Intelligent EdgeFabric

Best Practice

Issue	01
Date	2020-12-14



HUAWEI TECHNOLOGIES CO., LTD.

Copyright © Huawei Technologies Co., Ltd. 2020. 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 Technologies Co., Ltd.

Trademarks and Permissions


and other Huawei trademarks are trademarks 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 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

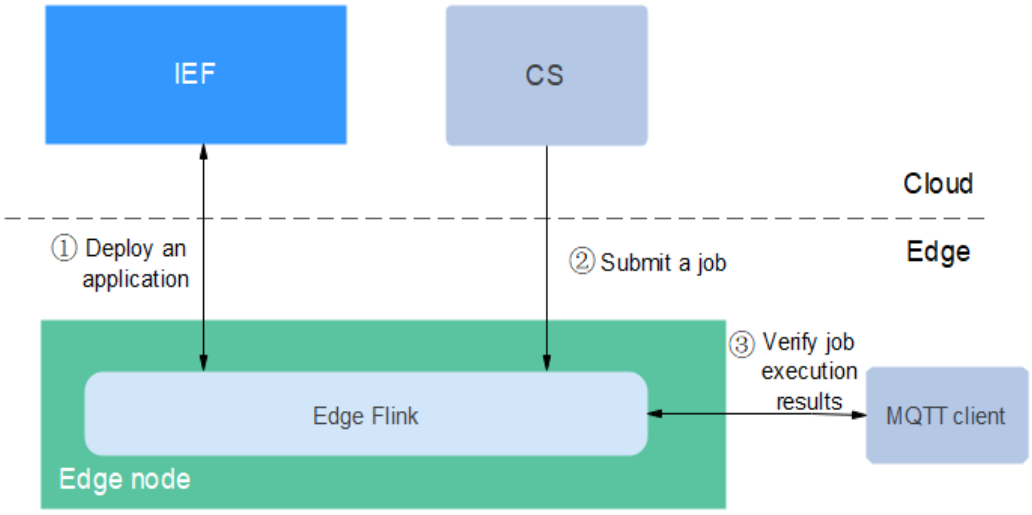
1 Analysis of Real-Time Streams at the Industrial IoT Edge..... 1

 Copyright © Huawei Technologies Co., Ltd.

1 Analysis of Real-Time Streams at the Industrial IoT Edge

In certain scenarios, data needs to be analyzed and processed near where data is generated when a large amount of data is generated on edge devices, which reduces the amount of data to be migrated to the cloud and improves real-time data processing. IEF can work with the Cloud Stream Service (CS). By deploying the stream computing applications on edge nodes, IEF extends the real-time stream computing capability from the cloud to the edge. Then, CS edits and delivers stream processing jobs to edge nodes. In this way, stream data can be quickly and accurately analyzed and processed at the edge.

Figure 1-1 Edge real-time stream analysis



Prerequisites

1. IEF and CS have been enabled.
2. An edge node has been registered and managed.

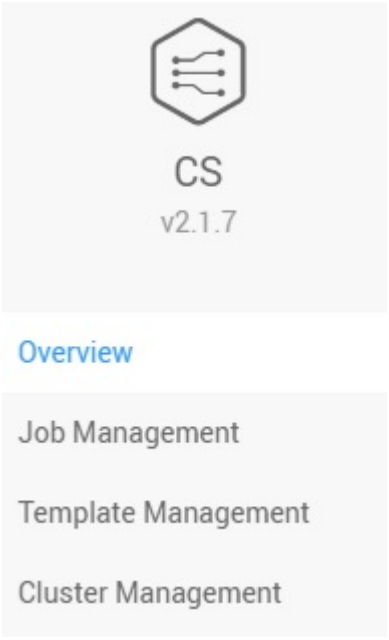
Deploying an Application

Step 1 Log in to the IEF console. In the navigation pane on the left, choose **Edge Marketplace** > **Edge Application Center**, and click **Deploy** on the Edge Flink

