BCS

Best Practices

Issue 01

Date 2023-03-01

HUAWEI

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 Building a Bank Consortium Blockchain Using Enhanced Hyperledger Fabric

1.1 Overview

Background

With the continuous advancements of IT, interbank business processes become standardized. However, some operations are redundant. For example, it takes a long time and tedious procedure to query users' basic information such as account and identity across banks, resulting in poor user experience.

Solution

Blockchain Service (BCS) provides consortium blockchains based on enhanced Hyperledger Fabric. After creating a consortium blockchain, you can invite other banks to join it. Consortium members can share data with each other in a trusted manner, which saves time for users and improves the efficiency for banks. Peers of each consortium member run in a separate VPC for independent and secure management.

This best practice uses a bank consortium blockchain as an example to explain the basics of consortium blockchains and how to form a consortium blockchain. It helps you quickly get started with Huawei Cloud BCS.

NOTE

This is a demo only and is not for actual use.

Scenario

- Business scenario and members:
  - **democraticBank** invites **civilizationBank** and **harmoniousBank** (invitees A and B) to join the bank consortium.
  - A customer registers accounts with **civilizationBank** and **harmoniousBank** without Know Your Customer (KYC) checks based on an existing account and KYC check results of **democraticBank**.

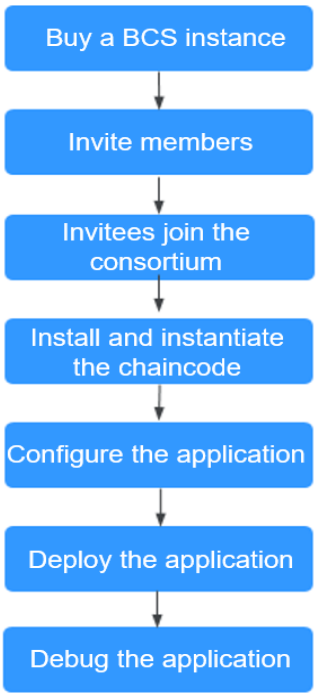
- Requirements

  Data such as user identities must be encrypted to protect the information against brute force attack. Quick information queries based on identities must be supported.
- Key challenges:
  - Security and privacy: Banks require no disclosure of customer privacy information to other banks.
  - High-performance retrieval: Retrieval performance (millisecond-level response) similar to that of legacy databases must be delivered to ensure user experience.
  - Easy-to-use interfaces: Simple and convenient interfaces must be provided to facilitate application development.

Process Flow

The following describes the general process of setting up a consortium blockchain.

Figure 1-1 Process flow



1. **Buy a BCS instance**.

   Buy a consortium blockchain and name the organization member **democraticBank**.
2. **Invite tenants to join the consortium**.

   After the consortium blockchain is created, organization member **democraticBank** invites **civilizationBank** and **harmoniousBank** to join the consortium blockchain.
3. **Invitees join the consortium**.

   civilizationBank and **harmoniousBank** join the consortium blockchain based on the invitation information.

4. **Install and instantiate the chaincode**.

   The chaincode must be installed by the inviting party and all invitees (in this demo, invitees A and B), and can be instantiated by any member in the consortium.

5. **Configure the application**.

   Before initiating transactions, the inviting party must download the administrator certificates of the orderer and organization **democraticBank**, and the certificates of invitees A and B with the private keys hidden.

6. **Deploy the application**.

   Install the bank consortium blockchain demo application on an Elastic Cloud Server (ECS).

7. **Debug the application**.

   After the application is deployed, you can test data transactions on the consortium blockchain.

1.2 Buying a BCS Instance

Buy a BCS instance and configure the basic parameters and network nodes.

Procedure

Step 1 Log in to the BCS console.

Step 2 On the **Instance Management** page, click **Buy** next to **Enhanced Hyperledger Fabric Instance**.

Step 3 Configure the basic settings as prompted. **Table 1-1** describes the parameters for the basic settings.

NOTICE

To ensure that the demo runs properly, set the parameters as described in the following table.

Table 1-1 Basic settings

Parameter	Description
Region	Retain the default value.
Enterprise Project	Select an existing enterprise project, for example, **default**. If you have not enabled the enterprise management service, this parameter is not displayed.
Instance Name	Enter **bank-union-demo**.
Edition	Select **Enterprise**.

Parameter	Description
Blockchain Type	Select **Consortium**.
Enhanced Hyperledger Fabric Version	Select **v2.2**.
Consensus Mechanism	Select **Raft (CFT)**.
Resource Access Initial Password	Enter a password.
Confirm Password	-

Step 4 Click **Next: Configure Resources**. Table 1-2 describes the resource parameters.

