Intelligent EdgeFabric

Best Practices

Issue	02
Date	2022-11-30

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.

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Contents

1 Connecting Edge Nodes to IEF Through Direct Connect or VPN

Scenario

If an edge node cannot access IEF through a public network, it can connect to Huawei Cloud Virtual Private Cloud (VPC) through **Direct Connect** or **Virtual Private Network (VPN)**, and then connect VPC to IEF through the private and secure channel provided by a **VPC endpoint**.

Connection Solution

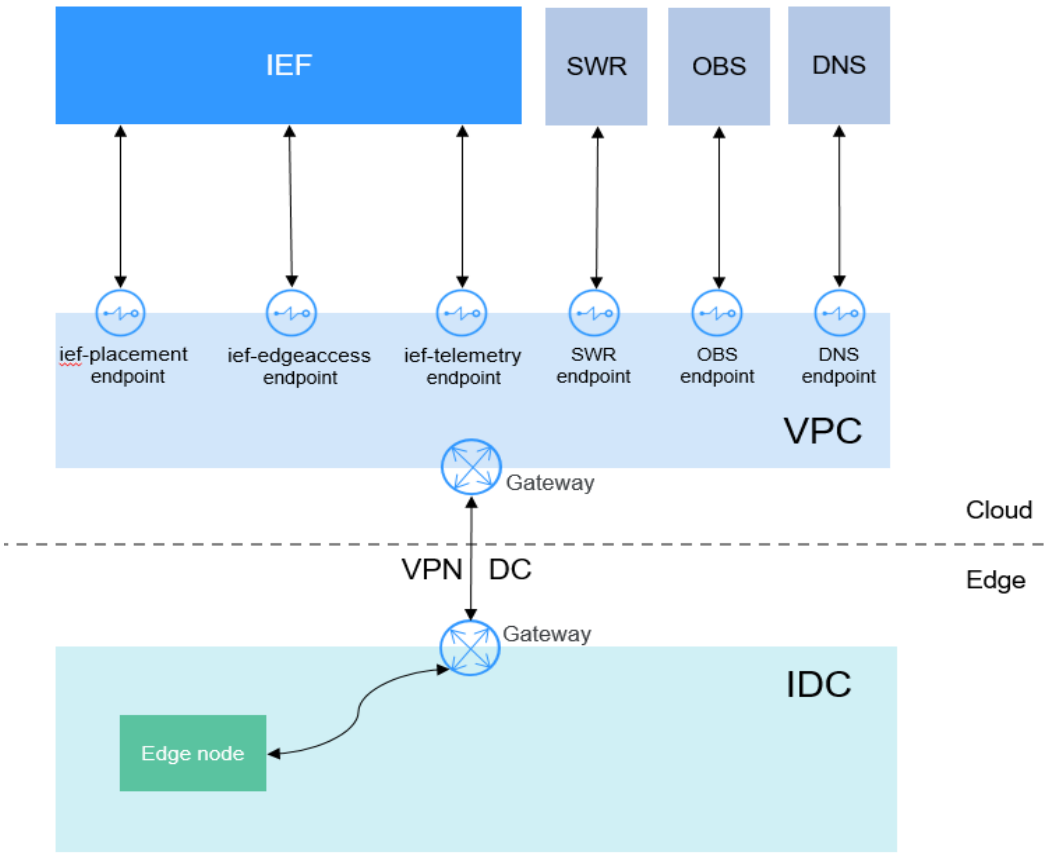
Before deploying applications on an edge node, ensure that the edge node can communicate with IEF, SWR and OBS. If the edge node cannot connect to IEF through a public network, connect the edge node to Huawei Cloud VPC through VPN or Direct Connect, and then enable VPC to access IEF, SWR and OBS through VPC endpoints. **Figure 1-1** shows the connection solution.

To connect edge nodes to IEF, you need to create the following three endpoints.

