

**Log Tank Service**

# **Best Practices**

**Issue**            01  
**Date**             2025-02-17



**Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2025. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

## **Trademarks and Permissions**



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

## **Notice**

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

---

# Contents

---

<b>1 Overview.....</b>	<b>1</b>
<b>2 Log Ingestion.....</b>	<b>3</b>
2.1 Collecting Logs from Third-Party Cloud Vendors, Internet Data Centers, and Other Huawei Cloud Regions to LTS.....	3
2.2 Collecting Kubernetes Logs from Third-Party Clouds, IDCs, and Other Huawei Cloud Regions to LTS.....	8
2.3 Collecting Syslog Aggregation Server Logs to LTS.....	14
2.4 Importing Logs of Self-built ELK to LTS.....	17
2.5 Using Flume to Report Logs to LTS.....	20
2.6 Collecting Zabbix Data Through ECS Log Ingestion.....	29
2.7 Collecting Logs from Multiple Channels to LTS.....	31
<b>3 Log Search and Analysis.....</b>	<b>35</b>
3.1 Analyzing Huawei Cloud ELB Logs on LTS.....	35
3.2 Viewing ELB Log Analysis Results on the LTS Dashboards.....	36
3.3 Analyzing Huawei Cloud WAF Logs on LTS.....	38
3.4 Embedding the LTS Log Query Page into a User-built System.....	39
<b>4 Log Transfer.....</b>	<b>52</b>
4.1 Changing File Time Zones for Log Transfer in a Batch.....	52
<b>5 Billing.....</b>	<b>57</b>
5.1 Collecting Statistics on LTS Expenses of Different Departments Based on Log Stream Tags.....	57

# 1 Overview

This document describes the following best practices of Log Tank Service (LTS):

**Table 1-1** Best practice overview

Category	Best Practice	Scenario
Log ingestion	<a href="#">Collecting Logs from Third-Party Cloud Vendors, Internet Data Centers, and Other Huawei Cloud Regions to LTS</a>	This practice describes how to collect Alibaba Cloud host logs to Huawei Cloud LTS. The method is similar to that of collecting logs from Internet Data Centers (IDCs) or across Huawei Cloud regions.
Log ingestion	<a href="#">Collecting Kubernetes Logs from Third-Party Clouds, IDCs, and Other Huawei Cloud Regions to LTS</a>	This practice describes how to collect Alibaba Cloud Kubernetes logs to Huawei Cloud LTS. The method is similar to that of collecting logs from IDCs or across Huawei Cloud regions.
Log ingestion	<a href="#">Collecting Syslog Aggregation Server Logs to LTS</a>	This practice describes how to use the syslog protocol to upload logs to LTS. You need to purchase an Elastic Cloud Server (ECS) as a syslog aggregation server. Syslog comes preinstalled by default on Linux servers. However, Huawei Cloud ECSs do not receive remote syslog writes by default. You need to enable this function.
Log ingestion	<a href="#">Importing Logs of Self-built ELK to LTS</a>	This practice describes how to use a custom Python script and ICAgent (LTS collector) to transfer logs from Elasticsearch to LTS.
Log ingestion	<a href="#">Using Flume to Report Logs to LTS</a>	This practice describes how to collect logs using Flume and report logs using Kafka provided by LTS.

Category	Best Practice	Scenario
Log ingestion	<a href="#">Collecting Zabbix Data Through ECS Log Ingestion</a>	This practice describes how to collect monitoring data from Zabbix to an LTS log stream.
Log ingestion	<a href="#">Collecting Logs from Multiple Channels to LTS</a>	This practice describes how to collect log data from multiple channels to LTS.
Log search and analysis	<a href="#">Analyzing Huawei Cloud ELB Logs on LTS</a>	This practice describes how to search for and analyze logs after Elastic Load Balance (ELB) logs are ingested to and structured in LTS.
Log search and analysis	<a href="#">Viewing ELB Log Analysis Results on the LTS Dashboards</a>	This practice describes how to display logs in dashboards after ELB logs are ingested to and structured in LTS.
Log search and analysis	<a href="#">Analyzing Huawei Cloud WAF Logs on LTS</a>	This practice describes how to search for and analyze logs after Web Application Firewall (WAF) logs are ingested to and structured in LTS.
Log search and analysis	<a href="#">Embedding the LTS Log Query Page into a User-built System</a>	This practice describes how to use the federation proxy mechanism of Identity and Access Management (IAM) for custom identity broker and embed a login link to your systems so you can view LTS logs in your systems without logging in to the Huawei Cloud console.
Log transfer	<a href="#">Changing File Time Zones for Log Transfer in a Batch</a>	This practice describes how to use Python scripts and LTS APIs to implement custom operations in a batch.
Billing	<a href="#">Collecting Statistics on LTS Expenses of Different Departments Based on Log Stream Tags</a>	This practice describes how to collect statistics on the LTS expenses of different departments in an enterprise. You can add tags to LTS log streams to distinguish business departments. LTS will add these tags to CDRs sent to the Billing Center.

# 2 Log Ingestion

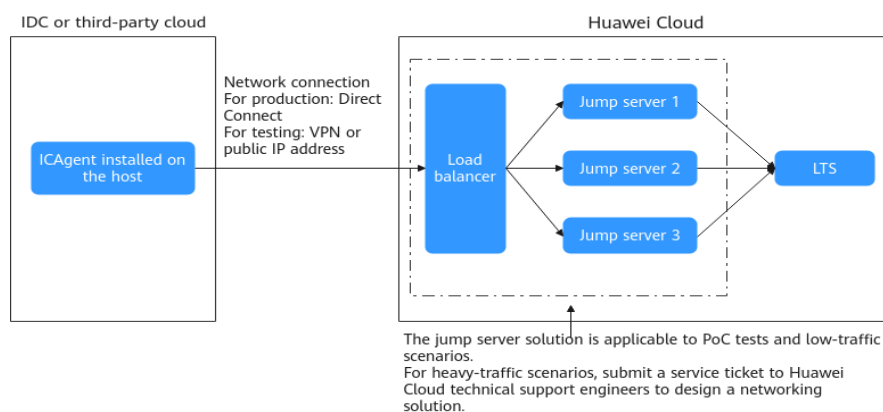
## 2.1 Collecting Logs from Third-Party Cloud Vendors, Internet Data Centers, and Other Huawei Cloud Regions to LTS

### Solution Overview

Cloud users often need to collect logs across clouds or regions. There are two typical scenarios:

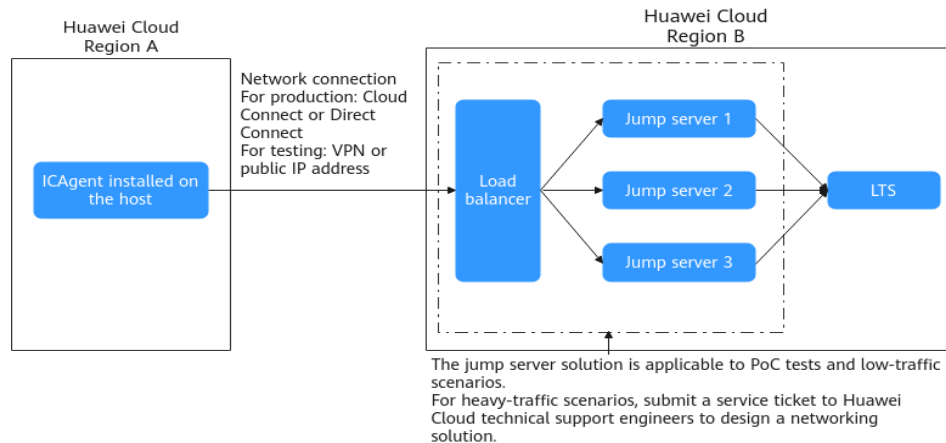
- Scenario 1: collecting logs from IDCs or third-party clouds to Huawei Cloud LTS

**Figure 2-1** Third-party cloud log collection



- Scenario 2: collecting logs from one Huawei Cloud region to LTS in another Huawei Cloud region

**Figure 2-2** Cross-region log collection



In both scenarios, you need to establish a network connection, install ICAgent, and follow the log ingestion wizard.

- **ICAgent:** the log collector of Huawei Cloud LTS. After being installed on a host, it collects logs from the host to LTS. Ensure that the time and time zone of your local browser are consistent with those of the host to install ICAgent.
- **Networking**
  - Scenario 1: Direct Connect is a typical method for connecting a customer-built IDC or third-party cloud to Huawei Cloud. If Direct Connect is unavailable, you can use a VPN or public IP address.
  - Scenario 2: Cloud Connect or Direct Connect is a typical method for interconnecting Huawei Cloud regions. You can also use a VPN or public IP address.
- **Jump server**
  - ICAgent installed in customer-built IDCs, third-party clouds, or other Huawei Cloud regions cannot directly access the network segment used by the Huawei Cloud management plane for log reporting, necessitating a jump server for data forwarding. Use the jump server solution for Proof of Concept (PoC) tests or when log traffic is light. If you do not want to use jump servers for heavy traffic scenarios in production environments, [submit a service ticket](#) to Huawei Cloud technical support to design a network passthrough solution.
  - A typical jump server configuration is 2 vCPUs and 4 GB memory, allowing it to forward traffic at approximately 30 MB/s. Configure a proper number of jump servers based on your log traffic and use a load balancer to distribute traffic among them.

This practice describes how to collect Alibaba Cloud host logs to Huawei Cloud LTS. The method is similar to that of collecting logs from customer-built IDCs or across Huawei Cloud regions.

Below are the steps to collect the logs from a Linux host in Alibaba Cloud's China (Beijing) region to LTS in Huawei Cloud's CN East-Shanghai1 region.

## Planning Resources

**Table 2-1** Planning resources

Region	Resource	Description
CN East-Shanghai 1	ECS	You are advised to use <b>CentOS 6.5 64bit</b> or later images. The minimum specifications are <b>1 vCPU   1 GB</b> and the recommended ones are <b>2 vCPUs   4 GB</b> .
	Load balancer	<ul style="list-style-type: none"> <li>• When buying a load balancer, select the same VPC as the ECS.</li> <li>• Create an EIP for connecting to the jump servers.</li> <li>• Buy the bandwidth based on the service requirements.</li> </ul>

### Purchasing a Load Balancer and an ECS as a Jump Server in Huawei Cloud CN East-Shanghai1

**Step 1** Log in to the ECS console and buy an ECS.

Before installing ICAgent on a non-Huawei Cloud host, buy an ECS as a jump server from Huawei Cloud.

**Step 2** Buy a load balancer, add TCP listeners, and associate a backend server group with it.

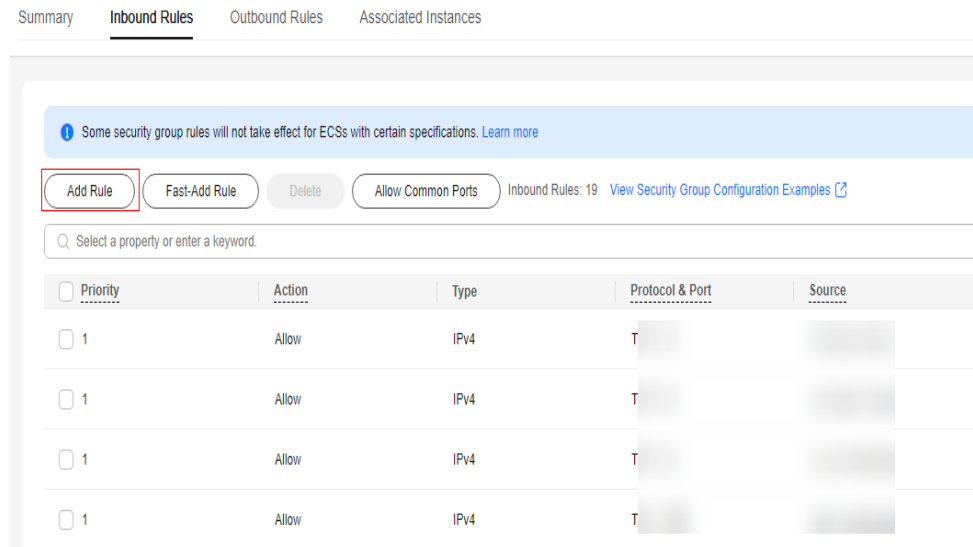
1. Add listeners for ports 30200, 30201, 8149, 8923, and 8102. For details, see [Adding a TCP Listener](#).
2. Add the jump server to a backend server group. For details, see [Backend Server](#).

**Step 3** Configure a security group rule for the jump server and open forwarding ports.

1. Modify the security group rule used by the jump server.
  - a. On the ECS console, click the name of the ECS used as the jump server to go to the details page.
  - b. On the **Security Groups** tab page, click a security group name to go to the details page.
  - c. Click the **Inbound Rules** tab and click **Add Rule**. Open the inbound ports 8149, 8102, 8923, 30200, 30201, and 80 to ensure that data can be transmitted from the non-Huawei Cloud host to the jump server.

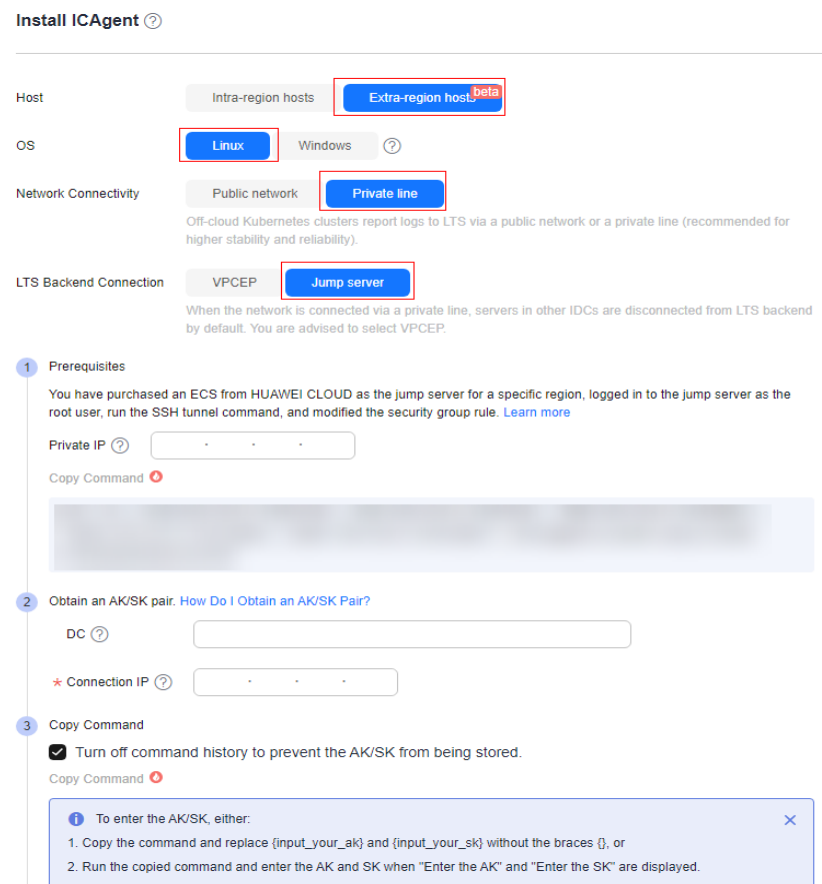


**Figure 2-3** Modifying a security group rule



2. On the LTS console, choose **Host Management > Hosts** in the navigation pane, and click **Install ICAgent** in the upper right corner. Set parameters as shown in the following figure. Set **Private IP** to the private IP address of the ECS to generate an installation password.

