C:\logstash-2.2.2 directory.

Confirm the Logstash launch program is C:\logstash-2.2.2\bin\logstash.bat.

Step 3: Install the plug-in used by Logstash to write logs to Log Service

Install the plug-in online or offline based on the machine’ s network environment.

Online installation

The plug-in is hosted by RubyGems. For further information, refer to here.

Run PowerShell or cmd.exe to go to the Logstash installation directory.

```
PS C:\logstash-2.2.2> .\bin\plugin install logstash-output-logservice
```

Offline installation

Download from the Logstash website. Go to the logstash-output-logservice page and click **Download** in the lower-right corner.

If the machine where logs are collected cannot access the Internet, copy the downloaded gem package to the C:\logstash-2.2.2 directory of the machine. Run PowerShell or cmd.exe to go to the Logstash installation directory.

```
PS C:\logstash-2.2.2> .\bin\plugin install C:\logstash-2.2.2\logstash-output-logservice-0.2.0.gem
```

Verification

```
PS C:\logstash-2.2.2> .\bin\plugin list
```

Verify that logstash-output-logservice exists in the plug-in list of the machine.

Step 4: Install NSSM

Download from the Logstash website. Go to the NSSM website to download the NSSM installation package.

After you download the installation package to the local machine, extract it to the C:\logstash-2.2.2\nssm-2.24 directory.

When logstash.bat is launched in PowerShell, the Logstash process will operate in the foreground. As Logstash is used for testing configuration and debugging collections, it is recommended to configure Logstash as a Windows service after debugging to enable Logstash to operate in the background, as well as start automatically upon power-on.

Run the following commands in PowerShell. For more NSSM usage instructions, refer to this document.

Add Logstash as a Windows service

This operation is typically performed when Logstash is deployed for the first time. If Logstash has been added, you can skip this operation.

32-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win32\nssm.exe install logstash "C:\logstash-2.2.2-win\bin\logstash.bat" "agent -f C:\logstash-2.2.2-win\conf"
```

64-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win64\nssm.exe install logstash "C:\logstash-2.2.2-win\bin\logstash.bat" "agent -f C:\logstash-2.2.2-win\conf"
```

Launch the service

If the configuration file in the Logstash conf directory is updated, stop the Logstash service and launch it again.

32-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win32\nssm.exe start logstash
```

64-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win64\nssm.exe start logstash
```

Stop the service

32-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win32\nssm.exe stop logstash
```

64-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win64\nssm.exe stop logstash
```

Modify the service

32-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win32\nssm.exe edit logstash
```

64-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win64\nssm.exe edit logstash
```

Delete the service

32-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win32\nssm.exe remove logstash
```

64-bit system

```
C:\logstash-2.2.2-win\nssm-2.24\win64\nssm.exe remove logstash
```

Plug-in parameters

logstash-input-file

The plug-in is used to collect log files in tail mode. For details, refer to logstash-input-file.

Note: path indicates the file path, which must use Unix separators, for example, C:/test/multiline/*.log. Otherwise, fuzzy match is not supported.

logstash-output-logservice

You can use the plug-in to collect logs to Log Service.

Parameter	Description
endpoint	Example: http://cn-shenzhen.log.aliyuncs.com. For details, refer to Log Service endpoint.
project	Name of a Log Service project
logstore	Logstore name
topic	Log topic name. The default value “null” can be applied.
source	Log source. If this parameter is set to null, the IP address of the local machine is used as the log source.
access_key_id	Access key ID of your Alibaba Cloud account.
access_key_secret	Access key secret of your Alibaba Cloud account.
max_send_retry	The maximum number of retries performed when data packets cannot be sent to Log Service due to an exception. Data packets with retry failures are discarded. The retry interval is 200 ms.

Create collection configuration

You can create a configuration file for each log type. The file name format is xxx.conf. For easier management, it is recommended to create these configuration files in the C:\logstash-2.2.2-win\conf\ directory.

After you create a configuration file in the C:\logstash-2.2.2-win\conf\ directory, restart Logstash to apply the file.

Note: The configuration file must be encoded as UTF-8 without BOM. You can download Notepad++ to modify the file encoding format.

IIS log

For details, refer to IIS log configuration.

CSV log

The system time when logs are collected is used as the uploaded log time. For details, refer to CSV log configuration.

Default log time

For CSV logs, the time in the log content is used as the uploaded log time. For details, refer to CSV log configuration.

General log

By default, the system time when logs are collected is used as the uploaded log time. Log fields are not parsed. Single-line logs and multiline logs are supported. For details, refer to General log configuration.

Verify configuration syntax

Run PowerShell or cmd.exe to go to the Logstash installation directory.

```
PS C:\logstash-2.2.2-win\bin> .\logstash.bat agent --configtest --config C:\logstash-2.2.2-win\conf\iis_log.conf
```

Verify data collection

Modify the collection configuration file. Add the temporary configuration item rubydebug in the output phase to output collected results to the console. Set the type field according to your needs.

```
output {
if [type] == "***" {
stdout { codec => rubydebug }
logservice {
...
}
```

```
}
}
```

Run PowerShell or cmd.exe to go to the Logstash installation directory and launch the process.

```
PS C:\logstash-2.2.2-win\bin> .\logstash.bat agent -f C:\logstash-2.2.2-win\conf
```

After verification, end the logstash.bat process and delete the temporary configuration item rubydebug.

Subsequent operations

When logstash.bat is launched in PowerShell, the Logstash process operates in the foreground. It is used for testing configuration and debugging collections. It is recommended that Logstash be configured as a Windows service after debugging to enable Logstash to operate in the background and start automatically upon power-on. For details about how to configure Logstash as a Windows service, refer to Configure Logstash as a Windows service.

Logstash provides multiple plug-ins to meet personalized requirements.

grok: Structurally parses a log into multiple fields through a regular expression.

json_lines, json: Structurally parses JSON logs.

date: Parses and converts the date and time fields of logs.

multiline: Defines complex types of multiline logs.

kv: Structurally parses logs of the key–value pair type.

Prompts on the console

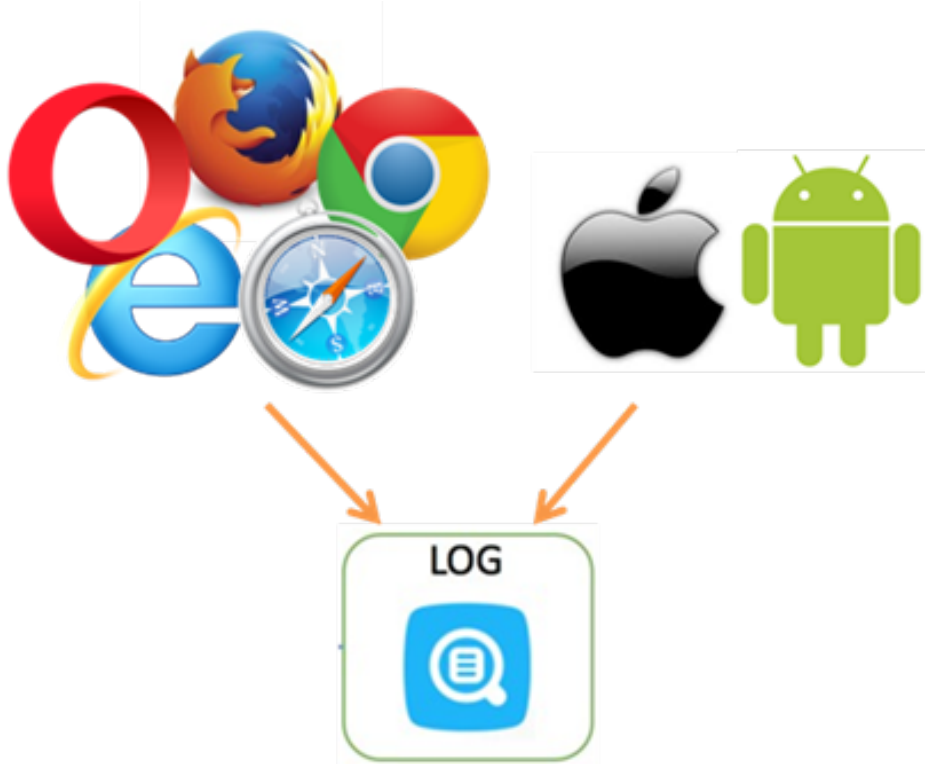
The following error does not affect functions and can be ignored: io/console not supported; tty will not be manipulated.

If other errors occur, it is recommended that you search Google or other Logstash forums for help.

Data with garbled characters in Log Service

Logstash supports UTF-8 encoding. Check whether input files are correctly encoded.

Through Web Tracking, you can collect data from HTML, H5, iOS, or Android platforms and customize dimensions and metrics in Log Service.



As shown in the preceding figure, the Tracking function can be used to collect user information from various browsers, iOS apps, and Android Apps (apart from iOS/Android SDK). For example:

- Browsers, operating systems, and resolutions used by users.
- Users' browsing behavior, for example, a users' clicking behaviors and purchasing behaviors on the website.
- The users' staying time in the app, and whether users are active.

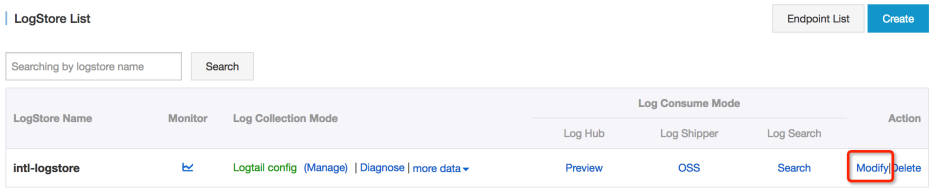
Note: Using Web Tracking means that this Logstore enables the anonymous write permission of the Internet, and dirty data may be generated.

Enable Web Tracking

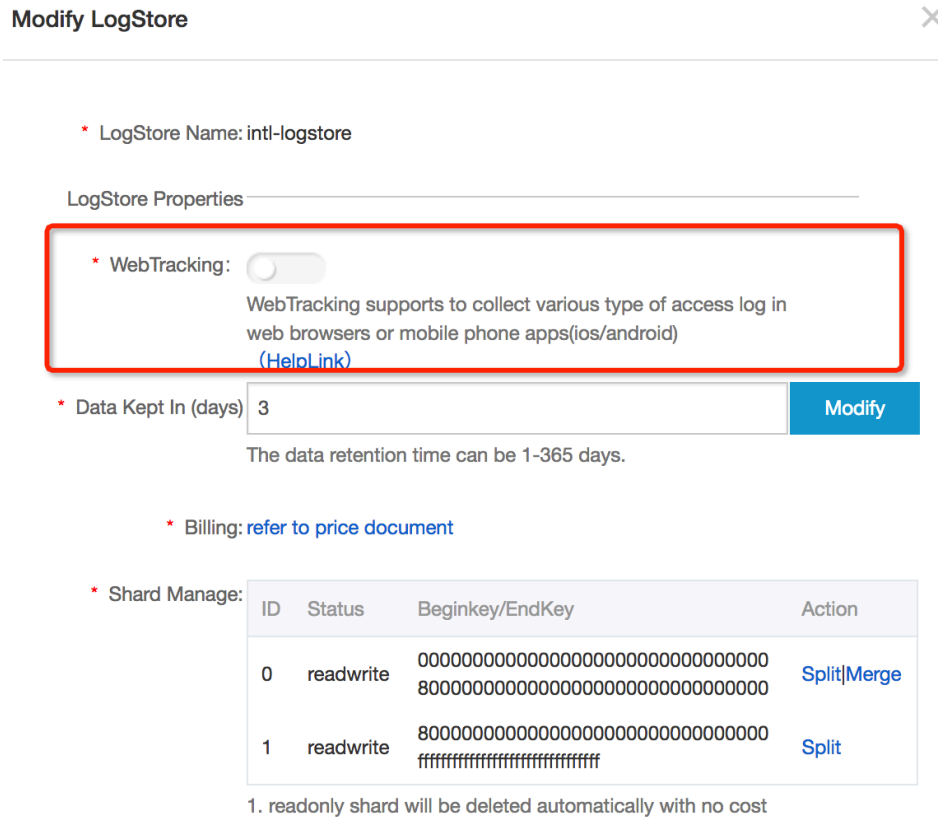
Before use, you need to enable the Web Tracking function of the Logstore.

Enable Web Tracking through the console

Select the Logstore where the Web Tracking function needs to be activated and click **Modify** on the right.



Turn on the **Web Tracking** switch.



Enable Web Tracking through Java SDK

```
import com.aliyun.openservices.log.Client;
import com.aliyun.openservices.log.common.LogStore;
import com.aliyun.openservices.log.exception.LogException;
public class WebTracking {
static private String accessId = "your accesskey id";
static private String accessKey = "your accesskey";
static private String project = "your project";
static private String host = "log service data address";
static private String logStore = "your logstore";
static private Client client = new Client(host, accessId, accessKey);
public static void main(String[] args) {
try {
//Activate the Tracking function on the created LogStore
LogStore logSt = client.GetLogStore(project, logStore).GetLogStore();
client.UpdateLogStore(project, new LogStore(logStore, logSt.GetTtl(), logSt.GetShardCount(), true));
```

```
//Disable the Tracking function
//client.UpdateLogStore(project, new LogStore(logStore, logSt.GetTtl(), logSt.GetShardCount(), false));
//Create a LogStore that supports the Tracking function
//client.UpdateLogStore(project, new LogStore(logStore, 1, 1, true));
}
catch (LogException e){
e.printStackTrace();
}
}
}
```

Upload data to the Logstore

After the Web Tracking function is enabled for Logstore, you can use any of the following three methods to upload data to the Logstore.

Use the HTTP GET request

```
curl --request GET 'http://${project}.${sls-
host}/logstores/${logstore}/track?APIVersion=0.6.0&key1=val1&key2=val2'
```

Parameter deifnitions:

- ${project}: Name of the project you activated in Log Service.
- ${sls-host}: Domain name of the region where your Log Service is located.
- ${logstore}: Name of the Logstore with the Web Tracking function enabled under ${project}.
- APIVersion=0.6.0: Reserved field, the parameter is required.
- key1=val1 and key2=val2: Key-value pairs to be uploaded to Log Service. Multiple pairs are supported, but you must ensure that the URL length is less than 16 KB.

Use the HTML img tag

```
<img src='http://${project}.${sls-host}/logstores/${logstore}/track.gif?APIVersion=0.6.0&key1=val1&key2=val2'/>
```

The parameter definitions are the same as those in **Use the HTTP GET request**.

Use the JS SDK

Copy loghub-tracking.js to the web directory, and add the following script on the page:

Click to Download

```
<script type="text/javascript" src="loghub-tracking.js" async></script>
```

Note: In order to keep page loading running, the script sends HTTP requests

asynchronously. If data needs to be sent several times in the page loading process, subsequent requests will overwrite the preceding HTTP request, and the browser shows the tracking request exits. Sending synchronous requests can help prevent this problem. To send requests synchronously, replace the following statement in the script:

```
this.httpRequest_.open("GET", url, true)
```

Replace the last parameter:

```
this.httpRequest_.open("GET", url, false)
```

Create a Tracker object.

The first parameter defines the endpoint, the second parameter defines the project, and the third parameter defines the Logstore.

```
var logger = new window.Tracker('cn-hangzhou-staging-intranet.sls.aliyuncs.com','ali-test-
tracking','web-tracking');
logger.push('customer', 'zhangsan');
logger.push('product', 'iphone 6s');
logger.push('price', 5500);
logger.logger();
logger.push('customer', 'lisi');
logger.push('product', 'ipod');
logger.push('price', 3000);
logger.logger();
```

After the preceding statement is executed, the following two logs are shown in Log Service:

```
customer:zhangsan
product:iphone 6s
price:5500


customer:lisi
product:ipod
price:3000
```

Consume data

After data is uploaded to Log Service, you can use Log Service to Ship data to MaxCompute or EMR to perform data analysis. You can also use the LogHub Client Library to consume data in Log Service.

LogHub Log4j Appender (source code)

Apache Log4j is an open source project which allows you to set the log output destination to console, file, GUI component, socket server, NT event recorder, or Unix Syslog daemon. You can set the output format and level of each log to control log generation with a finer granularity. These configurations can be performed through a configuration file without needing to modify application codes.

Log4j consists of the following three components:

- Log level: Includes ERROR, WARN, INFO, and DEBUG in a descending order of priority to indicate the importance of logs.
- Log output destination: Indicates whether logs are printed to the console or a file.
- Log output format: Indicates how the logs are displayed.

LogHub Log4j Appender allows you to set the log output destination to Alibaba Cloud Log Service. Though, LogHub Log4j Appender does not support defining the log output format. Logs written to Log Service is as follows.

```
level:ERROR
location:test.TestLog4jAppender.main(TestLog4jAppender.java:18)
message:test log4j appender
thread:main
time:2016-05-27T03:15+0000
```

where:

- level indicates the log level.
- location indicates the location of the log printing statement in code.
- message indicates the log content.
- thread indicates the thread name.
- time indicates the log printing time.

Highlights of LogHub Log4j Appender

- Logs collected by the agent will not be flushed into disk. Generated data is sent directly to Log Service through the network.
- For applications using Log4j to record logs, you only need to modify the configuration file to transfer logs to Log Service.
- LogHub Log4j Appender merges logs and sends logs at the same time, improving the network I/O efficiency.

Usage

Step 1: Introduce dependency to the Maven project.

```
<dependency>
```

```
<groupId>com.google.protobuf</groupId>
<artifactId>protobuf-java</artifactId>
<version>2.5.0</version>
</dependency>
<dependency>
<groupId>com.aliyun.openservices</groupId>
<artifactId>log-loghub-log4j-appender</artifactId>
<version>0.1.3</version>
</dependency>
```

Step 2: Modify the log4j.properties file and configure the root logger by using the following syntax. If the file does not exist, create it in the root directory of the project.

```
log4j.rootLogger = [level] , appenderName1, appenderName2, ...
```

Where:

- level indicates the log level. This is classified into ERROR, WARN, INFO, and DEBUG in descending order of priority. By defining a level, you enable the output of logs for this level in application programs. For example, if you set level to INFO, all the DEBUG-level logs in the application programs will not be printed.
- appenderName indicates the log output destination. You can specify multiple output destinations. Each appender belongs to a specific type with configuration parameters provided.

Configuration using the LogHub Appender:

```
log4j.rootLogger=WARN,loghub
log4j.appender.loghub = com.aliyun.openservices.log.log4j.LoghubAppender
log4j.appender.loghub.projectName = [you project]
log4j.appender.loghub.logstore = [you logstore]
log4j.appender.loghub.endpoint = [your project endpoint]
log4j.appender.loghub.accessKeyId = [your accesskey id]
log4j.appender.loghub.accessKey = [your accesskey]
```

The content enclosed in brackets are required. The following section provides parameter definitions.

Configuration parameters

The LogHub Log4j Appender provides the following configuration parameters. When the optional parameters are not set, the default values are applied.