- **ief-placement**: for managing and upgrading edge nodes
- **ief-edgeaccess**: for sending messages between edge nodes and cloud services
- **ief-telemetry**: for uploading monitoring data and logs on edge nodes

To connect edge nodes to SWR, you need to create one endpoint. To connect edge nodes to OBS, you need to create an OBS endpoint and a DNS endpoint. (OBS can only be accessed through the domain name when the OBS address is dynamically resolved by DNS.)

Figure 1-1 Connecting edge nodes to IEF through Direct Connect or VPN



Procedure

Step 1 Create a VPC.

For details, see **Creating a VPC**.

You can also use an existing VPC.

NOTICE

The VPC CIDR block cannot overlap the IDC CIDR block.

Step 2 Connect the edge node to a VPC through Direct Connect or VPN.

For details, visit the following links:

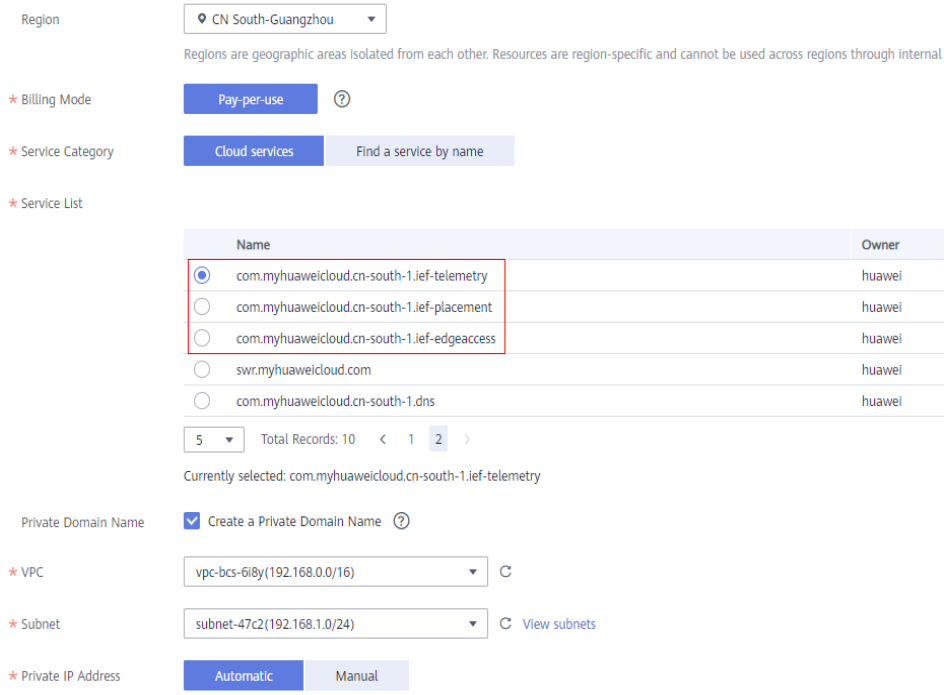
- VPN: **https://support.huaweicloud.com/intl/en-us/qs-vpn/vpn_qs_00008.html**
- Direct Connect: **https://support.huaweicloud.com/intl/en-us/qs-dc/en-us_topic_0145790541.html**

Step 3 Create IEF endpoints to enable the connection between the edge node and IEF.

You need to create three endpoints: **ief-placement**, **ief-edgeaccess**, and **ief-telemetry**. The procedure is as follows:

1. Log in to the **VPC Endpoint console** and click **Buy VPC Endpoint** in the upper right corner.
2. Select the IEF endpoints and VPC.