Table 1-2 Resource configurations

Parameter	Description
Environment Resources	Select **Custom**.
Cluster	Select **Create a new CCE cluster**.
AZ	Select **AZ1**.
ECS Specifications	Select the flavor for **4 vCPUs | 8 GB**.
ECS Quantity	Enter **1**.
High Availability	Select **No**.
VPC	Select **Automatically create VPC**.
Subnet	Select **Automatically create subnet**.
ECS Login Method	Select **Password**.
Password of Root User	If you do not enter a password here, the previously specified resource access initial password will be used.
Confirm password.	-
Use EIP of a CCE Node	Select **Yes**.
EIP Billed By	Select **Bandwidth**.
EIP Bandwidth	Set it to 5 Mbit/s.

Step 5 Click **Next: Configure Blockchain**. Table 1-3 describes the blockchain parameters.

Table 1-3 Blockchain configurations

Parameter	Description
Blockchain Configuration	Select **Custom**.

Parameter	Description
Blockchain Mgmt. Initial Password	If you do not enter a password here, the previously specified resource access initial password will be used.
Confirm Password	-
Volume Type	Select **SFS Turbo**.
Storage Capacity of Peer Organization (GB)	Retain the default value.
Ledger Storage	Select **File database (GoLevelDB)**.
Peer Organization	Create one peer organization **with the exact name and peer quantity as the following**: Name: **democraticBank**; quantity: **2**
Channel Configuration	Change the channel name to **testchannel** and add the created **democraticBank** organization to the channel. **NOTE** The channel must be named **testchannel**.
Orderer Quantity	Retain the default value.
Security Mechanism	Select **ECDSA**.
Configure Block Generation	Select **No**.
Enable Support for RESTful API	Select **No**.

Step 6 Click **Next: Confirm**.

Step 7 Confirm the configurations, confirm that you have read and agree to the agreement, and click **Pay Now**.

Wait for several minutes. After a message is displayed indicating successful installation, check the status of the instance. If it is **Normal**, the deployment is completed.

NOTE

If the creation fails, locate the failure cause by referring to the **purchase and deployment FAQs**.

----End

1.3 Inviting Tenants to Join a Consortium

After creating the consortium blockchain, you can invite tenants to join a channel to establish a consortium.

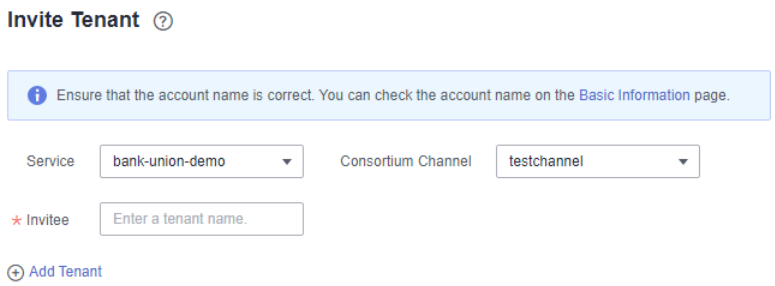
Procedure

Step 1 Log in to the BCS console.

Step 2 Choose **Member Management** in the navigation pane on the left. Click **Invite Tenant** in the upper right corner of the page.

Step 3 Select the BCS instance and channel you just created, and enter the invitee's name.

Figure 1-2 Inviting a tenant



Step 4 (Optional) Click **Add Tenant** to invite multiple tenants.

In this demo, **civilizationBank** ("invitee A") and **harmoniousBank** ("invitee B") are invited.

NOTE

A maximum of 40 tenants can be invited.

Step 5 Click **OK**. An invitation notification is sent to the invitees.

----End

1.4 Joining a Consortium

When you are invited to join a consortium blockchain, you will receive a notification. You can accept the invitation to join the consortium.

Procedure

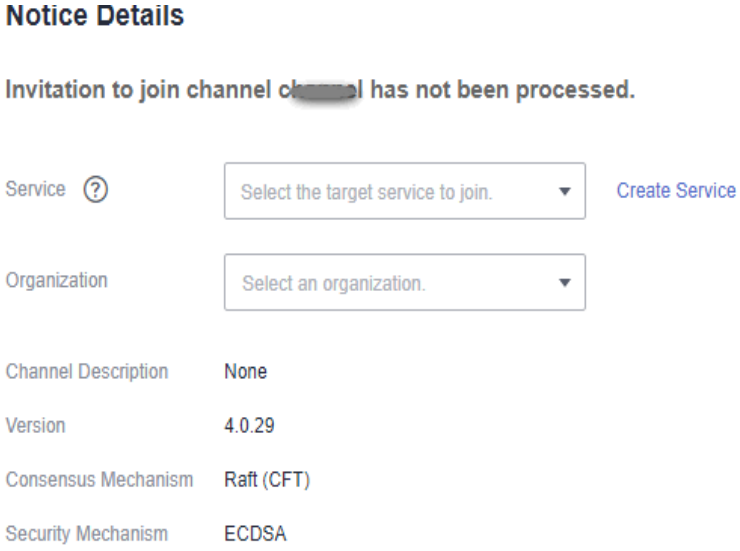
Step 1 Log in to the BCS console.