```
#(Required) Log Service project name
log4j.appender.loghub.projectName = [you project]
#(Required) Log Service Logstore name
log4j.appender.loghub.logstore = [you logstore]
#(Required) Log Service HTTP address
log4j.appender.loghub.endpoint = [your project endpoint]
```

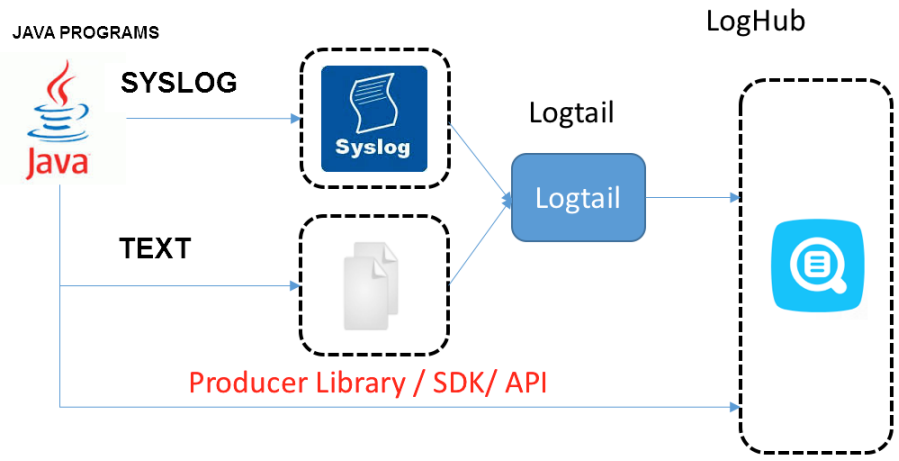
```
#(Required) User identity
log4j.appender.loghub.accessKeyId = [your accesskey id]
log4j.appender.loghub.accessKey = [your accesskey]
#This parameter is required when a temporary identity is used. If a non-temporary identity is used, this line of configuration can be deleted.
log4j.appender.loghub.stsToken=[your ststoken]
#(Optional) Timeout time (in milliseconds) for sending cached logs. Cached logs are sent immediately after the timeout time has elapsed.
log4j.appender.loghub.packageTimeoutInMS=3000
#(Optional) Maximum number of logs contained in each cached log packet. The parameter value must not exceed 4,096.
log4j.appender.loghub.logsCountPerPackage=4096
#(Optional) Maximum size (in bytes) of each cached log packet. The parameter value must not exceed 5 MB.
log4j.appender.loghub.logsBytesPerPackage = 5242880
#(Optional) Maximum memory (in bytes) available for appender instances. The default value is 100 MB.
log4j.appender.loghub.memPoolSizeInByte=1048576000
#(Optional) Number of I/O threads used to send log packets in the background. The default value is 1.
log4j.appender.loghub.ioThreadsCount=1
# (Optional) Time format of the logs that are output to Log Service. The time is formatted using SimpleDateFormat of Java. The default value is ISO8601.
log4j.appender.loghub.timeFormat=yyyy-MM-dd'T'HH:mmZ
log4j.appender.loghub.timeZone=UTC
```

LogHub Producer Library is a LogHub class library written for high concurrency Java applications. Producer Library and Consumer Library are used for LogHub read and write packaging to lower the threshold for data collection and consumption.

Problems solved by LogHub Producer Library:

- Logs collected by agents are not flushed into a disk. Once the data is generated, it is directly sent to the server through the network.
- High concurrency write operations on the client. For example, there are more than one hundred write operations within one second.
- Client computing logically separated from input/output (I/O). Logging does not affect the computing time used.

In the above scenarios, the Producer Library helps you reduce program development costs, aggregate write requests in batches, and asynchronously send them to the LogHub server. During the process, you can configure parameters for batch aggregation, server exception processing logic, and so on.



Comparison of the above access methods:

Access method	Advantages/disadvantages	Applicable scenarios
Log flushed into a disk + Logtail	Log collection decoupled from logging, no need to modify the code	Common scenarios
syslog + Logtail	Good performance (80 MB/S), the log is not flushed into the disk, but the syslog protocol needs to be supported	Syslog scenario
SDK direct transmission	Not flushed into the disk, and directly sent to the server; switching between the network IO and program IO needs to be properly processed	Log is not flushed into the disk
Producer Library	Not flushed into the disk, asynchronously merged and sent to the server, with good throughput	Log is not flushed into the disk and the client QPS is high

Note: Producer Library only supports the Java version, any other languages need to be developed.

Functions of the LogHub Producer Library

- Provides an asynchronous sending interface, thus ensuring thread security.
- The configuration of multiple projects can be added.
- The number of network IO threads used for sending can be configured.
- The number and size of logs merged into package can be configured.
- Memory usage is controllable. When the memory usage reaches the threshold value configured by the user, the send interface of producer will be blocked until there is idle memory.

Usage

The producer is used to perform the following steps:

Step 1: Add dependencies to the Maven project

```
<dependency>
<groupId>com.google.protobuf</groupId>
<artifactId>protobuf-java</artifactId>
<version>2.5.0</version>
</dependency>
<dependency>
<groupId>com.aliyun.openservices</groupId>
<artifactId>log-loghub-producer</artifactId>
<version>0.1.4</version>
</dependency>
```

Step 2： Configure ProducerConfig in the program

Parameters are defined below:

```
public class ProducerConfig
{
//Sending timeout time of the cached log, unit is in milliseconds. If the cache times out, the log will be sent immediately.
public int packageTimeoutInMS = 3000;
//The maximum quantity of logs contained in each cached log, which cannot exceed 4096.
public int logsCountPerPackage = 4096;
//The upper limit of the size of each cached log package, the unit is in bytes, and cannot exceed 5 MB.
public int logsBytesPerPackage = 5 * 1024 * 1024;
//The upper limit of memory that can be used by a single producer instance, the unit is in bytes.
public int memPoolSizeInByte = 1000 * 1024 * 1024;
//The maximum number of threads in the IO thread pool, used to send data to Log Service.
public int maxIOThreadSizeInPool = 50;
//If the specified shardhash method is used to send the log, this parameter needs to be set; otherwise, ignore this parameter. The backend merge thread will merge the data mapped to the same shard, and the shard is associated with a hash interval.
//During processing, the producer will map the hash transmitted by the user to the minimum value of the Hash interval associated with the shard. The producer regularly pulls the Hash interval associated with each shard from LogHub. The parameter `shardHashUpdateIntervalInMS` defines that the hash interval of the shard is updated in milliseconds.
public int shardHashUpdateIntervalInMS = 10 * 60 * 1000;
//The number of retries if sending fails. If the number of retries exceeds this value, the exception will be used as the parameter of callback and handed over to the user for handling.
public int retryTimes = 3;
}
```

Step 3： Inherit ILogCallback

Callback is used to process the log sending result, including successful sending and exception. You can choose not to process the result. In this way, it is not required to inherit ILogCallback.

Step 4： Create a producer instance and call the send interface to send data

Example

main:

```
public class ProducerSample {
private final static int ThreadsCount = 25;

public static String RandomString(int length) {
String str = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
Random random = new Random();
StringBuffer buf = new StringBuffer();
for (int i = 0; i < length; i++) {
int num = random.nextInt(62);
buf.append(str.charAt(num));
}
return buf.toString();
}

public static void main(String args[]) throws InterruptedException {
ProducerConfig producerConfig = new ProducerConfig();
//Use the default configuration to create a producer instance
final LogProducer producer = new LogProducer(producerConfig);
//Add configurations of multiple projects
producer.setProjectConfig(new ProjectConfig("your project 1",
"endpoint", "your accesskey id", "your accesskey"));
producer.setProjectConfig(new ProjectConfig("your project 2",
"endpoint", "your accesskey id", "your accesskey",
"your sts token"));
//Update the configuration of project 1
producer.setProjectConfig(new ProjectConfig("your project 1",
"endpoint", "your new accesskey id", "your new accesskey"));
//Delete the configuration of project 2
producer.removeProjectConfig("your project 2");
//Generate a log collection, used for testing
final Vector<Vector<LogItem>> logGroups = new Vector<Vector<LogItem>>();
for (int i = 0; i < 100000; ++i) {
Vector<LogItem> tmpLogGroup = new Vector<LogItem>();
LogItem logItem = new LogItem((int) (new Date().getTime() / 1000));
logItem.PushBack("level", "info" + System.currentTimeMillis());
logItem.PushBack("message", "test producer send perf "
+ RandomString(50));
logItem.PushBack("method", "SenderToServer " + RandomString(10));
tmpLogGroup.add(logItem);
logGroups.add(tmpLogGroup);
}
//Call the send interface concurrently to send the log
Thread[] threads = new Thread[ThreadsCount];
for (int i = 0; i < ThreadsCount; ++i) {
threads[i] = new Thread(null, new Runnable() {
Random random = new Random();

public void run() {
```

```
int j = 0, rand = random.nextInt(99999);
while (++j < Integer.MAX_VALUE) {
producer.send("project 1", "logstore 1", "topic",
"source ip", logGroups.get(rand),
new CallbackSample("project 1", "logstore 1", "topic", "source ip", null, logGroups.get(rand), producer));
}
}
}, i + "");
threads[i].start();
}
//Wait for the sending thread to quit
Thread.sleep(1 * 60 * 60 * 1000);
//Actively refresh the cached log that has not been sent
producer.flush();
//Disables the background IO thread; the close action will send the data cached in the memory at the time of
calling.
producer.close();
}
}
```

callback:

```
public class CallbackSample extends ILogCallback {
//Save the data to be sent; retry if an exception occurs
public String project;
public String logstore;
public String topic;
public String shardHash;
public String source;
public Vector<LogItem> items;
public LogProducer producer;
public int retryTimes = 0;
public CallbackSample(String project, String logstore, String topic,
String shardHash, String source, Vector<LogItem> items, LogProducer producer) {
super();
this.project = project;
this.logstore = logstore;
this.topic = topic;
this.shardHash = shardHash;
this.source = source;
this.items = items;
this.producer = producer;
}

public void onCompletion(PutLogsResponse response, LogException e) {
if (e != null) {
//Print the exception
System.out.println(e.GetErrorCode() + ", " + e.GetErrorMessage() + ", " + e.GetRequestId());
//Three retries at most
if(retryTimes++ < 3)
{
producer.send(project, logstore, topic, source, shardHash, items, this);
}
}
else{
```

```
System.out.println("send success, request id: " + response.GetRequestId());
}
}

}
```

Common log formats

The Apache log format and directory are specified in the /etc/apache2/httpd.conf configuration file.

Log format

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
```

The above statements define two log formats: combined and common.

The following statement writes logs into the specified file in the combined format.

```
CustomLog "/var/log/apache2/access_log" combined
```

Sample log:

```
192.168.1.2 - - [02/Feb/2016:17:44:13 +0800] "GET /favicon.ico HTTP/1.1" 404 209 "http://localhost/x1.html"
"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.97
Safari/537.36"
```

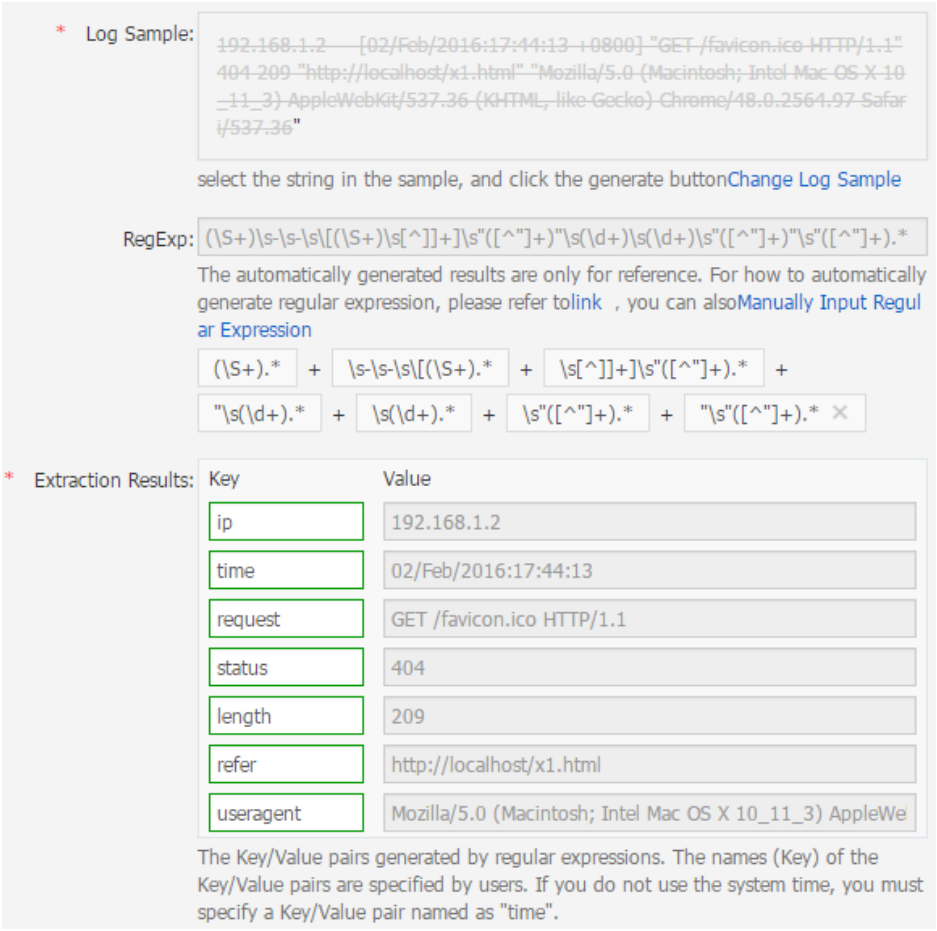
Apache log format details:

```
%a remote_ip
%A local_ip
%B size
%b size
%D time_taken_ms
%f filename
%h remote_host
%H protocol
%l ident
%m method
%p port
%P pid
"%q" url_query
"%r" request
```

```
%s status
%>s status
%t time
%T time_taken
%u remote_user
%U url_stem
%v server_name
%V canonical_name
%I bytes_received
%O bytes_sent
"%{User-Agent}i" user_agent
"%{Referer}i" referer
```

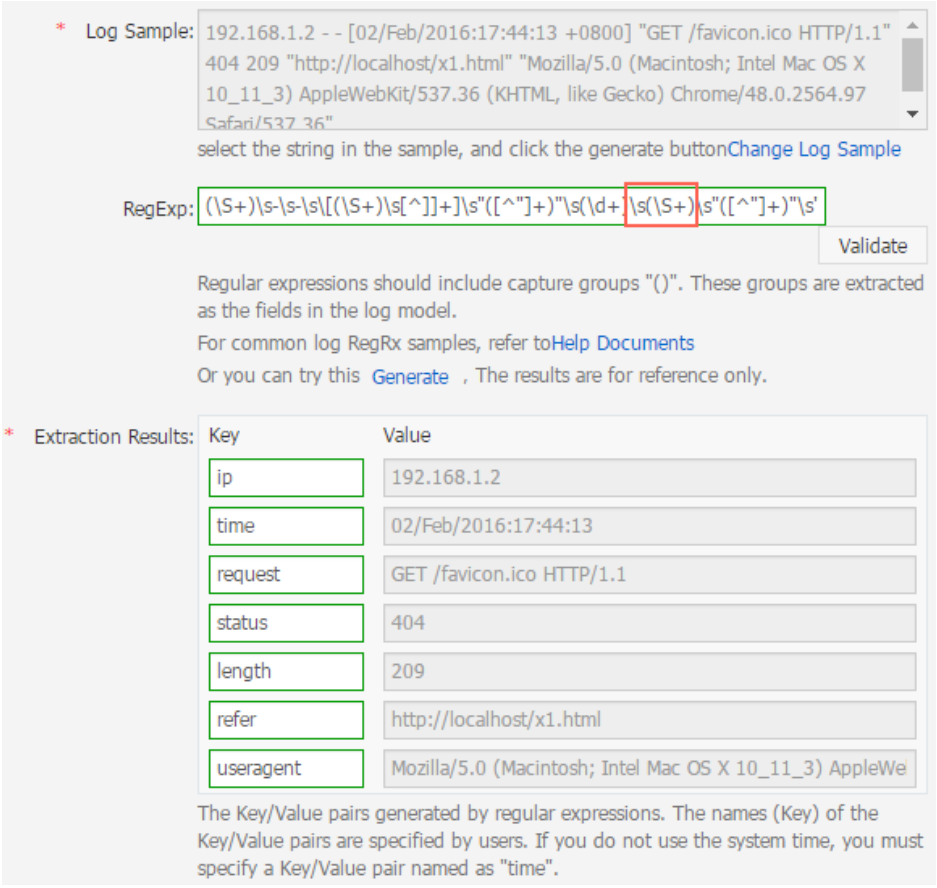
Apache log collection by Log Service

For details about the standard process, refer to **Quick start**. Manually edit the regular expression after it is automatically generated.



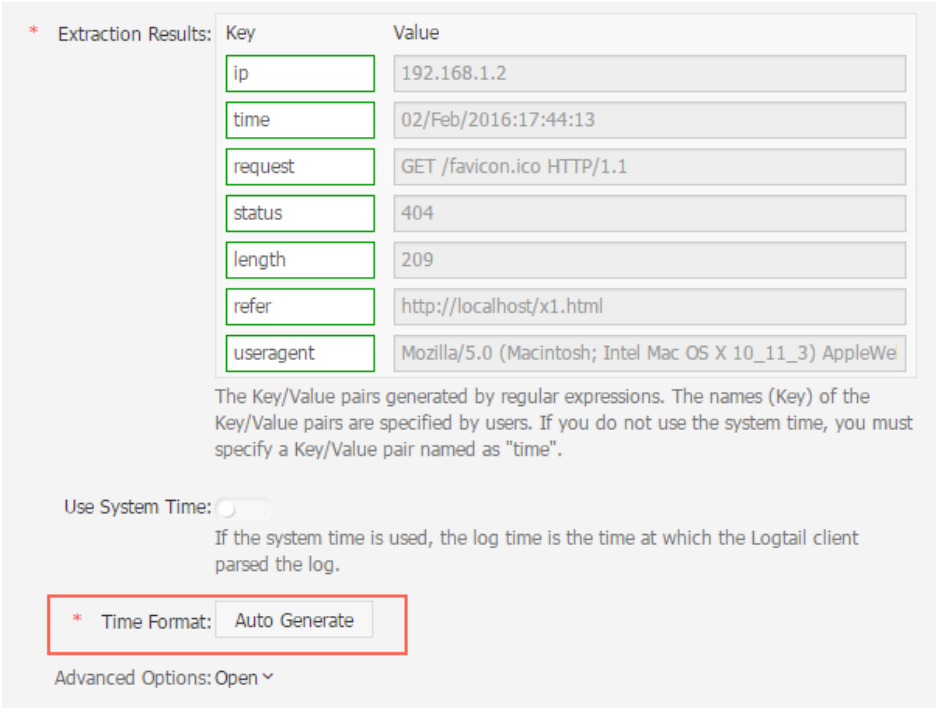
After all fields are extracted, set the name of each field and adjust the regular expression for general applicability.

Click **Manual Input Regular Expression** to adjust as follows.



The length field is numeric, but can be filled in with a hyphen. In this case, replace the matched result (\d+) with (\S+). If other fields do not belong to the defined type, make similar replacements.

After replacing, click **Validate**. If the regular expression is correct, extracted results are displayed. Manually adjust the regular expression if it is incorrect.



Assign a defined field to each extracted result. For example, name a field “time”. Click **Auto Generate** next to **Time Format**. Then click **Next**.

After Logtail configuration is completed, push the configuration to the client.

The Nginx log format and directory are specified in the /etc/nginx/nginx.conf configuration file.

Log format

```
log_format main  '$remote_addr - $remote_user [$time_local] "$request" '
'$request_time $request_length '
'$status $body_bytes_sent "$http_referer" '
'"$http_user_agent"';
```

Above, the **main** log format is defined.