Figure 1-2 Creating IEF endpoints

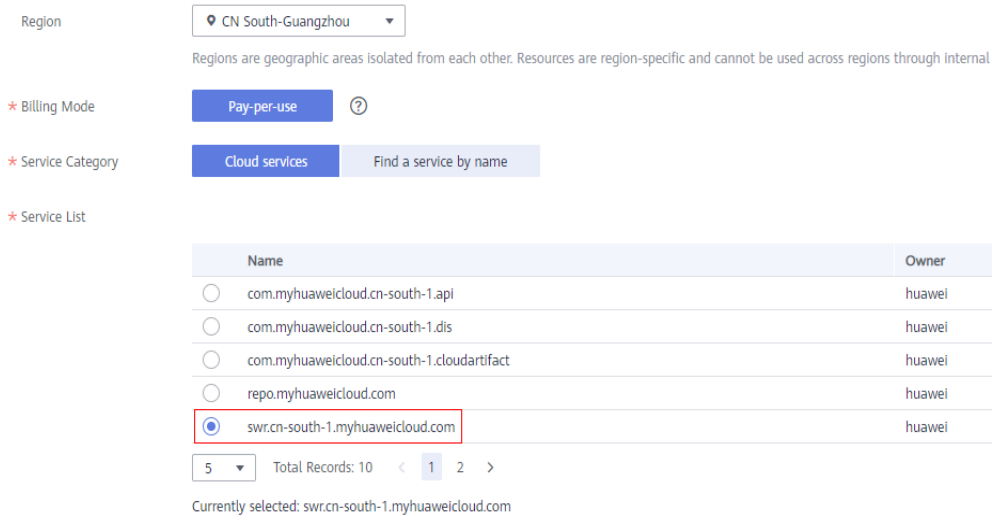


3. Click **Next**, confirm the information, and click **Submit**.

Step 4 Create an SWR endpoint so that the edge node can pull container images from SWR.

The procedure is the same as that of **creating IEF endpoints**.

Figure 1-3 Creating an SWR endpoint



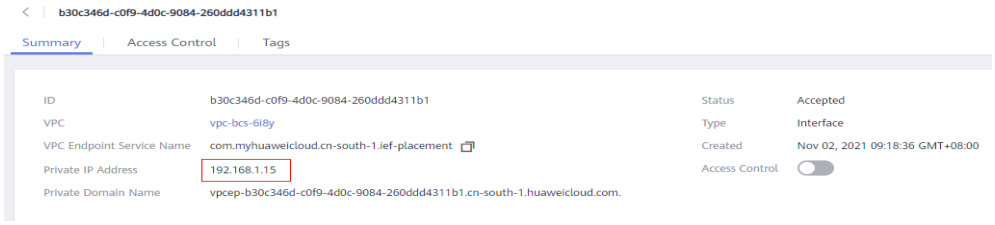
Step 5 Create DNS and OBS endpoints for edge nodes to access OBS.

For details, see **Accessing OBS**.

Step 6 Add the hosts configuration for the edge node.

Query the four IP addresses of IEF and SWR endpoints and add them in the **/etc/hosts** file of the edge node.

Figure 1-4 Querying IP addresses of the endpoints



Open the **/etc/hosts** file and add the IP addresses at the end of the file so that the domain names for accessing IEF and SWR point to the IP addresses of the corresponding endpoints.

NOTICE

Change the IP addresses and domain names based on the site requirements. The IP addresses are the ones obtained in the preceding step, and the domain names vary depending on the region. For details, see **Domain Name**.

```
192.168.2.20    ief2-placement.cn-north-1.myhuaweicloud.com
192.168.2.142   ief2-edgeaccess.cn-north-1.myhuaweicloud.com
192.168.2.106   ief2-telemetry.cn-north-1.myhuaweicloud.com
192.168.2.118   swr.cn-north-1.myhuaweicloud.com
```

Step 7 Register the edge node and use IEF to manage the edge node. For details, see **Edge Node Overview**.

----End

Domain Name

NOTE

ief-edgeaccess of the platinum edition has an independent address, which is the value of **Access Domain** on the **Dashboard** page of the IEF console.