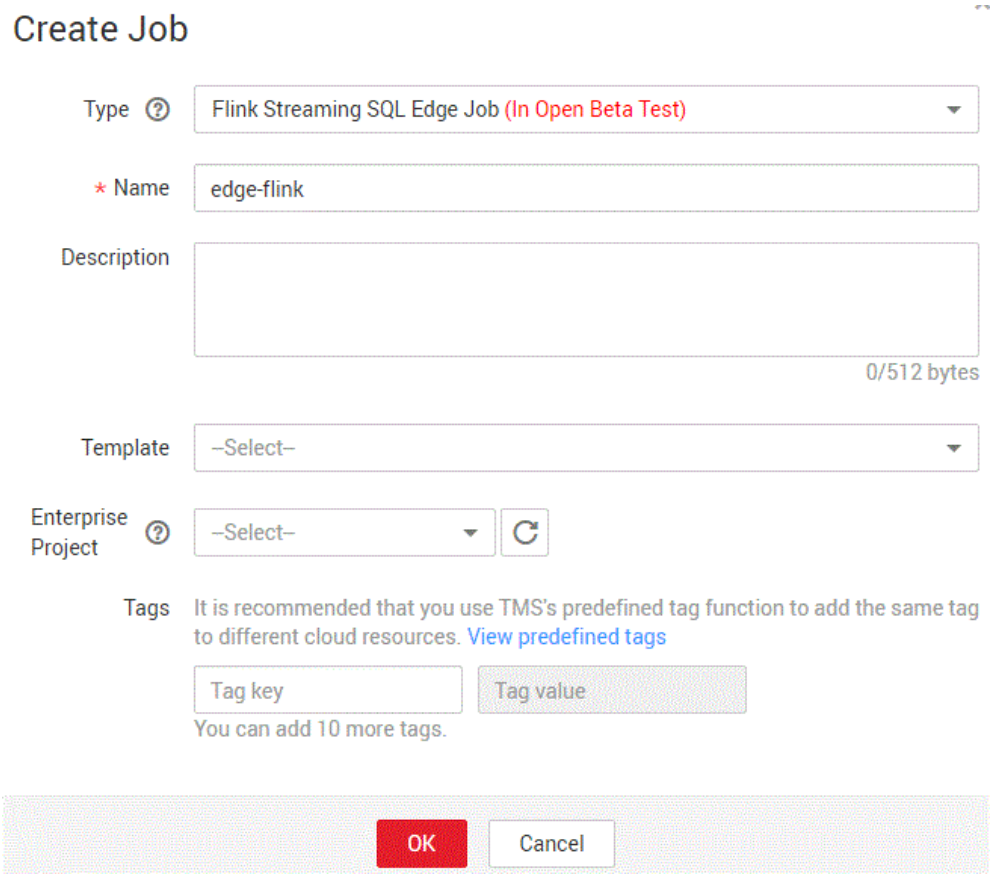
 Copyright © Huawei Technologies Co., Ltd.

card. On the page displayed, associate an edge node and specify parameters. The parameter values cannot be smaller than their default values. Otherwise, the deployment will fail.

Step 2 Log in to the CS console (**https://console.huaweicloud.com/cs/?locale=en-us#/dashboard**). In the navigation pane, choose **Job Management**.



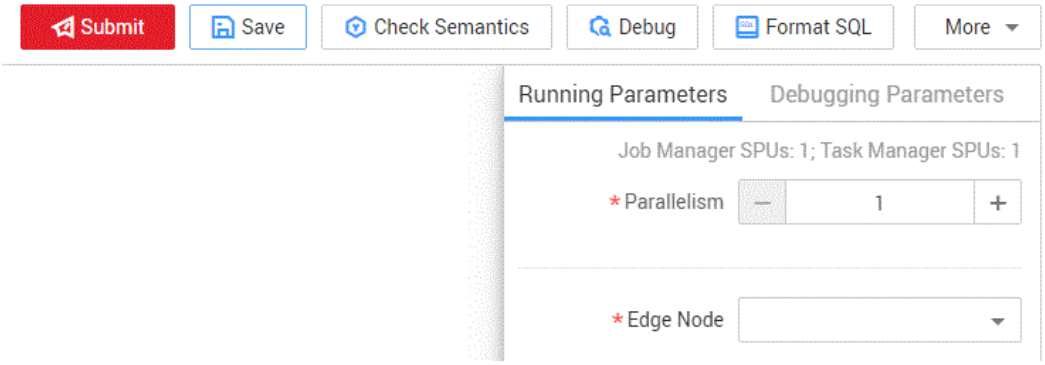
Step 3 Click **Create Job**. The **Create Job** page is displayed.