The following statement defines how to use the **main** log format and defines the name of the file where to write the log data.

```
access_log /var/logs/nginx/access.log main
```

Sample log:

```
192.168.1.2 - - [10/Jul/2015:15:51:09 +0800] "GET /ubuntu.iso HTTP/1.0" 0.000 129 404 168 "-" "Wget/1.11.4 Red Hat modified"
```

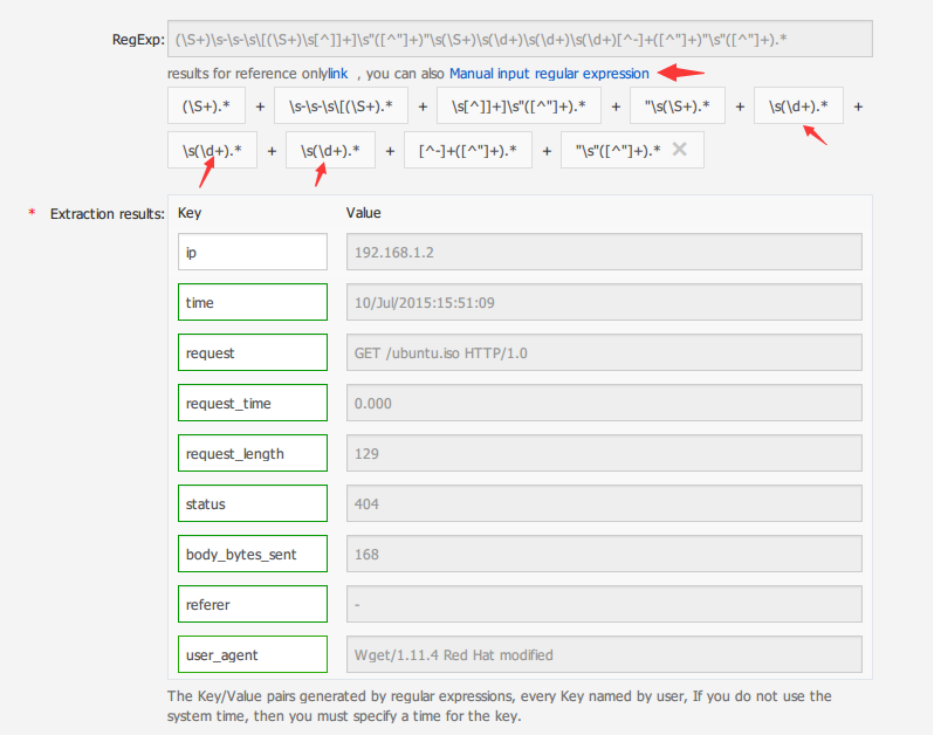
Nginx log format details:

Field	Explanation
remoteaddr	The IP address of the agent.
remote_user	The username of the agent.
request	The requested URL and HTTP protocol.
status	The request status.
bodybytessent	The number of bytes (not including the size of the response header) sent to the agent. This variable is used with bytes_sent in modlogconfig of the Apache module.
connection	The connection serial number.
connection_requests	The number of requests received over a connection.
msec	The log write time, measured in seconds and precise to milliseconds.
pipe	Whether requests are sent via the HTTP

	pipeline. p indicates requests are sent via the HTTP pipeline. . indicates requests are not sent via the HTTP pipeline.
httpreferer	The webpage link from which access is directed.
“http_user_agent”	Information about the agent’ s browser. http_user_agent must be enclosed by double quotation marks.
requestlength	The length of a request, including the request line, request header, and request body.
request_time	The request processing time, which is measured in seconds and precise to milliseconds. The time starts when the first byte is sent to the agent and ends when the logs are written after the last character is sent to the agent.
[$time_local]	The local time when the general log format is applied. This variable must be enclosed in brackets.

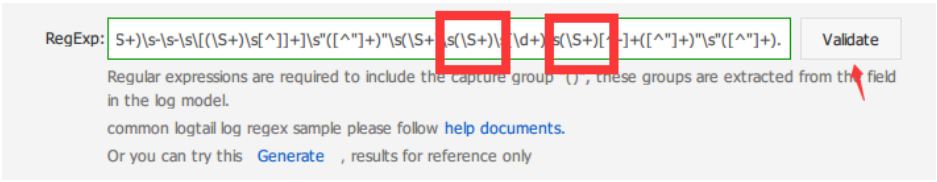
Nginx log collection by Log Service

For details about the standard process, refer to Quick start. Manually adjust the regular expression after it is automatically generated.



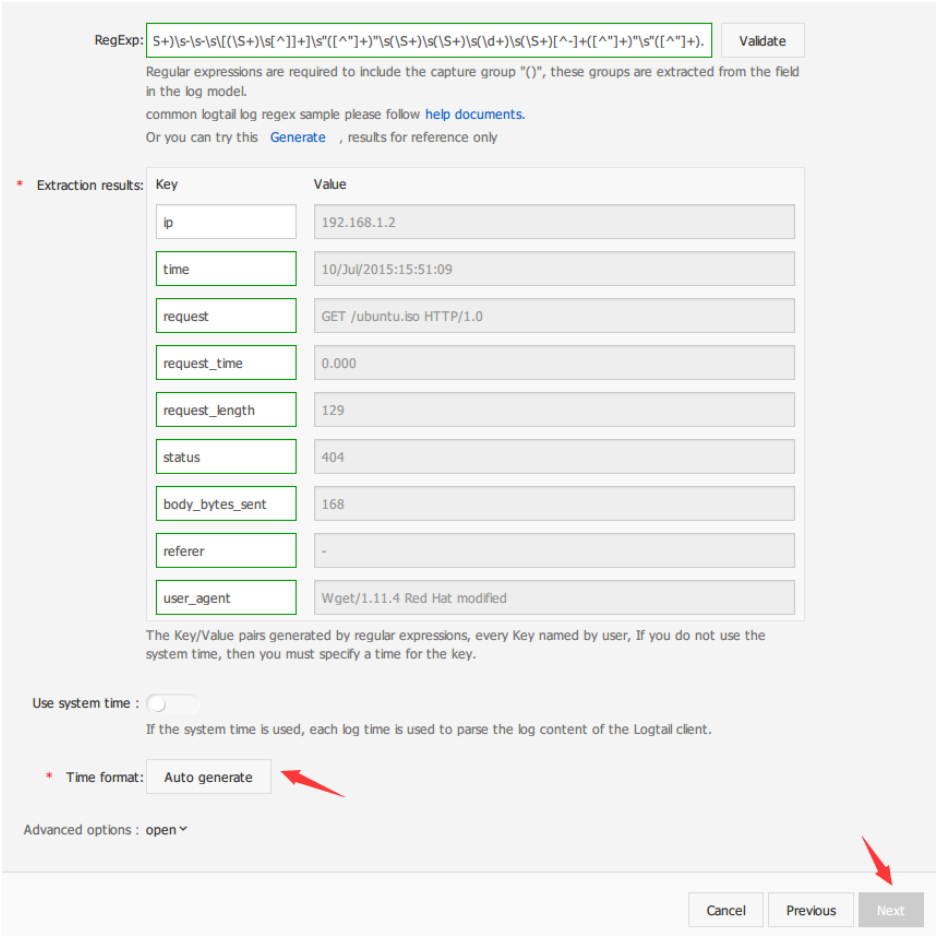
After the regular expression generator extracts all the fields, set the name of each field and adjust the

regular expression for general applicability. Click **Manual Input Regular Expression** to adjust as follows.



The request_length and body_bytes_sent fields are numeric, but can be filled in with a hyphen. In this case, replace the matched result (\d+) with (\S+). If other fields do not belong to the defined type, make similar replacements.

After replacing, click **Validate**. If the regular expression is correct, extracted results are displayed. Adjsut the regular expression if it is incorrect.



Regular Expression for this case :

```
(\S+)\s-\s-\s\[(\S+)\s[^\]]+\]\s"(\S+)\s(\S+)\s(\S+)"\s(\S+)\s(\S+)\s(\S+)\s(\S+)\s"([^"]+)"\s"([^"]+)"
```

Python logging module provides a general logging system for use by third-party modules or applications. The logging module provides different log levels and records logs by different methods

including file, HTTP GET/POST, SMTP, and Socket. You can customize a log recording method as needed. The logging module is the same as Log4j except that they have different implementation details. The logging module provides the logger, handler, filter, and formatter features.

Python log format

The formatter specifies the log record output format. The formatter construction mode are defined by two parameters: message format string and message date string. The parameters are optional.

Python sample log:

```
import logging
import logging.handlers

LOG_FILE = 'tst.log'

handler = logging.handlers.RotatingFileHandler(LOG_FILE, maxBytes = 1024*1024, backupCount = 5) # Instantiate the handler
fmt = '%(asctime)s - %(filename)s:%(lineno)s - %(name)s - %(message)s'

formatter = logging.Formatter(fmt) # Instantiate the formatter
handler.setFormatter(formatter) # Add the formatter to the handler

logger = logging.getLogger('tst') # Obtain the logger named tst
logger.addHandler(handler) # Add the handler to the logger
logger.setLevel(logging.DEBUG)

logger.info('first info message')
logger.debug('first debug message')
```

Output sample log:

```
2015-03-04 23:21:59,682 - log_test.py:16 - tst - first info message
2015-03-04 23:21:59,682 - log_test.py:17 - tst - first debug message
```

The formatter is configured in the %(key)s format, that is, replacing the dictionary keywords. The following keywords are provided.

Format	Description
%(name)s	Name of the logger (logging channel).
%(levelno)s	Numeric logging level for the message (DEBUG, INFO, WARNING, ERROR, CRITICAL).
%(levelname)s	Text logging level for the message ('DEBUG', 'INFO', 'WARNING', 'ERROR', 'CRITICAL').
%(pathname)s	Full path of the source file where the logging call was issued (if available).
%(filename)s	File name portion of the path name.

%(module)s	Module (name portion of the file name).
%(funcName)s	Name of function containing the logging call.
%(lineno)d	Source line number where the logging call was issued (if available).
%(created)f	Time when the LogRecord was created, in UNIX standard time format and seconds calculated from 1970-1-1 00:00:00 UTC.
%(relativeCreated)d	Time in milliseconds when the LogRecord was created, relative to the time the logging module was loaded.
%(asctime)s	Human-readable time when the LogRecord was created. By default this is of the form “2003-07-08 16:49:45,896” (the numbers after the comma are millisecond portion of the time).
%(msecs)d	Millisecond portion of the time when the LogRecord was created.
%(thread)d	Thread ID (if available).
%(threadName)s	Thread name (if available).
%(process)d	Process ID (if available).
%(message)s	The logged message.

Python log collection by Logtail

For details about the standard process, refer to Quick start.

The automatically generated regular expression is based on the sample log and does not cover every log type. Therefore, you need to tune the regular expression after it is generated.

A regular expression and other information for a common Python log is shown below.

Log sample:

```
2016-02-19 11:03:13,410 - test.py:19 - tst - first debug message
```

Regular expression:

```
(\d+-\d+-\d+\s\S+)\s+-\s+([^:]+):(\d+)\s+-\s+(\w+)\s+-\s+(.*)
```

Log format:

```
```

```
%(asctime)s - %(filename)s:%(lineno)s - %(levelno)s %(levelname)s %(pathname)s %(module)s
%(funcName)s %(created)f %(thread)d %(threadName)s %(process)d %(name)s - %(message)s
```

Log output:

```
2016-02-19 11:06:52,514 - test.py:19 - 10 DEBUG test.py test <module> 1455851212.514271
139865996687072 MainThread 20193 tst - first debug message
```

Regular expression:

```
(\d+-\d+-\d+\s\S+)\s-\s([^:]+):(\d+)\s+-
\s+(\d+)\s+(\w+)\s+(\S+)\s+(\w+)\s+(\S+)\s+(\S+)\s+(\d+)\s+(\w+)\s+(\d+)\s+(\w+)\s+-\s+(.*)
```

Log collection methods

Log Service supports collection of log4j logs in two ways:

- By using LogHub log4j Appender
- By using Logtail to collect log4j log files

LogHub log4j Appender

For detailed introductions and operations, refer to log4j appender.

Log collection by Logtail

The following shows the sample log of log4j default format printed to a file.

```
2013-12-25 19:57:06,954 [10.207.37.161] WARN impl.PermanentTairDaoImpl - Fail to Read Permanent
Tair,key:e:470217319319741_1,result:com.example.tair.Result@172e3ebc[rc=code=-1, msg=connection error or
timeout,value=,flag=0]
```

Matching of the beginning of a line in multiline logs (the beginning of a line is expressed by IP information):

```
\d+-\d+-\d+\s.*
```

Regular expression used to extract log information:

```
(\d+-\d+-\d+\s\d+:\d+:\d+,\d+)\s\[([^\]]*)\]\s(\S+)\s+(\S+)\s-\s(.*)
```

Time conversion format:

```
%Y-%m-%d %H:%M:%S
```

Sample log extraction result:

Key	Value
time	2013-12-25 19:57:06,954
ip	10.207.37.161
level	WARN
class	impl.PermanentTairDaoImpl
message	Fail to Read Permanent Tair,key:e:470217319319741_1,result:com.example.tair.Result@172e3ebc[rc=code=-1, msg=connection error or timeout,value=,flag=0]

By default, Node.js logs are printed to the console, making data collection and troubleshooting inconvenient. The Log4js package is provided to print logs to files and customize log formats.

```
var log4js = require('log4js');
log4js.configure({
appenders: [
{
type: 'file', //file that is output
filename: 'logs/access.log',
maxLogSize: 1024,
backups:3,
category: 'normal'
}
]
});
var logger = log4js.getLogger('normal');
logger.setLevel('INFO');
logger.info("this is a info msg");
logger.error("this is a err msg");
```

The logs outputted to a file are shown below.

```
[2016-02-24 17:42:38.946] [INFO] normal - this is a info msg
[2016-02-24 17:42:38.951] [ERROR] normal - this is a err msg
```

Log4js has six output levels: trace, debug, info, warn, error, and fatal.

Node.js log collection by Log Service

Sample 1

Log sample

```
[2016-02-24 17:42:38.946] [INFO] normal - this is a info msg
```

Regular expression

```
\[([^]]+)]\s\[([^\]]+)]\s(\w+)\s-(.*)
```

Extracted fields

time, level, loggerName, message

Sample 2

Log sample

```
[2016-01-31 12:02:25.844] [INFO] access - 42.120.73.203 - - "GET /user/projects/ali_sls_log?ignoreError=true
HTTP/1.1" 304 - "http://
aliyun.com/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/48.0.2564.97 Safari/537.36"
```

Regular expression

```
\[([^]]+)]\s\[(\w+)]\s(\w+)\s-\s(\S+)\s-\s-\s"([^"]+)"\s(\d+)[^"]+("[^"]+)"\s"([^"]+).*
```

Extracted fields

time, level, loggerName, ip, request, status, referer, user_agent

Default WordPress log format

Raw sample log:

```
172.64.0.2 - - [07/Jan/2016:21:06:39 +0800] "GET /wp-admin/js/password-strength-meter.min.js?ver=4.4 HTTP/1.0"
200 776 "http://wordpress.c4a1a0aecdb1943169555231dcc4adfb7.cn-hangzhou.alicontainer.com/wp-
admin/install.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/47.0.2526.106 Safari/537.36"
```

Matching of the beginning of a line in multiline log (the beginning of a line is expressed by IP

information):

```
\d+\.\d+\.\d+\.\d+\s-\s.*
```

Regular expression used to extract log information:

```
(\S+) - - \[([^\]]*)] "(\S+) ([^"]+)" (\S+) (\S+) "([^"]+)" "([^"]+)"
```

Time conversion format:

```
%d/%b/%Y:%H:%M:%S
```

Sample log extraction results:

Key	Value
ip	127.64.0.2
time	07/Jan/2016:21:06:39 +0800
method	GET
url	/wp-admin/js/password-strength-meter.min.js?ver=4.4 HTTP/1.0
status	200
length	776
ref	http://wordpress.c4a1a0aecdb1943169555231dcc4adfb7.cn-hangzhou.alicontainer.com/wp-admin/install.php
user-agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.106 Safari/537.36

Delimiter logs use linefeeds as the boundary. Each natural line is a log.

The fields of each log are connected by fixed separators (for example, tabs, spaces, bars, commas, semicolons, and other single characters). Fields containing separators are enclosed by a pair of double quotation marks.

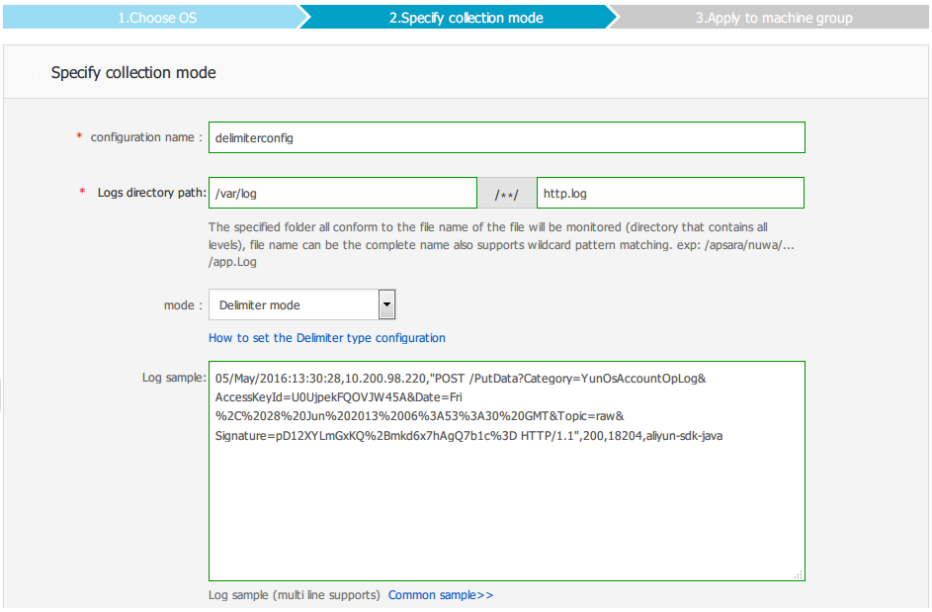
Common delimiter logs include CSV and TSV.

Sample log

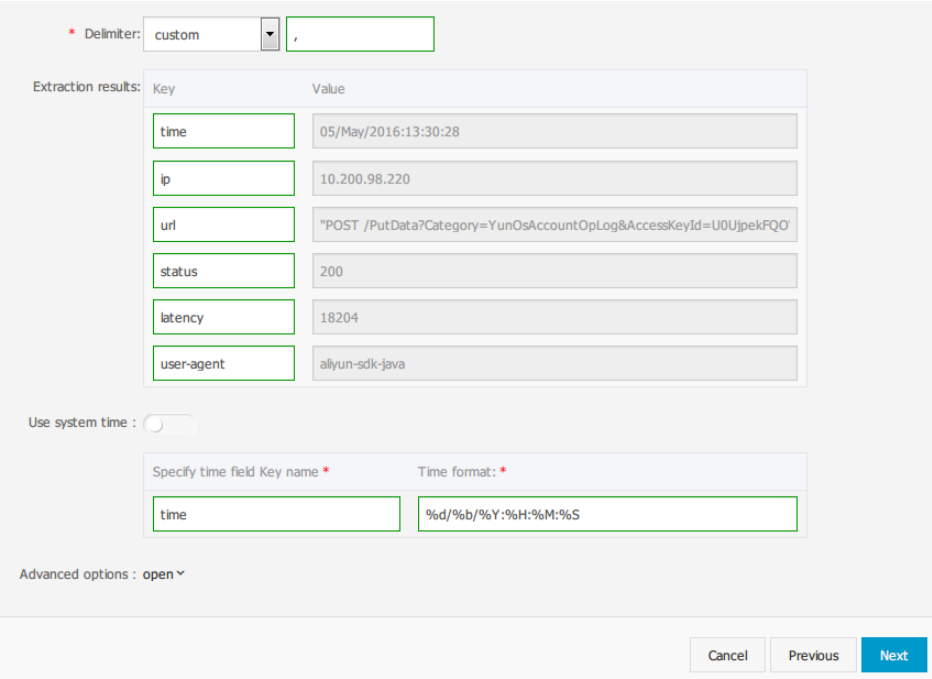
```
05/May/2016:13:30:28,10.200.98.220,"POST
/PutData?Category=YunOsAccountOpLog&AccessKeyId=U0UjpekFQOVJW45A&Date=Fri%2C%2028%20Jun%2020
```

```
13%2006%3A53%3A30%20GMT&Topic=raw&Signature=pD12XYLmGxKQ%2Bmkd6x7hAgQ7b1c%3D
HTTP/1.1",200,18204,aliyun-sdk-java
05/May/2016:13:31:23,10.200.98.221,"POST
/PutData?Category=YunOsAccountOpLog&AccessKeyId=U0UjpekFQOVJW45A&Date=Fri%2C%2028%20Jun%2020
13%2006%3A53%3A30%20GMT&Topic=raw&Signature=pD12XYLmGxKQ%2Bmkd6x7hAgQ7b1c%3D
HTTP/1.1",401,23472,aliyun-sdk-java
```

Collection configuration



The preceding sample log uses commas as separators and contains six columns: time, ip, url, status, latency, and user-agent.