Region	Name	Domain Name
CN North-Beijing1	ief-placement	ief2-placement.cn-north-1.myhuaweicloud.com
	ief-edgeaccess	ief2-edgeaccess.cn-north-1.myhuaweicloud.com
	ief-telemetry	ief2-telemetry.cn-north-1.myhuaweicloud.com
	swr	swr.cn-north-1.myhuaweicloud.com

Region	Name	Domain Name
CN North-Beijing4	ief-placement	ief2-placement.cn-north-4.myhuaweicloud.com
	ief-edgeaccess	ief2-edgeaccess.cn-north-4.myhuaweicloud.com
	ief-telemetry	ief2-telemetry.cn-north-4.myhuaweicloud.com
	swr	swr.cn-north-4.myhuaweicloud.com
CN South-Guangzhou	ief-placement	ief-placement.cn-south-1.myhuaweicloud.com
	ief-edgeaccess	ief-edgeaccess.cn-south-1.myhuaweicloud.com
	ief-telemetry	ief-telemetry.cn-south-1.myhuaweicloud.com
	swr	swr.cn-south-1.myhuaweicloud.com
CN East-Shanghai1	ief-placement	ief-placement.cn-east-3.myhuaweicloud.com
	ief-edgeaccess	ief-edgeaccess.cn-east-3.myhuaweicloud.com
	ief-telemetry	ief-telemetry.cn-east-3.myhuaweicloud.com
	swr	swr.cn-east-3.myhuaweicloud.com
CN East-Shanghai2	ief-placement	ief2-placement.cn-east-2.myhuaweicloud.com
	ief-edgeaccess	ief2-edgeaccess.cn-east-2.myhuaweicloud.com
	ief-telemetry	ief2-telemetry.cn-east-2.myhuaweicloud.com
	swr	swr.cn-east-2.myhuaweicloud.com

2 Connecting to the MQTT Broker Using the Open-source C Language Library

Scenario

MQTT is a message protocol in publish/subscribe mode. It works even on remote devices with poor hardware performance and in poor network conditions.

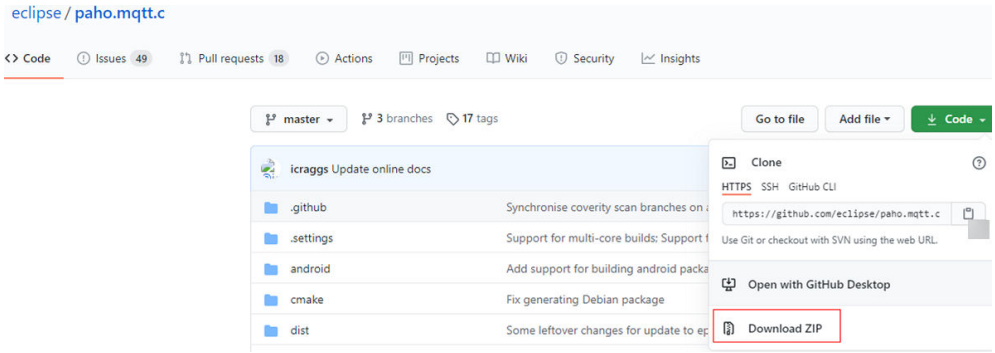
This article describes how to connect and use IEF's built-in MQTT broker using an open-source C language library, Eclipse Paho C Client Library.

Procedure

Step 1 Prepare a Linux computer and download the source code.

git clone https://github.com/eclipse/paho.mqtt.c.git

You can also download the ZIP package from **https://github.com/eclipse/paho.mqtt.c** and decompress it.



Step 2 Install the compilation dependency tool.

For Ubuntu, run the following commands:

apt-get install build-essential gcc make cmake cmake-gui cmake-curses-gui

apt-get install fakeroot fakeroot devscripts dh-make lsb-release

apt-get install libssl-dev

apt-get install ninja-build

For CentOS, run the following commands:

yum install build-essential gcc make cmake cmake-gui cmake-curses-gui

yum install fakeroot fakeroot devscripts dh-make lsb-release

yum install openssl-devel

yum install ninja-build

Step 3 Modify the sample code.