**Figure 2-4** Installing ICAgent



3. Copy the command, log in to the jump server as user **root**, run the SSH tunneling command, and enter the password of user **root** as prompted.
4. Run the **netstat -lnp | grep ssh** command to check whether the corresponding TCP ports are being listened to. If the command output similar to the following is returned, the ports are open.
  - Enter **http://Jump server IP address** in the address bar of a browser. If the access is successful, the security group rule has taken effect.
  - If the jump server is powered off and then restarted, run the installation command generated on the ICAgent installation page again. If you use the jump server in a production environment, configure the SSH tunneling command to run upon system startup.

Figure 2-5 Viewing ports

```
root@ecs-fcfc ~# netstat -lnp | grep ssh
tcp        0      0 0.0.0.0:22          0.0.0.0:*           LISTEN      1546/sshd
tcp        0      0 192.168.1.1:22    0.0.0.0:*           LISTEN      6161/sshd
tcp        0      0 192.168.1.1:22    0.0.0.0:*           LISTEN      6161/sshd
tcp        0      0 192.168.1.1:22    0.0.0.0:*           LISTEN      6161/sshd
tcp        0      0 192.168.1.1:22    0.0.0.0:*           LISTEN      6161/sshd
tcp        0      0 192.168.1.1:22    0.0.0.0:*           LISTEN      6161/sshd
tcp        0      0 192.168.1.1:22    0.0.0.0:*           LISTEN      6161/sshd
tcp6       0      0 :::22              :::*                 LISTEN      1546/sshd
```

----End

## Installing ICAgent on an Alibaba Cloud Host

- Step 1** Obtain an AK/SK. For details, see [How Do I Obtain an Access Key \(AK/SK\)?](#)
- Step 2** Enter the connection IP address of the jump server (that is, the EIP of the ECS) to generate the ICAgent installation command.
  - Replace the AK/SK in the command with the correct AK/SK. Otherwise, ICAgent cannot be installed.
  - **Connection IP:** For EIP connection, use the EIP of the jump server. For VPC peering connection, use the internal IP address of the VPC where the jump server locates.
- Step 3** Log in to the Alibaba Cloud host as user **root** and run the ICAgent installation command. If the message **ICAgent install success** is displayed, ICAgent is successfully installed.

If you use LTS to collect logs across Huawei Cloud regions, for example, collecting logs from the CN East-Shanghai1 region to the CN South-Guangzhou region, you need to buy a load balancer and an ECS used as a jump server in CN South-Guangzhou, and then run the ICAgent installation command on the jump server in CN East-Shanghai1.

Figure 2-6 Checking the ICAgent installation status

```
root@i22z[alibm]17h[calaz-0] c[0] http://[raent-re-act-1-nb-re-act-3-shuweicloud.com/raent-11m/one-agent-install] sh % one-agent-install] sh 64 8161m-
-east-3 b)
#28ad8db
% total % Received % Xferd Average Speed Time Time Time Current
          Dload Upload Total Spent Left Speed
100 7400 100 7400 0 0 35922 0 --:--:-- --:--:-- --:--:-- 35922
start to install ICAgent...
begin to download install package from icag
myhuaweicloud.com.
download success.
start install package.
start install ICAgent...
daemon
start
starting ICAgent...
ICAgent install success.
```

**Step 4** Choose **Host Management** > **Hosts** in the navigation pane of the LTS console and check whether the ICAgent status is **Running**.

----End

## Ingesting Logs to LTS

**Step 1** Log in to the LTS console and choose **Host Management** > **Host Groups** in the navigation pane. Click **Create Host Group**. On the displayed page, enter a host group name and select hosts.

**Step 2** Configure a log ingestion rule. For details, see [Ingesting ECS Text Logs to LTS](#).

----End

## Viewing a Log Stream

On the **Log Management** page of LTS, click the target log stream to go to its details page. If there are logs, the Alibaba Cloud logs have been reported to LTS.

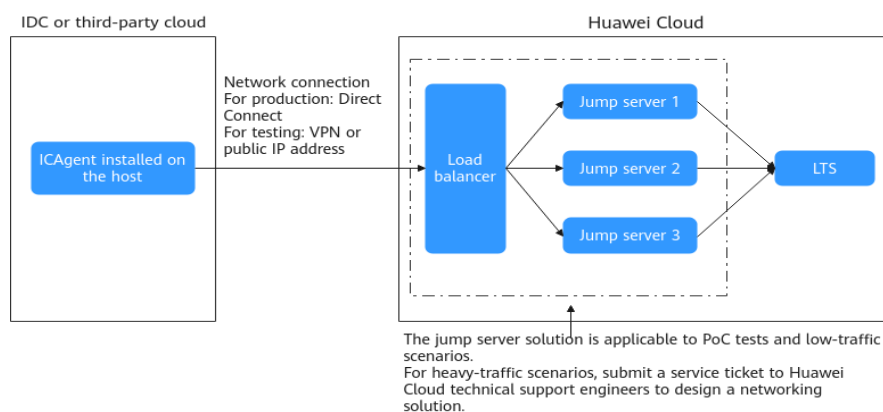
# 2.2 Collecting Kubernetes Logs from Third-Party Clouds, IDCs, and Other Huawei Cloud Regions to LTS

## Solution Overview

Cloud users often need to collect Kubernetes logs across clouds or regions. There are two typical scenarios:

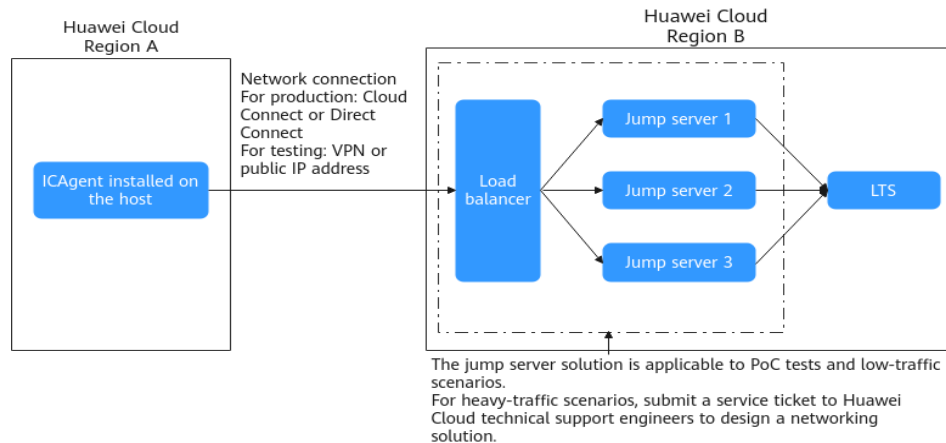
- Scenario 1: collecting logs from IDCs or third-party clouds to Huawei Cloud LTS

**Figure 2-7** Third-party cloud log collection



- Scenario 2: collecting logs from one Huawei Cloud region to LTS in another Huawei Cloud region

**Figure 2-8** Cross-region log collection



In both scenarios, you need to establish a network connection, install ICAgent, and follow the log ingestion wizard.

- **ICAgent:** the log collector of Huawei Cloud LTS. After being installed on a host, it collects logs from the host to LTS. Ensure that the time and time zone of your local browser are consistent with those of the host to install ICAgent.
- **Networking**
  - Scenario 1: Direct Connect is a typical method for connecting a customer-built IDC or third-party cloud to Huawei Cloud. If Direct Connect is unavailable, you can use a VPN or public IP address.
  - Scenario 2: Cloud Connect or Direct Connect is a typical method for interconnecting Huawei Cloud regions. You can also use a VPN or public IP address.
- **Jump server**
  - ICAgent installed in customer-built IDCs, third-party clouds, or other Huawei Cloud regions cannot directly access the network segment used by the Huawei Cloud management plane for log reporting, necessitating a jump server for data forwarding. Use the jump server solution for Proof of Concept (PoC) tests or when log traffic is light. If you do not want to use jump servers for heavy traffic scenarios in production environments, [submit a service ticket](#) to Huawei Cloud technical support to design a network passthrough solution.
  - A typical jump server configuration is 2 vCPUs and 4 GB memory, allowing it to forward traffic at approximately 30 MB/s. Configure a proper number of jump servers based on your log traffic and use a load balancer to distribute traffic among them.

This practice describes how to collect Alibaba Cloud host logs to Huawei Cloud LTS. The method is similar to that of collecting logs from customer-built IDCs or across Huawei Cloud regions.

Below are the steps to collect the logs from a Linux host in Alibaba Cloud's China (Beijing) region to LTS in Huawei Cloud's CN East-Shanghai1 region.

## Planning Resources

Table 2-2 Planning resources

Region	Resource	Description
CN East-Shanghai1	ECS	You are advised to use <b>CentOS 6.5 64bit</b> or later images. The minimum specifications are <b>1 vCPU   1 GB</b> and the recommended ones are <b>2 vCPUs   4 GB</b> .
	Load balancer	<ul style="list-style-type: none"> <li>• When buying a load balancer, select the same VPC as the ECS.</li> <li>• Create an EIP for connecting to the jump servers.</li> <li>• Buy the bandwidth based on the service requirements.</li> </ul>

### Purchasing a Load Balancer and an ECS as a Jump Server in Huawei Cloud CN East-Shanghai1

**Step 1** Log in to the ECS console and buy an ECS.

Before installing ICAgent on a non-Huawei Cloud host, buy an ECS as a jump server from Huawei Cloud.

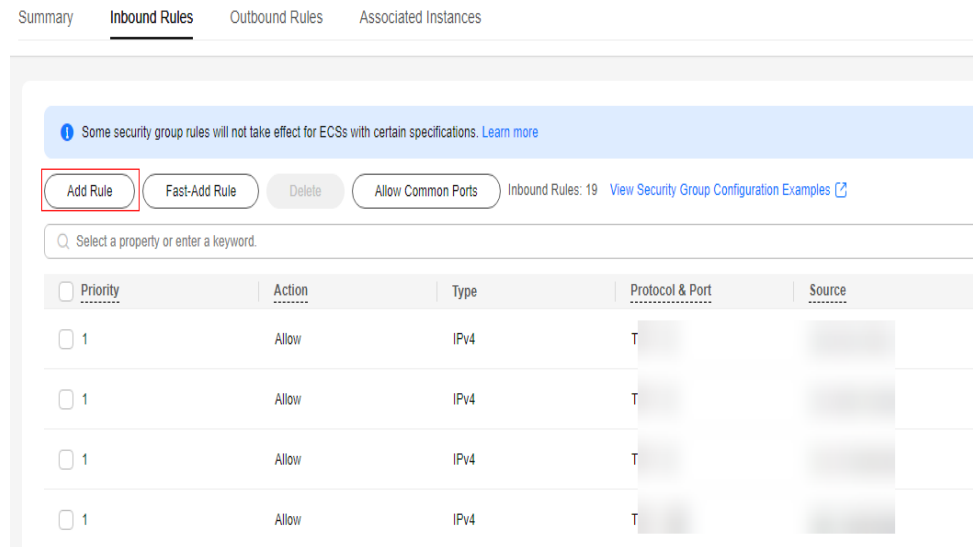
**Step 2** Buy a load balancer, add TCP listeners, and associate a backend server group with it.

1. Add listeners for TCP ports 30200, 30201, 8149, 8923, and 8102. For details, see [Adding a TCP Listener](#).
2. Add the jump server to a backend server group. For details, see [Backend Server](#).

**Step 3** Configure a security group rule for the jump server and open forwarding ports.

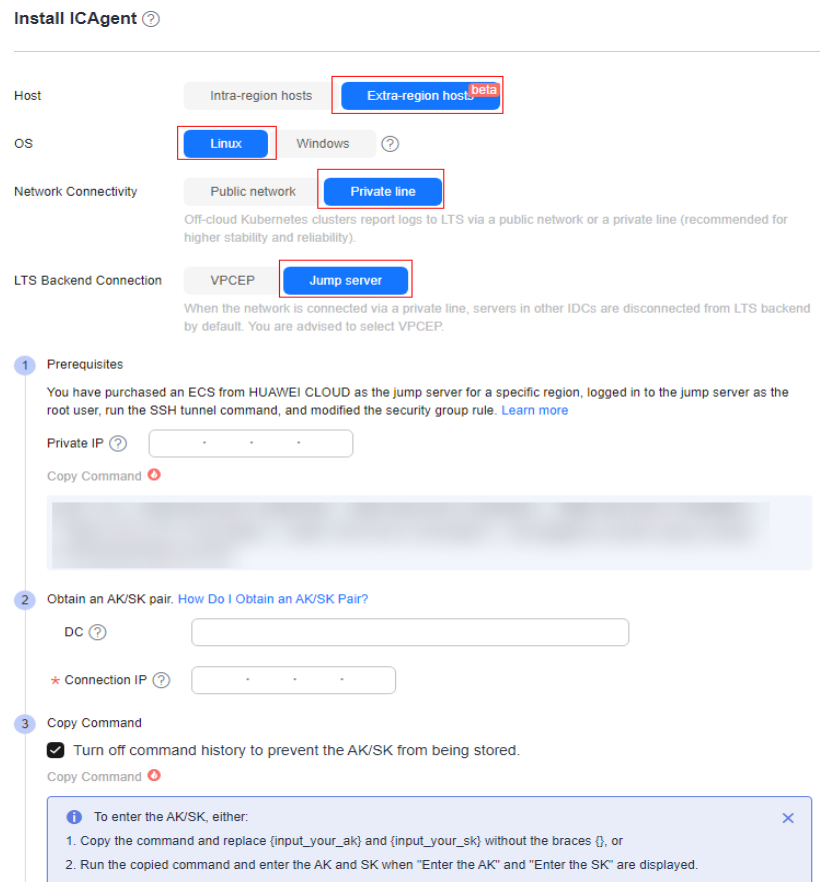
1. Modify the security group rule used by the jump server.
  - a. On the ECS console, click the name of the ECS used as the jump server to go to the details page.
  - b. On the **Security Groups** tab page, click a security group name to go to the details page.
  - c. Click the **Inbound Rules** tab and click **Add Rule**. Open the inbound ports 8149, 8102, 8923, 30200, 30201, and 80 to ensure that data can be transmitted from the non-Huawei Cloud host to the jump server.