You can use the system time as the time of a log or use a column of the log as the time (for example,

the time field 05/May/2016:13:30:29). For date format setup, refer to Logtail date formats.

Additional log format information

Separator

A delimiter log is divided into several fields by separators. The format requirements are as follows:

- Separators must be single characters, such as tabs (\t), spaces, bars (|), commas (,), and semicolons (;).
- Separators of the multi-character type are not allowed, for example, || and &&&.
- Double quotation marks (“) are used as quotes, but not separators.

Quote

Fields containing separators are enclosed by a pair of double quotation marks. Use an escape character preceding a double quotation mark not used to enclose a field.

Double quotation marks used as quotes

When commas are used as separators, quotes must be located adjacent to the separators. Modify the format if there are spaces, tabs, and other characters between them. A sample log is shown below.

```
1997,Ford,E350,"ac, abs, moon",3000.00
```

The log is parsed into five fields.

Field 1	Field 2	Field 3	Field 4	Field 5
1997	Ford	E350	ac, abs, moon	3000.00

Processing of double quotation marks in log fields as escape characters

In the following sample log, the third field contains double quotation marks not used as quotes.

```
1999,Chevy,"Venture ""Extended Edition, Very Large""","",5000.00
```

The log is parsed into five fields:

Field 1	Field 2	Field 3	Field 4	Field 5
1999	Chevy	Venture “Extended Edition, Very Large”		5000.00

That is, the double quotation marks are either used separately as quotes on the boundary of a field or used in a pair (“”) as data within a field. For other cases that are not compliant with the delimiter

log format definition, parse fields in other modes (such as simple mode and regular mode).

A JSON formatted log can be written in two types of structures:

- Object: A collection of name–value pairs.
- Array: An ordered list of values.

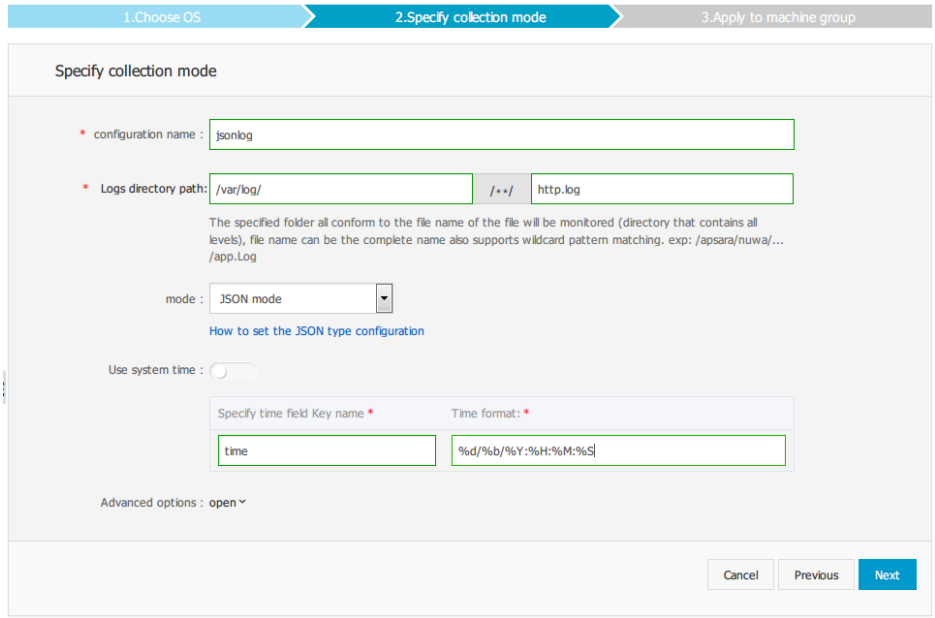
Logtail supports JSON logs of the object type. Logtail automatically extracts the keys and values from the first layer of an object as the names and values of fields, respectively. The field value belongs to the object, array, or basic type, for example, a string or number.

Logtail does not support automatic parsing of non-object data (for example, JSON arrays). You can use regular expressions for field extraction or use the simple mode for log collection by line.

Sample log

```
{"url": "POST
/PutData?Category=YunOsAccountOpLog&AccessKeyId=U0UjpekFQOVJW45A&Date=Fri%2C%2028%20Jun%2020
13%2006%3A53%3A30%20GMT&Topic=raw&Signature=pD12XYLmGxKQ%2Bmkd6x7hAgQ7b1c%3D HTTP/1.1",
"ip": "10.200.98.220", "user-agent": "aliyun-sdk-java", "request": {"status": "200", "latency": "18204"}, "time":
"05/May/2016:13:30:28"}
{"url": "POST
/PutData?Category=YunOsAccountOpLog&AccessKeyId=U0UjpekFQOVJW45A&Date=Fri%2C%2028%20Jun%2020
13%2006%3A53%3A30%20GMT&Topic=raw&Signature=pD12XYLmGxKQ%2Bmkd6x7hAgQ7b1c%3D HTTP/1.1",
"ip": "10.200.98.210", "user-agent": "aliyun-sdk-java", "request": {"status": "200", "latency": "10204"}, "time":
"05/May/2016:13:30:29"}
```

Collection configuration



You can use the system time as the time of a log or use a field value within a JSON object as the time (for example, the time field 05/May/2016:13:30:29). For date format setup, refer to Logtail date formats.

ThinkPHP is a Web application development framework based on the PHP language.

ThinkPHP sample log

The following logging method is used in ThinkPHP:

```
<?php
Think\Log::record('D model class not found for method instantiation' );
?>
```

The printed log using this method is shown below:

```
[ 2016-05-11T21:03:05+08:00 ] 30.9.181.163 /index.php
INFO: [ app_init ] --START--
INFO: Run Behavior\BuildLiteBehavior [ RunTime:0.000014s ]
INFO: [ app_init ] --END-- [ RunTime:0.000091s ]
INFO: [ app_begin ] --START--
INFO: Run Behavior\ReadHtmlCacheBehavior [ RunTime:0.000038s ]
INFO: [ app_begin ] --END-- [ RunTime:0.000076s ]
INFO: [ view_parse ] --START--
INFO: Run Behavior\ParseTemplateBehavior [ RunTime:0.000068s ]
INFO: [ view_parse ] --END-- [ RunTime:0.000104s ]
INFO: [ view_filter ] --START--
INFO: Run Behavior\WriteHtmlCacheBehavior [ RunTime:0.000032s ]
INFO: [ view_filter ] --END-- [ RunTime:0.000062s ]
INFO: [ app_end ] --START--
INFO: Run Behavior\ShowPageTraceBehavior [ RunTime:0.000032s ]
INFO: [ app_end ] --END-- [ RunTime:0.000070s ]
ERR: D model class not found for method instantiation
```

ThinkPHP log collection by Log Service

ThinkPHP logs are multiline logs in varying modes. The following fields can be extracted from these logs: time, IP address of the visitor, accessed URL, and printed message. Because the message mode is not fixed, the message is packaged into a field containing multiple lines of information.

In Log Service, select multiline mode and fill in the following regular expression at the beginning of the line.

```
\[\s\d+-\d+-\w+:\d+:\d+\+\d+:\d+\s.*
```

Regular expression:

```
\[\s(\d+-\d+-\w+:\d+:\d+)[^:]+:\d+\s]\s+(\S+)\s(\S+)\s+(.*)
```

Time expression:

```
%Y-%m-%dT%H:%M:%S
```

Sample log

View IIS log configurations, select the W3C format (default field setting), and save the format to put it into effect.

```
2016-02-25 01:27:04 112.74.74.124 GET /goods/list/0/1.html - 80 - 66.249.65.102
Mozilla/5.0+(compatible;+Googlebot/2.1;++http://www.google.com/bot.html) 404 0 2 703
```

Collection configuration

```
input {
file {
type => "iis_log_1"
path => ["C:/inetpub/logs/LogFiles/W3SVC1/*.log"]
start_position => "beginning"
}
}

filter {
if [type] == "iis_log_1" {
#ignore log comments
if [message] =~ "^#" {
drop {}
}

grok {
# check that fields match your IIS log settings
match => ["message", "%{TIMESTAMP_ISO8601:log_timestamp} %{IPORHOST:site} %{WORD:method}
%{URIPATH:page} %{NOTSPACE:querystring} %{NUMBER:port} %{NOTSPACE:username} %{IPORHOST:clienthost}
%{NOTSPACE:useragent} %{NUMBER:response} %{NUMBER:subresponse} %{NUMBER:scstatus}
%{NUMBER:time_taken}"]
}

date {
match => [ "log_timestamp", "YYYY-MM-dd HH:mm:ss" ]
timezone => "Etc/UTC"
}

useragent {
```

```
source=> "useragent"
prefix=> "browser"
}

mutate {
remove_field => [ "log_timestamp"]
}
}
}

output {
if [type] == "iis_log_1" {
logservice {
codec => "json"
endpoint => "***"
project => "***"
logstore => "***"
topic => ""
source => ""
access_key_id => "***"
access_key_secret => "***"
max_send_retry => 10
}
}
}
```

Note:

- The configuration file must be encoded as UTF-8 without BOM. You can download Notepad++ to modify the file encoding format.
- path indicates the file path, which must use Unix separators, for example, C:/test/multiline/*.log. Otherwise, fuzzy match is not supported.
- The type field must be modified in a unified manner and kept consistent across the file. If a machine has multiple Logstash configuration files, the type field in each file must be unique. Otherwise, data cannot be processed properly.

Related plug-ins: file and grok.

Restart Logstash to apply configurations

Create a configuration file in the conf directory and restart Logstash to apply the file. For details, refer to Collect Windows logs through Logstash.

Use the system time as the uploaded log time

Sample log

```
10.116.14.201,-,2/25/2016,11:53:17,W3SVC7,2132,200,0,GET,project/shenzhen-
```

```
test/logstore/logstash/detail,C:\test\csv\test_csv.log
```

Collection configuration

```
input {
file {
type => "csv_log_1"
path => ["C:/test/csv/*.log"]
start_position => "beginning"
}
}

filter {
if [type] == "csv_log_1" {
csv {
separator => ","
columns => ["ip", "a", "date", "time", "b", "latency", "status", "size", "method", "url", "file"]
}
}
}

output {
if [type] == "csv_log_1" {
logservice {
codec => "json"
endpoint => "***"
project => "***"
logstore => "***"
topic => ""
source => ""
access_key_id => "***"
access_key_secret => "***"
max_send_retry => 10
}
}
}
```

Note:

- The configuration file must be encoded as UTF-8 without BOM. You can download Notepad++ to modify the file encoding format.
- path indicates the file path, which must use Unix separators, for example, C:/test/multiline/*.log. Otherwise, fuzzy match is not supported.
- The type field must be modified in a unified manner and kept consistent across the file. If a machine has multiple Logstash configuration files, the type field in each file must be unique. Otherwise, data cannot be processed properly.

Related plug-ins: file and csv.

Restart Logstash to apply configurations

Create a configuration file in the conf directory and restart Logstash to apply the file. For details, refer to Collect Windows logs through Logstash.

Use the log field content as the uploaded log time

Sample log

```
10.116.14.201,-,Feb 25 2016 14:03:44,W3SVC7,1332,200,0,GET,project/shenzhen-test/logstore/logstash/detail,C:\test\csv\test_csv_withtime.log
```

Collection configuration

```
input {
file {
type => "csv_log_2"
path => ["C:/test/csv_withtime/*.log"]
start_position => "beginning"
}
}

filter {
if [type] == "csv_log_2" {
csv {
separator => ","
columns => ["ip", "a", "datetime", "b", "latency", "status", "size", "method", "url", "file"]
}
date {
match => [ "datetime" , "MMM dd YYYY HH:mm:ss" ]
}
}
}

output {
if [type] == "csv_log_2" {
logservice {
codec => "json"
endpoint => "***"
project => "***"
logstore => "***"
topic => ""
source => ""
access_key_id => "***"
access_key_secret => "***"
max_send_retry => 10
}
}
}
```

Note:

- The configuration file must be encoded as UTF-8 without BOM. You can download

Notepad++ to modify the file encoding format.
- path indicates the file path, which must use Unix separators, for example, C:/test/multiline/*.log. Otherwise, fuzzy match is not supported.
- The type field must be modified in a unified manner and kept consistent across the file. If a machine has multiple Logstash configuration files, the type field in each file must be unique. Otherwise, data cannot be processed properly.

Related plug-ins: file, csv, and date.

Restart Logstash to apply configurations

Create a configuration file in the conf directory and restart Logstash to apply the file. For details, refer to Collect Windows logs through Logstash.

Sample log

```
2016-02-25 15:37:01 [main] INFO com.aliyun.sls.test_log4j - single line log
2016-02-25 15:37:11 [main] ERROR com.aliyun.sls.test_log4j - catch exception!
java.lang.ArithmeticException: / by zero
at com.aliyun.sls.test_log4j.divide(test_log4j.java:23) ~[bin/:?]
at com.aliyun.sls.test_log4j.main(test_log4j.java:13) [bin/:?]
2016-02-25 15:38:02 [main] INFO com.aliyun.sls.test_log4j - normal log
```

Collection configuration

```
input {
file {
type => "common_log_1"
path => ["C:/test/multiline/*.log"]
start_position => "beginning"
codec => multiline {
pattern => "^\d{4}-\d{2}-\d{2} \d{2}:\d{2}:\d{2}"
negate => true
auto_flush_interval => 3
what => previous
}
}
}

output {
if [type] == "common_log_1" {
logservice {
codec => "json"
endpoint => "***"
project => "***"
logstore => "***"
topic => ""
source => ""
```

```
access_key_id => "***"
access_key_secret => "***"
max_send_retry => 10
}
}
}
```

Note:

- The configuration file must be encoded as UTF-8 without BOM. You can download Notepad++ to modify the file encoding format.
- path indicates the file path, which must use Unix separators, for example, C:/test/multiline/*.log. Otherwise, fuzzy match is not supported.
- The type field must be modified in a unified manner and kept consistent across the file. If a machine has multiple Logstash configuration files, the type field in each file must be unique. Otherwise, data cannot be processed properly.
- For a singleline log file, remove the codec => multiline line.

Restart Logstash to apply configurations

Create a configuration file in the conf directory and restart Logstash to apply the file. For details, refer to Collect Windows logs through Logstash.

Unity3D is an integrated game development tool compatible with multiple platforms. Developed by Unity Technologies, this tool allows a player to easily create various interactive content such as 3D video game, architectural visualization, and real-time 3D animation.

You can use the Web Tracking function of Log Service to collect Unity3D logs conveniently. Collection of Unity Debug.Log is used as an example below to explain how to collect Unity logs into Log Service.

Step 1 Activate the Web Tracking function

For details, refer to Tracking function of Log Service.

Step 2 Register Unity3D LogHandler

Create a C# file LogOutputHandler.cs in the Unity editor, copy the following code, and modify the three member variables in it, which are the name of the log project, name of the log library (logstore), and the address of the log project (serviceAddr). Here, serviceAddr can be found in Official Documentation of Log Service.

```
using UnityEngine;
using System.Collections;

public class LogOutputHandler : MonoBehaviour
{
```

```
//Register the HandleLog function on scene start to fire on debug.log events
public void OnEnable()
{
Application.logMessageReceived += HandleLog;
}

//Remove callback when object goes out of scope
public void OnDisable()
{
Application.logMessageReceived -= HandleLog;
}

string project = "your project name";
string logstore = "your logstore name";
string serviceAddr = "http address of your log service project";

//Capture debug.log output, send logs to Loggly
public void HandleLog(string logString, string stackTrace, LogType type)
{
string parameters = "";
parameters += "Level=" + WWW.EscapeURL(type.ToString());
parameters += "&";
parameters += "Message=" + WWW.EscapeURL(logString);
parameters += "&";
parameters += "Stack_Trace=" + WWW.EscapeURL(stackTrace);
parameters += "&";
//Add any User, Game, or Device MetaData that would be useful to finding issues later
parameters += "Device_Model=" + WWW.EscapeURL(SystemInfo.deviceModel);

string url = "http://" + project + "." + serviceAddr + "/logstores/" + logstore + "/track?APIVersion=0.6.0&" +
parameters;
StartCoroutine(SendData(url));
}

public IEnumerator SendData(string url)
{
WWW sendLog = new WWW(url);
yield return sendLog;
}
}
```

The above code can be used to send logs to Alibaba Cloud Log Service asynchronously. In the example, you can add more fields to be collected.

Step 3 Generate the Unity log

In the project, create the LogglyTest.cs file, and add the following code.

```
using UnityEngine;
using System.Collections;

public class LogglyTest : MonoBehaviour {
```

```
void Start () {
Debug.Log ("Hello world");
}
}
```

Step 4 View logs on the Log Service console

After the above steps are completed, run the Unity program. You can see the logs that you have sent on the Log Service console.

Conclusion

The above example provides the methods for collecting the Debug.Log or logs such as Debug.LogError and Debug.LogException. The component object model of the Unity and its program crash API, and other types of Log APIs can be used to conveniently collect the device information on the client.

Real-time comsumption

Logs collected to the LogHub of Log Service can be consumed in the following three methods.

Method	Scenario	Real-time support	Storage period
Real-time consumption (LogHub)	Stream Compute and real-time computing	Real-time (< 10 ms)	365 days (to request a longer storage period, please submit a ticket to Alibaba Cloud Technical Support)
Index query (LogSearch)	Online query of recent hot data	Real-time (1s [in 99.9% cases] to 3s)	365 days (contact us for a longer storage period)
Post storage (LogShipper)	Full log storage for offline analysis	5–30 minutes	The storage period depends on the specific storage system.

Real-time consumption

Logs are consumed after they are written. Log consumption and log query require log read capability. Logs in a shard are consumed as follows.

1. Obtain a cursor based on a set of criteria, such as time, Begin, and End.
2. The system reads logs based on the cursor and step, and returns the next cursor.
3. Continuously move the cursor to consume logs.

In addition to basic APIs, Log Service provides SDKs, Storm spout, Spark client, and Web console for log consumption.

- Log consumption by the Spark client: Refer to EMR SDK/Log Service chapter.
- Log consumption by the Storm spout: Log Service provides LogHub Storm spout to enable the interconnection between Storm and LogHub.
- Log consumption by Stream Compute: Interconnection with LogHub is enabled by a data source created in Stream Compute.Log consumption by the LogHub client lib: The LogHub Consumer Library is an advanced mode of automatic shard allocation when multiple LogHub consumers simultaneously consume Logstores.
- Log consumption by SDKs: Log Service provides SDKs in multiple languages (Java and Python) with support for log consumption APIs. For more information about SDKs, refer to Log Service SDKs.
- Access log statistics image: Free Docker image for real-time analysis of common logs.

Index query

- Log query on the Log Service console: Refer to Log query.
- Log query using the SDKs or APIs of Log Service: Log Service provides HTTP-enabled REST APIs. The APIs support a full featured log query. For details, refer to Log Service APIs.

Shipping and storage

- Ship to OSS: stores logs for a long period or analyzes logs with E-MapReduce.
- Ship to Table Store: stores logs through Table Store (NoSQL).

The Log Service console provides a dedicated preview page to directly preview a portion of the logs on the Logstore in your browser.

Procedure

Log on to the Log Service console.

Select the desired project and click the project name or **Manage** on the right.

On the **Logstore List** page, select the desired Logstore and click **Preview**.

Specify the Logstore shard to be queried and the log time range. Then, click **Preview**.

The first 10 packets in the specified time interval are displayed.

View the consumption progress on the console

For more details of viewing the consumption progress on the console, please refer to Document description.

By default, Spark Streaming and Storm spout use a consumer library to consume LogHub data.

Application scenarios

The LogHub Consumer Library is an advanced mode of automatic shard allocation when multiple LogHub consumers simultaneously consume Logstores.

For example, the LogHub Consumer Library automatically handles the shard load balancing logic and consumer failover among multiple consumers in Storm and Spark. You can then focus on your business development without concern for shard allocation, checkpoints, and failover.