Add the following line to the **src/samples/ paho_cs_pub.c** file of the source code.

ssl_opts.enableServerCertAuth =0;

```
94                  ssl_opts.CApath = opts.capath;
95                  ssl_opts.keyStore = opts.cert;
96                  ssl_opts.trustStore = opts.cafile;
97                  ssl_opts.privateKey = opts.key;
98                  ssl_opts.privateKeyPassword = opts.keypass;
99                  ssl_opts.enabledCipherSuites = opts.ciphers;
100                 ssl_opts.enableServerCertAuth = 0;
101                 conn_opts.ssl = &ssl_opts;
```

Add the following line to the **src/samples/paho_cs_sub.c** file.

```
91                  ssl_opts.CApath = opts.capath;
92                  ssl_opts.keyStore = opts.cert;
93                  ssl_opts.trustStore = opts.cafile;
94                  ssl_opts.privateKey = opts.key;
95                  ssl_opts.privateKeyPassword = opts.keypass;
96                  ssl_opts.enabledCipherSuites = opts.ciphers;
97                  ssl_opts.enableServerCertAuth = 0;
98                  conn_opts.ssl = &ssl_opts;
```

Step 4 Compile the sample program.

mkdir /tmp/build.paho

cd /tmp/build.paho

cmake -GNinja -DPAHO_BUILD_STATIC=TRUE -DPAHO_BUILD_SHARED=FALSE -DPAHO_WITH_SSL=TRUE -DPAHO_BUILD_SAMPLES=TRUE *{paho.mqtt.c directory}*

ninja package

{paho.mqtt.c directory} is the directory where the source code **paho.mqtt.c** is stored, for example, **/root/work/paho.mqtt.c**.

Step 5 Go to the directory where the compiled sample program is located, and copy the generated binary files **paho_cs_pub** and **paho_cs_sub** to the edge node.

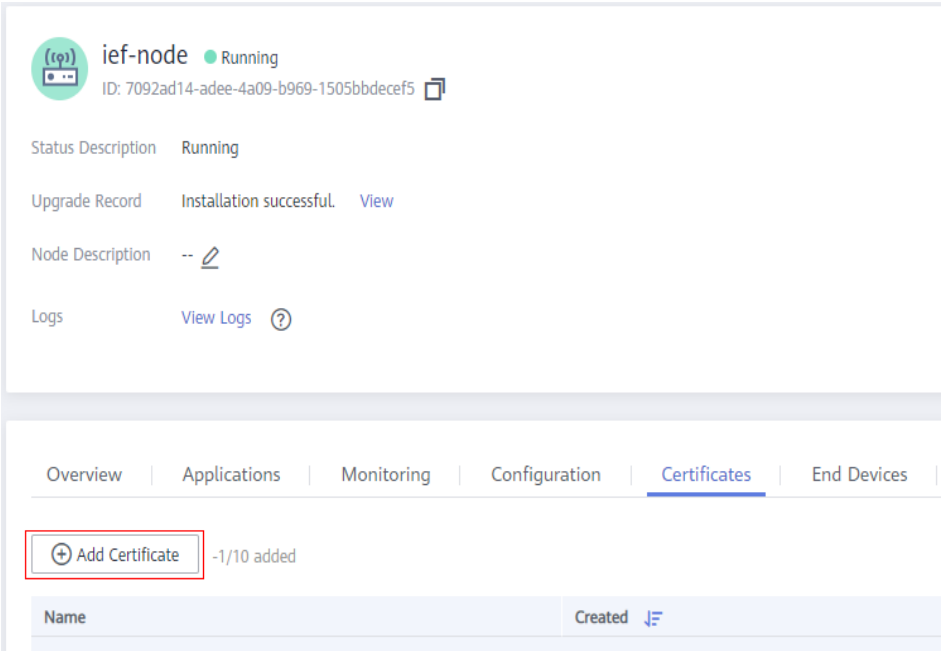
cd /tmp/build.paho/src/samples/

```
(base) root@cci-clustermanager-xsw:/tmp/build.paho/src/samples# ls
CMakeFiles            MQTTAsync_publish       MQTTAsync_subscribe  MQTTClient_publish_async  paho_c_pub  paho_cs_sub
cmake_install.cmake   MQTTAsync_publish_time  MQTTClient_publish   MQTTClient_subscribe      paho_cs_pub  paho_c_sub
```


Step 6 Download the edge node certificate.