Step 2 Choose **Notification Management** in the navigation pane on the left. Locate the notification and click **View Details** in the **Operation** column.

Step 3 Create a BCS instance.

1. Click **Create Instance** and use the new BCS instance to join the channel.

Figure 1-3 Creating instances



2. Specify the BCS instance parameters as prompted.

Table 1-4 Parameters

Parameter	Setting
Billing Mode	Select **pay-per-use**.
Region	Select the same region as the inviting party.
Instance Name	Enter the same name as the BCS instance of the inviting party, for example, **bank-union-demo**.
Edition	Select **Enterprise**.
Blockchain Type	Select **Consortium**.
Cluster Type	Select **CCE cluster**.
Container Cluster	Select an existing container cluster.
Volume Type	Select **SFS**.
Network Storage	Select an existing SFS file system.
Ledger Storage	Retain the default value (**GoLevelDB**), which is used by the inviting party.
Peer Organization	Create one peer organization named **civilizationBank**.
Consensus Mechanism	Retain the default value, which is used by the inviting party.
Enable Data Aging on Orderers	Select **No**.
Cross-AZ Scheduling	Select **No**.

Parameter	Setting
Security Mechanism	Retain the default value.
Version	Retain the default value, which is used by the inviting party.
Blockchain Mgmt. Initial Password	Enter a password.
Use EIP of a CCE Node	Select **Yes**.

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

   Wait for several minutes. After a message is displayed indicating successful creation, check the status of the instance and organizations. If they are **Normal**, the deployment is completed.

Step 4 After creating the BCS instance, confirm the organization in the **Notice Details** window and click **Accept** to join the consortium blockchain.

Step 5 Other invitees can repeat **Step 1** to **Step 4** to join the consortium.

The operations and parameter settings for other invitees are the same as described in the previous steps, except the peer organization names. In this demo, invitee B should set the name of the peer organization to **harmoniousBank**.

----End

1.5 Installing and Instantiating a Chaincode

A chaincode must be installed on all peers in a channel and instantiated on any one peer. To use the same chaincode, channel members must specify the same name and version for the chaincode during chaincode installation.

NOTICE

- The chaincode must be installed by the inviting party and all invitees (in this demo, invitees A and B).
- The chaincode name and version specified by all parties must be consistent.
- The chaincode can be instantiated by any member in the consortium.

Installing a Chaincode

Step 1 Log in to the BCS console.

Step 2 In the navigation pane on the left, click **Instance Management**.

Step 3 On a target instance card, click **Manage Blockchain**.

Step 4 On the login page, enter the username and password, and click **Log In**.

NOTE

- The username is **admin**, and the password is the **Blockchain Mgmt. Initial Password** set when you created the BCS instance. If you have not set this password, use the resource access initial password. For security purposes, change the password periodically.
- If you use the Internet Explorer, you may fail to open the **Blockchain Management** login page and see a message indicating that the certificate is untrusted. In this case, you can click **here** to resolve the problem.

Step 5 Click **Install Chaincode** in the upper left corner of the page.

Step 6 Enter the chaincode name and version number, select the peers where the chaincode is to be installed, specify the programming language of the chaincode, and add the chaincode file, as described in **Table 1-5**.

Table 1-5 Parameters

Parameter	Setting
Chaincode Name	Enter **fabbank**.
Chaincode Version	Enter **1.0**.
Ledger Storage	File database (GoLevelDB)
Select All Peers	Select the checkbox.
Organization & Peer	All peers have been selected automatically.
Language	Select **Golang**.
Chaincode File	Log in to the BCS console and choose **Use Cases**. Download **Bank_Demo** in the **SDK Demo for a Bank Consortium Blockchain** area.
Chaincode Description	Enter a description of the chaincode.
Code Security Check	This option is displayed only when the chaincode language is Golang. Enable this option to check chaincode security.

Step 7 Click **Install** to install the chaincode.

----End

Instantiating a Chaincode

Step 1 After installing the chaincode, click **Instantiate** in the **Operation** column of the chaincode list.

Step 2 Specify the channel for instantiation, chaincode version, endorsement policy, endorsing organizations, and chaincode parameters, as described in **Table 1-6**.

Table 1-6 Parameters

Parameter	Setting
Chaincode Name	Enter **fabbank**.
Channel	Select **testchannel**.
Chaincode Version	Select **1.0**.
Initialization Function	Enter **init**.
Chaincode Parameters	-
Endorsement Policy	Select **Endorsement from any of the following organizations**.
Endorsing Organizations	Select all the three organizations.
Privacy Protection	Select **No**.

Step 3 Click **Instantiate**.

Wait for 2 to 3 minutes and refresh the page. Click **View more** to check the instantiation status.