For example, three consumption instances A, B, and C are enabled for Stream Compute through Storm. When there are 10 shards, the system allocates three shards each to A and B and four shards to C. Then,

- If A is down, the system allocates the data not consumed by A to B and C through load balancing mode. When A recovers, the system balances out the loads of A, B, and C.
- If two consumption instances named D and E are added, the system performs load balancing to ensure that each instance consumes two shards.
- If shards are merged or split, the system performs load balancing to account for the merged or split shards.
- After read-only shards are consumed, the system performs load balancing to account for the remaining shards.

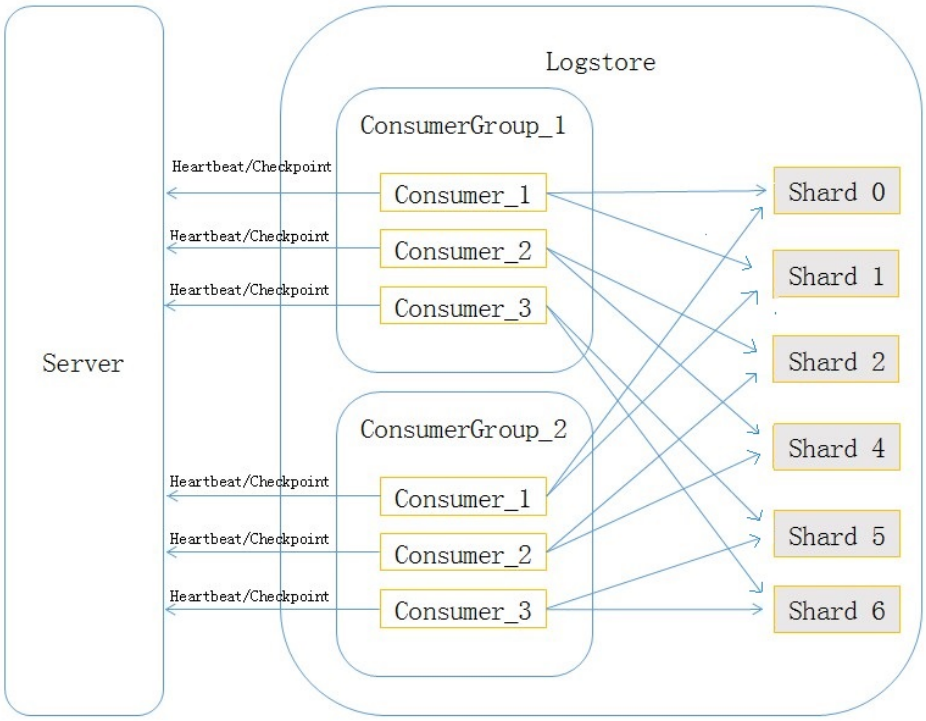
No missing or duplicate data occurs in the preceding processes. You only need to complete the following coding:

1. Set configuration parameters.
2. Write the log processing code.
3. Enable consumption instances.

The LogHub Consumer Library allows you to focus on data processing without concern for issues such as load balancing, consumption breakpoint saving, sequential consumption, and consumption exception handling. Therefore, the LogHub Consumer Library is highly recommended for data consumption.

Terms

The LogHub Consumer Library has four important concepts: consumer group, consumer, heartbeat, and checkpoint. Their relationships are described as follows.



consumer group

Consumer groups are sub-resources of Logstores. Consumers with the same consumer group name consume data from the same Logstore, but the data consumed by each consumer is different. Up to five consumer groups can be created under one Logstore, but the group names must be unique within the Logstore. Different consumer groups under the same Logstore consume data independently.

```
{
"order":boolean,
"timeout": integer
}
```

- order: Indicates whether to consume data with identical keys sequentially based on the write time.

- timeout: Indicates the timeout time (measured in seconds) for consumers in a consumer group. If a consumer does not send a heartbeat packet within the required timeout time, the consumer is considered to be timed out and Log Service determines that the consumer will go offline.

consumer

Each consumer is allocated with several shards and can consume data in these shards. Consumers in the same consumer group must have unique names.

heartbeat

Consumers must periodically send keep-alive heartbeat packets to Log Service.

checkpoint

Consumers must periodically save the endpoint of their shard consumption to Log Service. After a shard is transferred from one consumer to another, the target consumer obtains the shard consumption breakpoint from Log Service and consumes data starting from the breakpoint.

LogHub Consumer Library usage

Implement the following two interface classes of the LogHub Consumer Library:

- ILogHubProcessor: Each shard corresponds to an instance, and each instance consumes the data of a specific shard.
- ILogHubProcessorFactory: Generates the instance implementing the ILogHubProcessor interface.

Set parameters.

Enable one or more ClientWorker instances.

Maven address

```
<dependency>
<groupId>com.google.protobuf</groupId>
<artifactId>protobuf-java</artifactId>
<version>2.5.0</version>
</dependency>
<dependency>
```

```
<groupId>com.aliyun.openservices</groupId>
<artifactId>loghub-client-lib</artifactId>
<version>0.6.5</version>
</dependency>
```

Example

Main function

```
public static void main(String args[])
{
LogHubConfig config = new LogHubConfig(...);

ClientWorker worker = new ClientWorker(new SampleLogHubProcessorFactory(), config);

Thread thread = new Thread(worker);
//The ClientWorker instance runs automatically after the thread is executed and extends the Runnable interface.
thread.start();
Thread.sleep(60 * 60 * 1000);
//The shutdown function of the ClientWorker instance is called to exit the consumption instance. The associated thread is stopped automatically.
worker.shutdown();
//Multiple asynchronous tasks are generated when the ClientWorker instance is running. You are advised to wait 30s until all tasks are exited after shutdown.
Thread.sleep(30 * 1000);
}
```

ILogHubProcessor and ILogHubProcessorFactory implementation samples

ILogHubProcessor indicates the consumption instance corresponding to a specific shard. Note the data consumption logic during the development process. The same ClientWorker instance consumes data serially, and only one ILogHubProcessor instance is generated. The ClientWorker instance calls the shutdown function of ILogHubProcessor upon exit.

```
public class SampleLogHubProcessor implements ILogHubProcessor
{
private int mShardId;
// Records the last persistent checkpoint time.
private long mLastCheckTime = 0;

public void initialize(int shardId)
{
mShardId = shardId;
}

// Master logic of data consumption
public String process(List<LogGroupData> logGroups,
ILogHubCheckPointTracker checkPointTracker)
```

```
{
for(LogGroupData logGroup: logGroups)
{
LogGroup lg = logGroup.GetLogGroup();
System.out.println("source ip:" + lg.getSource());
System.out.println("topic: " + lg.getTopic());
for(Log log: lg.getLogsList())
{
StringBuilder content = new StringBuilder();
content.append(log.getTime() + "\t");
for(Content cont: log.getContentsList())
{
content.append(cont.getKey() + "=" + cont.getValue()+ "\t");
}
System.out.println(content.toString());
}
}
long curTime = System.currentTimeMillis();
// Writes checkpoints to the Log Service every 60s. If a ClientWorker instance crashes during the 60s period,
// the new ClientWorker instance consumes data starting from the last checkpoint. Duplicate data may exist.
if (curTime - mLastCheckTime > 60 * 1000)
{
try
{
//When the parameter is set to true, checkpoints are immediately updated to Log Service. When the parameter is
set to false, checkpoints are locally cached. The default update interval is 60s.
//The background updates checkpoints to Log Service.
checkPointTracker.saveCheckPoint(true);
}
catch (LogHubCheckPointException e)
{
e.printStackTrace();
}
mLastCheckTime = curTime;
}
else
{
try
{
checkPointTracker.saveCheckPoint(false);
}
catch (LogHubCheckPointException e)
{
e.printStackTrace();
}
}
// "null" indicates that data is properly processed. If you need to roll back to the last checkpoint for retry, you can
return checkPointTracker.getCheckpoint().
return null;
}
// The ClientWorker instance calls this function upon exit, during which you can perform cleanup.
public void shutdown(ILogHubCheckPointTracker checkPointTracker)
{
//Saves the consumption breakpoint to Log Service.
try {
checkPointTracker.saveCheckPoint(true);
```

```
} catch (LogHubCheckPointException e) {
e.printStackTrace();
}
}
}
```

ILogHubProcessorFactory is used to generate ILogHubProcessor.

```
public class SampleLogHubProcessorFactory implements ILogHubProcessorFactory
{
public ILogHubProcessor generatorProcessor()
{
// Generates a consumption instance.
return new SampleLogHubProcessor();
}
}
```

Configuration

```
public class LogHubConfig
{
//Default interval when the ClientWorker instance pulls data.
public static final long DEFAULT_DATA_FETCH_INTERVAL_MS = 200;
//Name of a consumer group. The name can be 3 to 63 characters in length, and can include English letters, digits, underscores (_), and hyphens (-). It cannot be null and must begin and end with lowercase letters or numbers.
private String mConsumerGroupName;
//Name of a consumer. Consumers in the same consumer group must have unique names.
private String mWorkerInstanceName;
//Address of the LogHub data interface.
private String mLogHubEndPoint;
//Project name
private String mProject;
//LogStore name
private String mLogStore;
//Access key ID of the cloud account.
private String mAccessId;
//Access key of the cloud account.
private String mAccessKey;
//Indicates the consumption start point of a shard when the shard's checkpoint is not recorded in Log Service. The value does not take effect if Log Service records valid checkpoint information. mCursorPosition can be set to BEGIN_CURSOR, END_CURSOR, or SPECIAL_TIMER_CURSOR. BEGIN_CURSOR indicates that consumption starts from the first data entry of the shard. END_CURSOR indicates that consumption starts from the last data entry at the current time. SPECIAL_TIMER_CURSOR is used in conjunction with mLoghubCursorStartTime to indicate a specific data consumption start time.
private LogHubCursorPosition mCursorPosition;
//When mCursorPosition is set to SPECIAL_TIMER_CURSOR, data consumption will start at a specific time measured in seconds.
private int mLoghubCursorStartTime = 0;
// The interval (measured in milliseconds) of LogHub data acquisition by polling. The smaller the interval, the faster data is extracted. The default value is DEFAULT_DATA_FETCH_INTERVAL_MS. It is recommended to set the interval to a value greater than 200 ms.
private long mDataFetchIntervalMillis;
```

```
// The interval (measured in milliseconds) when the ClientWorker instance sends heartbeat packets to Log Service.
The recommended value is 10,000 ms.
private long mHeartBeatIntervalMillis;
//Sequential consumption or not.
private boolean mConsumeInOrder;
}
```

Important notes

consumerGroupName in LogHubConfig indicates a consumer group. Consumers with the same consumerGroupName are differentiated by workerInstance name. They consume the shards of the same Logstore.

```
Assume that a LogStore has four shards numbered from 0 to 3.
There are three ClientWorker instances with the following consumerGroupName and workerinstance name settings:
<consumer_group_name_1 , worker_A> ,
<consumer_group_name_1 , worker_B> ,
<consumer_group_name_2 , worker_C>
The ClientWorker instances are allocated with the following shards:
<consumer_group_name_1 , worker_A>: shard_0, shard_1
<consumer_group_name_1 , worker_B>: shard_2, shard_3
<consumer_group_name_2 , worker_C>: Shard_0, shard_1, shard_2, shard_3 # The ClientWorker instances with different consumer group names consume data independently.
```

Ensure that the ILogHubProcessor process () interface is correctly implemented and exited.

The saveCheckPoint () interface of ILogHubCheckPointTracker indicates that the data processing is complete, regardless of whether the transferred parameter is set to true or false. If the parameter is set to true, checkpoint persistency is implemented immediately in Log Service. If the parameter is set to false, checkpoints are synchronized to Log Service every 60s.

RAM authorization is required if the access key ID and access key secret of a sub-account are configured in LogHubConfig.

Action	Resource
log:GetCursorOrData	acs:log:**${regionName}**:**${projectOwnerAliUid}**:project/**${projectName}**/logstore/**${logstoreName}**
log:CreateConsumerGroup	acs:log:**${regionName}**:**${projectOwnerAliUid}**:project/**${projectName}**/logstore/**${logstoreName}**/consumergroup/*
log:ListConsumerGroup	acs:log:**${regionName}**:**${projectOwnerAliUid}**:project/**${projectName}**/logstore/**${logstoreN**

	ame}/consumergroup/*
log:ConsumerGroupUpdateCheckPoint	acs:log:**${regionName}**:**${projectOwnerAliUid}**:project/**${projectName}**/logstore/**${logstoreName}**/consumergroup/**${consumerGroupName}**
log:ConsumerGroupHeartBeat	acs:log:**${regionName}**:**${projectOwnerAliUid}**:project/**${projectName}**/logstore/**${logstoreName}**/consumergroup/**${consumerGroupName}**
log:GetConsumerGroupCheckPoint	acs:log:**${regionName}**:**${projectOwnerAliUid}**:project/**${projectName}**/logstore/**${logstoreName}**/consumergroup/**${consumerGroupName}**

Consumer group is an advanced real-time consumption data mode. It can provide automatic Logstore consumption load balancing for multiple consumer instances. Both Spark Streaming and Storm use consumer group as their basic mode.

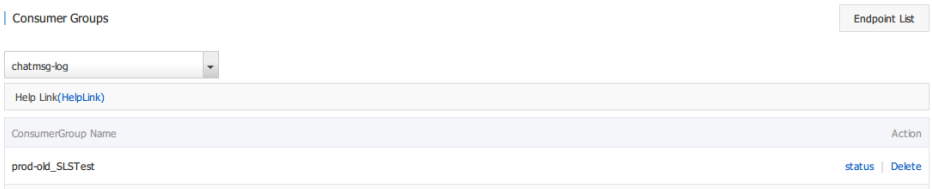
View consumption progress on the console

Log on to the Log Service console.

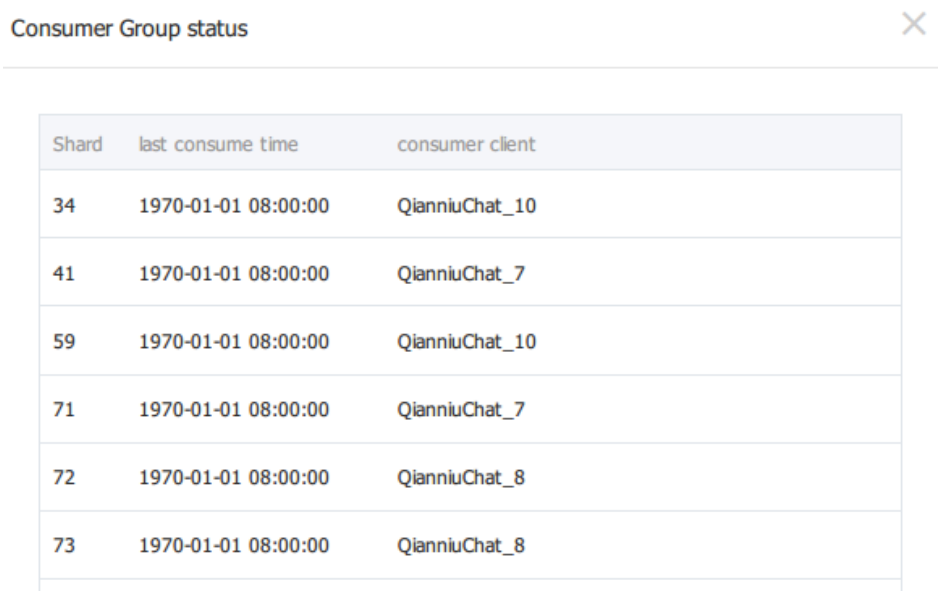
Select the desired project and click the project name or **Manage** on the right.

In the left navigation bar, click **LogHub - Consume** > **Consumer Group**.

On the **Consumer Groups** page, select a Logstore to see whether the collaborative consumption function has been activated.



After selecting the specified consumer group, click **Consumption Status** to view the data consumption progress for each shard.



As shown in the preceding figure, the page shows 6 shards of this Logstore, corresponding to 3 consumers. Here, the most recent data consumption time is shown for each consumer. With the data consumption time, you can determine if the current data processing can keep up with data production. If processing lags behind (that is, data consumption is slower than data production), it is recommended you increase the number of consumers.

Using API/SDKs to view consumption progress

Java SDK is used as an example to show how to get the consumption status using API.

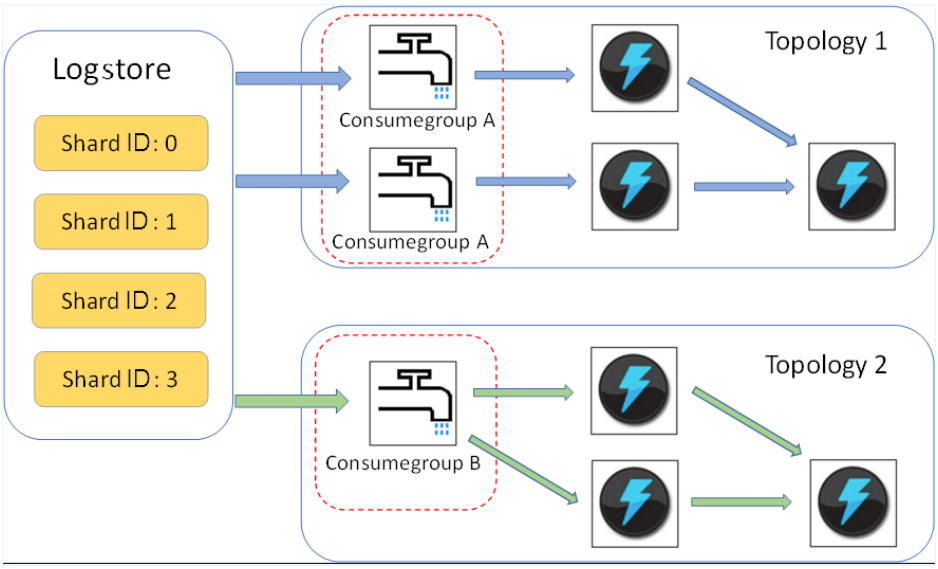
```
package test;

import java.util.ArrayList;

import com.aliyun.openservices.log.Client;
import com.aliyun.openservices.log.common.Consts.CursorMode;
import com.aliyun.openservices.log.common.ConsumerGroup;
import com.aliyun.openservices.log.common.ConsumerGroupShardCheckPoint;
import com.aliyun.openservices.log.exception.LogException;

public class ConsumerGroupTest {
static String endpoint = "";
static String project = "";
static String logstore = "";
static String accesskeyId = "";
static String accesskey = "";
public static void main(String[] args) throws LogException {
Client client = new Client(endpoint, accesskeyId, accesskey);
//Retrieve all consumer groups for this Logstore. If no consumer groups exist, the consumerGroups length is 0.
ArrayList<ConsumerGroup> consumerGroups;
try{
consumerGroups = client.ListConsumerGroup(project, logstore).GetConsumerGroups();
}
```

```
catch(LogException e){
if(e.GetErrorCode() == "LogStoreNotExist")
System.out.println("this logstore does not have any consumer group");
else{
//internal server error branch
}
return;
}
for(ConsumerGroup c: consumerGroups){
//Print consumer group attributes, including names, heartbeat timeout, and consumption order.
System.out.println("Name: " + c.getConsumerGroupName());
System.out.println("Heartbeat timeout: " + c.getTimeout());
System.out.println("Consumption order: " + c.isInOrder());
for(ConsumerGroupShardCheckPoint cp: client.GetCheckPoint(project, logstore,
c.getConsumerGroupName()).GetCheckPoints()){
System.out.println("shard: " + cp.getShard());
//Reformat the returned time to be accurate to the millisecond in long integer.
System.out.println("Last data consumption time: " + cp.getUpdateTime());
System.out.println("Consumer name: " + cp.getConsumer());

String consumerPrg = "";
if(cp.getCheckPoint().isEmpty())
consumerPrg = "Consumption not started";
else{
//Unix timestamp, in seconds; note the format during output.
try{
int prg = client.GetPrevCursorTime(project, logstore, cp.getShard(), cp.getCheckPoint()).GetCursorTime();
consumerPrg = "" + prg;
}
catch(LogException e){
if(e.GetErrorCode() == "InvalidCursor")
consumerPrg = "Invalid, the previous consumption time has exceeded the data lifecycle in the LogStore";
else{
//internal server error
throw e;
}
}
}
System.out.println("Consumption progress: " + consumerPrg);

String endCursor = client.GetCursor(project, logstore, cp.getShard(), CursorMode.END).GetCursor();
int endPrg = 0;
try{
endPrg = client.GetPrevCursorTime(project, logstore, cp.getShard(), endCursor).GetCursorTime();
}
catch(LogException e){
//do nothing
}
//Unix timestamp, in seconds; note the format during output
System.out.println("Arrival time of last line of data: " + endPrg);
}
}
}

}
```

LogHub of Log Service provides an efficient and reliable log channel for collecting log data through Logtail and SDKs. You can access real-time systems such as Spark Streaming and Storm to consume the data written to LogHub.