**Figure 2-9** Modifying a security group rule



2. On the LTS console, choose **Host Management** > **Hosts** in the navigation pane, and click **Install ICAgent** in the upper right corner. Set parameters as shown in the following figure. Set **Private IP** to the private IP address of the ECS to generate an installation password.

**Figure 2-10** Installing ICAgent



3. Copy the command, log in to the jump server as user **root**, run the SSH tunneling command, and enter the password of user **root** as prompted.
4. Run the **netstat -lnp | grep ssh** command to check whether the corresponding TCP ports are being listened to. If the command output similar to the following is returned, the ports are open.
  - Enter **http://Jump server IP address** in the address bar of a browser. If the access is successful, the security group rule has taken effect.
  - If the jump server is powered off and then restarted, run the installation command generated on the ICAgent installation page again. If you use the jump server in a production environment, configure the SSH tunneling command to run upon system startup.

Figure 2-11 Viewing ports

```
root@ecs-fcfc ~]# netstat -lnp | grep ssh
tcp        0      0 0.0.0.0:22          0.0.0.0:*          LISTEN     1546/sshd
tcp        0      0 0.0.0.0:22          0.0.0.0:*          LISTEN     1546/sshd
tcp        0      0 192.168.1.1:22     192.168.1.1:*      LISTEN     1546/sshd
tcp        0      0 192.168.1.1:22     192.168.1.1:*      LISTEN     1546/sshd
tcp        0      0 192.168.1.1:22     192.168.1.1:*      LISTEN     1546/sshd
tcp        0      0 192.168.1.1:22     192.168.1.1:*      LISTEN     1546/sshd
tcp        0      0 192.168.1.1:22     192.168.1.1:*      LISTEN     1546/sshd
tcp6       0      0 :::22              :::*                LISTEN     1546/sshd
```

----End

## Configuring Log Ingestion

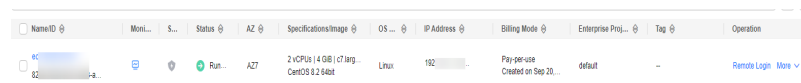
For a Kubernetes cluster, simply install ICAgent on one node, not all nodes.

Obtain an AK/SK in advance. For details, see [How Do I Obtain an Access Key \(AK/SK\)?](#)

### Step 1 Configure the jump server.

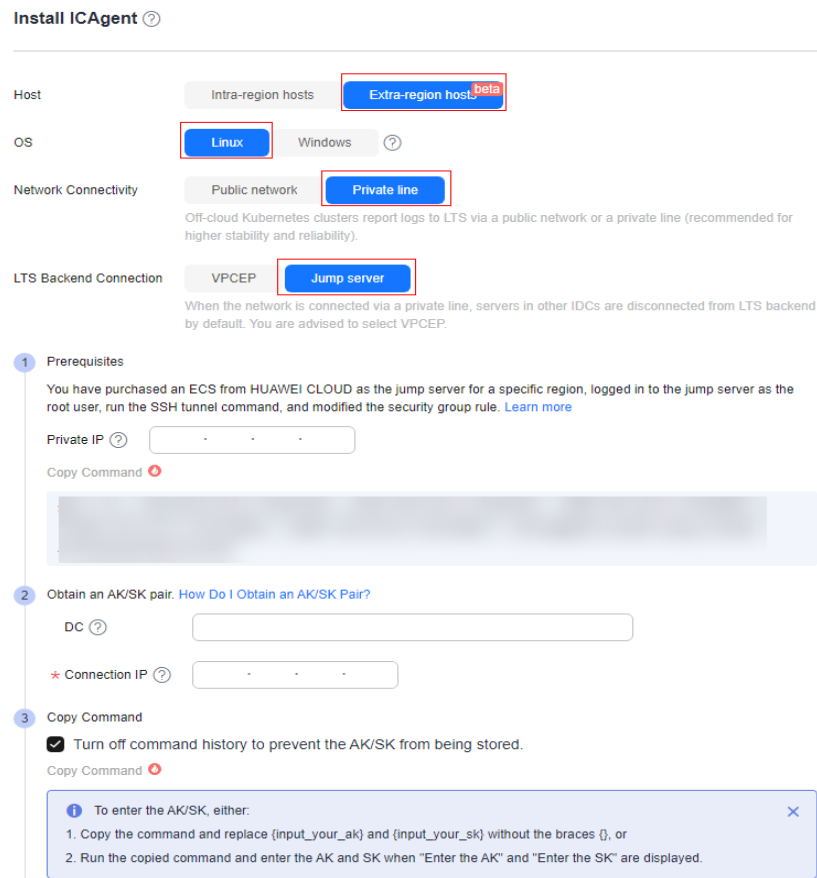
1. On the ECS console, locate the jump server and obtain its private IP address.

Figure 2-12 Obtaining the private IP address



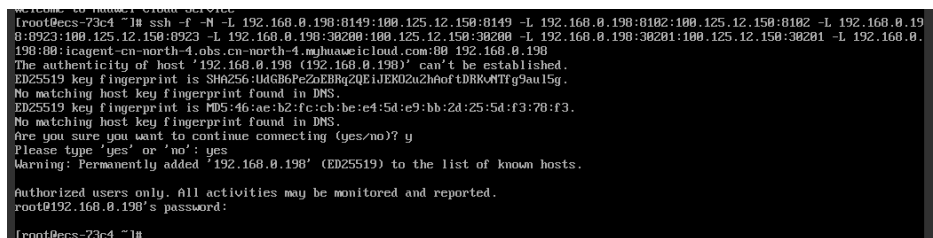
2. On the LTS console, choose **Host Management > Hosts** in the navigation pane and click **Install ICAgent**. On the page displayed, set parameters as follows, set **Private IP** to the private IP address of the ECS to generate an installation command, and copy the command.

Figure 2-13 Installing ICAgent



3. Log in to the ECS, run the command copied in the previous step, and enter the node password as prompted. If no error is reported, the installation is successful.

Figure 2-14 Running the generated installation command



4. On the **Install ICAgent** page, set **Connection IP** to the public IP address of the jump server. Ensure that the checkbox next to **Turn off command history to prevent the AK/SK from being stored** is selected.
5. Copy the ICAgent installation command and run it on the jump server. Enter the AK and SK of the current account as prompted. If the message **ICAgent install success** is displayed, ICAgent is successfully installed.

**Step 2** Configure a log ingestion rule. For details, see [Ingesting Self-Built Kubernetes Application Logs to LTS](#).

----End



## Viewing a Log Stream

On the **Log Management** page of LTS, click the target log stream to go to its details page. If there are logs, the Alibaba Cloud Kubernetes logs have been reported to LTS.

## 2.3 Collecting Syslog Aggregation Server Logs to LTS

### Introduction

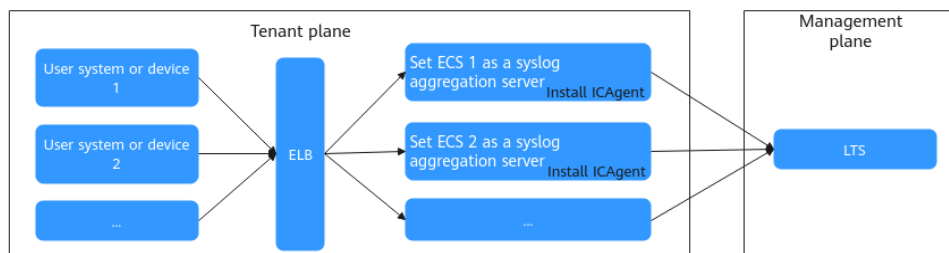
System Logging Protocol (Syslog) is a data protocol used by network devices to collect logs to a logging server. It can record log messages of multiple event types and is supported by almost all network devices, such as routers, switches, and printers. Even Unix-like servers can generate syslog messages to record user logins, firewall events, and Apache or Nginx access.

Syslog defines related format specifications based on RFC5424 and RFC3164. RFC3164 was released in 2001, and RFC5424 was an upgraded version released in 2009. The new version is compatible with the old version and solves many problems of the old version. Therefore, RFC 5424 is recommended.

This section describes how to use the syslog protocol to upload logs to LTS. You need to buy an ECS as a syslog aggregation server. Rsyslog comes preinstalled by default on Linux servers. However, Huawei Cloud ECSs do not receive remote syslog writes by default. You need to enable this function.

### Solution Overview

Figure 2-15 Solution flowchart



- You can buy a Linux ECS and configure it as a syslog aggregation server to receive log data from other devices. If the size of a log received by a syslog server exceeds 1,024 bytes, the log will be truncated.
- The log processing rate of a single syslog server is 10 MB/s. To process a large number of logs or ensure high reliability, you can buy multiple ECSs as syslog servers and configure load balancers for distributing traffic.
- You need to install ICAgent on syslog servers and configure log collection rules to collect logs to LTS.

### Planning Resources

Buy two ECSs. One serves as a syslog aggregation server, and the other serves as a service ECS to simulate user systems or devices to send logs.

## Buying an ECS

**Step 1** Log in to the management console and choose **Compute > Elastic Cloud Server**.

**Step 2** Buy an ECS as a syslog aggregation server.

You are advised to use **CentOS 6.5 64bit** or later images. The recommended specifications are 2 vCPUs and 4 GB of memory.

**Step 3** Log in to the syslog server as user **root** to install ICAgent.

1. Allow TCP ports 30200, 30201, 8149, 8923, and 8102 in the outbound rules of the syslog server, and then allow UDP port 514 in the inbound rules as the default listening port.
2. Go to the LTS console and choose **Host Management > Hosts** in the navigation pane.
3. On the page displayed, click **Install ICAgent**. Set **OS** to **Linux** and **Host** to **Intra-region hosts**. Then, select **Obtain AK/SK** for **Installation Mode**. Click **Copy Command** to copy the ICAgent installation command and manually replace the AK/SK.
4. Log in to the syslog server as user **root** and run the ICAgent installation command. If the message **ICAgent install success** is displayed, ICAgent is successfully installed. You can then view the ICAgent status by choosing **Host Management** in the navigation pane of the LTS console and then clicking **Hosts**.

**Step 4** Enable the rsyslog listening and receiving functions.

By default, rsyslog of Huawei Cloud ECSs does not receive remote syslog writes. You need to manually enable this function.

1. Log in to the ECS.
2. Modify the rsyslog configuration file.  
`vi /etc/rsyslog.conf`
3. Add the following content to the configuration file to enable TCP and UDP remote receiving:

```
# Provides UDP syslog reception
$ModLoad imudp
$UDPServerRun 514
# Provides TCP syslog reception
$ModLoad imtcp
$InputTCPServerRun 514
```
4. Save the settings. Click **More > Restart** in the **Operation** column to restart the ECS.
5. Run any of the following commands. If the command output is normal, the service is working.

Run **service rsyslog status** to check whether the rsyslog running status is running.

**Figure 2-16** Checking the rsyslog status

```
[root@ecs-syslog-hed270223 ~]# service rsyslog status
Redirecting to /bin/systemctl status rsyslog.service
● rsyslog.service - System Logging Service
   Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-09-08 15:55:17 CST; 2 days ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
   Main PID: 4162 (rsyslogd)
   CGroup: /system.slice/rsyslog.service
           └─4162 qusrzshin/rsyslogd -n
```

Run **systemctl status rsyslog** to check whether the rsyslog running status is running.

Figure 2-17 Rsyslog status

```
[root@ecs-syslog-hwd270223 ~]# systemctl status rsyslog
● rsyslog.service - System Logging Service
   Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-09-08 15:55:17 CST; 2 days ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
   Main PID: 4162 (rsyslogd)
   CGroup: /system.slice/rsyslog.service
           └─4162 /usr/sbin/rsyslogd -n
```

Run **netstat -anp | grep 514** to check whether the listening function is enabled.

Figure 2-18 Checking whether listening is enabled

```
[root@ecs-e859 log]# netstat -anp | grep 514
tcp        0      0 0.0.0.0:514          0.0.0.0:*           LISTEN    988/rsyslogd
tcp6      0      0 :::514              :::*                 LISTEN    988/rsyslogd
udp        0      0 0.0.0.0:514          0.0.0.0:*           988/rsyslogd
udp6      0      0 :::514              :::*                 988/rsyslogd
```

**Step 5** Configure syslog collection.

1. Choose **Host Management > Host Groups** in the LTS navigation pane and click **Create Host Group**. On the displayed page, enter a host group name and select hosts.
2. Choose **Log Ingestion** in the navigation pane. On the displayed page, click **ECS (Elastic Cloud Server)**.
3. Select a log stream.
4. Select host groups.
5. Set the collection path to **/var/log/messages**.

**Step 6** Log in to the service ECS for verification.

After your service system or device generates logs, you can view the logs on the LTS console. Log in to the service ECS and run the **logger -n x.x.x.x -P 514 testremotelog** command to send syslog messages to the aggregation server. *x.x.x.x* indicates the IP address (public or private) of the syslog server. **testremotelog** indicates the log content, which can be customized.

After the command is executed, you can view the log in the configured log group and log stream.

Alternatively, log in to the syslog aggregation server and check whether the **testremotelog** log exists in **/var/log/messages**.

```
tail -f /var/log/messages
```

Figure 2-19 Checking whether the testremotelog log exists

```
[root@ecs-d1aa etc]# tail -f /var/log/messages
Feb 6 11:20:30 ecs-syslog root
Feb 6 11:22:05 ecs-syslog root 00
Feb 6 11:22:08 ecs-syslog root 11
Feb 6 11:22:11 ecs-syslog root 22
Feb 6 11:28:55 ecs-syslog root testremotelog
Feb 6 11:28:56 ecs-syslog root testremotelog
Feb 6 11:28:57 ecs-syslog root testremotelog
Feb 6 11:28:57 ecs-syslog root testremotelog
Feb 6 11:28:58 ecs-syslog root testremotelog
Feb 6 11:28:58 ecs-syslog root testremotelog
```

**Step 7** Use multiple syslog servers and load balancers to implement load balancing.

The log processing rate of a single syslog server is 10 MB/s. To process a large number of logs, you can use multiple syslog servers and load balancers.

1. Create syslog aggregation servers and install ICAgent.
2. Create a load balancer. For details, see [Using ELB to Distribute Traffic to a Web Application Across ECSs](#).
3. Add listeners for TCP/UDP ports and port 514. For details, see [Adding a Listener](#).
4. Add backend servers to the backend server group. For details, see [Backend Server Group](#).

----End

## 2.4 Importing Logs of Self-built ELK to LTS

### Solution Overview

ELK is an acronym that stands for Elasticsearch, Logstash, and Kibana. Together, these three tools provide a most commonly used log analysis and visualization solution in the industry.