----End

1.6 Configuring the Application

Before initiating transactions, the inviting party must download the administrator certificates of the orderer and organization **democraticBank**, and the certificates in which the private keys are hidden by invitees A and B.

Encrypt the private keys in the downloaded certificates for storage.

Prerequisites

An ECS has been created. For more information on how to create ECSs, see **Elastic Cloud Server User Guide**.

Downloading SDK Configurations and Certificates

Step 1 On the **Instance Management** page, click **Download Client Configuration** on the card containing the **bank-union-demo** instance of the inviting party.

Step 2 Select **SDK Configuration File** and set the parameters as described in **Table 1-7**.

NOTICE

To ensure that the demo runs properly, set the parameters as described in the following table.

Table 1-7 SDK file parameters

Parameter	Setting
Chaincode Name	Enter **fabbank**.
Certificate Root Path	Enter **/opt/bank/src/bank/conf/crypto**. **NOTE** **/opt/bank/src/bank/conf/crypto** is the internal path of the container and is mounted to the **/root/bankuniondemo** path of the ECS.
Channel	Retain the automatically selected **testchannel**.
Member	Retain the default settings. Do not select all organizations and peers.

Step 3 Select **Orderer Certificate** and **Peer Certificates**. Under **Peer Certificates**, retain the default selection for **Peer Organization** and select **Administrator certificate**. Click **Download**.

Step 4 Decompress the downloaded file. Duplicate the **sdk-config.yaml** file with another two copies, and name the three files **democraticBank.yaml**, **civilizationBank.yaml**, and **harmoniousBank.yaml**. Then, save the files to the **/root/bankuniondemo** directory on the ECS.

NOTE

- Create the **bankuniondemo** directory in the **/root** directory of the ECS.
- According to **Installing and Instantiating a Chaincode**, the chaincode installed by all parties (inviting party **democraticBank**, invitee A **civilizationBank**, and invitee B **harmoniousBank**) must be the same one. Therefore, you only need to download the inviting party's SDK configuration file **sdk-config.yaml** to obtain the chaincode and certificate path information. Then, duplicate the file with two copies, and rename the three files.

Step 5 On the card containing the instance of invitee A, click **Download Client Configuration**, and select **Peer Certificates** and **Administrator certificate** to download the administrator certificate of the **civilizationBank** organization.

Step 6 Invitee A deletes the private key files (**server.key** file in the **tls** directory and files in the **keystore** folder in the **msp** directory) from the administrator certificate of organization **civilizationBank**.

Figure 1-4 Deleting the server.key file from the tls directory

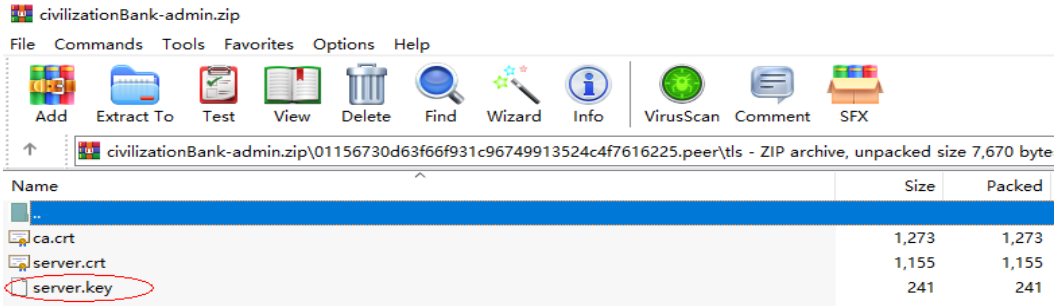
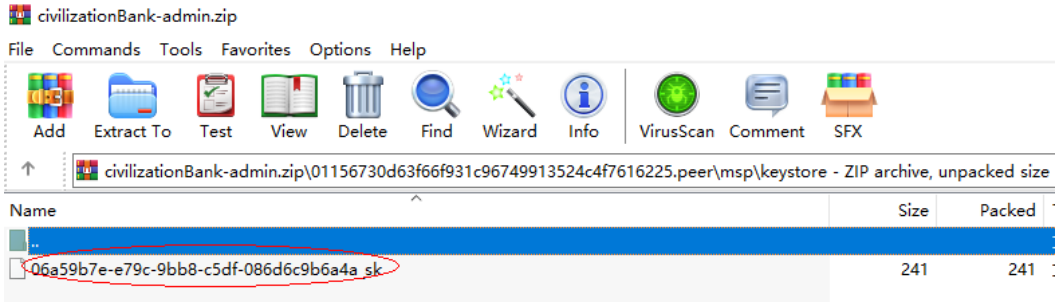


Figure 1-5 Deleting the keystore file from the msp directory



Step 7 Invitee B repeats **Step 5** to **Step 6** to download the administrator certificate of organization **harmoniousBank** and delete the private key files.