The LogHub Storm spout feature reads data from LogHub in real time, reducing Storm users' cost for LogHub consumption.

Basic architecture and process



- In the preceding figure, LogHub Storm spouts are enclosed in the red dotted boxes. Each Storm topology has a group of spouts to read data from a Logstore. The spouts in different topologies are independent of each other.
- Each topology is identified by a unique LogHub consumer group name. The LogHub client lib is used for load balancing and automatic failover among the spouts in the same topology.
- Spouts read data from LogHub in real time, send data to the bolt nodes of the topology, and periodically save consumption endpoints as checkpoints to the LogHub server.

Note:

- To prevent misuse, each Logstore supports up to five consumer groups. You can use the DeleteConsumerGroup interface of the Java SDK to delete unused consumer groups.
- It is recommended to have an equal number of spouts and shards. Otherwise, a single spout may be unable to process a large amount of data.
- If a shard contains a large amount of data exceeding the processing capability of a single spout, you can use the shard split interface to reduce the per-shard data volume.
- Dependency on the Storm ACK mechanism is mandatory in LogHub spouts to confirm that spouts correctly send messages to bolts. Therefore, bolts must call ACK for such confirmation.

Example

Spout (used for topology creation)

```
public static void main( String[] args )
{
String mode = "Local"; // Uses the local test mode.

String conumser_group_name = ""; // Each topology must be assigned a unique consumer group name. The name can be 3 to 63 characters in length, and can include English letters, digits, underscores (_), and hyphens (-). It cannot be null and must begin and end with lowercase letters or numbers.
String project = ""; // Project of the Log Service
String logstore = ""; // LogStore of the Log Service
String endpoint = ""; // Domain of the Log Service
String access_id = ""; // User's access key
String access_key = "";

// Configurations required for creating a LogHub Storm spout.
LogHubSpoutConfig config = new LogHubSpoutConfig(conumser_group_name,
endpoint, project, logstore, access_id,
access_key, LogHubCursorPosition.END_CURSOR);

TopologyBuilder builder = new TopologyBuilder();

// Creates a LogHub Storm spout.
LogHubSpout spout = new LogHubSpout(config);

// In the actual condition, the number of spouts may be equal to the number of LogStore shards.
builder.setSpout("spout", spout, 1);
builder.setBolt("exclaim", new SampleBolt()).shuffleGrouping("spout");


Config conf = new Config();
conf.setDebug(false);
conf.setMaxSpoutPending(1);

// The serialization method LogGroupDataSerializSerializer of LogGroupData must be configured explicitly when Kryo is used for data serialization and deserialization.
Config.registerSerialization(conf, LogGroupData.class, LogGroupDataSerializSerializer.class);

if (mode.equals("Local")) {
logger.info("Local mode...");

LocalCluster cluster = new LocalCluster();

cluster.submitTopology("test-jstorm-spout", conf, builder.createTopology());

try {
Thread.sleep(6000 * 1000); //waiting for several minutes
} catch (InterruptedException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
```

```
cluster.killTopology("test-jstorm-spout");
cluster.shutdown();

} else if (mode.equals("Remote")) {

logger.info("Remote mode...");

conf.setNumWorkers(2);
try {
StormSubmitter.submitTopology("stt-jstorm-spout-4", conf, builder.createTopology());
} catch (AlreadyAliveException e) {
// TODO Auto-generated catch block
e.printStackTrace();
} catch (InvalidTopologyException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
} else {
logger.error("invalid mode: " + mode);
}

}

}
```

Sample code of bolts that consume data (only the content of each log is printed)

```
public class SampleBolt extends BaseRichBolt {
private static final long serialVersionUID = 4752656887774402264L;

private static final Logger logger = Logger.getLogger(BaseBasicBolt.class);

private OutputCollector mCollector;

@Override
public void prepare(@SuppressWarnings("rawtypes") Map stormConf, TopologyContext context,
OutputCollector collector) {

mCollector = collector;
}

@Override
public void execute(Tuple tuple) {

String shardId = (String) tuple
.getValueByField(LogHubSpout.FIELD_SHARD_ID);

@SuppressWarnings("unchecked")
List<LogGroupData> logGroupDatas = (ArrayList<LogGroupData>)
tuple.getValueByField(LogHubSpout.FIELD_LOGGROUPS);

for (LogGroupData groupData : logGroupDatas) {
// Each LogGroup consists of one or more logs.
```

```
LogGroup logGroup = groupData.GetLogGroup();
for (Log log : logGroup.getLogsList()) {
StringBuilder sb = new StringBuilder();
// Each log has a time field and multiple key–value pairs.
int log_time = log.getTime();
sb.append("LogTime:").append(log_time);
for (Content content : log.getContentsList()) {
sb.append("\t").append(content.getKey()).append(":")
.append(content.getValue());
}
logger.info(sb.toString());
}
}
// The dependency on the Storm ACK mechanism is mandatory in LogHub spouts to confirm that spouts correctly
send messages
// to bolts. Therefore, bolts must call ACK for such confirmation.
mCollector.ack(tuple);
}


@Override
public void declareOutputFields(OutputFieldsDeclarer declarer) {
//do nothing
}
}
```

Maven

Use the following code for versions earlier than Storm 1.0 (for example, 0.9.6).

```
<dependency>
<groupId>com.aliyun.openservices</groupId>
<artifactId>loghub-storm-spout</artifactId>
<version>0.6.5</version>
</dependency>
```

Use the following code for Storm 1.0 and later versions.

```
<dependency>
<groupId>com.aliyun.openservices</groupId>
<artifactId>loghub-storm-1.0-spout</artifactId>
<version>0.1.2</version>
</dependency>
```

StreamCompute can be used to directly consume data in LogHub after data sources of the LogHub type are created.

```
CREATE STREAM TABLE source_test_galaxy ( $schema ) WITH ( type='loghub', endpoint=$endpoint,
accessId=$loghub_access_id, accessKey=$loghub_access_key, projectName=$project, logstore=$logstore );
```

Parameter list:

- $schema: The keys in logs mapped to the columns in the StreamCompute table, for example, name STRING, age STRING, id STRING.
- $endpoint: Data access point, that is, the access point in a specific region.
- $loghub_access_id: Access ID of the account (or sub-account) with the read permission.
- $loghub_access_key: Access key of the account (or sub-account) with the read permission.
- $project: Project where data is located.
- $logstore: Logstore where data is located.

Example:

```
CREATE STREAM TABLE source_test_galaxy ( name STRING, age STRING, id STRING ) WITH ( type='loghub',
endpoint='http://cn-hangzhou-intranet.log.aliyuncs.com', accessId='mock_access_id',
accessKey='mock_access_key', projectName='ali-cloud-streamtest', logstore='stream-test' );
```

Ship logs

To post data in LogHub to storage products, such as OSS, Table Store, or MaxCompute (coming soon), LogShipper provides a complete status API and automatic retry function. This means you only need to complete configuration on the console.

Source

Use the LogHub function to create a Logstore, which contains real-time data.

Output

- OSS (Object Storage Service):
    - Instructions
    - Procedure
    - The format on OSS can be processed through Hive. E-MapReduce is recommended.
- Table Store (NoSQL data storage service):
    - Operating Procedure

Scenario

- Interconnect to data warehouse

Advantages

- Large scale: Supports transfers of petabytes of data per day
- Stable: Automatic + API retry; real-time posting information is learned through API/Web
- Real-time: Minute-level latency
- Free of charge

LogShipper can automatically archive the logs in a Logstore to an OSS Bucket for activating such uses as:

- OSS allows you to configure the lifecycle for the data stored in it as needed. You can then set a long-term for storing logs.
- OSS data can be consumed through a user-defined program, as well as other systems (for example, E-MapReduce).

Using LogShipper to ship logs to an OSS Bucket provides advantages, such as:

Ease of use

You can use Log Service to synchronize the data in a Logstore to OSS after completing a simple configuration setup.

Avoid repetitive collections

Log collection involves the process of centralizing the logs. This means you do not have to collect the logs again before importing them to the OSS budget.

Full reuse of log grouping

LogShipper automatically ships logs in different Projects and Logstores to different OSS Bucket directories, facilitating data management.

Here, an example is used to illustrate how to use the OSS shipping function of Log Service. Assume there are two primary accounts of Alibaba Cloud user: A and B.

Note:

- A and B may be the same account. In this case, RAM authorization is recommended.
- Log Service project and OSS Bucket must be in the same region. Cross-region data shipping is not supported.

With primary account A, project-a is activated in the Shenzhen region of Log Service, and a Logstore is created to store the nginx access log.

With primary account B, bucket-b is created in the Shenzhen region of OSS.

The nginx access log of project-a of Log Service will be archived in the prefix-b directory of bucket-b.

To implement the OSS shipping function through Log Service, you need to perform two steps (RAM authorization and shipping rule configuration).

Step 1: Resource Access Management (RAM) authorization

Quick authorization

Use primary account B to log on to Alibaba Cloud console, and activate Resource Access Management (RAM) free of charge.

Click RAM quick authorization page and confirm the granting of permissions of writing all OSS Buckets. Log Service writes the OSS Bucket of B in place of A.

View and modify the role

Use primary account B to log on to Alibaba Cloud console Resource Access Management (RAM). Access **Role Management** and view the role (for quick authorization, the created role is AliyunLogDefaultRole by default).

If A and B are different accounts, go to **Change Role**. If A and B are the same account, the created role of quick authorization can be directly used.

Record the role Arn (for example, acs:ram::1323431815622349:role/aliyunlogdefaultrole), as it needs to be provided to primary account A for the creation of OSS shipping rule.

By default, quick authorization grants B the permission of writing all OSS Buckets. For an extended access control list, go to **Change Policy** introduced below.

Authorization of creating shipping rule using a subaccount

After the role creation and authorization processes are completed for B, primary account A has the right to use the role created by primary account B to write data into OSS Bucket. This is provided that primary account A creates the shipping rule.

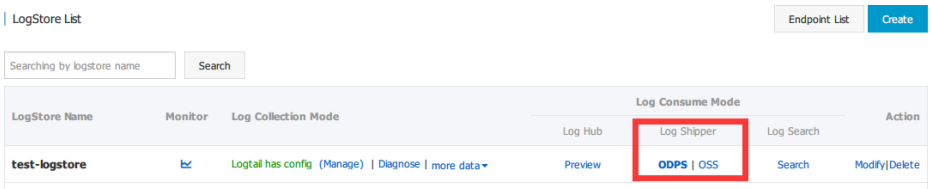
To use a sub-account a_1 of primary account A to create the shipping rule as the role, go to **PassRole Authorization**.

Step 2: Configure OSS shipping rule

Log on to theLog Service console.

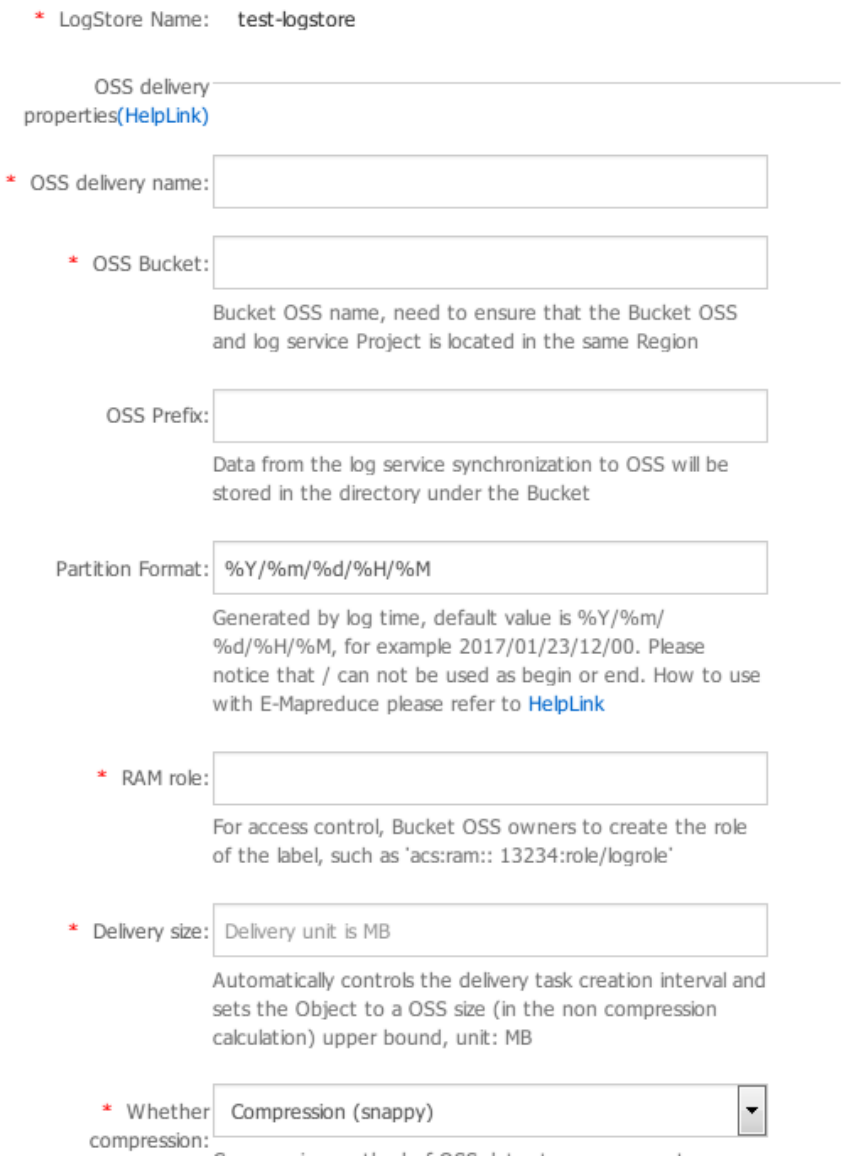
Select a project, and click the project name or **Manage** on the right.

Select the desired Logstore. Click **OSS** under the LogShipper column.



Click **Enable**, and set the following OSS shipping configuration, and click **Confirm**.

- OSS Shipping Name: Name of the created shipping. The OSS shipping name can be 3 to 63 characters in length and can include lowercase letters, digits, hyphens (-), and underscores (_). It must begin and end with a lowercase letter or digit.
- OSS Bucket: OSS Bucket name. Ensure that the OSS Bucket and Log Service project are in the same region.
- OSS Prefix: The data synchronized from Log Service to OSS will be stored in this Bucket directory.
- Partition Format: Use %Y, %m, %d, %H, and %M to format the creation time of the shipping task to generate the shard string (for information about formatting, refer to strptime API). This allows to define the directory hierarchy of the OSS Object file, where a back slash (/) indicates an OSS level. The following table is an example illustrating how to define the OSS target file path of OSS Prefix and shard format.
- RAM Role: Used for ACL, it is an identifier of the role created by the OSS Bucket owner. For example, acs:ram::1323431815622349:role/aliyunlogdefaultrole.
- Delivery Size: Automatic control of the interval for creating shipping tasks, with the maximum size of one OSS Object (not compressed) set. The unit is in MBs.
- Compression: Compression of OSS data storage. It can be none or snappy. None indicates the original data is not compressed, and snappy indicates the data is compressed through the snappy algorithm. Snappy can reduce the space used for OSS Bucket storage.
- Delivery Time: The interval after which a shipping task is generated. The default value is 300, and the unit is in s.



Log Service concurrently implements data shipping at the backend. Large amounts of data may be processed by multiple shipping threads. Each shipping thread will jointly determine the frequency of task generation based on the size and time. When either condition is met, the shipping thread will create the task.

Each shipping task will be written into an OSS file, with the path format of oss:// OSS-BUCKET/OSS-PREFIX/PARTITION-FROMAT_RANDOM-ID(.snappy). A shipping task created at 2017/01/20 19:50:43 is taken as an example to explain the use of shard format.

OSS Bucket	OSS Prefix	Shard Format	OSS File Path
test-bucket	test-table	%Y/%m/%d/%H/%M	oss://test-bucket/test-table/2017/01/20/19/50/43_1484913043351525351_2850008
test-bucket	log_ship_oss_exampl	%Y/%m/%d/log_%H	oss://test-

	e	%M%s	bucket/log_ship_oss_example/2017/01/20/log_195043_1484913043351525351_2850008
test-bucket	log_ship_oss_example	%Y%m%d/%H	oss://test-bucket/log_ship_oss_example/20170120/19_1484913043351525351_2850008
test-bucket	log_ship_oss_example	%Y%m%d/	oss://test-bucket/log_ship_oss_example/20170120/_1484913043351525351_2850008
test-bucket	log_ship_oss_example	%Y%m%d%H	oss://test-bucket/log_ship_oss_example/2017012019_1484913043351525351_2850008

Log shipping task management

After the OSS shipping function is enabled, Log Service will regularly start the shipping task in the background. You can view the shipping task status on the console.

Through **Log Shipping Task Management**, you can:

View LogShipper tasks from the past two days, along with their statuses. The status of a shipping task can be Success, In Progress, or Failed. Failed indicates that the shipping task has encountered an error due to external reasons and cannot be retried. In this case, you must manually troubleshoot the issue.

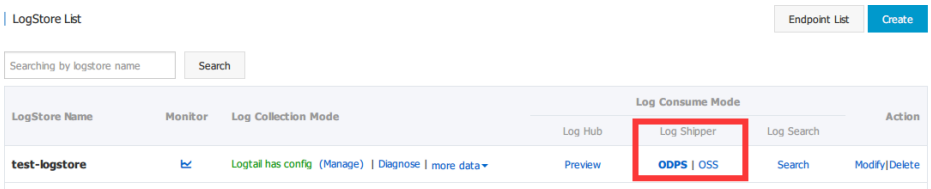
For failed shipping tasks (created within the last two days), you can view the external cause for the failure in the task list. After you successfully troubleshoot the cause, you can retry failed tasks separately or in batches.

Procedure

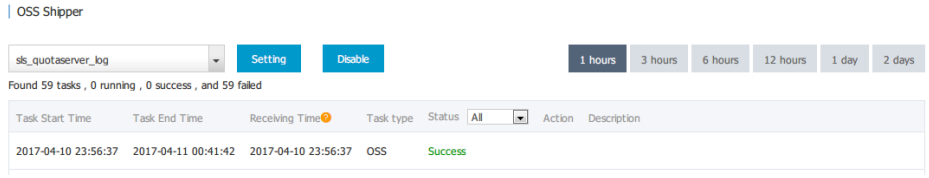
Log on to the Log Service console.

Select the desired project, and click the project name or **Manage** on the right.

Select the desired Logstore. Click **OSS** under the LogShipper column.



You can view the status of the shipping task.



If the implementation of the shipping task fails, a corresponding error will be displayed on the console. Based on the policy, the system default is that a retry will be performed. You can also manually retry the operation.

Task retry

Typically, log data will be synchronized in OSS 30 minutes after it is written into the Logstore.

By default, Log Service will retry the task(s) of the latest two days based on the backoff policy. The minimum interval for retry is 15 minutes. A task that has failed once can be retried in 15 minutes, a task that failed twice can be retried in 30 minutes (2 x 15 minutes), and a task that failed three times can be retried in 60 minutes (2 x 30 minutes).

