

CEC
2.5.0.0.0

Agent Integration--Agent Connection Bar Integration (JS)

Issue 01
Date 2024-03-01



Copyright © Huawei Technologies Co., Ltd. 2024. 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



HUAWEI 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.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: <https://www.huawei.com>

Email: support@huawei.com

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 Overview	1
2 Integration Principles	4
3 Integration Procedure	5
4 Preparations	7
4.1 Resource Preparations	7
4.2 Enabling the Voice and Video Agent Function in the CEC	9
5 Integration Development	18
5.1 Authentication Before Sign-In	18
5.2 Obtaining CEC Account Information	21
5.3 Developing a Token Generation Mechanism and Verification API	23
5.4 Developing an Integration Page	25
5.4.1 Core Code Analysis	25
5.4.2 JavaScript Code Example	26
5.5 Making Outbound Calls in One-Click Mode	28
5.5.1 What Do I Do If Error Code 500 Is Displayed for a CORS Request and Third-Party Authentication Fails	29
6 (Optional) Developing Other Functions	31
6.1 Listening to Connection Events	31
6.2 Implementing the One-Click Outbound Call Function	33
7 Test and Verification	35

1 Overview

The CEC provides a lightweight connection bar that can be directly integrated. You can easily integrate the voice processing capability of agents into the common operation GUI of employees as the agent voice and video service channels in the original customer service system. This saves the construction and maintenance costs of the infrastructures such as the ACD and computer telecom integration platform.

The lightweight connection bar provides the functions of voice and video call handling and agent status control. It has the following features:

- It is lightweight, easy to be integrated into different platforms, and does not occupy the main pages of portals and workbenches.
- It is easy to operate. Agents can sign in to the platform, answer inbound calls, transfer calls, mute calls, switch the status, or seek help in one-click mode.

You can integrate the core functions of the CEC into a third-party system by referring to this document. The agent sign-in and sign-out, and connection control (that is, lightweight connection bar) pages are integrated in iFrame mode, as shown in [Figure 1-1](#) and [Figure 1-2](#).

NOTE

Lightweight integration does not support sign-in of the same agent in multiple places, browsers, or environments. If the same agent signs in in multiple places, browsers, or environments, events may be lost or abnormal responses may be returned. If the page is abnormal in this case, exit or disable sign-in in multiple places, browsers, or environments, and refresh the page.