 Copyright © Huawei Technologies Co., Ltd.

Step 4 Set **Type** to **Flink Streaming SQL Edge Job** and enter the job name. The **Description**, **Template**, and **Labels** field are optional. A job can be quickly created if you use an existing template.

Step 5 Click **OK**. The **Edit** page is displayed.

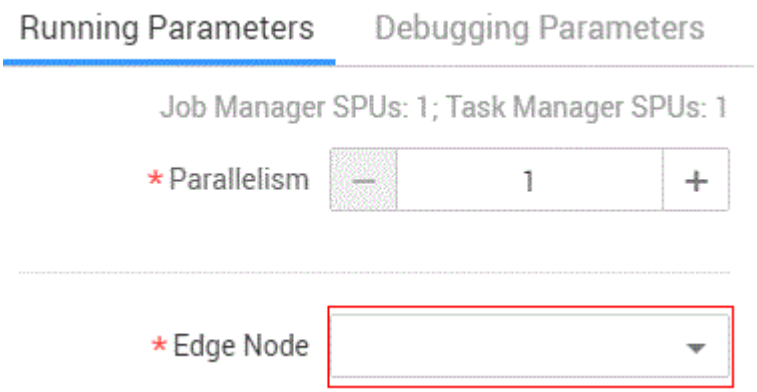


Step 6 Edit the Flink streaming SQL edge job as required to process data generated on edge devices. Currently, **type** can be set to **edgehub** and **encode** can be set to **json** or **csv**. For details about the SQL syntax, see **Syntax Constraints**.

Example: Exporting the names and scores of students who scored 80 or higher

```
create source stream student_scores(name string, score int) with (
type = "edgehub",
topic = "abc",
encode = "json",
json_config = "score = student.score; name=student.name"
);
create sink stream excellent_students(name string, score int) with (
type = "edgehub",
topic = "abcd",
encode = "csv",
field_delimiter = ","
);
insert into excellent_students select name, score from student_scores where score >= 80;
```

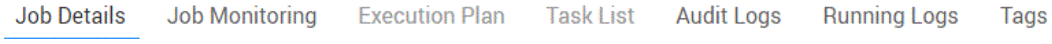
Step 7 On the right of the page, set **Parallelism** and **Edge Node**. **Parallelism** specifies the number of tasks where CS jobs can simultaneously run, and **Edge Node** specifies the edge node to which the job belongs. You can select multiple edge nodes to deploy the job.



Step 8 Click **Submit**. On the displayed **Job Configuration List** page, click **OK** to submit and start the job. You can view the job running status on the **Job Management**

Copyright © Huawei Technologies Co., Ltd.

page and click a specific job to view the job details, job monitoring information, and job execution plans.



----End

Verifying Job Running

Step 1 Log in to a node that interwork with edge nodes. Then, install Mosquitto.

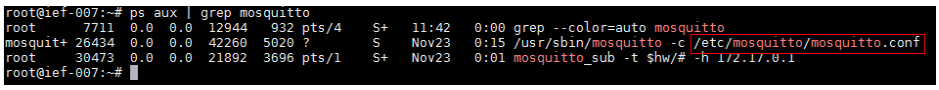
Mosquitto implements the communication among the algorithm, container, and MQTT Broker.