Step 8 The inviting party stores the downloaded **bank-union-demo-orderer** administrator certificate, the administrator certificate of organization **democraticBank**, and certificates of organizations **civilizationBank** and **harmoniousBank** (in which the private key files have been deleted by the invitees) to the **/root/bankuniondemo** directory of the prepared ECS.

----End

1.7 Deploying the Application

This section describes how to deploy the application.

Deploying the Application

Step 1 To obtain the image, go to the BCS console and choose **Use Cases**. Download **Bank_Image** in the **SDK Demo for a Bank Consortium Blockchain** area.

Step 2 Ensure that the **docker** and **unzip** commands exist before logging in to ECS.

Run the **cd /root/bankuniondemo** command to upload the downloaded image to the directory.

Run the **vi startserver.sh** command to create the **startserver.sh** file. Add the following content to the **startserver.sh** file and save the file:

```
#!/bin/sh
# Obtain the current path.
path=`pwd`
# Obtain the IP address.
localip=`ifconfig eth0 | grep "inet addr" | awk '{ print $2}' | awk -F: '{print $2}'`

# Check whether unzip and docker commands exist.
if type unzip > /dev/null 2>&1; then
    echo "Unzip command is exists!"
else
    echo "No unzip command, please install unzip before start this demo"
    exit 1
fi

if type docker > /dev/null 2>&1; then
    echo "Docker exists!"
else
    echo "No docker, please install docker before start this demo"
    exit 1
fi
```



```
# Check whether the certificate directory exists.
if [ -e ./crypto ];then
rm -rf crypto
fi

# Decompress the certificate file.
echo "Start unziping certificate ...."

cerzip=$(ls|grep .zip)
for i in $cerzip;
do
unzip -n $i -d crypto/
done
echo "Unzip certificate Done!!"

# Configure the certificate path.
Folder="$path/crypto"
for file in ${Folder}/*
do
    temp_file=`basename $file`

    mv -f $path/crypto/$temp_file/* $path/crypto/
done

# Check whether the required certificate files are ready.
echo "Start checking if the required files are ready"
if [ -e $path/democraticBank.yaml ]&&[ -e $path/civilizationBank.yaml ]&&[ -e $path/
harmoniousBank.yaml ];then
echo "Check required yaml files OK!!"
else
echo "Please check if the required yaml files are ready!!"
exit
fi

# Restart the system if the password is forgotten.
read -p "Is this the first time you deploy the application? If yes please insert 'y', if you want to reset your
password please insert 'n': " choice
echo $choice
if [ $choice = "n" ];then
docker rm $(docker stop $(docker ps -a -q --filter ancestor=bank --format="{{.ID}}"))
echo "Resetting password..."
fi

# Set the password.
read -p "Please set your password: " test
echo $test
if [ ${#test} -lt 4 -o ${#test} -gt 24 ];then echo "password length should in [4, 24].";exit;fi

#load images
if [ -e $path/bankv3.tar.gz ];then
echo "Start loading images..."
docker load -i bankv3.tar.gz
echo "Api-server images is exists,Skip load!"
fi

# Start the service.
echo "Start server ....."
docker run --env TESTPWD=$test -p 8080:8080 -d -it -v $path/crypto:/opt/bank/src/bank/conf/crypto -v
$path/democraticBank.yaml:/opt/bank/src/bank/conf/democraticBank.yaml -v $path/
harmoniousBank.yaml:/opt/bank/src/bank/conf/harmoniousBank.yaml -v $path/civilizationBank.yaml:/opt/
bank/src/bank/conf/civilizationBank.yaml bank

RET_CODE=`echo $?`
if [ $RET_CODE -eq 0 ]; then
     echo "Start server success!"
     echo -e "please login \033[32mhttp://EIP:8080/adminLogin \033[0mto visit Bank Management System"
     echo -e "or login \033[32mhttp://EIP:8080/userLogin \033[0mto visit Bank Customer System"
else
```



```
    echo "Failed to start server! Please check if everything ok"
fi
```

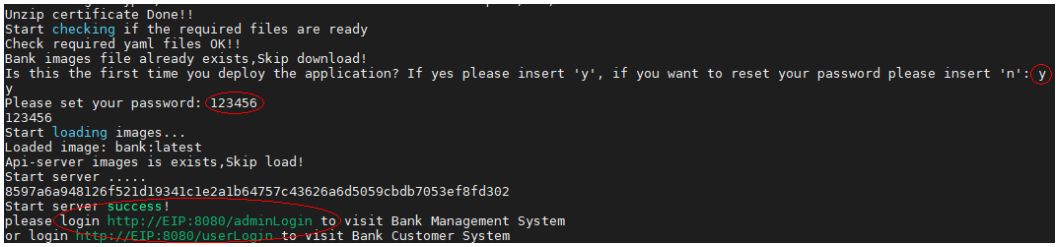
Run the following script:

bash startserver.sh

Step 3 After you see "Is this the first time you deploy the application?", enter **y** for yes, or enter **n** if you want to change the password. Then, insert your password (**123456** is used as an example in the following figure).