1. Log in to the IEF console, choose **Managed Resources** > **Edge Nodes** in the navigation pane, and click an edge node name on the displayed page to view the edge node details page. Click the **Certificates** tab, and click **Add Certificate**.

   Figure 2-1 Adding a certificate

2. In the displayed dialog box, enter the certificate name and click **OK**.
3. Copy the downloaded certificate to the edge node and decompress it.

Step 7 Run the sample program.

For example, to publish a message to a specified topic, run the following commands to view the **paho_cs_pub** command instructions.

```
(base) root@cci-clustermanager-xsw:/tmp/build.paho/src/samples# ./paho_c_pub
Eclipse Paho MQTT C publisher

Library information:
Product name: Eclipse Paho Asynchronous MQTT C Client Library
Version: 1.3.8
Build level: 2021-01-20T14:05:21Z
OpenSSL version: OpenSSL 1.1.0l  10 Sep 2019
OpenSSL flags: compiler: gcc -DDSO_DLFCN -DHAVE_DLFCN_H -DNDEBUG -DOPENSSL_THREADS -DOPENSSL_NO_STATIC_ENGI
BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DSHA256_ASM -DSHA512_ASM -DRC4_ASM -DM
H_ASM -DECP_NISTZ256_ASM -DPADLOCK_ASM -DPOLY1305_ASM -DOPENSSLDIR="\"/usr/local/ssl\"" -DENGINESDIR="\"/us
OpenSSL build timestamp: built on: reproducible build, date unspecified
OpenSSL platform: platform: linux-x86_64
OpenSSL directory: OPENSSLDIR: "/usr/local/ssl"

Usage: paho_c_pub [topicname] [-t topic] [-c connection] [-h host] [-p port]
       [-q qos] [-i clientid] [-u username] [-P password] [-k keepalive_timeout]
       [-V MQTT-version] [--quiet] [--trace trace-level]
       [-r] [-n] [-m message] [-f filename]
       [--maxdatalen len] [--message-expiry seconds] [--user-property name value]
       [--will-topic topic] [--will-payload message] [--will-qos qos] [--will-retain]
       [--cafile filename] [--capath dirname] [--cert filename] [--key filename]
       [--keypass string] [--ciphers string] [--insecure]

  -t (--topic)        : MQTT topic to publish to
  -c (--connection)   : connection string, overrides host/port e.g wss://hostname:port/ws.  Use this option
                        rather than host/port to connect with TLS and/or web sockets. No default.
  -h (--host)         : host to connect to.  Default is localhost.
  -p (--port)         : network port to connect to. Default is 1883.
  -q (--qos)          : MQTT QoS to publish with (0, 1 or 2). Default is 0.
  -V (--MQTTversion)  : MQTT version (31, 311, or 5).  Default is 311.
  --quiet             : do not print error messages.
  --trace             : print internal trace ("error", "min", "max" or "protocol").
```

The following is an example of publishing a message:

./paho_cs_pub -c ssl://127.0.0.1:8883 -q 0 **-m** "xxx" **-t** "aaa" **--cert** /root/mqtt_cert/xOEMIsYVpw_private_cert.crt **--key** /root/mqtt_cert/xOEMIsYVpw_private_cert.key

Running this command will send a message whose content is **xxx** to the **aaa** topic in the built-in MQTT broker. **127.0.0.1:8883** is the address of the built-in MQTT broker of the edge node. **/root/mqtt_cert/xOEMIsYVpw_private_cert.crt** and **/root/mqtt_cert/xOEMIsYVpw_private_cert.key** are the edge node certificates.

----End