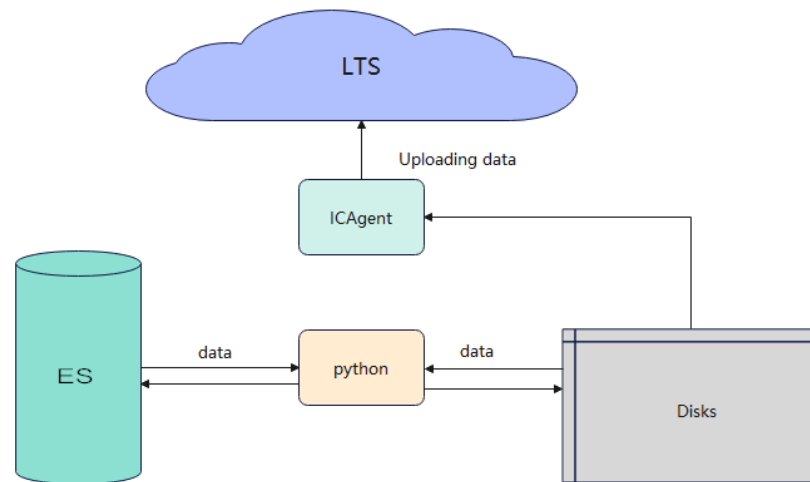
- Elasticsearch is an open-source, distributed, and RESTful search and analysis engine based on Lucene.
- Logstash is an open-source data processing pipeline on the server side. It allows you to collect and transform data from multiple sources in real time, and then send the data to your repository. It is usually used to collect, filter, and forward logs.
- Kibana is an open-source platform for data analysis and visualization, enabling you to create dashboards and search and query data. It is usually used together with Elasticsearch.

LTS outperforms the ELK solution in terms of function diversity, costs, and performance. For an in-depth comparison, see [What Are the Advantages of LTS Compared with Self-built ELK Stack?](#) This section describes how to use custom Python scripts and ICAgent to migrate logs from Elasticsearch to LTS.

ICAgent can be installed on ECSs to collect their log files. With this function, you can import Elasticsearch logs to LTS.

You can flush Elasticsearch data to ECSs using Python scripts, and then collect the flushed log files to LTS using its log ingestion function.

Figure 2-20 Solution flowchart



## Importing Logs of Self-built ELK to LTS

- Step 1** Log in to the [LTS console](#).
- Step 2** [Install ICAgent](#) on the ECS.
- Step 3** Configure ECS log ingestion on the LTS console. For details, see [Ingesting ECS Text Logs to LTS](#).
- Step 4** Prepare for script execution. The following example is for reference only. Enter your actual information.
  - If you use Python for the first time, you need to install the Python environment.
  - If you use Elasticsearch for the first time, you need to install the Python data package of the corresponding Elasticsearch version. Elasticsearch 7.10.1 is used in this solution test.  

```
pip install elasticsearch==7.10.1
```
  - Elasticsearch used in this solution test is created by Huawei Cloud Search Service (CSS).
- Step 5** Run the python script for constructing index data. If the index already has data, skip this step and go to [Step 6](#).

The python script must be executed on the ECS and named `xxx.py`. The following is an example of constructing data:

Modify the following italic fields as required. In this example, 1,000 data records with the content **This is a test log,Hello world!!!\n** are inserted.

- **index**: name of the index to be created. It is **test** in this example.
- **es**: URL for accessing Elasticsearch. It is **http://127.0.0.1:9200** in this example.

```

from elasticsearch import Elasticsearch
def creadIndex(index):
    mappings = {
        "properties": {
            "content": {
                "type": "text"
            }
        }
    }

```

```
    }
  }
  es.indices.create(index=index, mappings=mappings)
def reportLog(index):
  i = 0
  while i < 1000:
    i = i + 1
    body = {"content": "This is a test log,Hello world!!!\n"}
    es.index(index=index,body=body)
if __name__ == '__main__':
  # Index name
  index = 'test'
  # Link to Elasticsearch
  es = Elasticsearch("http://127.0.0.1:9200")
  creadIndex(index)
  reportLog(index)
```

**Step 6** Construct the Python read and write script to write Elasticsearch data to the disk. The output file path must be the same as that configured in the log ingestion rule.

The script must be executed on the ECS and named *xxx.py*. The following is an example of the script for writing data to the disk:

Modify the following italic fields as required.

- **index**: index name. It is **test** in this example.
- **pathFile**: absolute path for writing data to the disk. It is **/tmp/test.log** in this example.
- **scroll\_size**: size of the index rolling query. It is **100** in this example.
- **es**: URL for accessing Elasticsearch. It is **http://127.0.0.1:9200** in this example.

```
from elasticsearch import Elasticsearch
def writeLog(res, pathFile):
  data = res.get('hits').get('hits')
  i = 0
  while i < len(data):
    log = data[i].get('_source').get('content')
    file = open(pathFile, 'a', encoding='UTF-8')
    file.writelines(log)
    i = i + 1
  file.flush()
  file.close()
if __name__ == '__main__':
  # Index name
  index = 'test'
  # Output file path
  pathFile = '/tmp/' + index + '.log'
  # Size of the scrolling query. The default value is 100.
  scroll_size = 100
  # Link to Elasticsearch
  es = Elasticsearch("http://127.0.0.1:9200")
  init = True
  while 1:
    if (init == True):
      res = es.search(index=index, scroll="1m", body={"size": scroll_size})
      init = False
    else:
      scroll_id = res.get("_scroll_id")
      res = es.scroll(scroll="1m", scroll_id=scroll_id)
    if not res.get('hits').get('hits'):
      break
    writeLog(res, pathFile)
```

**Step 7** Ensure that Python has been installed and run the following command on the ECS to write the Elasticsearch index data to the disk:

```
python xxx.py
```

**Step 8** Check whether the data was successfully queried and written into the disk.

In this example, the path for writing data to the disk is `/tmp/test.log`. Replace it with your actual path. Run the following command to check whether the data has been written to the disk:

```
tail -f /tmp/test.log
```

**Step 9** Log in to the LTS console. On the **Log Management** page, click the target log stream to go to its details page. If log data is displayed on the **Log Search** tab page, log collection is successful.

----End

## 2.5 Using Flume to Report Logs to LTS

Flume is a reliable, high-availability, and distributed system for collecting, aggregating, and transporting massive logs. It allows you to customize various data senders in a log system for better data collection. Flume can also process data simply and write data to various data receivers.

You can collect logs using Flume and report logs using the Kafka protocol supported by LTS. The following are some common data collection scenarios:

1. [Using Flume to Collect Text Logs to LTS](#)
2. [Using Flume to Collect Database Table Data to LTS](#)
3. [Using Flume to Collect Logs Transmitted with the Syslog Protocol to LTS](#)
4. [Using Flume to Collect Logs Transmitted with the TCP/UDP Protocol to LTS](#)
5. [Using Flume to Collect Device Management Data Reported with the SNMP Protocol to LTS](#)
6. [Using Default Interceptors to Process Logs](#)
7. [Using a Custom Interceptor to Process Logs](#)
8. [Enriching Logs with External Data Sources and Reporting the Logs to LTS](#)

### Prerequisites

- The Java Development Kit (JDK) has been installed on the host.
- Flume has been installed and the JDK path has been configured in the Flume configuration file.

### Using Flume to Collect Text Logs to LTS

You can add the `conf` file to use Flume to collect text logs and report them to LTS by referring to the following example. For details about the following parameters, see [Using Kafka to Report Logs to LTS](#).

```
#Named
a1.sources = r1
a1.channels = c1
a1.sinks = k1

#Source
a1.sources.r1.type = TAILDIR
a1.sources.r1.channels = c1
```

```
a1.sources.r1.filegroups = f1
a1.sources.r1.filegroups.f1 = /tmp/test.txt
a1.sources.r1.fileHeader = true
a1.sources.r1.maxBatchCount = 1000

#Channel
a1.channels.c1.type = memory
a1.channels.c1.capacity = 10000
a1.channels.c1.transactionCapacity = 100

#Sink
a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required.username="${projectId}" password="${accessKey}#${accessSecret}";

#Bind
a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

## Using Flume to Collect Database Table Data to LTS

You can use Flume to collect database table data and report it to LTS to monitor table data changes. For details about the following parameters, see [Using Kafka to Report Logs to LTS](#).

**Step 1** Download the [flume-ng-sql-source](#) plug-in and compress it into a JAR package named **flume-ng-sql-source.jar**. Before packaging, ensure that the version of flume-ng-core in the POM file is the same as that of Flume to be installed. Then, place the JAR package in the **lib** directory in the Flume package installation path, for example, **FLUME\_HOME/lib**. Replace **FLUME\_HOME** with the actual installation path.

**Step 2** Add the MySQL driver to the **FLUME\_HOME/lib** directory.

1. Download the MySQL driver package.  
wget https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.35.tar.gz
2. Decompress the driver package and compress it into a JAR package.  
tar xzf mysql-connector-java-5.1.35.tar.gz
3. Place the JAR package in the **FLUME\_HOME/lib/** directory.  
cp mysql-connector-java-5.1.35-bin.jar FLUME\_HOME/lib/

**Step 3** Add the **conf** file for collecting MySQL data.

```
# a1 indicates the agent name.
# source indicates the input source of a1.
# channels indicates a buffer.
# sinks indicates the output destination of a1. In this example, Kafka is used.
a1.channels = c1
a1.sources = r1
a1.sinks = k1

#source
a1.sources.r1.type = org.keedio.flume.source.SQLSource
# Connect to MySQL: Replace {mysql_host} with the IP address of your VM and {database_name} with the
database name. You can run the ifconfig or ip addr command to obtain the IP address.
# Add ?useUnicode=true&characterEncoding=utf-8&useSSL=false to the URL. Otherwise, the connection
may fail.
a1.sources.r1.hibernate.connection.url = jdbc:mysql://{mysql_host}:3306/{database_name}?
useUnicode=true&characterEncoding=utf-8&useSSL=false
```



```
# Hibernate Database connection properties
# MySQL account, which generally is root.
a1.sources.r1.hibernate.connection.user = root
# Enter your MySQL password.
a1.sources.r1.hibernate.connection.password = xxxxxxxx
a1.sources.r1.hibernate.connection.autocommit = true
# MySQL driver.
a1.sources.r1.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
a1.sources.r1.hibernate.connection.driver_class = com.mysql.jdbc.Driver
# Store the status file.
a1.sources.r1.status.file.path = FLUME_HOME/bin
a1.sources.r1.status.file.name = sqlSource.status
# Custom query
# Replace {table_name} with the name of the data table to be collected. You can also use the following
method:
a1.sources.r1.custom.query = select * from {table_name}

#Sink
a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required.username=${projectId} password=${accessKey}#${accessSecret}";

a1.channels.c1.type = memory
a1.channels.c1.capacity = 10000
a1.channels.c1.transactionCapacity = 10000
a1.channels.c1.byteCapacityBufferPercentage = 20
a1.channels.c1.byteCapacity = 800000
```

**Step 4** After being started, Flume collects table data from the database to LTS.

----End

## Using Flume to Collect Logs Transmitted with the Syslog Protocol to LTS

Syslog is a protocol used to transmit log messages on an IP network. Flume collects logs transmitted using syslog and reports them to LTS. For details about the following parameters, see [Using Kafka to Report Logs to LTS](#).

- To receive UDP logs, add the **conf** file for collecting logs transmitted using syslog. Example:

```
a1.sources = r1
a1.sinks = k1
a1.channels = c1

a1.sources.r1.type=syslogudp
# host_port indicates the port number of the syslog server.
a1.sources.r1.port = {host_port}
# host_ip indicates the IP address of the syslog server.
a1.sources.r1.host = {host_ip}
a1.sources.r1.channels = c1

a1.channels.c1.type = memory

#Sink
a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
```

```
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="${projectId}" password="${accessKey}#${accessSecret}";
a1.sinks.k1.channel = c1
```

- To receive TCP logs, add the **conf** file for collecting logs transmitted using syslog. Example:

```
a1.sources = r1
a1.sinks = k1
a1.channels = c1

a1.sources.r1.type=syslogtcp
# host_port indicates the port number of the syslog server.
a1.sources.r1.port = {host_port}
# host_ip indicates the IP address of the syslog server.
a1.sources.r1.host = {host_ip}
a1.sources.r1.channels = c1

a1.channels.c1.type = memory

#Sink
a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="${projectId}" password="${accessKey}#${accessSecret}";
a1.sinks.k1.channel = c1
```

## Using Flume to Collect Logs Transmitted with the TCP/UDP Protocol to LTS

You can use Flume to collect logs transmitted with the TCP/UDP protocol and report them to LTS. For details about the following parameters, see [Using Kafka to Report Logs to LTS](#).

- To collect TCP port logs, add the **conf** file of the collection port. Example:

```
a1.sources = r1
a1.sinks = k1
a1.channels = c1

a1.sources.r1.type = netcat
a1.sources.r1.bind = 0.0.0.0
a1.sources.r1.port = {host_port}

a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="${projectId}" password="${accessKey}#${accessSecret}";

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

- To collect UDP port logs, add the **conf** file of the collection port. Example:

```
a1.sources = r1
a1.sinks = k1
```

```
a1.channels = c1

a1.sources.r1.type = netcatudp
a1.sources.r1.bind = 0.0.0.0
a1.sources.r1.port = {host_port}

a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="${projectId}" password="${accessKey}#${accessSecret}";

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

## Using Flume to Collect Device Management Data Reported with the SNMP Protocol to LTS

You can use Flume to collect device management data reported with SNMP and send the data to LTS. For details about the following parameters, see [Using Kafka to Report Logs to LTS](#).

- Listen to port number 161 of the SNMP communication by adding the **conf** file for receiving SNMP logs. Example:

```
a1.sources = r1
a1.sinks = k1
a1.channels = c1

a1.sources.r1.type = netcatudp
a1.sources.r1.bind = 0.0.0.0
a1.sources.r1.port = 161

a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="${projectId}" password="${accessKey}#${accessSecret}";

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

- Listen to port number 162 of SNMP trap communication by adding the **conf** file for receiving SNMP logs. Example:

```
a1.sources = r1
a1.sinks = k1
a1.channels = c1

a1.sources.r1.type = netcatudp
a1.sources.r1.bind = 0.0.0.0
a1.sources.r1.port = 162
```

```
a1.sinks.k1.type = org.apache.flume.sink.kafka.KafkaSink
a1.sinks.k1.kafka.topic = ${logGroupId}_${logStreamId}
a1.sinks.k1.kafka.bootstrap.servers = ${ip}:${port}
a1.sinks.k1.kafka.producer.acks = 0
a1.sinks.k1.kafka.producer.security.protocol = SASL_PLAINTEXT
a1.sinks.k1.kafka.producer.sasl.mechanism = PLAIN
a1.sinks.k1.kafka.producer.compression.type = gzip
a1.sinks.k1.kafka.producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="${projectId}" password="${accessKey}#${accessSecret}";

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

## Using Default Interceptors to Process Logs

In Flume, an interceptor is a simple pluggable component between a source and channels. Before a source writes received events to channels, the interceptor can convert or delete the events. Each interceptor processes only the events received by a given source.

- **Timestamp interceptor**

This interceptor inserts a timestamp into Flume event headers. If no interceptor is used, Flume receives only messages. To configure the connection between a source and a timestamp interceptor, set **type** to **timestamp** and the class name full path **preserveExisting** to **false**. You can also set **preserveExisting** to **true**, indicating the value of the timestamp header will not be replaced if the header already exists in the event. Configuration example:

```
a1.sources.r1.interceptors = timestamp
a1.sources.r1.interceptors.timestamp.type=timestamp
a1.sources.r1.interceptors.timestamp.preserveExisting=false
```

- **Regex filtering interceptor**

This interceptor filters out unnecessary logs during collection or selectively collects logs that meet a specified regular expression. Set **type** to **REGEX\_FILTER**. If you set **excludeEvents** to **false**, events that match a specified regular expression are collected. If you set **excludeEvents** to **true**, matched events are deleted and unmatched events are collected. Example configuration for connecting a source to a regex filtering interceptor:

```
a1.sources.r1.interceptors = regex
a1.sources.r1.interceptors.regex.type=REGEX_FILTER
a1.sources.r1.interceptors.regex.regex=(today)|(Monday)
a1.sources.r1.interceptors.regex.excludeEvents=false
```

In this way, the configured interceptor collects only the logs containing **today** or **Monday** in the log messages.

- **Search and replace interceptor**

This interceptor provides a simple string-based search and replacement function based on Java regular expressions. Configuration example:

```
# Interceptor alias
a1.sources.r1.interceptors = search-replace
# Interceptor type. The value must be search_replace.
a1.sources.r1.interceptors.search-replace.type = search_replace
```

```
# Delete characters from the event body and match the event content based on a specified regular
expression.
a1.sources.r1.interceptors.search-replace.searchPattern = today
# Replace the matched event content.
a1.sources.r1.interceptors.search-replace.replaceString = yesterday
# Set a character set. The default value is utf8.
a1.sources.r1.interceptors.search-replace.charset = utf8
```

## Using a Custom Interceptor to Process Logs

The following uses Java as an example to describe how to customize an interceptor in Flume. **FLUME\_HOME** specifies where Flume is installed. It is **/tools/flume** in this example and needs to be replaced with the actual Flume installation directory.

### Step 1 Create a Maven project and introduce the Flume dependency.

Ensure that the dependency matches the Flume version in the target cluster.

```
<dependencies>
  <dependency>
    <groupId>org.apache.flume</groupId>
    <artifactId>flume-ng-core</artifactId>
    <version>1.10.1</version>
    <scope>provided</scope>
  </dependency>
</dependencies>
```

The dependency does not need to be packaged into the final JAR package. Therefore, set the scope to **provided**.

### Step 2 Create a class to implement the **Interceptor** interface and related methods.

- **initialize()** method: initializes the interceptor, reads configuration information, and establishes connections.
- **intercept(Event event)** method: intercepts and processes a single event, and receives an event object as input and returns a modified event object.
- **intercept(List<Event> list)** method: processes events in a batch, and intercepts and processes the event list.
- **close ()** method: closes the interceptor, releases the resources, and closes the connections.

```
import org.apache.flume.Event;
import org.apache.flume.interceptor.Interceptor;

import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.List;

public class TestInterceptor implements Interceptor {
    @Override
    public void initialize() {
    }
    @Override
    public Event intercept(Event event) {
        // Obtain event data.
        String eventData = new String(event.getBody(), StandardCharsets.UTF_8);
        // Check whether the event data contains a specified string.
        if (eventData.contains("hello")) {
            // If an event contains the specified string, the event is excluded and null is returned.
            return null;
        }
    }
}
```

```
        return event;
    }

    @Override
    public List<Event> intercept(List<Event> events) {
        // Create a new list to store the processed events.
        List<Event> interceptedEvents = new ArrayList<>();
        for (Event event : events) {
            Event interceptedEvent = intercept(event);
            if (interceptedEvent != null) {
                interceptedEvents.add(interceptedEvent);
            }
        }
        return interceptedEvents;
    }

    @Override
    public void close() {
    }
}
}
```

**Step 3** Build an interceptor. The creation and configuration of an interceptor is usually implemented in Builder mode. The complete code is as follows:

```
import org.apache.flume.Context;
import org.apache.flume.Event;
import org.apache.flume.interceptor.Interceptor;

import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.List;

public class TestInterceptor implements Interceptor {
    @Override
    public void initialize() {
    }
    @Override
    public Event intercept(Event event) {
        // Obtain event data.
        String eventData = new String(event.getBody(), StandardCharsets.UTF_8);
        // Check whether the event data contains a specified string.
        if (eventData.contains("hello")) {
            // If an event contains the specified string, the event is excluded and null is returned.
            return null;
        }
        return event;
    }
}

@Override
public List<Event> intercept(List<Event> events) {
    List<Event> interceptedEvents = new ArrayList<>();
    for (Event event : events) {
        Event interceptedEvent = intercept(event);
        if (interceptedEvent != null) {
            interceptedEvents.add(interceptedEvent);
        }
    }
    return interceptedEvents;
}

@Override
public void close() {
}

// Build an interceptor.
public static class Builder implements Interceptor.Builder {
```

```
@Override
public void configure(Context context) {

}

@Override
public Interceptor build() {
    return new TestInterceptor();
}

}
}
```

**Step 4** Convert the interceptor to a JAR package and upload it to the **lib** folder in your Flume installation directory.

**Step 5** Compile the configuration file and configure the custom interceptor in it.

When configuring the full class name of the interceptor, note that the format is *Full class name of the interceptor*\$Builder.

```
# Configuration of the interceptor.
#Definition of the interceptor.
a1.sources.r1.interceptors = i1
# Full class name of the interceptor.
a1.sources.r1.interceptors.i1.type = TestInterceptor$Builder
```

**Step 6** Run Flume.

----End

KV parsing logs: Strings are separated by spaces and converted into strings of the Map type.

```
public class TestInterceptor implements Interceptor {
    @Override
    public void initialize() {
    }

    @Override
    public Event intercept(Event event) {
        // Obtain event data.
        String eventData = new String(event.getBody(), StandardCharsets.UTF_8);
        Map<String, Object> splitMap = new HashMap<>();
        String[] splitList = eventData.split(" ");
        for (int i = 0; i < splitList.length; i++) {
            splitMap.put("field" + i, splitList[i].trim());
        }
        eventData.setBody(splitMap.toString().getBytes(StandardCharsets.UTF_8));
        return event;
    }

    @Override
    public List<Event> intercept(List<Event> events) {
        List<Event> interceptedEvents = new ArrayList<>();
        for (Event event : events) {
            Event interceptedEvent = intercept(event);
            if (interceptedEvent != null) {
                interceptedEvents.add(interceptedEvent);
            }
        }
        return interceptedEvents;
    }

    @Override
    public void close() {
    }
}
```

## Enriching Logs with External Data Sources and Reporting the Logs to LTS

An event is the basic unit for Flume to transmit data from a source to a destination. An event consists of a header and a body. The header stores certain attributes of the event in the K-V structure, and the body stores the data in the byte array format.

If an external data source is available and you want to enrich the log content, such as modifying or deleting log content, or adding fields, add the content changes to the event's body so that Flume can report them to LTS. The following example shows how to use Java to extend the log content. For details about the following parameters, see [Using Kafka to Report Logs to LTS](#).

```
import com.alibaba.fastjson.JSONObject;

import org.apache.flume.Context;
import org.apache.flume.Event;
import org.apache.flume.interceptor.Interceptor;

import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.List;

public class TestInterceptor implements Interceptor {
    @Override
    public void initialize() {
    }

    @Override
    public Event intercept(Event event) {
        // Obtain the event data, convert the original data into JSON strings, and add extra fields.
        String eventData = new String(event.getBody(), StandardCharsets.UTF_8);
        JSONObject object = new JSONObject();
        object.put("content", eventData);
        object.put("workLoadType", "RelipcaSet");
        eventData = object.toJSONString();
        event.setBody(eventData.getBytes(StandardCharsets.UTF_8));
        return event;
    }

    @Override
    public List<Event> intercept(List<Event> events) {
        List<Event> interceptedEvents = new ArrayList<>();
        for (Event event : events) {
            Event interceptedEvent = intercept(event);
            if (interceptedEvent != null) {
                interceptedEvents.add(interceptedEvent);
            }
        }
        return interceptedEvents;
    }

    @Override
    public void close() {
    }
}
```

## 2.6 Collecting Zabbix Data Through ECS Log Ingestion

Zabbix is a common open-source monitoring system with various alarm rules. LTS collects monitoring data from Zabbix to log streams. This section describes how LTS collects Zabbix data through ECS log ingestion.



## Prerequisites

- Prepare an ECS for log collection. For details, see [Purchasing an ECS](#). If you already have an available ECS, skip this step.
- Zabbix has been downloaded and installed on the ECS. For details, see [Download and install Zabbix](#).

## Configuring a Path for Storing Monitoring Data

Zabbix saves monitoring data on the server where Zabbix is located. You can perform the following steps to set a monitoring data storage path:

1. Log in to the server where Zabbix is located.
2. Open the **zabbix\_server.conf** file.  

```
vim /etc/zabbix/zabbix_server.conf
```
3. Set the data storage path in the file.  

```
ExportDir=/tmp/
```
4. Restart the Zabbix service for the configuration to take effect.  

```
systemctl restart zabbix-server
```

After the configuration takes effect, Zabbix generates a file (with the extension `.ndjson`) in the `/tmp` directory to save monitoring data.

## Ingesting ECS Logs to LTS

**Step 1** Choose **Log Ingestion > Ingestion Center** in the navigation pane and click **ECS (Elastic Cloud Server)**.

**Step 2** The page for selecting a log stream is displayed.

1. Select a log group from the drop-down list of **Log Group**, for example, **lts-group-ECS**.
2. Select a log stream from the drop-down list of **Log Stream**, for example, **lts-topic-ECS**.
3. Click **Next: (Optional) Select Host Group**.

**Step 3** Select host groups and click **Next: Configurations**.

**Step 4** Set the collection path to `/tem/**/*ndjson` and retain the default values for other parameters. For details, see [Ingesting ECS Text Logs to LTS](#).

**Figure 2-21** Configuring the collection

**Basic Settings**

\* Collection Configuration Name ⓘ  
Zabbix [Import Configuration](#)

\* Collection Paths  
/tem/\*\*/\*.ndjson [Add Custom Wrapping Rule](#)  
+ Add Collection Path  
[Guidelines for Adding Collection Paths of Linux and Windows Hosts. To avoid incorrect paths, use path verification](#)

Allows multiple file collection  
  
After enabled, the same log file on the same host can be collected to multiple log streams. This function requires a valid ICAgent version. [Version Details](#)  
After disabled, the collection path cannot be configured repeatedly. That is, the collection path of the same log file on the same host can be configured only once even if the log file is stored in different log streams.  
**The Windows scenario is not supported.**

Set Collection Filters ⓘ

Collect Windows Event Logs ⓘ

**Step 5** Click **Next: Index Settings**. On the displayed page, retain the default parameter settings. After configuring the index, you can query and analyze logs. For more information, see [Setting Indexes](#).

**Step 6** Click **Submit**. After the ingestion configuration is complete, click **Back to Ingestion Configurations**. An ingestion configuration will be displayed on the **Ingestion Management** page.

**Step 7** After the log ingestion is configured, you can view the reported logs on the LTS console in real time.

Click the log stream name in the **Log Stream** column of the target ingestion rule to access the stream details page.

**Step 8** Click the **Real-Time Logs** tab to view logs in real time.

Logs are reported to LTS once every five seconds. You may wait for at most five seconds before the logs are displayed.

----End

## 2.7 Collecting Logs from Multiple Channels to LTS

Log ingestion is the process of collecting log data generated during the running of applications or services and storing the data to specified locations for subsequent analysis and use. Log data includes system running status, error information, and user operation records. Log data is important for system O&M, troubleshooting, and service analysis.

LTS enables real-time log ingestion via various methods. Logs can be collected using ICAgent, ingested from cloud services, or reported to LTS via custom software, APIs, or SDKs, meeting requirements in different scenarios.

- More than 40 Huawei Cloud services
- Web, iOS, Android, and applets
- Open-source software, such as Logstash, Flume, and Beats

- Standard protocols, such as HTTP, HTTPS, syslog, and Kafka

## Applications Scenarios

This practice uses a food take-out and delivery platform as an example. This platform allows order foods via websites, apps, WeChat, and Alipay. Once restaurants confirm and prepare the orders, the platform notifies nearby delivery drivers, who then deliver the foods to customers.

During data-based operations, data collection is challenging for the following reasons:

- **Multiple channels:** Data is scattered across channels, including advertisers and promoters.
- **Various devices:** Data comes from websites, official accounts, mobile phones, browsers (web and mobile pages), etc.
- **Heterogeneity:** Data sources include VPCs, customer-built IDCs, and Huawei Cloud ECSs.
- **Multiple development languages:** Different languages are used, with Java for the core system, Nginx for the frontend server, and C++ for the backend payment system.
- **Diverse architectures:** Restaurant devices run on both x86 and Arm.

In this case, you can use LTS to collect internal and external logs for [central management](#).

- [Collecting Promotion Logs](#)
- [Collecting Device User Logs](#)
- [Collecting Server Logs](#)
- [Collecting Logs from Various Network Environments](#)

## Centrally Managing Logs

1. On the LTS console, create a log group. For details, see [Managing Log Groups](#).
2. Create log streams for logs generated by different data sources. For details, see [Managing Log Streams](#). The following log streams are for reference only.
  - **wechat-server** for storing WeChat server access logs
  - **wechat-app** for storing WeChat server application logs
  - **wechat-error** for storing error logs
  - **alipay-server**
  - **alipay-app**
  - **deliver-app** for storing the app statuses of delivery drivers
  - **deliver-error** for storing error logs
  - **web-click** for storing logs of HTML5 page clicks
  - **server-access** for storing server-side access logs
  - **server-app** for storing server application logs
  - **coupon** for storing application coupon logs

- **pay** for storing payment logs
- **order** for storing order logs

## Collecting Promotion Logs

Restaurants usually use the following promotion ways:

- Issue coupons when customers sign up.
- Distribute coupons via QR codes on flyers, web pages or other channels.

Before promotion, set the following registration server address to generate a QR code for flyers or web pages. When a customer scans the QR code for registration, the platform recognizes that the customer is accessing it from the specific source and records a log.