Figure 1-1 Voice call page

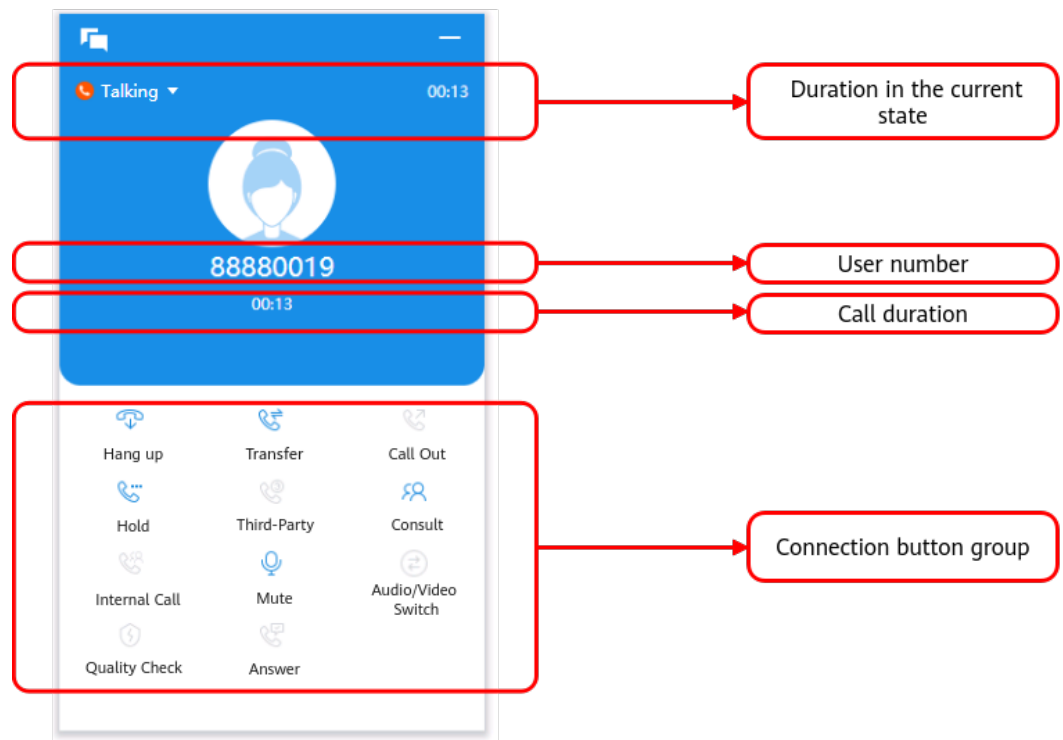
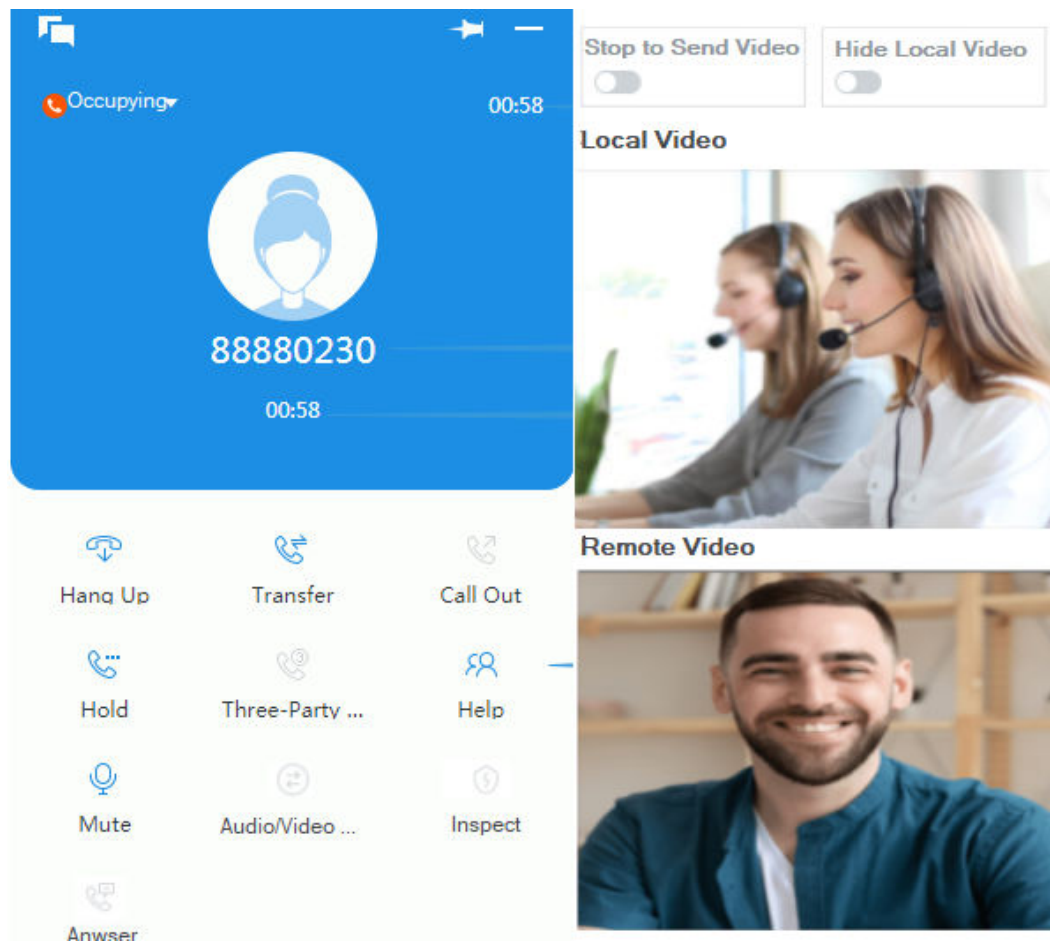