1. Install Mosquitto.
   - In the Ubuntu OS, run the following commands to install Mosquitto:
     ```
     apt-get install mosquitto
     systemctl start mosquitto
     systemctl enable mosquitto
     ```
   - In the CentOS OS, run the following commands to install Mosquitto:
     ```
     yum install epel-release
     yum install mosquitto
     systemctl start mosquitto
     systemctl enable mosquitto
     ```

   NOTE

   For more information about Mosquitto installation, visit the Mosquitto official website **https://mosquitto.org/download/**.

2. Configure Mosquitto.
   a. If a firewall is deployed, add a firewall filtering rule.
      ```
      firewall-cmd --add-port=1883/tcp --permanent
      firewall-cmd --add-port=8883/tcp --permanent
      systemctl restart firewalld
      ```
   b. Run the following command to query the directory where the Mosquitto configuration file is located:
      ```
      ps aux | grep mosquitto
      ```
      The directory of the Mosquitto configuration file is shown in the following figure:

   c. Run the **vim** command to open the configuration file.
      ```
      vim Directory of the Mosquitto configuration file
      (for example, vim /etc/mosquitto/mosquitto.conf)
      ```
   d. By default, Mosquitto listens on port 1883 of the whole network segment. Add the following lines at the end of the configuration file to limit the IP addresses on which Mosquitto listens:
      ```
      bind_address 172.17.0.1
      listener 1883 127.0.0.1
      ```
   e. Query the directory of the Mosquitto startup file and modify the file for the init.d and systemd systems. The file directory is shown in the following figure:
      ```
      systemctl status mosquitto
      ```

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-2 Directory for storing the Mosquitto startup file



- For the init.d system, run the following command to open the Mosquitto startup file:

  ```
  vim /etc/init.d/mosquitto
  ```

  Add **docker** after **# Required-Start: $remote_fs $syslog**.

  Figure 1-3 Modifying the Mosquitto startup file for the init.d system

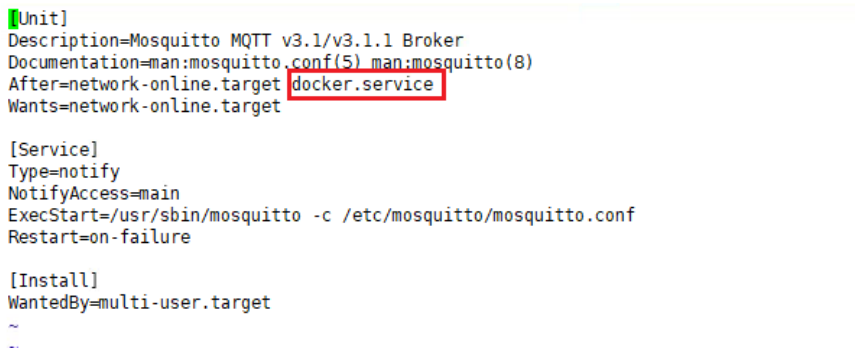
  

- For the systemd system, run the following command to open the Mosquitto startup file:

  ```
  vim /usr/lib/systemd/system/mosquitto.service
  ```

  Add the Docker startup dependency by adding **docker.service** after **After=network-online.target**.

  Figure 1-4 Adding the Docker startup dependency

f. Restart Mosquitto.

Copyright © Huawei Technologies Co., Ltd.

```
systemctl daemon-reload
systemctl restart mosquitto
```

g. Check whether Mosquitto is started.

```
systemctl status mosquitto
```

If **active (running)** is displayed in the command output, Mosquitto is running properly.

NOTE

For more information about Mosquitto configuration, visit the Mosquitto official website **https://mosquitto.org/**.

Step 2 Run the following command to subscribe to data:

mosquitto_sub -h 127.0.0.1 -t *abcd*

In the command, **abcd** is the topic name defined in the job.

Step 3 Open a new window, and run the following command to publish data:

mosquitto_pub -h 127.0.0.1 -t *abc* **-m '{"student":{"score":90,"name":"1bc2"}}'**

In the command, **abcd** is the topic name defined in the job.

After data is published, the data subscribed to in **Step 2** can be received.

----End

Copyright © Huawei Technologies Co., Ltd.