```
http://example.com/login?source=10012&ref=kd4b
```

When the server side receives a request, the server generates a log like the following one:

```
2024-06-20 19:00:00 e41234ab342ef034,102345,5k4d,467890
```

Promotion logs can be collected in the following ways:

- Applications output logs to hard disks, and ICAgent collects logs. For details, see [Ingesting ECS Text Logs to LTS](#).
- Application logs are written to LTS via SDKs. For details, see [SDK Overview](#).

## Collecting Device User Logs

LTS collects logs from various devices, including web browsers, iOS, Android, Baidu applets, WeChat applets, DingTalk applets, and quick apps. You can quickly integrate LTS's mobile SDKs to your devices to enable functions such as cache sending, retry upon exceptions, and batch sending.

Collection solution:

- Mobile devices: Use iOS and Android SDKs to ingest logs.
- Arm devices: Use Native C for cross compilation.
- Restaurant devices: Use SDKs for x86 devices, and Native C for cross compilation for Arm devices.

You can use the following methods to collect device user logs:

- Create a log collection task to write logs to a specified log stream.
- Ingest CCE logs to LTS to collect logs generated in Docker. For details, see [Ingesting CCE Application Logs to LTS](#).
- Use SDKs to write C#, Python, Java, PHP, and C logs into LTS. For details, see [SDK Overview](#).

## Collecting Server Logs

The following server logs are for reference only.

- Syslog logs

```
Aug 31 11:07:24 zhouqi-mac WeChat[9676]: setupHotkeyListenning event NSEvent: type=KeyDown  
loc=(0,703) time=115959.8 flags=0 win=0x0 winNum=7041 ctxt=0x0 chars="u" unmodchars="u"  
repeat=0 keyCode=32
```

- Application debug logs

```
__FILE__:build/release64/sls/shennong_worker/ShardDataIndexManager.cpp  
__LEVEL__:WARNING  
__LINE__:238  
__THREAD__:31502  
offset:816103453552  
saved_cursor:1469780553885742676  
seek count:62900  
seek data redo  
log:pangu://localcluster/redo_data/41/example/2016_08_30/250_1472555483  
user_cursor:1469780553885689973
```

- Trace logs

```
[2013-07-13 10:28:12.772518] [DEBUG] [26064] __TRACE_ID__:661353951201 __item__:  
[Class:Function]_end__ request_id:1734117 user_id:124 context:.....
```

You can use the following methods to collect server logs:

- Create a log collection task to write logs to a specified log stream.
- Ingest CCE logs to LTS to collect logs generated in Docker. For details, see [Ingesting CCE Application Logs to LTS](#).
- Use SDKs to write C#, Python, Java, PHP, and C logs into LTS. For details, see [SDK Overview](#).

## Collecting Logs from Various Network Environments

LTS provides access points in each region, enabling three access methods:

- Intranet (classic network): Cloud services in the same region as LTS can access LTS via the intranet, which features high-bandwidth links and is the recommended method.
- Public network (classic network): Any access is allowed. The access speed depends on the link quality. In this case, HTTPS is recommended for transmission security.
- Private network (VPC): Cloud services in the same region as LTS can access LTS through VPCs.

# 3 Log Search and Analysis

---

## 3.1 Analyzing Huawei Cloud ELB Logs on LTS

### Solution Overview

When distributing external traffic, ELB logs details of HTTP and HTTPS requests, such as URIs, client IP addresses and ports, and status codes.

You can use ELB access logs for auditing or search for logs by time and keyword. You can also obtain external access statistics by running SQL aggregation queries. For example, you can check the number of requests with 404 responses within a certain day, or analyze the unique visitors (UVs) or page views (PVs) within a week.

### Planning Resources


You have purchased and used a Huawei Cloud load balancer.


### Restrictions

ELB access logs only record layer 7 requests sent to the dedicated and shared load balancers. Requests to layer 4 shared load balancers are not logged.

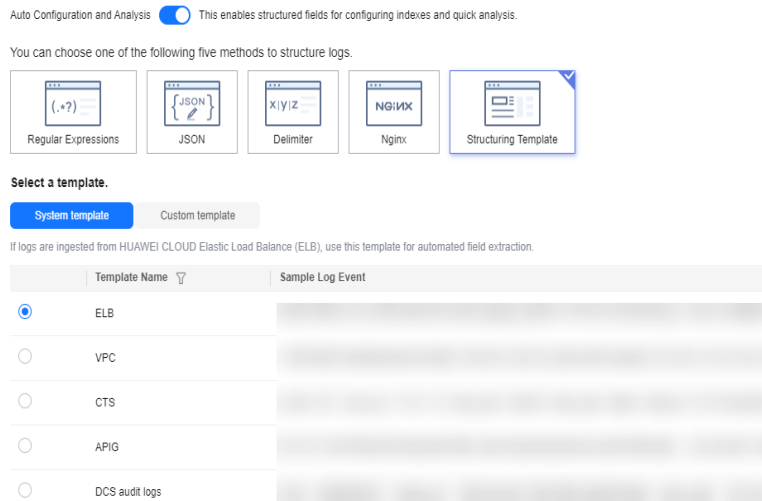
### Analyzing Huawei Cloud ELB Logs on LTS

**Step 1** Ingest ELB access logs to LTS. For details, see [Access Logging](#).

**Step 2** Click  in the upper left corner and choose **Management & Governance > Log Tank Service**.

**Step 3** On the **Log Management** page, click a log stream name. On the log stream details page displayed, click  in the upper right corner. Go to the **Cloud Structuring Parsing** tab page, retain the default setting (enabled) of **Auto Configuration and Analysis**, click **Structuring Template**, select the ELB system template, and click **Save**. For details about quick analysis, see [Quick analysis](#).

**Figure 3-1** Selecting the ELB structuring template



**Step 4** On the log stream details page, click the **Log Analysis** tab and run SQL queries and analysis. For details about how to visualize query results, see [Setting Cloud Structuring Parsing](#).

- To count the PVs within a week, run the following SQL statement:  

```
select count(*) as pv
```
- To count the UVs within a week, run the following SQL statement:  

```
select count(distinct remote_port) as uv
```
- Statistics on 2xx/3xx/4xx/5xx (return codes) returned by all URIs in one day are collected to show the service execution result. The SQL query and analysis statements are as follows:  

```
select host, router_request_uri as url, count(*) as pv,
sum(case when status >= 200 and status < 300 then 1 else 0 end ) as "2xx times",
sum(case when status >= 300 and status < 400 then 1 else 0 end ) as "3xx times",
sum(case when status >= 400 and status < 500 then 1 else 0 end ) as "4xx times",
sum(case when status >= 500 and status < 600 then 1 else 0 end ) as "5xx times"
group by host, router_request_uri
order by pv desc
limit 100
```

The query results can be displayed in a table, or bar, line, pie, or number chart. For details, see [Statistical Charts](#).

----End

## 3.2 Viewing ELB Log Analysis Results on the LTS Dashboards

You can run SQL statements to query and analyze ELB logs ingested to LTS in real time, save the results as charts, and view the charts in dashboards.

### Prerequisites

- ELB logs have been collected. For details, see [Ingesting ELB Logs to LTS](#).
- Log structuring has been configured. For details, see [Setting Cloud Structuring Parsing](#).

## Restrictions


- Up to 100 charts can be created for a log stream.
- Up to 50 charts can be added to a dashboard.
- Up to 10 filters can be added for a dashboard.
- Up to 100 dashboards can be created for a HUAWEI ID.
- Up to 100 dashboard templates can be created for a HUAWEI ID.
- Up to 200 dashboard groups can be created for a HUAWEI ID.
- Up to 200 dashboard template groups can be created for a HUAWEI ID.

## Procedure

1. [Creating a Visual Chart](#)
2. [Adding a Chart to a Dashboard](#)
3. [Adding a Filter](#)

## Creating a Visual Chart

**Step 1** Use SQL statements to query and analyze logs.

1. Log in to the LTS console. In the navigation pane, choose **Log Management**.
2. In the log group list, click  on the left of a group, and click the desired log stream to go to its details page.
3. Click the **Log Analysis** tab. On the tab page displayed, select a time range, enter a SQL statement in the SQL search box, and click **Searching Logs**.
  - If the number of logs generated within the specified time range exceeds 1 billion, iterative query is triggered so you can view all logs in multiple queries. The message **Query status: Results are accurate** is displayed.
  - You can select various charts to display the query result.
  - For details about SQL queries, see [SQL Analysis Syntax](#).

**Step 2** Create a chart.

1. Click **Create**.  
You can also click **Save** to save the current query result as a new chart.
2. On the **Create Chart** page, set parameters as required.  
You can enable **Add to Dashboard** to add the new chart to a dashboard group.
3. Click **OK**.

**Step 3** View the chart.

Click **Show Chart** to view the chart.


----End

## Adding a Chart to a Dashboard

You can add a chart to a dashboard in either of the following ways:

**Method 1:**



**Step 1** Hover your cursor over a chart name, and  will be displayed. Move the cursor onto the icon, and choose **Add to Dashboard** from the hover menu.

**Step 2** In the dialog box displayed, select the target dashboard.

**Step 3** Click **OK**.

----End

#### **Method 2:**

**Step 1** Create a dashboard.

1. In the navigation pane, choose **Dashboards**.
2. On the page displayed, select a dashboard group under **Dashboards**.
3. Click **Add Dashboard**. On the **Create Dashboard** page, set parameters as required.

For details about dashboard parameters, see [Visualizing Logs in Dashboards](#).

**Step 2** Add a chart to the dashboard.

1. On the **Create Dashboard** page, click **Add Chart**. On the page displayed, select the desired [chart](#).
2. Click **OK**.

----End


## **Adding a Filter**

To add a filter based on the configured variables, perform the following steps:

**Step 1** In the navigation pane, choose **Dashboards**.

**Step 2** On the page displayed, select a dashboard group under **Dashboards**.

**Step 3** Click the name of the desired dashboard. Its details page is displayed.

**Step 4** Click . On the **Filter** page, set parameters and click **OK**.

For details about filter parameters, see [Adding a Filter](#).

**Step 5** Adjust the page layout and click **Save**.

**Step 6** Check the filtering result.

----End

## **3.3 Analyzing Huawei Cloud WAF Logs on LTS**

### **Solution Overview**

WAF examines all HTTP and HTTPS requests to detect and block attacks such as SQL injections, cross-site scripting (XSS), Trojan upload, and command or code injections. You can check the access and attack logs for real-time decision-making, device O&M, and service trend analysis.

## Analyzing WAF Logs in LTS

You can [analyze collected WAF logs in LTS](#).

## 3.4 Embedding the LTS Log Query Page into a User-built System

Log query pages can be embedded into your systems. You can use the federation proxy mechanism of Identity and Access Management (IAM) for custom identity broker and embed a login link to your systems so you can view LTS logs in your systems without logging in to the Huawei Cloud console.

### Application Scenarios

- With this function, you can log in to LTS from a user-built system without entering a password. However, you still need to enter a username and password when logging in to the Huawei Cloud LTS console.
- You can quickly integrate the query and analysis capabilities of LTS in external systems (such as an internal O&M or operations system).
- You do not need to manage multiple Huawei Cloud IAM users, facilitating log data sharing and viewing.

### Embedding the LTS Log Query Page into a User-built System

Create an identity broker and an agency in IAM, and then embed the LTS log query page into your system.

**Step 1** Log in to the IAM console, for example, as **DomainA**.

**Step 2** Create an IAM user group (for example, **GroupC**) on the **User Groups** page and grant the **Agent Operator** permissions in global service to the user group. Users granted these permissions can only switch to the delegated account to access the authorized services. For details, see [Creating a User Group and Assigning Permissions](#).

**Step 3** Create an IAM user (for example, **UserB**) on the IAM console and add the user to **GroupC** by referring to [Adding Users to a User Group](#).

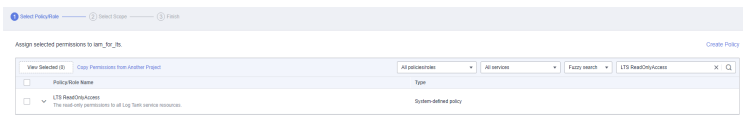
Ensure that the IAM user can use LTS through **programmatic access** or **on the console**. For details about how to change the IAM user access mode, see [Modifying IAM User Information](#).

**Step 4** In the navigation pane, choose **Agencies**. Then, click **Create Agency** in the upper right corner.

**Step 5** Configure agency parameters.

1. For example, set **Agency Name** to **iam\_for\_lts**, **Agency Type** to **Account**, **Delegated Account** to **DomainA**, and **Validity Period** to **Unlimited**, and click **Next**.
2. Set the minimum authorization scope by selecting the **LTS ReadOnlyAccess** permissions, which grant users read-only access to query LTS data without the ability to modify LTS settings, and click **Next**.

**Figure 3-2** Selecting a policy/role



3. Specify the authorization scope, select **Region-specific projects**, select the corresponding region as required, and click **OK**.

**Step 6** Use tools such as Postman to obtain the **X-Subject-LoginToken** parameter. (The following figures are for reference only.)

1. Obtain the **X-Subject-Token** of **UserB** using the account and password.

API type: POST

API URL: Enter **https://Endpoint/v3/auth/tokens**, select the user-defined format for the parameters, and enter the following parameters: **name** indicates the tenant name, username, and tenant name from top to bottom, and **password** indicates the user password.

An endpoint is the **request address** for calling an API. Endpoints vary depending on services and regions. For the endpoints of IAM, see [Regions and Endpoints](#).