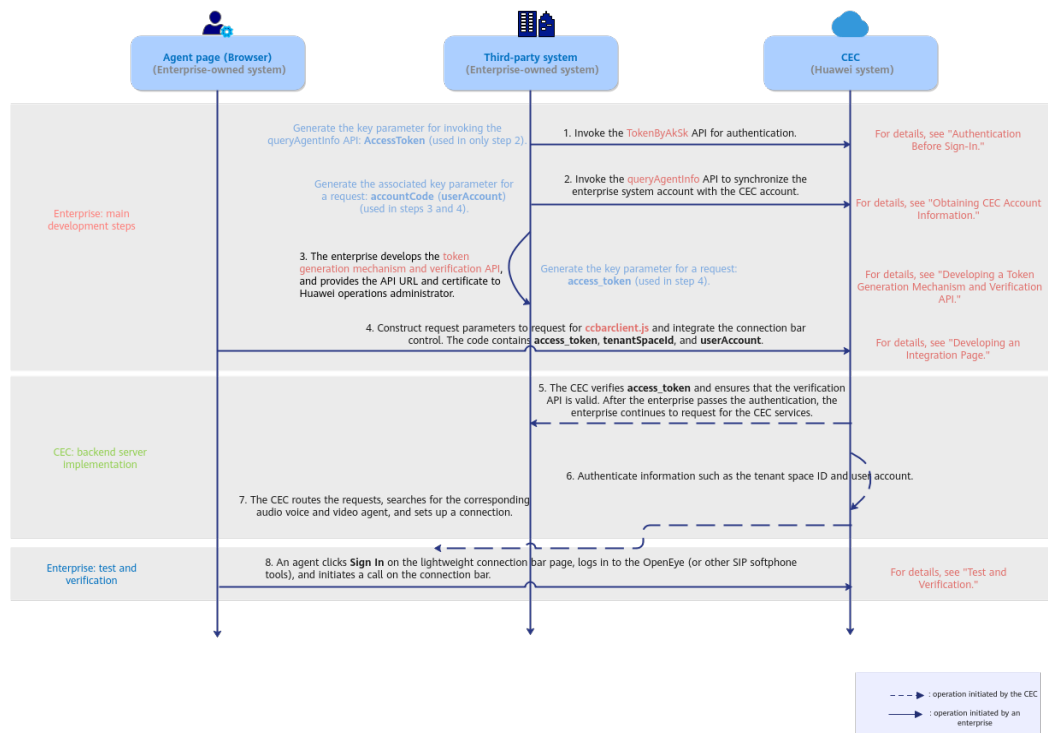
Figure 1-2 Video call page



2 Integration Principles

The lightweight connection bar can be quickly and efficiently integrated into third-party systems. You can learn about main integration principles by referring to [Figure 2-1](#).

Figure 2-1 Lightweight connection bar integration principles



OpenEye: multimedia soft terminal of the CEC, which can implement the call function. You can also sign in to the CEC using other tools, such as WebRTC and the mobile app, to make calls.

Softphone number: calling number, which is the login account of the OpenEye.

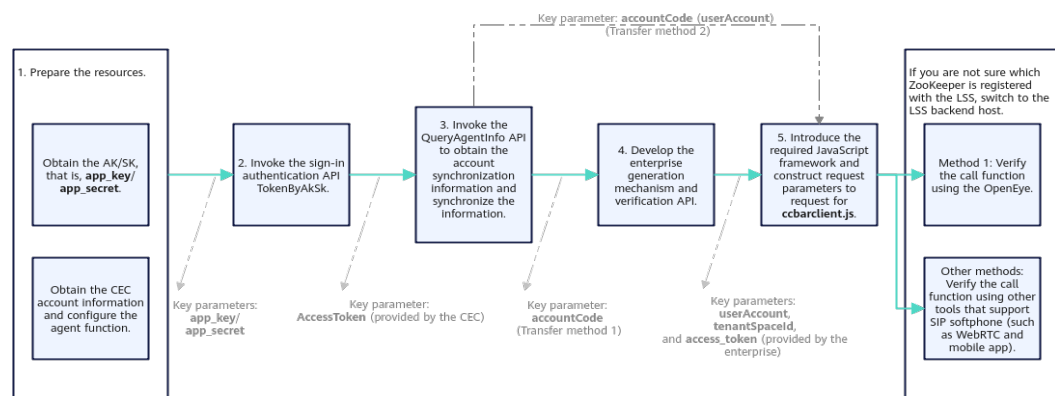
NOTE

Currently, WebRTC registration does not support HTTP. The client page must use HTTPS.