The command output is similar to the following:

Figure 1-6 Command output

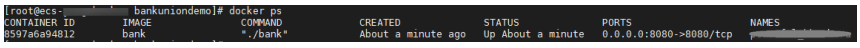


Step 4 Run the following command to check whether the container has started:

docker ps

If the following information is displayed, the container has started. Otherwise, check the configuration.

Figure 1-7 Message displayed



----End

Adding a Security Group

- If you deploy the bankuniondemo application on your own server, skip this procedure.
- If you deploy the bankuniondemo application on an ECS on Huawei Cloud, create a security group and then add security group rules to enable access to port 8080 of the server. For details, see **Creating a Security Group**. **Table 1-8** describes the security group rule.

Table 1-8 Parameters of a security group rule

Parameter	Setting	Description
Protocol	Select **TCP**.	Specifies the network protocol.
Direction	Set the inbound rules.	Specifies the direction in which the security group rule takes effect. Inbound rules control external access to the ECS.

Parameter	Setting	Description
Port number range	Create a rule, and set the port to **8080**.	Specifies the port range for which a rule takes effect.
Source	Select **IP Address**, and enter **0.0.0.0/0**.	This parameter is required for inbound rules.

- The EIP must be the IP address of the server where the bankuniondemo application is deployed or the EIP of the ECS.

1.8 Debugging the Application

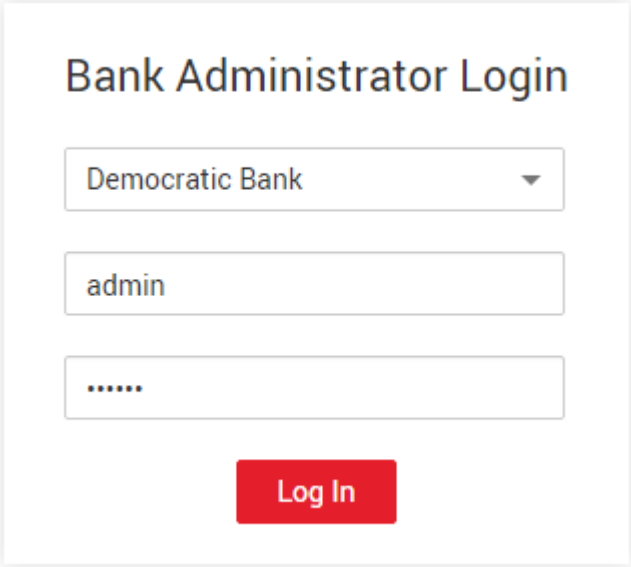
After deploying the application, the administrator can enter customer information and the customer can apply to open an account.

Entering Customer Information (by the Bank Administrator)

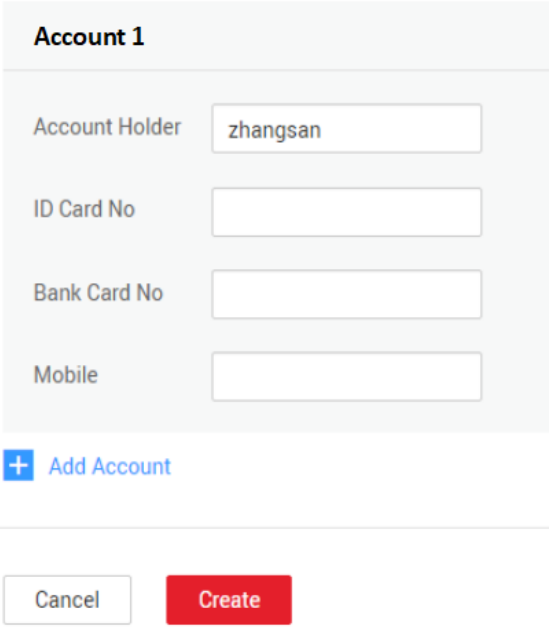
Step 1 After the demo application is successfully deployed, visit **http://*EIP*:8080/adminLogin** to access the administrator-facing page of the application.