To immediately retry a failed task, click **Retry All Failed Tasks** on the console or specify a task and retry it through API/SDK.

Failed task error

Error information about failed tasks is as follows.

Error information	Troubleshooting method
UnAuthorized	Unauthorized. Check: - Whether the OSS user has created the role. - Whether the account ID in role description is correct. - Whether the role has been granted OSS Bucket write permission. - Whether role-arn is correctly configured.
ConfigNotExist	Configuration does not exist. It is usually caused by the deletion of a shipping rule. If the rule is recreated, retry to resolve the problem.
InvalidOssBucket	OSS Bucket does not exist. Check : - Whether the OSS Bucket is in the same region as the Log Service project.

	- Whether the Bucket name is correctly configured.
InternalServerError	Internal error of Log Service. Retry to resolve the problem.

Step: View data in OSS

OSS data can be accessed through the console, API/SDK, and other methods.

To access OSS data through the console, start the OSS service, select your desired Bucket, and click **Object Management** to view the data shipped from Log Service.

Object address

Address format

oss:// OSS-BUCKET/OSS-PREFIX/PARTITION-FROMAT_RANDOM-ID

Field description

- OSS-BUCKET and OSS-PREFIX indicate the OSS Bucket name and directory prefix respectively, and are configured by the user. INCREMENTID is a random number added by the system.
- PARTITION-FORMAT is defined as %Y/%m/%d/%H/%M, where %Y, %m, %d, %H, and %M indicate year, month, day, hour, and minute respectively. They are calculated by the shipping task server through strptime API.
- RANDOM-ID is the unique identifier of a shipping task.

Directory time

Assume the following:

- Data is shipped to OSS every 5 minutes.
- The shipping task is created at 2016-06-23 00:00:00.
- The data to be shipped is the data written into Log Service at 2016-06-22 23:55.

To analyze the complete log of the full day of June 22, 2016, in addition to all objects in the 2016/06/22 directory, you also need to check whether the objects in the first 10 minutes in the 2016/06/23/00/ directory contain the log with the date of 2016-06-22.

Object storage format

- JSON
- Parquet

OSS storage addresses

Compression type	File suffix	OSS file address example
Do Not Compress	None	oss://oss-shipper-shenzhen/ecs_test/2016/01/26/20/54_1453812893059571256_937
snappy	.snappy	oss://oss-shipper-shenzhen/ecs_test/2016/01/26/20/54_1453812893059571256_937.snappy

Do not compress

Object is combined by multiple logs. Each line of the file indicates a log in the JSON format. An example is as follows.

```
{"__time__":1453809242,"__topic__":"","__source__":"10.170.148.237","ip":"10.200.98.220","time":"26/Jan/2016:19:54:02 +0800","url":"POST /PutData?Category=YunOsAccountOpLog&AccessKeyId=U0UjpekFQOVJW45A&Date=Fri%2C%2028%20Jun%202013%2006%3A53%3A30%20GMT&Topic=raw&Signature=pD12XYLmGxKQ%2Bmkd6x7hAgQ7b1c%3D HTTP/1.1","status":"200","user-agent":"aliyun-sdk-java"}
```

snappy-compressed

Implemented through C++ of snappy official website (Snappy.Compress method), and obtained by compressing the data in none format at the file level. .snappy file can be decompressed to obtain the corresponding file in none format.

Decompress through C++ Lib

On the Snappy website, download Lib, and execute Snappy.Uncompress to decompress.

Decompress using Java Lib

xerial snappy-java, use Snappy.Uncompress orSnappy.SnappyInputStream (SnappyFramedInputStream is not supported).

```
<dependency>
<groupId>org.xerial.snappy</groupId>
<artifactId>snappy-java</artifactId>
<version>1.0.4.1</version>
<type>jar</type>
<scope>compile</scope>
</dependency>
```

Note: Version 1.1.2.1 may not support the decompression of certain files due to the existence of bug, which is fixed in Version 1.1.2.6.

Snappy.Uncompress

```
String fileName = "C:\\My download\\36_1474212963188600684_4451886.snappy";
RandomAccessFile randomFile = new RandomAccessFile(fileName, "r");
int fileLength = (int) randomFile.length();
randomFile.seek(0);
byte[] bytes = new byte[fileLength];
int byteread = randomFile.read(bytes);
System.out.println(fileLength);
System.out.println(byteread);
byte[] uncompressed = Snappy.uncompress(bytes);
String result = new String(uncompressed, "UTF-8");
System.out.println(result);
```

Snappy.SnappyInputStream

```
String fileName = "C:\\My download\\36_1474212963188600684_4451886.snappy";
SnappyInputStream sis = new SnappyInputStream(new FileInputStream(fileName));
byte[] buffer = new byte[4096];
int len = 0;
while ((len = sis.read(buffer)) != -1) {
System.out.println(new String(buffer, 0, len));
}
```

Unzipping tool under Linux environment

For Linux environments, we provide specific tool to decompress the snappy file, click to download: snappy_tool.

```
./snappy_tool 03_1453457006548078722_44148.snappy 03_1453457006548078722_44148
compressed.size: 2217186
snappy::Uncompress return: 1
uncompressed.size: 25223660
```

Authorization of Role management

Through quick authorization, the description of the AliyunLogDefaultRole role created by default for primary account B of OSS user is as follows.

```
{
"Statement": [
{
"Action": "sts:AssumeRole",
"Effect": "Allow",
```

```
"Principal": {
"Service": [
"log.aliyuncs.com"
]
}
}
],
"Version": "1"
}
```

If the primary account A of Log Service is the same as the primary account B of OSS user, retain the default role description, that is, log.aliyuncs.com is equivalent to B_ALIYUN_ID@log.aliyuncs.com.

Otherwise, log on to Account Management > Security Settings to view the ID of account A and modify the role description, to add A_ALIYUN_ID@log.aliyuncs.com (multiple roles can be added). Assume that the ID of primary account A is 1654218965343050.

```
{
"Statement": [
{
"Action": "sts:AssumeRole",
"Effect": "Allow",
"Principal": {
"Service": [
"1654218965343050@log.aliyuncs.com",
"log.aliyuncs.com"
]
}
}
],
"Version": "1"
}
```

Based on this role description, A has the right to use Log Service to obtain the temporary Token to operate the resources of B. For more details, refer to ACL STS.

Authorization of Policy management

Through quick authorization, the role AliyunLogDefaultRole is granted AliyunLogRolePolicy by default and has the permission of writing all OSS buckets of B.

Authorization description of AliyunLogRolePolicy is as follows.

```
{
"Version": "1",
"Statement": [
{
"Action": [
"oss:PutObject"
],
"Resource": "*",
```

```
"Effect": "Allow"
}
]
}
```

For finer RAM granularity, cancel the AliyunLogRolePolicy authorization to the role AliyunLogDefaultRole, create a policy with finer granularity, and then grant its permission to the role AliyunLogDefaultRole.

Authorization of Role to sub-account by primary account

By default, after the primary account B of the OSS user has created the role and granted permission to the primary account A of Log Service, A can use the role to create OSS shipping configuration through Log Service.

If the sub-account a_1 of A needs to use the role to create the log and ship the log data to OSS, the primary account A needs to grant the PassRole permission to the sub-account a_1.

You can access Alibaba Cloud RAM console and use the primary account A to grant the AliyunRAMFullAccess permission to the sub-account a_1. As a result, a_1 will have all RAM permissions, and will be able to use B to configure the rule for shipping from Log Service to OSS for the role created by A.

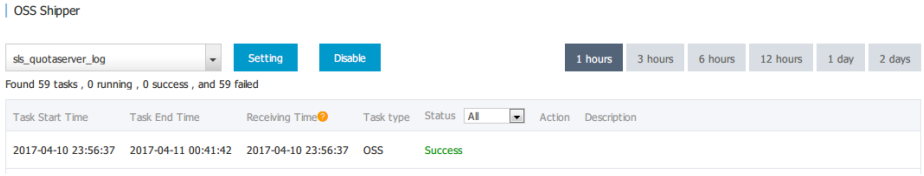
AliyunRAMFullAccess has greater permissions. To control the permission range of a_1, use A to grant only required permissions for shipping to OSS to a_1. Action and Resource of authorization policy can be modified, as shown in the following example.

```
{
"Statement": [
{
"Action": "ram:PassRole",
"Effect": "Allow",
"Resource": "acs:ram::1323431815622349:role/aliyunlogdefaultrole"
}
],
"Version": "1"
}
```

View OSS shipping task status

Log Service can be used to ship logs to OSS. This helps you store data that must be retained for a long period of time, or is to be jointly consumed by other systems (such as E-MapReduce). As described in Ship Logs to OSS, you can determine whether a Logstore should enable the function according to your requirements. Once this function is enabled, Log Service will regularly ship the logs written to this Logstore to the corresponding bucket in OSS. In order to detail shipping progress and

conveniently handle online problems, the Log Service console provides the **LogShipper Task Management** page.



As shown in the preceding figure, you can use the LogShipper task management page to:

- View LogShipper tasks from the past two days, along with their statuses. The shipping task status can be **Success**, **In Progress**, or **Failed**. **Failed** indicates that the shipping task encountered an error. For more information, refer to the **Shipping task management** section of Ship logs to OSS.
- For failed shipping tasks (created within the last two days), you can view the external cause for the failure in the task list. After you fix the problem, you can retry failed tasks separately or in batches.

Log search

Log Service supports log query by keyword. To use this function you must first create an index in the Logstore.

After the index is created, and the query by keyword function is enabled (disabled by default), you can perform a query by keyword, for example:

```
1. KEY_1 OR KEY_?2
Logs containing the keyword KEY_1 or satisfying KEY_?2 (for example, KEY_12 and KEY_22)
2. KEY_1 AND KEY_*2
Logs both containing KEY_1 and satisfying KEY_*2 (for example, KEY_222 and KEY_123452)
3. KEY_1 NOT KEY_2
Logs containing KEY_1 but without KEY_2
4. (KEY_1 OR KEY_2) AND KEY_3 NOT KEY_4
Logs containing KEY_1 or KEY_2, and containing KEY_3 but without KEY_4
```

For complete syntax for log query, refer to API/Syntax for log query.

Configuration instructions

Based on specific query requirements, you can select the appropriate index method that meets query requirements and helps reduce cost.

- If no key name (Key) needs to be specified for any query:
  - You may set **Full Text Index Attrributes** only.
  - No need to set **Key/Value Index Attrributes**.
- If a key name (Key) needs to be specified for some queries:
  - Create a key value index for the specific key (Key) based on the requirement.

Procedure

Log on to the Log Service console.

Select the desired project, and click the project name or **Manage** on the right.

Select the desired Logstore and click **Search** under the LogSearch column.

Click **Enable Index** in the upper-right corner.

Enter the index configuration information.

Assume that the log content in the Logstore is as follows:

Time/IP	Content
(1)17-04-11 00:46:00 1.1.1.1 Context View	**agent**:test&agent **code**:internalError **error**:internalError **message**:a,b;c;D-F

Full Text Index Attributes

Case Sensitive

false means case insensitive, and sample logs can be queried through query of INTERNALERROR or internalerror. **true** means sample logs can be queried through internalError only.

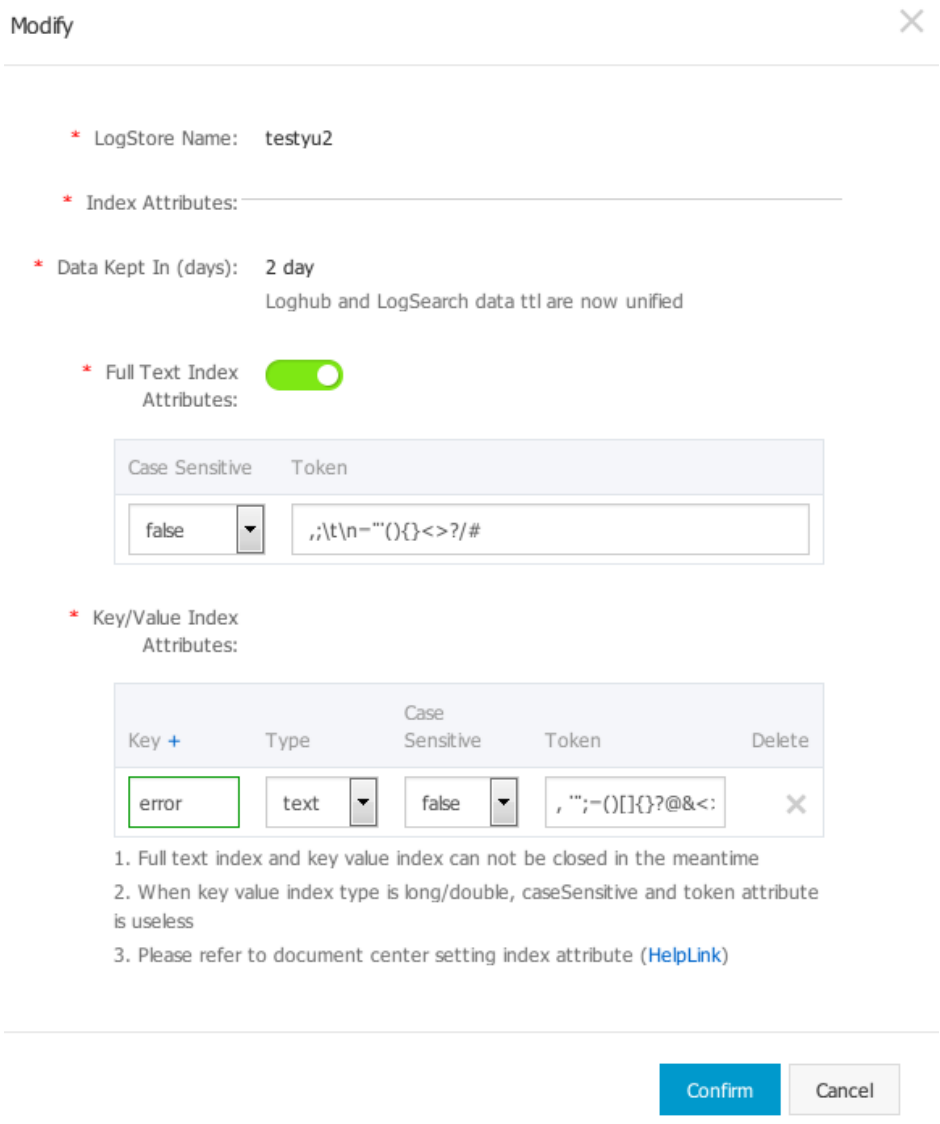
Token

Based on the specified single character, the log content is segmented into several keywords. Take the following sample log as an example:

Log content:a,b;c;D-F
If you set the separator to comma, semicolon, and dash, 5 keywords are obtained through segmentation:
"a" "b" "c" "D" "F"

Key/Value Index Attributes

The content corresponding to all the keys in the query log will be queried through the default index, and all will be queried if one is hit. For example, if internalError is queried in the sample log, the two keys (error and code) both meet this query condition. If you only need to query the log content with the error of internalError, you need to set the key value index, as shown in the following image.



Here, **Key** is the specific field Key of the log content specified by you, and the other two properties **Case Sensitive** and **Token** are consistent with the functions in **Full text index properties**. After the index properties shown in the preceding figure are created, the following log content with the error field of internalError can be

obtained by the following query.

```
error:internalError
```

Modify index settings

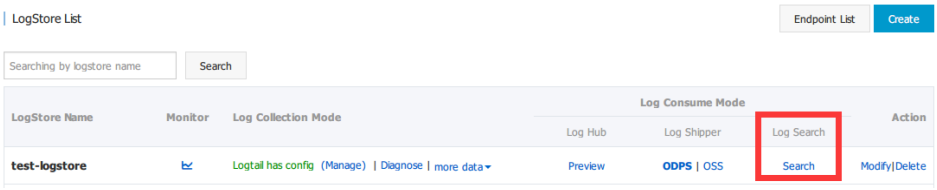
After creating a log index, you can modify the index settings on the Log Service Console.

Procedure

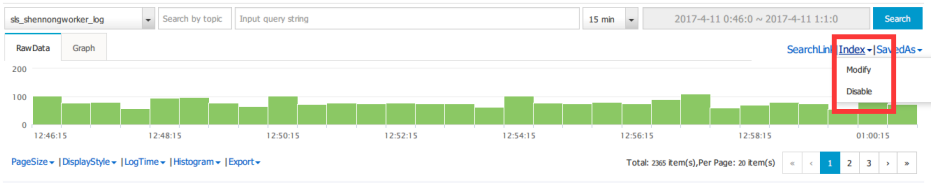
Log on to the Log Service console.

Select the desired project, and click the project name or **Manage** on the right.

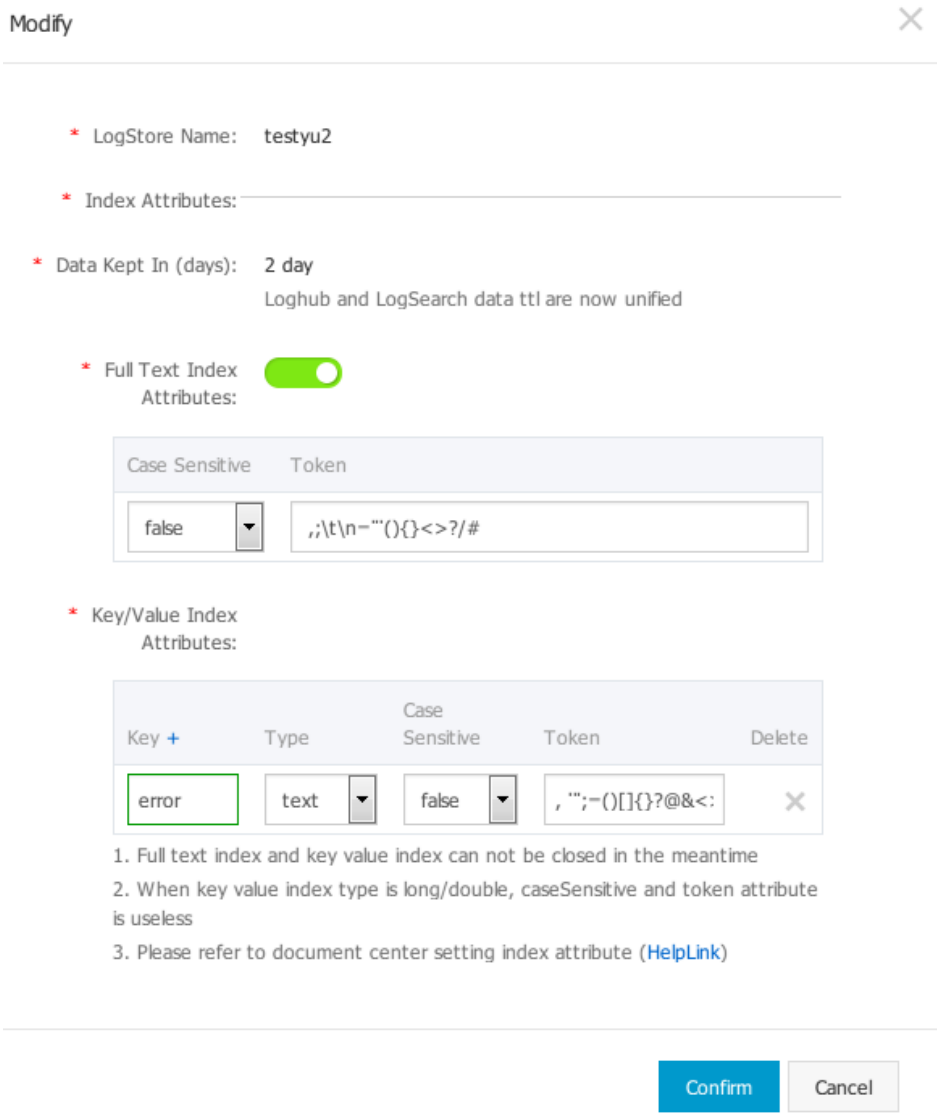
Select the desired LogStore and click **Search** under the Log Search column.