3 Integration Procedure

After learning about integration principles, you can perform integration development by referring to [Figure 3-1](#).

Figure 3-1 Procedure



- Step 1** Before the development, prepare resources and configure the voice and video agent functions in the CEC. For details, see [4 Preparations](#).
- Step 2** Invoke the sign-in authentication API TokenByAkSk. For details, see [5.1 Authentication Before Sign-In](#).
- Step 3** Invoke the QueryAgentInfo API to query softphone information about all agents and synchronize sign-in account information to your business system, including agent IDs, softphone numbers, and softphone passwords. For details, see [5.2 Obtaining CEC Account Information](#).
- Step 4** Develop a token generation mechanism and provide a verification API for the AICC. After the authentication is successful, the AICC sends the token to a third-party system. The third-party system checks whether the request is sent by the system. For details about development requirements, see [5.3 Developing a Token Generation Mechanism and Verification API](#).

NOTICE

Huawei uses HTTPS to ensure the security of information transmission channels. Third-party systems must ensure that the developed authentication functions have security protection capabilities, such as password complexity verification, anti-brute force cracking, and anti-DoS attack.

- Step 5** Construct request parameters to request for **ccbarclient.js** to integrate the lightweight connection control. For details, see [5.4 Developing an Integration Page](#).
- Step 6** Use the OpenEye terminal to test and verify whether the integration is successful. For details, see [7 Test and Verification](#). You can also use other tools that support SIP softphone, such as WebRTC and mobile app, to verify the integration.

----End

NOTICE

Only OAuth authentication is supported between the CEC and enterprise authentication systems. The WebRTC registration does not support the HTTP protocol. Therefore, the client page must use the HTTPS protocol.

4 Preparations

This section describes the preparations that need to be completed in the CEC before integration development.

[4.1 Resource Preparations](#)

[4.2 Enabling the Voice and Video Agent Function in the CEC](#)

4.1 Resource Preparations

Before the integration, prepare the following resources.

1. You have applied for tenant information to the CEC. The CEC operations administrator has added tenant information and provided the following information to you.

Table 4-1 Initial parameters to be obtained

Parameter	Description
userAccount	Account for signing in to the CEC.
access_token	Password for signing in to the CEC.
tenantSpaceId	Tenant space ID generated by the system after your tenant space (CEC) is successfully created. To obtain the tenant space ID, sign in to your tenant space and choose Configuration Center > System Management > Tenant Information .
app_key	App key corresponding to the app created in the API Fabric (CEC API management center). After a tenant space is provisioned, the CEC operations administrator will send the key value at the same time. Keep the key value properly.

Parameter	Description
app_secret	App secret corresponding to the app created in the API Fabric. After a tenant space is provisioned, the CEC operations administrator will send the secret value at the same time. Keep the secret value properly.

2. Use the tenant administrator account and password to sign in to your CEC and change the initial password on the sign-in page.
3. Confirm that the voice and video feature has been enabled for your tenant space (CEC):

Sign in to the tenant space and choose **Configuration Center > System Management > Tenant Information**.

Check the number of voice and video agents, as shown in [Figure 4-1](#). If the number is 0, the voice and video agent feature is disabled. Contact the CEC operations administrator to enable the feature.

Figure 4-1 Checking the voice and video agent feature

Tenant Info

Tenant Name agent_ljy	Company
Creation Time 2022-07-12	Expiration Date 2029-10-17
VDN ID 184	TenantId 202207125666
Time Zone UTC+	Time Zone Offset 00:00
DST Disabled	

Contact Method

Mobile Number 17322339997	Email linjinyi1@huawei.com
------------------------------	-------------------------------

Resource Information

Voice Agents 2	Max. Concurrent Voice Calls 10
Video Agent Quantity 2	Audio IVR Channel Quantity 2
Video IVR Channel Quantity 2	TTS Quantity 0
ASR Quantity 0	Recording Retention Period 12months
Number of Multimedia Agents 2	Versatile Agents 0

- (Optional) Download the OpenEye client to verify the call function. For details, see [Getting Started > Making the First Call](#). You can also use other tools that support SIP softphone, such as WebRTC and mobile app.