NOTE

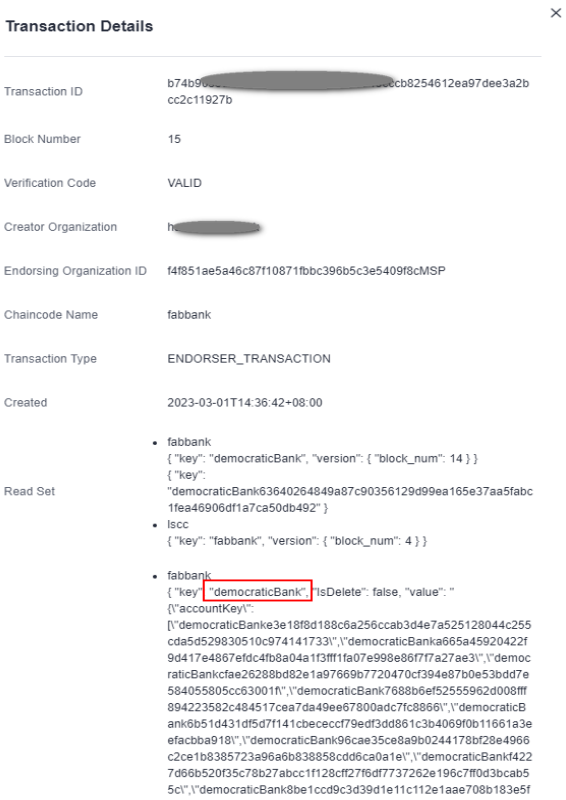
- The EIP must be the IP address of the server where the demo application is deployed or the EIP of the ECS.
- Default username: **admin**; password: the password inserted during **application deployment**. (This information is for the demo only.)



Step 2 Create an account and enter the customer information. For example, the information to be entered for the Democratic Bank includes the name, Resident Identity Card number, bank card number, and mobile number.



Step 3 After the account is created, the administrator can view the account creation record in **Block Browser**.



----End

Applying for an Account (by a Customer)

Step 1 Visit **http://*EIP*:8080/userLogin** to open the customer-facing page of the application.

NOTE

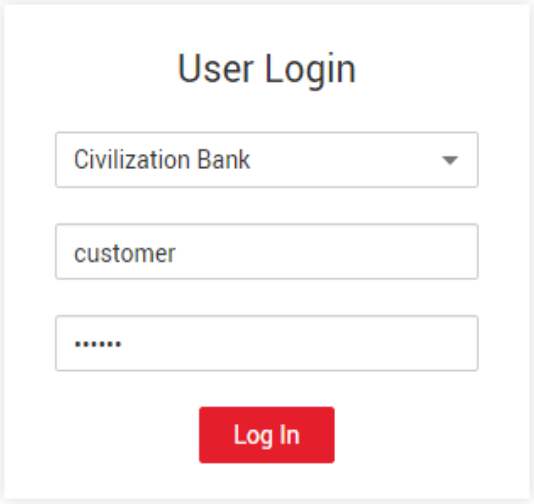
- The EIP must be the IP address of the server where the demo application is deployed or the EIP of the ECS.
- Default username: **customer**; password: the password inserted during application deployment. (This information is for the demo only.)

Step 2 Select Civilization Bank and log in.

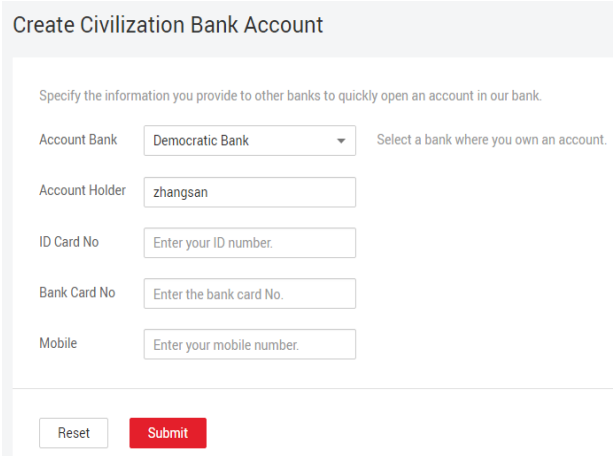
The account application page of the Civilization Bank is displayed. The customer can open an account with Civilization Bank based on the Democratic Bank account.

In this process, the customer's identity is shared across the blockchain. Once the customer opens a bank account, the reviewed trusted identity information is recorded to the blockchain and is encrypted and shared to other banks. Then, the customer can apply for accounts with other banks without repeating the approval process.

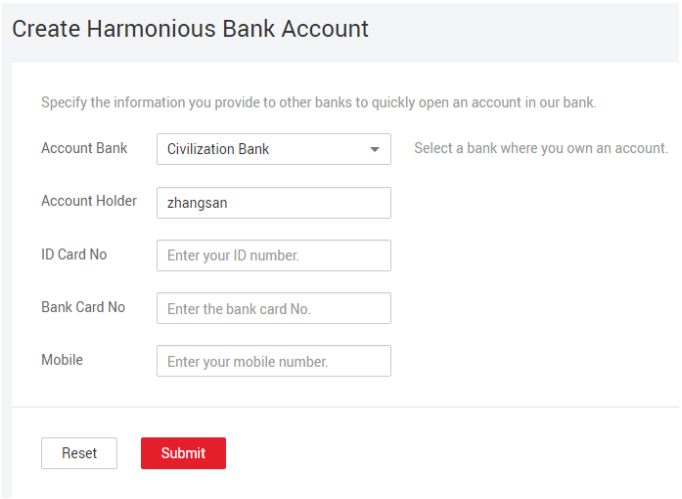
Logging in to the Civilization Bank system