```
{
  "auth": {
    "identity": {
      "methods": [
        "password"
      ],
      "password": {
        "user": {
          "domain": {
            "name": "xxxxxxx"
          },
          "name": "xxxxxxx",
          "password": "xxxxxxx"
        }
      }
    },
    "scope": {
      "domain": {
        "name": "xxxxxxx"
      }
    }
  }
}
```

Obtain the **X-Subject-Token** field in the response header.

### Figure 3-3 Response

General  
Request URL: https://iam.myhuaweicloud.com/v3/auth/tokens  
Request Method: POST  
Status Code: 201

Response Headers:  
cache-control: no-cache, no-store, must-revalidate  
connection: keep-alive  
content-length: 5482  
content-type: application/json; charset=UTF-8  
date: Tue, 26 Sep 2023 07:29:37 GMT  
expires: Thu, 01 Jan 1970 00:00:00 GMT  
pragma: no-cache  
server: CloudWAF  
strict-transport-security: max-age=31536000; includeSubdomains;  
x-content-type-options: nosniff  
x-download-options: noopen  
x-frame-options: SAMEORIGIN  
x-iam-trace-id: token\_cn-north-4\_null\_f8530fd2e48e21cc953d448988219b639  
x-request-id: f8530fd2e48e21cc953d448988219b639  
**x-subject-token**  
MIIRSQYJKoZihvcNAQcCoIlROJCCETCYCAQExDTALBglghkgBZQMEAgEwg9bBgkqhkiG9w0BBwGggg9MBIIPSHsidG9r:  
a3jvRmc3TudvocQQBq+4-QlhbppckgY1M3LS7pFv0vW2rGJEPAYrK9V+tb5zBaH5RwE1rfMl99PxmSGSFhLh9EUH6WMM9:  
+Zk1Y26HpaQqrrTKKOG9+PYPRW02ktSvgPaDJoeWIMiyF-5T0Ng3BT3srVFWZb3uPWjhM0Ls2r6w==  
x-xss-protection: 1; mode=block;

2. Obtain the temporary access key based on the **X-Subject-Token** obtained in **1**.

Add the **X-Auth-Token** field to the request header and set its value to the value of **X-Subject-Token** obtained in **1**.

### Figure 3-4 Obtaining a temporary access key

Key	Value
X-Auth-Token	MIIRSQYJKoZihvcNAQcCoIlROJCCETCYCAQExDTALBglghkgBZQMEAgEwg9bBgkqhkiG9w0BBwGggg9MBIIPSHsidG9r: a3jvRmc3TudvocQQBq+4-QlhbppckgY1M3LS7pFv0vW2rGJEPAYrK9V+tb5zBaH5RwE1rfMl99PxmSGSFhLh9EUH6WMM9: +Zk1Y26HpaQqrrTKKOG9+PYPRW02ktSvgPaDJoeWIMiyF-5T0Ng3BT3srVFWZb3uPWjhM0Ls2r6w==

API type: POST

API URL: Enter **https://Endpoint/v3.0/OS-CREDENTIAL/securitytokens**, select the user-defined format for the parameters, and enter the following parameters: **agency\_name** indicates the agency name, **domain\_name** indicates the tenant name, **duration\_seconds** indicates the token expiration time (in second), and **name** indicates the username.

```
{
  "auth": {
    "identity": {
      "methods": [
        "assume_role"
      ],
      "assume_role": {
        "agency_name": "iam_for_its",
        "domain_name": "xxxxxx",
        "duration_seconds": 86400,
        "session_user": {
          "name": "xxxxxx"
        }
      }
    }
  }
}
```

Obtain the temporary access key from the response body.

**Figure 3-5** Obtaining a temporary access key

```
{
  - credential: {
    access: "ZMC5PD5C5IE5V10X4JCE",
    expires_at: "2023-09-27T07:33:18.912000Z",
    secret: "IOA5hKWDuxLYN3uJLUOGqB9g2RDvOFdkRty32h7X",
    securitytoken: "gQqjbi1ub3J0aC00iZMdAa3kx9GOIg0zTTob5wvpFPee-hVQjagvQfE_ε
XhCXSmJw79obJuQVHeLA0SGiPTey_4OBI-5OmBwDuYXgLiXMcTIS4XoXBAxqo4hYR
QGvI4heEj3X834BlpfOApOBLA1433er9ViO6Gz_qio48jXSSyPBQ2i993320D3IBWUA0r
XEIJtk5OplOYWU56DmPHNDvaX1AwxwTzsXGg29dLW27L-RVvp6wN9WGvbgWKJ
iQkAjAMnx6_ajfmcptquc7ibB1.JsoF8vB5baQ8eOKpsSypCqLiSY7vhWgicykmmKUcW_)
uNqz24LzPaxaUZEv9sMeJK9MIq7dfccachmDw5wXGGwQZzIV8bT2GZr15xd0qipVbM
RdefvTQWYon1Qzc3pL5pkw7Qn491FN9rJqpG6IkXiSjihyMY6smZEmBVpRQd75CHUI
1E6YRCvEkQxtCtmqolLuRDzd6-lpEjEKEutLR_fHLPGeOvCmkAklytgkCag-_zFneRlvhr
U19ttPcyVRxsbppknFbox2jVGVyrlHI4GvvfEfZbOYAAQ0jIPgGctfwGaUm8slQCyyBPjP+
XK8UDV8uioCv5QNMkjXLCXiAaW7bshSITqn66b9LCOp36q_CvqfCn2XgWmMzHP2vI
  }
}
```

3. Obtain the login **X-Subject-LoginToken** based on the temporary access key obtained in 2.

API type: POST

API URL: Enter **https://Endpoint/v3.0/OS-AUTH/securitytoken/logintokens**, select the user-defined format for the parameters, and enter the following parameters: The values of **access**, **secret**, and **id** are the values of **access**, **secret**, and **securitytoken** returned in 2, respectively. **duration\_seconds** indicates the token expiration time (in second).

```
{
  "auth" : {
    "securitytoken" : {
      "access" : "xxxxxx",
      "secret" : "xxxxxx",
      "id" : "xxxxxx",
      "duration_seconds" : 43200
    }
  }
}
```

Obtain the **X-Subject-LoginToken** field in the response header.

**Figure 3-6** Obtaining X-Subject-LoginToken

```

General:
Request URL: https://iam.myhuaweicloud.com/v3.0/OS-AUTH/securitytoken/logintokens
Request Method: POST
Status Code: 201

Response Headers:
cache-control: no-cache, no-store, must-revalidate
connection: keep-alive
content-length: 529
content-type: application/json; charset=UTF-8
date: Tue, 26 Sep 2023 07:34:56 GMT
expires: Thu, 01 Jan 1970 00:00:00 GMT
pragma: no-cache
server: CloudWAF
strict-transport-security: max-age=31536000; includeSubdomains;
x-content-type-options: nosniff
x-download-options: noopen
x-frame-options: SAMEORIGIN
x-iam-trace-id: token_cn-north-4_null_dfa3dffde609d11e6f9f5d2bdc669f7e
x-request-id: dfa3dffde609d11e6f9f5d2bdc669f7e
x-subject-logintoken: MIIIEEgYJKoZIhvcNAQcCoIIEAzCCA-
8CAQExDTALBgIghkgBZQMEAgEwgglkBgkqhkiG9w0BBwGggglVBIIICEXsibG9naW50b2tlibl6eyJkb21haW5l
mDmgm7xaRF7MPveGMBMj8worNmn8r+NCKfKGYUpXgHbCFIdnaFbl9YGZWCBBNyul1zTcdlXjK-YZrB5lLs(
WcdOcOQAQWEFVTju9iGnCh6ve3ESULb5+61FQGtkoQ7dXlTjobYlMl5rjnmHSsnKmvblI5eJpsFGddV1nTFC
WDq8ZzMtpZRe8B5NTvOwXvCq5KKBBeup+e6EXGZ2S6uT7THuXYFRuQBIGCJLRsHsC40vw54yAKNOzvTR
x-xss-protection: 1; mode=block;
    
```

**Step 7** Construct a proxy URL based on the **X-Subject-LoginToken** obtained in **3** to complete password-free login.

The rules for constructing a proxy URL are as follows:

```

https://auth.huaweicloud.com/authui/federation/login?
service={target_console_url}&logintoken={logintoken}&idp_login_url={enterprise_s
ystem_loginURL}
    
```

**Table 3-1** URL parameters

Parameter	Description
{target_console_url}	URLEncode encoding result of the LTS address description. For details, see <a href="#">LTS URL</a> .
{logintoken}	URLEncode encoding result of <b>X-Subject-LoginToken</b> obtained in <b>3</b> .
{enterprise_system_login URL}	(Optional) URLEncode encoding result of the customer's page address. When the loginToken verification fails, the page is displayed.

- The preceding three parameters must be encoded using URLEncode. Otherwise, password-free login may fail.
- To perform URLEncode encoding, open a browser, press **F12** to enter the developer mode, select **console**, enter **encodeURIComponent("\*")**, and press **Enter** to view the returned URLEncode value. \* indicates the information to be encoded.

The value of *{target\_console\_url}* is the URLEncode code of the URL of the LTS frontend service. The URL before encoding is as follows. [Table 3-2](#) describes the parameters.

```
https://console-intl.huaweicloud.com/lts/?
region={regionId}&cfModuleHide=header_sidebar_floatlayer#/lts/
logEventsLeftMenu/events?
groupId={groupId}&topicId={topicId}&epsId={epsId}&condition={condition}
```

**Table 3-2** Parameters

Parameter	Description
{regionId}	Region ID. After logging in to the console, obtain the region ID from the address bar of the browser.
{groupId}	Log group ID.
{topicId}	Log stream ID.
{epsId}	ID of the enterprise project of a log stream. If there is no enterprise project, the value is <b>0</b> .
{condition}	Log search criteria, for example, <b>name:a and age:12 and addr:xx</b> . <ul style="list-style-type: none"> <li>Optional</li> <li>The format of a single keyword is <i>key:value</i>.</li> <li>Separate keywords with and.</li> <li>A keyword cannot contain semicolons (;) or colons (:).</li> <li>A keyword that contains special characters (+, =, ?, #, %, and &amp;) must be converted into a hexadecimal value, that is, an ASCII code starting with % (%2B, %3D, %3F, %23, %25, and %26).</li> </ul>

**Step 8** After the preceding steps are complete, you can log in to LTS from your user-built system without entering a password.

Use the following iframe embedding. The value of **src** is the proxy URL obtained in [Step 7](#).

The iframe embedding function requires that browsers allow third-party cookies. The setting procedure varies with browsers. For the Chrome browser, choose **Settings > Privacy and security > Third-party cookies > Allow third-party cookies**.

```
<body>
  <iframe src="target_url" width="100%" height="96%" id="ltsiframePage"></iframe>
</body>
```

----End

## LTS URL

1. The basic URL of the Log Tank Service (LTS) homepage is as follows.

```
https://console-intl.huaweicloud.com/lts/?
region={regionId}&cfModuleHide=header_sidebar_floatlayer_rightsidebar#/cts/manager/groups
```

**Table 3-3** Parameters

Parameter	Mandatory	Type	Description
regionId	Yes	String	Region ID. After logging in to the console, obtain the region ID from the address bar of the browser.

2. The basic URL of the log search page is as follows.

```
https://console-intl.huaweicloud.com/lts/?
region={regionId}&cfModuleHide=header_sidebar_floatlayer_rightsidebar#/cts/logEventsLeftMenu/
events?
groupId={groupId}&topicId={topicId}&epsId={epsId}&hideHeader={hideHeader}&fastAnalysisCollapsed
={fastAnalysisCollapsed}&hideDashboard={hideDashboard}&hideFeedback={hideFeedback}&isFoldLabel
={isFoldLabel}&hideStreamName={hideStreamName}&showK8sFilter={showK8sFilter}&clusterId={clus
terId}&hideBarChart={hideBarChart}&hideTabs={hideTabs}&condition={condition}
```

**Table 3-4** Parameters

Parameter	Mandatory	Type	Default Value	Description
regionId	Yes	String	No	Region ID. After logging in to the console, obtain the region ID from the address bar of the browser.
groupId	Yes	String	No	Log group ID.
topicId	Yes	String	No	Log stream ID.
epsId	No	String	No	ID of the enterprise project of a log stream. If there is no enterprise project, the value is <b>0</b> .
hideHeader	No	Boolean	false	Whether to hide the list on the left and the horizontal log stream list on the top. If yes, set this parameter to <b>true</b> . This parameter takes effect only for iframe embedding.
fastAnalysisCollapsed	No	Boolean	false	Whether to collapse quick analysis. If yes, set this parameter to <b>true</b> .
hideDashboard	No	Boolean	false	Whether to hide the dashboard creation icon. If yes, set this parameter to <b>true</b> .
hideFeedback	No	Boolean	false	Whether to hide the comment button. If yes, set this parameter to <b>true</b> .



Parameter	Mandatory	Type	Default Value	Description
isFoldLabel	No	Boolean	true	Whether to display the <b>label</b> field in a new line in the log table. If yes, set this parameter to <b>true</b> .
hideStreamName	No	Boolean	false	Whether to hide the log stream name. If yes, set this parameter to <b>true</b> .
showK8sFilter	No	Boolean	false	Whether to display the container log filter criteria. For container log search, you can set this parameter to <b>true</b> .
clusterId	No	String	None	Cluster ID. This parameter is mandatory only when <b>showK8sFilter</b> is set to <b>true</b> .
hideBarChart	No	Boolean	false	Whether to collapse the log quantity statistics chart by default. If yes, set this parameter to <b>true</b> .
hideTabs	No	Boolean	false	Whether to hide the <b>Log Search</b> , <b>Log Analysis</b> , and <b>Real-Time Logs</b> tabs. By default, the tabs are not hidden. To hide them, set this parameter to <b>true</b> .
hideShare	No	Boolean	false	Whether to hide the sharing button. By default, the button is not hidden. To hide it, set this parameter to <b>true</b> . This parameter is available only in CN North-Beijing4.
keepOnline	No	Boolean	false	Whether to keep the login state. If you want to stay logged-in and do not log out, set this parameter to <b>true</b> .
condition	No	String	None	Log search criteria, for example, <b>name:a and age:12 and addr:xx</b> . <ul style="list-style-type: none"> <li>• Optional</li> <li>• The format of a single keyword is <i>key.value</i>.</li> <li>• Separate keywords with and.</li> <li>• A keyword cannot contain semicolons (;) or colons (:).</li> <li>• A keyword that contains special characters (+, =, ?, #, %, and &amp;) must be converted into a hexadecimal value.</li> </ul>

3. The basic URL of the visualized log search page is as follows.

```
https://console-intl.huaweicloud.com/lts/?
region={regionId}&cfModuleHide=header_sidebar_floatlayer_rightsidebar#/cts/logEventsLeftMenu/
events?visualization=true&groupId={groupId}&topicId={topicId}&epsId={epsId}&sql={sql}
```

**Table 3-5** Parameters

Parameter	Mandatory	Type	Default Value	Description
regionId	Yes	String	No	Region ID. After logging in to the console, obtain the region ID from the address bar of the browser.
groupId	Yes	String	No	Log group ID.
topicId	Yes	String	No	Log stream ID.
epsId	No	String	No	ID of the enterprise project of a log stream. If there is no enterprise project, the value is <b>0</b> .
hideHeader	No	Boolean	false	Whether to hide the list on the left and the horizontal log stream list on the top. If yes, set this parameter to <b>true</b> .
sql	No	String	No	SQL query statement, for example, SELECT count (*).

- The basic URL of the dashboard page is as follows.