Click **Index** > **Modify** in the upper-right corner.



Modify index settings and click **Confirm**.



The index mode of Log Service allows you to quickly query massive logs in a Logstore and retrieve the desired log data. After the log is written into the Logstore, you can use the console to Create an index for the log content in the Logstore.

Index and query characteristics are as follows:

- Large scale: Support of petabytes of data per day
- Low latency: Data can be queried within 3 seconds after data is written
- Return in seconds: 1 billion-level log can be returned in one second

Note:

- Both the consumption log and query log mean "Read" log.
- After querying the specific log, the user using Logtail can use the Context query function.

Query logs by using the console

You can retrieve the same result when making a query on the Log Service console as by using APIs and SDK. Meanwhile, the console also provides visual interactive interfaces that help you perform query and understand the query results.

- Graphic user interfaces (bar charts) are provided to display the distribution of query results. Bar charts of different colors are used to indicate whether the log query results are complete.
- The flexible page flipping and sorting mechanism is provided to help you browse the returned original log data.
- The visual interaction mode is provided to help you correct the query condition. For example, you can change the time range for query by dragging on the bar charts or clicking keywords in the original log data to add key search words.
- The Context query function is provided to help positioning.

Operating procedure

Log on to the Log Service console.

Select the desired project, and click the project name or **Manage** on the right.

Select the desired Logstore and click **Search** under the LogSearch column.

You can set the query conditions (Logstore, topic, keyword, time interval, and time range), and click **Search** in the upper-right corner to query log.

Query results

You can query logs through the console, SDK or API. For any log query method, you must specify the queried LogStore (and the LogStore must belong to the specified project) and queried time range (the time range cannot exceed the log storage cycle of the LogStore). In addition, you can also specify the information such as the queried log topic, the maximum number of returned log records, and the Offset of returned logs.

To meet different data query requirements, Log Service provides two types of query: querying log distribution and querying original logs, as detailed below:

- Querying log distribution (GetHistograms): Query the number of logs that meet the query condition and the distribution of these logs in time regions (specifically, the number of logs that meet the query condition in each sub-time region) without returning the specific original log data.
- Querying original logs (GetLogs): Query logs that meet the query condition with all original log data returned.

Query latency

In Log Service, the logs written in the LogStore are classified into real-time data and historical data by the log timestamp. The specific definitions and their impact on the query results are as follows.

- Real-time data: The time point for a log is the current time point for the sever (-180 seconds, 900 seconds]. For example, if the log time is UTC 2014-09-25 12:03:00 and the receiving time of the server is UTC 2014-09-25 12:05:00, then an index is created in real time when this log is written in the LogStore, and the latency from writing into the LogStore to query success is within 3 seconds.
- Historical data: The time point in the log is the current time point of the server [-7 x 86400 seconds, -900 seconds). For example, if the log time is UTC 2014-09-25 12:00:00 and the receiving time of the server is UTC 2014-09-25 12:05:00, then this log is processed as the historical data. The maximum latency of this type of logs from being written into the LogStore to query success is 5 minutes.

If the data volume of logs to be searched for your query is very large (for example, the time span is long and your log data volume is very large), the data cannot be searched completely by one query request. In this case, Log Service will return the existing data to you, and inform you that the query results are not complete in the returned results. At the same time, the server will cache query results within 15 minutes. When a portion of the query request results are cache hits, the server will continue to scan the log data that are not cache hits for this request. To reduce your workload of merging multiple query results, Log Service merges the hit query results in the cache and the new hit results of the current query and returns them to you. Therefore, Log Service allows you to call this interface repeatedly using the same parameters to obtain the final complete results.

Using SDKs to query logs

Log Service provides SDKs in multiple languages (Java, .NET, PHP, and Python). The SDKs of all these languages support the full-function log query interface. For more information about the SDKs, refer to Log Service SDKs.

Use APIs to query logs

Log Service provides HTTP-enabled REST APIs. The APIs support full featured log query. For details, refer to Log Service APIs.

Query syntax

Refer to the section about Log query syntax. For the index, refer to the section about Index settings.

Query syntax

Log Service provides a set of query syntaxes used to express query conditions to help you easily query logs. You can specify query conditions through the GetLogs and GetHistograms interfaces in the Log Service API or on the query page of the Log Service console. This section details query condition syntax.

Full text index and key/value index

You can create an index for the LogStore in two modes:

- Full text index: The entire line of log is queried as a whole, without differentiating the key and value (Key, Value).
- Key/value index: Query is performed when Key is specified, for example, FILE:app, Type:action. All the contained strings under this key will be hit.

Syntax keyword

Log Service query conditions support the following keywords:

Name	Meaning
and	Binary operator, in the format of query1 and query2, indicating the intersection of the query results of query1 and query2. If there is no syntax keyword between the words, the relation between the words is and by default.
or	Binary operator, in the format of query1 or query2, indicating the union set of the query results of query1 and query2.
not	Binary operator, in the format of query1 not query2, indicating a result that meets query1 and does not meet query2, that is, query1–query2. If only not query1 exists, it indicates that logs not containing the query results of query1 are selected.
(,)	Left and right brackets are used to merge one or multiple sub-queries into one query to improve the query priority in the brackets.
:	Used to query the key-value pair. term1:term2 forms a key-value pair. If the key or value contains spaces, quotation marks need to be used to include the entire key or value.
“	Converts a keyword into a common query character. Any term in the left and right quotation marks will be queried and will not be used as a syntax keyword. Or all the terms in the left and right quotation marks are regarded as a whole in the key-value query.

\	Escape character. Used to escape quotation marks; the quotation marks after escaping indicate the symbols themselves and will not be considered as escape characters, for example, "\"".
|	Pipeline operator, indicating more computing based on the previous computing, for example, query1 | timeslice 1h | count.
timeslice	Time slice operator indicates the length of time during which the data is regarded as a whole for computing.The methods of use are timeslice 1h, timeslice 1m, and timeslice 1s, which respectively indicate 1 hour, 1 minute, and 1 second as a whole. For example, query1 | timeslice 1h | count indicates querying the query condition, and the total times with 1 hour as the time spice are returned.
count	Count operator, indicating the number of logs.
*	Fuzzy query keyword, used to replace 0 or multiple characters. For example: que*; all the hit words starting with que will be returned.
?	Fuzzy query keyword, used to replace one character. For example, qu?ry; all the hit words starting with beginning qu, ending with ry, and with a character in the middle.
__topic__	Query the data under a certain topic. Under the new syntax, the data of 0 or more topics can be queried in the query, for example, __topic__:mytopicname.
__tag__	Query a tag value under a tag key, for example, __tag__:tagkey:tagvalue.
source	Query the data of an IP, for example, source:127.0.0.1.

Note:

- Syntax keywords are case-insensitive.
- Priorities of syntax keywords are sorted in descending order as follows : > " > ( ) > and not > or.
- Log Service reserves the right to use the following keywords: sort asc desc group by avg sum min max limit. If you need to use the following keywords, use quotation marks to contain them.

Query examples

1. Logs that contains a and b at the same time: a and b or a b

2. Logs that contain a or b: a or b
3. Logs that contain a but no b: a not b
4. Those in all the logs that contain no a: not a
5. Query the logs that contain a and b, but no c: a and b not c
6. Logs that contain a or b and must contain c: (a or b) and c
7. Logs that contain a or b, but no c: (a or b) not c
8. Logs that contain a and b and may contain c: a and b or c
9. Logs whose FILE field contains apsara: FILE:apsara
10. Logs whose FILE field contains apsara and shennong: FILE:"apsara shennong" or FILE:apsara FILE: shennong or FILE:apsara and FILE:shennong
11. Logs containing and: and
12. Logs with the FILE field containing apsara or shennong: FILE:apsara or FILE:shennong
13. Logs with the file info field containing apsara: "file info":apsara
14. Logs that contain quotation marks: \"
15. Logs starting with shen: shen*
16. Query all the logs starting with shen under the FILE field: FILE:shen*
17. Query the logs starting with shen, ending with ong end and with a character in the middle: shen?ong
18. Query all the logs starting with shen and aps: shen* and aps*
19. Query the distribution of logs starting with shen, with the time slice as 20 minutes: shen*| timeslice 20m | count
20. Query all the data under topic1 and topic2: __topic__:topic1 or __topic__ : topic2
21. Query all the data of tagvalue2 under tagkey1: __tag__ : tagkey1 : tagvalue2

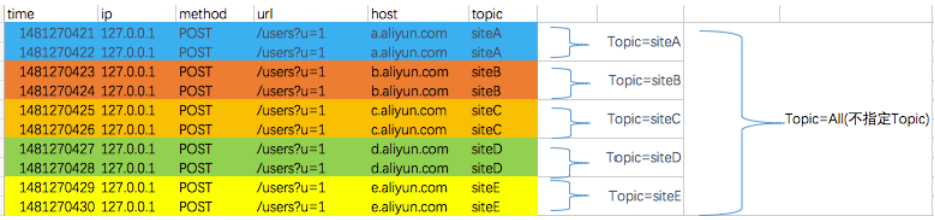
Additional query types

Specified or cross-topic query

Each LogStore can be divided into one or more subspaces according to the topic. During query, the query range can be limited for the specified topic to increase the speed. This means that users with a level-2 classification requirement for the LogStore are recommended to use topic to divide the LogStore.

When one or more topics are specified to perform query, query is only implemented in the topic that meets the conditions. However, if no topic is entered, the data under all the topics is queried by default.

Example in which topics are used to classify logs under different domain names:



Topic query example:

- The data under all the topics can be queried. The data of all the topics will be queried if no topic is specified in the query syntax and parameter.
- Topic can be queried in the query. The query syntax is __topic__:topicName. The old mode is still supported at the same time. The topic is specified in the URL parameter.
- Multiple topics can be queried, for example, __topic__:topic1 or __topic__:topic2 indicates the union of data under topic1 and topic2.

Histogram query

The added syntax of Log Service provides the custom interval function. The query syntax is as follows:

```
where_condition | timeslice 1[hms] |count
h stands for the unit of hour
m stands for the unit of minute
s stands for the unit of second
```

The interval size can be adjusted by changing the timeslice parameter. For example, to query the data of 2 hours, the corresponding results of different timeslice parameters are as follows:

timeslice parameter	Number of Histogram intervals	Size of each interval
1h	2	1 hour
30m	4	30 minutes
2m	60	2 minutes
30s	240	30 seconds

When you expand a log file, each log records an event. The log and its corresponding event do not exist independently. Several consecutive logs can then be used to review a specific process of a whole event sequence.

Log context query specifies the log source (machine + files) and the log in it, and searches a certain number of records before (the text preceding) and after (the text following) this log in the original file. This provides an easy method for clarifying the cause of a problem, and how to rectify it, under a DevOps scenario.

The Log Service console provides a specific page for query. You can use a browser to view the context information in the original file of the specified log, similar to turning the pages of the original log file.

Application scenarios

For example, an O2O take-out website will record the transaction track of an order in the program

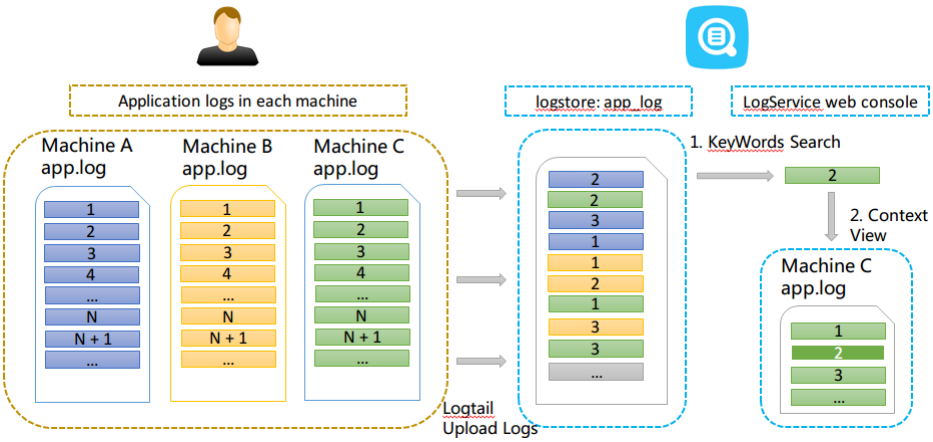
log on the server:

User login > **Browse articles** > **Click articles** > **Add to shopping cart** > **Place an order** > **Pay for the order** > **Deduct payment** > **Generate an order**

If the order cannot be placed, the persons responsible for operation and customer service must quickly locate the cause of the problem. In a conventional context query, the administrator adds the machine login permission to related members, and then the investigator logs in to each machine where applications are deployed in turn and uses the order ID as the keyword to search application log files.

For Log Service context query, you can use Log Service to implement this function as follows.

1. Install the log collection client Logtail on the server, and add the machine group and log collection configuration on the console. Then, Logtail will start to upload the incremental logs.
2. Access the console log query page of Log Service, specify the time range, and find the order failure log according to the order ID.
3. Page up with the found error log as a benchmark till other related log information is found (for example: credit card deduction fails).



Advantages

- There is no intrusion into the application, and you do not need to change the log file format.
- The specified log context information of any machine and file can be viewed on the Log Service console, avoiding issues when logging in to each machine to view the log file.
- In combination with the time cue of event occurrence, if context query is performed after a suspicious log in the specified time segment of the Log Service console is located, you always get twice the results with half the effort.
- There is zero data loss as a result of insufficient server storage or log file rotation, and can view historical data on the Log Service console at any time.

Prerequisites

- Use Logtail to collect logs. Upload data to the LogStore; only machine group creation and collection configuration is required.
- Activate the Log index query function.

For more information about how to query log context, refer to the introduction in Console context query.

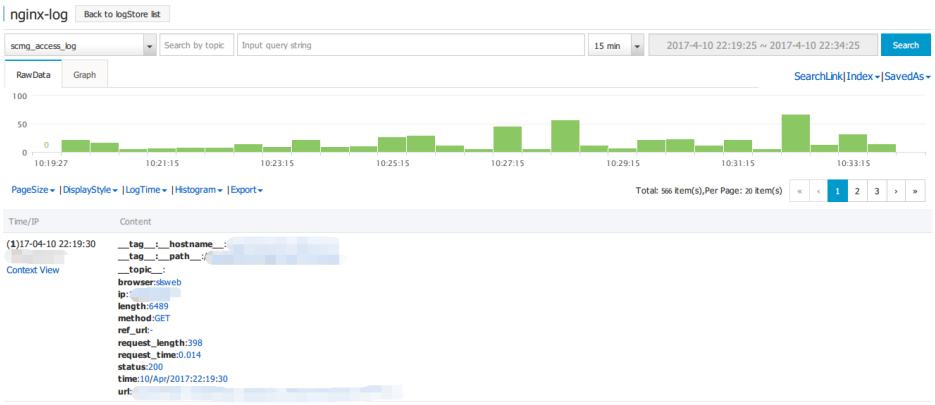
Procedure

Log on to the Log Service console.

Select the desired project and click the project name or **Manage** on the right.

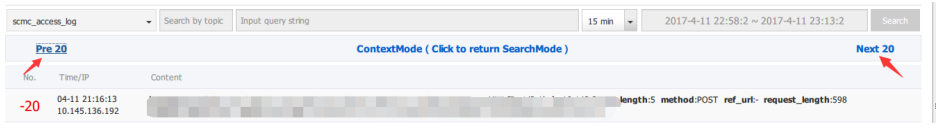
On the **Logstore List** page, select the desired Logstore and click **Search** in the LogSearch column.

If there is a **Context View** link to the left of a log returned on the query results page, this indicates that the log supports the context query function.



Select a log and click **Context View**.

This will switch the console from common search mode to context query mode.



Scroll to view the context information of the selected log. If you need to view an extended range of information, click **Prev 20** or **Next 20** for preceding and following results, respectively.

DevOps scenario log query solution comparison: ELK (search class), Hadoop/Hive

To handle the increasingly demanding software and service delivery needs, many enterprises are switching over to a DevOps mode. This allows to implement departmental collaboration, respond to customers' requirements, and conduct continuous delivery through collaboration between development (Dev) and O&M support (Ops).

Logs plays an important support role in aspects, such as problem investigation, security audit, and operation support, which means an appropriate log solution is important to DevOps.

To effectively compare LogSearch against ELK and Hadoop/Hive type solutions, the following aspects are measured:

- Latency: When users can perform a query after the log is generated.
- Query capability: The data volume is scanned by unit time.
- Query function: The keyword query, condition combination query, fuzzy query, numerical comparison, and context query.
- Elasticity: The rapid response to a rise of hundred times of traffic.
- Cost: The cost per GB.
- Reliability: The security of log data.

Common solutions and comparisons

- Self-built ELK: Comparison through Elastic, LogStash, and Kibana.
- Offline Hadoop + Hive: The data is stored in Hadoop, and Hive or Presto is used for query (not analysis).
- Use the Log Service (LogSearch).

Several solutions are compared through the application log and Nginx access log as an example (10 GBs per day).

Function item	ELK type system	Hadoop + Hive	Log service
Latency that can be queried	1-60 seconds (controlled by refresh_interval)	several minutes to several hours	1-3 seconds
Query latency	Less than 1 second	In minutes	Less than 1 second
Super large query	Several seconds to several minutes	In minutes	In seconds (query 1 billion logs)
Keyword query	Supported	Supported	Supported
Fuzzy query	Supported	Supported	Supported
Context query	Not supported	Not supported	Supported
Numerical comparison	Supported	Supported	Not supported currently

			(ETA:2017/2)
Consecutive string query	Supported	Supported	Not supported
Elasticity	Prepare machine in advance	Prepare machine in advance	10 times of resizing in seconds
Write cost	RMB5/GB for write; no charge for query	No charge for write; RMB0.3/GB for one query	RMB0.5/GB for write; no charge for query
Storage cost	≤ RMB3.36/GB * days	≤ RMB0.035/GB * days	≤ RMB0.016/GB * days
Reliability	Set the number of copies	Set the number of copies	SLA > 99.9%, data > 99.99999999%

Log Service Monitor

You can view Log Service monitoring information through the CloudMonitor console.

Monitoring metrics

Log Service provides the following metric data:

- Metric data available through CloudMonitor
  - Log reading/writing in Logstores
  - Shard-level metric data (coming soon)
  - Logs collected by agents (Logtail)
- Metric data available through the console
  - Current point of real-time subscription consumption (Spark Streaming, Storm, and Consumer Library)
  - OSS/MaxCompute posting status

The following describes how to view metric data through CloudMonitor. For details about viewing metric data through the console, refer to Check the collaborative consumer group progress and Check the LogShipper status.

View monitoring information through CloudMonitor

Method 1: Use Logstore to access CloudMonitor.

Log on to the Log Service console.

Select the desired project and click the project name or **Manage** on the right side.

Select the desired Logstore and click the metrics icon in the monitor column to access Cloud Monitor.

Method 2: Use CloudMonitor to view Log Service.

Log on to the **Cloud Monitor console**.

Click **Cloud Service Monitoring** > **Log Service** in the left navigation bar.

Select the desired project and Logstore, and click **Monitoring chart**.

Access control RAM

To access the Log Service console as a sub-user, perform the following three steps. For a detailed document, refer to **RAM console**.

Step 1: Create a sub-user

Log on to the **RAM console**.

Click **Users** in the left navigation bar.

Click **New User** in the upper-right corner.

Input user information, select **Automatically generate an Access Key for this user**, and click **OK**.

Click the newly created user to go to its corresponding user details page. Click **Enable Console Login**, set a user login password, and click **OK**.

Step 2: Grant the sub-user permission to access Log Service resources

In this example, the sub-user is granted the permission for accessing only the parent account resources.

In the **User Management** page, find the corresponding sub-user. Click **Authorization**.

In the displayed dialog box, select **AliyunLogReadOnlyAccess** and click **OK**.

Step 3: Log on to the console as the sub-user

After performing steps 1 and 2, the sub-user has the permission to access the Log Service console.

Return to the Overview page of the RAM console. Click the link for login as a sub-user, and use the newly created user name and password to log in.