Opening a Civilization Bank account based on the Democratic Bank account

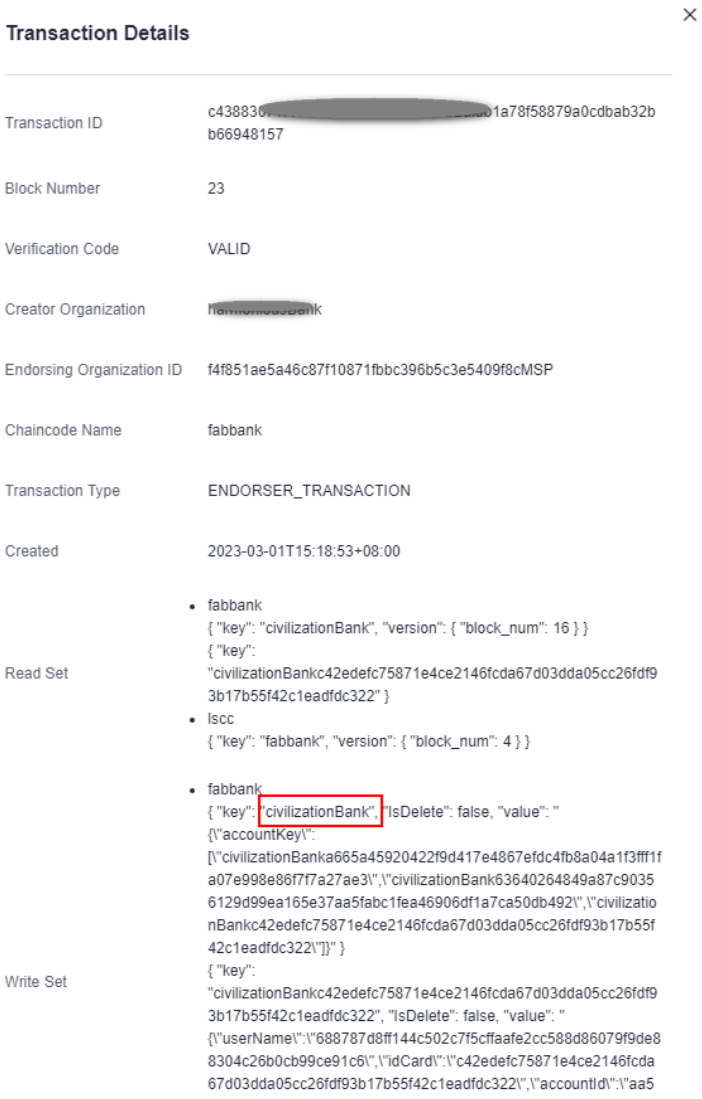


Opening a Harmonious Bank account based on the Civilization Bank account



If the account is opened successfully, the system displays a message indicating that the bank account has been opened successfully based on the account information of other banks.

Step 3 After the account is opened, the customer can view the account opening record in **Block Browser**.



Transaction Details

Transaction ID	c43883[redacted]1a78f58879a0cdbab32bb66948157
Block Number	23
Verification Code	VALID
Creator Organization	[redacted]k
Endorsing Organization ID	f4f851ae5a46c87f10871fbbc396b5c3e5409f8cMSP
Chaincode Name	fabbank
Transaction Type	ENDORSER_TRANSACTION
Created	2023-03-01T15:18:53+08:00
Read Set	• fabbank { "key": "civilizationBank", "version": { "block_num": 16 } } { "key": "civilizationBankc42edefc75871e4ce2146fcda67d03dda05cc26fdf93b17b55f42c1eadfdc322" } • lscc { "key": "fabbank", "version": { "block_num": 4 } }
Write Set	• fabbank { "key": "civilizationBank", "IsDelete": false, "value": "{\"accountKey\":[\"civilizationBanka665a45920422f9d417e4867efdc4fb8a04a1f3fff1fa07e998e86f7f7a27ae3\",\"civilizationBank63640264849a87c90356129d99ea165e37aa5fabc1fea46906df1a7ca50db492\",\"civilizationBankc42edefc75871e4ce2146fcda67d03dda05cc26fdf93b17b55f42c1eadfdc322\"]}" } { "key": "civilizationBankc42edefc75871e4ce2146fcda67d03dda05cc26fdf93b17b55f42c1eadfdc322", "IsDelete": false, "value": "{\"userName\":\"688787d8ff144c502c7f5cffaafe2cc588d86079f9de88304c26b0cb99ce91c6\",\"idCard\":\"c42edefc75871e4ce2146fcda67d03dda05cc26fdf93b17b55f42c1eadfdc322\",\"accountId\":\"aa5

----End