```
https://console-intl.huaweicloud.com/lts/?
region={regionId}&cfModuleHide=header_sidebar_floatlayer_rightsidebar#/cts/manager/dashboard?
dashboardId={dashboardId}&hideDashboardList={hideDashboardList}&showCurrentdashboardGroup={
showCurrentdashboardGroup}&streamId={streamId}&streamDisabled={streamDisabled}&readOnly={re
adonly}&filter=key1:value1,value2;key2:value3,value4&autoFresh={autoFresh}
```

**Table 3-6** Parameters

Parameter	Mandatory	Type	Default Value	Description	Example Value
regionId	Yes	String	No	Region ID. After logging in to the console, obtain the region ID from the address bar of the browser.	region=xx-xx-xx

Parameter	Mandatory	Type	Default Value	Description	Example Value
dashboardId	No	String	None	ID of the dashboard to be displayed. The default value is "". Add this parameter when you want to display a dashboard by default.	dashboardId=xxxxxxx
hideDashboardList	No	Boolean	false	Indicates whether to hide the dashboard drop-down list box. By default, the drop-down list box is not hidden. To hide it, set this parameter to <b>true</b> . Set this parameter to <b>true</b> when you want to hide the dashboard drop-down list box.	hideDashboardList=true
showCurrentdashboardGroup	No	Boolean	false	Indicates whether to display only the dashboard of the current group or template. The default value is <b>false</b> . Set this parameter to <b>true</b> when you want to display only the dashboard of the current group or template. Note: If <b>hideDashboardList</b> is set to <b>true</b> , this parameter is invalid.	showCurrentdashboardGroup=true

Parameter	Mandatory	Type	Default Value	Description	Example Value
streamId	No	String	None	Log stream ID: The default value is "". This parameter applies only to dashboard templates. Add this parameter when you want to select a specified log stream by default.	streamId=xxxxxx
streamDisabled	No	Boolean	false	By default, log streams can be selected from the log stream drop-down list. If you set this parameter to <b>true</b> , log streams cannot be selected from the drop-down list. This parameter applies only to dashboard templates. Add this parameter when you want to disable the log stream drop-down list.	streamDisabled=true

Parameter	Mandatory	Type	Default Value	Description	Example Value
filter	No	String	None	<p>Filter parameter. The value is the name of the filter to be selected and the selected item.</p> <p><b>key1</b> and <b>key2</b> indicate the filter names. <b>value1</b> and <b>value2</b> indicate the values to be selected for <b>key1</b>. <b>value3</b> and <b>value4</b> indicate the values to be selected for <b>key2</b>. Separate filters by semicolons (;), and selected items by commas (,).</p> <p>Add this parameter when the keys and values of some filters need to be selected by default on the embedded dashboard page.</p>	filter=key1:value1,value2;key2:value3,value4
readonly	No	Boolean	false	<p>Indicates whether the scenario is read-only. In the read-only scenario, operation-related buttons are hidden, for example, creating a filter and adding, modifying, or deleting a dashboard.</p> <p>Add this parameter when you only need to display the dashboard and do not need the operation permission.</p>	readonly=true

Parameter	Mandatory	Type	Default Value	Description	Example Value
autoFresh	No	String	No	<p>Scheduled refresh interval. The default value is "".</p> <p>Add this parameter when you need to specify the default scheduled refresh interval. Currently, the refresh interval can be <b>0m</b> (irregular refresh), <b>1m</b> (scheduled refresh per 1 min), <b>5m</b> (scheduled refresh per 5 min), or <b>15m</b> (scheduled refresh per 15 min).</p>	autoFresh=1m

# 4 Log Transfer

## 4.1 Changing File Time Zones for Log Transfer in a Batch

For recurring tasks like configuring log ingestion, alarm rules, and log transfer, batch operations are not supported on the LTS console. You can use Python scripts and LTS APIs to perform custom operations in a batch.

### Scenario

If you have created 1,000 rules for log transfer to OBS but set all file time zones to **(UTC) Coordinated Universal Time**, you need to change it to UTC+08:00. Currently, the LTS console does not allow batch modifying log transfer rules. Manually modifying each transfer rule will be time-consuming.

### Prerequisites

1. Prepare a Linux host.
2. Check the API document.
  - Obtain information about all transfer tasks by calling the log transfer querying API.
  - Change the time zones configured for transfer tasks by calling the log transfer updating API.
3. Test API functions in API Explorer, which provides API search and platform debugging capabilities.
4. Install the Python SDK on the host by referring to the API Explorer sample code.
  - Python [SDK dependency address](#) and [SDK usage description](#):

```
pip install huaweicloudsdklts
```
  - API Explorer provides sample code for calling APIs using Python. Example:

```
# coding: utf-8
import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdklts.v2.region.lts_region import LtsRegion
from huaweicloudsdkcore.exceptions import exceptions
```

```

from huaweicloudsklts.v2 import *
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has
    # great security risks. It is recommended that the AK and SK be stored in ciphertext in
    # configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before
    # running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the
    # local environment
    /* Hardcoded or plaintext AK and SK are risky. For security, encrypt your AK and SK and store
    them in the configuration file or as environment variables.
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    credentials = BasicCredentials(ak, sk)
    client = LtsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(LtsRegion.value_of("xx")) \
        .build()
    try:
        request = ListTransfersRequest()
        response = client.list_transfers(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)

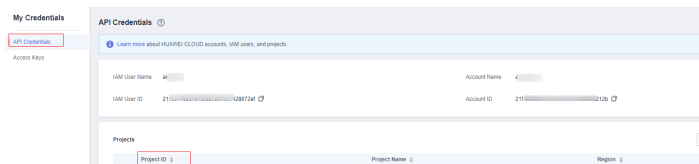
```

## Changing File Time Zones for Log Transfer in a Batch

**Step 1** Obtain the parameter and replace it with the actual value in the code.

- Obtain an AK/SK pair of the Huawei account.
- Obtain the project ID. For details, see [API Credentials](#).

**Figure 4-1** Obtaining the project ID



- Obtain the value of **Region&iam\_Endpoint** from [Regions and Endpoints](#).

**Table 4-1** Endpoints

Region Name	Region	Endpoint	Protocol
AP-Bangkok	ap-southeast-2	lts.ap-southeast-2.myh uaweicloud.com	HTTPS
AP-Singapore	ap-southeast-3	lts.ap-southeast-3.myh uaweicloud.com	HTTPS
CN-Hong Kong	ap-southeast-1	lts.ap-southeast-1.myh uaweicloud.com	HTTPS



- Obtain the time zone and time zone ID.

**Table 4-2** Common time zones

Time Zone	Time Zone ID
UTC-12:00	Etc/GMT+12
UTC-11:00	Etc/GMT+11
UTC-10:00	Pacific/Honolulu
UTC-09:00	America/Anchorage
UTC-08:00	America/Santa_Isabel
UTC-07:00	America/Chihuahua
UTC-06:00	America/Chicago
UTC-05:00	America/New_York
UTC-04:00	America/Santiago
UTC-03:00	America/Sao_Paulo
UTC-02:00	Etc/GMT+2
UTC-01:00	Atlantic/Azoresjavik
UTC+00:00	Europe/London
UTC+01:00	Europe/Parist
UTC+02:00	Europe/Istanbul
UTC+03:00	Europe/Minsk
UTC+04:00	Europe/Moscow
UTC+05:00	Asia/Tashkent
UTC+06:00	Asia/Almaty
UTC+07:00	Asia/Bangkok
UTC+08:00	Asia/Shanghai
UTC+09:00	Asia/Tokyo
UTC+10:00	Asia/Yakutsk
UTC+11:00	Asia/Vladivostok
UTC+12:00	Pacific/Fiji
UTC+13:00	Pacific/Apia

**Step 2** Run the following command on the host to check whether the **huaweicloudsdkcore** and **huaweicloudsklts** packages have been installed:

```
pip list | grep huaweicloudsdk
```

- If they have been installed, the command output displays information about **huaweicloudsdk**.
- If not, no information is returned. Run the following command to install them:

```
pip install huaweicloudsdkcore huaweicloudsklts
```

**Step 3** Run the **vi lts\_python.py** command on the host to create the **lts\_python.py** file. Then, copy the following code to the file to batch change the time zones of files to be transferred to OBS.

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsklts.v2 import *
from huaweicloudsklts.v2.region.lts_region import LtsRegion
/* Hardcoded or plaintext AK and SK are risky. For security, encrypt your AK and SK and store them in the
configuration file or as environment variables.
if __name__ == "__main__":
AK = "your ak"
SK = "your sk"
PROJECT_ID = "your project id"
REGION = "your region"
IAM_ENDPOINT = "iam_endpoint"

OBS_TIME_ZONE = "the time_zone you want to change"
OBS_TIME_ZONE_ID = "time_zone_id"

credentials = BasicCredentials(AK, SK, PROJECT_ID).with_iam_endpoint(IAM_ENDPOINT)

client = LtsClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(LtsRegion.value_of(REGION)) \
    .build()

# 1.get obs transfer task
try:
    request = ListTransfersRequest()
    request.log_transfer_type = "OBS"
    response = client.list_transfers(request)
    obs_transfer_num = len(response.log_transfers)
    task_list = response.log_transfers
    print("#### get {} obs transfer task ####".format(obs_transfer_num))
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)

# 2.set obs transfer task obs_time_zone to UTC+08:00
CNT = 1
while len(task_list):
    transfer_task = task_list.pop()
    print("There are still {} progress: \n".format(len(task_list)), transfer_task)
    try:
        if transfer_task.log_transfer_info.log_transfer_detail.obs_time_zone == OBS_TIME_ZONE:
            CNT += 1
            continue
        request = UpdateTransferRequest()
        transfer_task.log_transfer_info.log_transfer_detail.obs_time_zone = OBS_TIME_ZONE
        transfer_task.log_transfer_info.log_transfer_detail.obs_time_zone_id = OBS_TIME_ZONE_ID
        request.body = UpdateTransferRequestBody(
            log_transfer_info=transfer_task.log_transfer_info,
            log_transfer_id=transfer_task.log_transfer_id
        )
        response = client.update_transfer(request)
        CNT += 1
```

```
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
    task_list.append(transfer_task)
except exceptions.ServerResponseException as e:
    print({
        "target": transfer_task.log_streams,
        "reason": e
    })
    task_list.append(transfer_task)
```

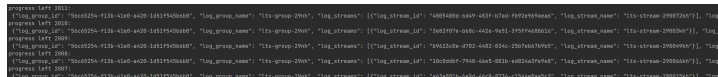
**Step 4** Run the Python script on the host to batch change the time zones of files to be transferred to OBS.

```
nohup python lts_python.py > lts_python.log &
```

**Step 5** View execution logs to check whether the Python script has been executed and whether the files' time zones have been changed.

```
tail -f lts_python.log
```

**Figure 4-2** Viewing execution logs



----End

# 5 Billing

---

## 5.1 Collecting Statistics on LTS Expenses of Different Departments Based on Log Stream Tags

To collect statistics on the LTS expenses of different departments in an enterprise, you can add tags to LTS log streams to distinguish business departments. LTS will add these tags to CDRs sent to the Billing Center. You can download LTS billing details by navigating to **Billing Center > Billing > Expenditure Details**. Then, you can use resource tags to aggregate and analyze expenses of different departments, providing a basis for enterprise expense allocation.

### Prerequisites

The function of reporting CDRs by log stream is available only to whitelist users. To use log stream tags to aggregate and analyze departmental expenses in LTS, [submit a service ticket](#).

### Solution


Log streams are managed in log groups. When you add a tag to a log group, **Apply to Log Stream** is enabled by default to automatically add the tag to streams in this group. This feature allows you to differentiate LTS expenses by department.

This practice uses departments aa and bb as examples. First, add the **group=groupaa** tag to department aa's log group and the **group=groupbb** tag to department bb's log group. Then, export bills from the Billing Center and perform statistical analysis with Excel.

The prices mentioned in the following are only for reference. The actual prices are subject to those in [Price Calculator](#).

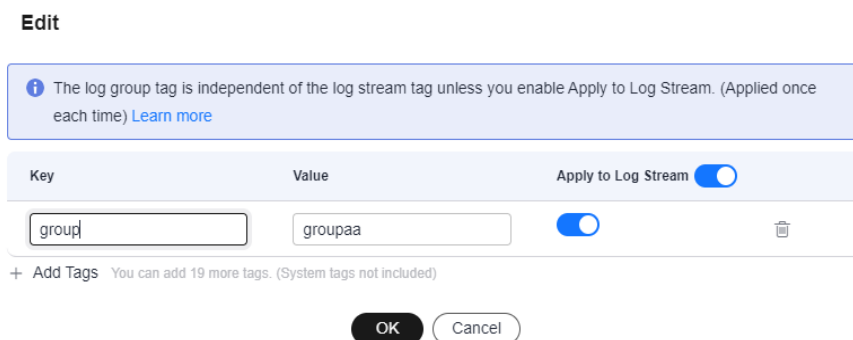
## Collecting Statistics on LTS Expenses of Different Departments Based on Log Stream Tags


**Step 1** Log in to the LTS console.

**Step 2** Move the cursor to the **Tags** column of the log group created for department aa and click .

**Step 3** In the displayed dialog box, click **Add Tags**, enter key **group** and value **groupaa**, retain the default setting (enabled) of **Apply to Log Stream**, and click **OK**.

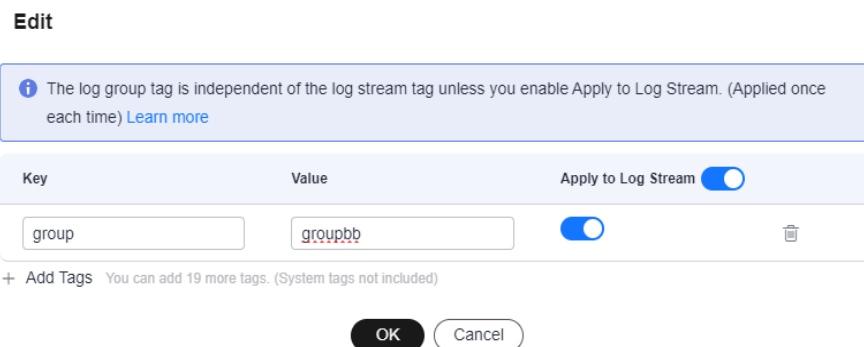
**Figure 5-1** Adding a tag for department aa



**Step 4** Move the cursor to the **Tags** column of the log group created for department bb and click .

**Step 5** In the displayed dialog box, click **Add Tags**, enter key **group** and value **groupbb**, retain the default setting (enabled) of **Apply to Log Stream**, and click **OK**.

**Figure 5-2** Adding a tag for department bb



**Step 6** After the tags are added, wait for about one hour for CDRs to be generated.

**Step 7** On the top menu bar, choose **More > Billing > Bills**. The **Dashboard** page is displayed.

**Step 8** Choose **Expenditure Details** in the navigation pane, select **Usage**, set **Data Period** to **Details**, and select **Service Type: Log Tank Service LTS** in the filter box. For details, see [Bill Details](#).

**Step 9** Click **Export**. On the displayed page, set a custom export scope and export the expenditure details to the local PC. For details, see [Exporting Bills](#).

**Step 10** In the exported Excel file, filter tags in the **Resource Tag** column to view the expenditure details of departments aa and bb.

The actual prices are subject to those in [Price Calculator](#).

----End