4.2 Enabling the Voice and Video Agent Function in the CEC

Procedure

Step 1 Sign in to the CEC as a tenant administrator.

Step 2 Add a voice and video skill queue.

- Choose **Configuration Center > Employee Center > Skill Queue**. The skill queue management page is displayed, as shown in [Figure 4-2](#).

NOTE

A maximum of 1000 skill queues can be created by default.

- Click **New** and set parameters. For details about the parameters, see [Table 4-2](#).

Figure 4-2 Page for configuring a skill queue

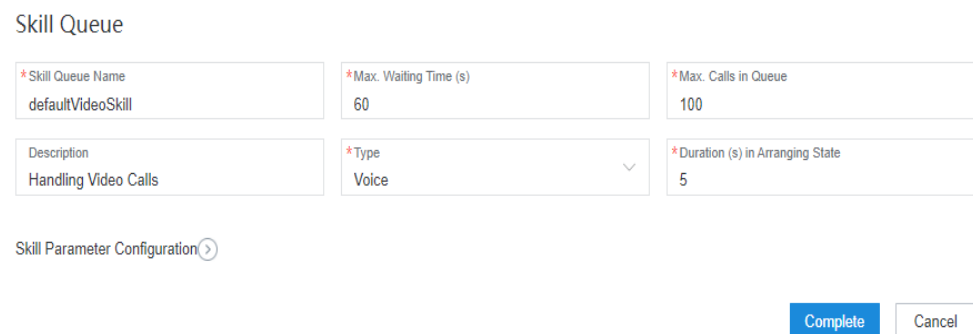


Table 4-2 Parameters for configuring a skill queue

Parameter	Mandatory or Not	Description
Skill Queue Name	Yes	The value can contain a maximum of 20 characters and cannot contain spaces.
Max. Waiting Time (s)	Yes	The default value is 60 . The unit is second. The value ranges from 1 to 60000.
Max. Calls in Queue	Yes	The default value is 100 . The value ranges from 1 to 10000.

Parameter	Mandatory or Not	Description
Description	Yes	The value can contain a maximum of 50 characters.
Type	Yes	The options are as follows: <ul style="list-style-type: none"> ● Voice: A voice skill queue handles voice businesses. ● Multimedia: A multimedia skill queue handles multimedia businesses. ● Video: A video skill queue handles video businesses. ● Voice Click to Dial: A voice click-to-dial skill queue is used together with multimedia businesses. During a text chat with an agent, a customer can directly make a voice call to the agent. ● Video Click to Dial: A video click-to-dial skill queue is used together with multimedia businesses. During a text chat with an agent, a customer can directly make a video call to the agent. <p>NOTE Click-to-dial skill queues apply only to the web channel.</p> The default value is Voice .
Duration (s) in Arranging State	Yes	Duration during which an agent is in wrap-up state after a call ends. The default value is 5 . After this duration, the agent enters the idle state and can answer calls from customers. The value ranges from 0 to 3600.

Parameter	Mandatory or Not	Description
Skill Parameter Configuration	No	<p>Personalized configurations, which are the processing policies when a customer calls a skill queue and the call cannot be connected. The options are as follows:</p> <ul style="list-style-type: none"> ● Skill Timeout <ul style="list-style-type: none"> – Process Method: Processing policy when queuing times out because no idle agent can answer the call. <ul style="list-style-type: none"> ▪ Release (default) ▪ Transfer – Device Type: If Process Method is set to Transfer, you need to configure the skill queue or IVR flow to which the call is transferred. <ul style="list-style-type: none"> ▪ Skill Queue ▪ IVR ● Skill Busy <ul style="list-style-type: none"> – Process Method: Processing policy when the customer is queuing because no idle agent can answer the call, or the number of queuing customers exceeds the upper limit. <ul style="list-style-type: none"> ▪ Release (default) ▪ Transfer – Device Type: If Process Method is set to Transfer, you need to configure the skill queue or IVR flow to which the call is transferred. <ul style="list-style-type: none"> ▪ Skill Queue ▪ IVR ● Skill NoAgents <ul style="list-style-type: none"> – Process Method: Processing policy when no agent can answer the call because no agent is on duty. <ul style="list-style-type: none"> ▪ Release (default) ▪ Transfer

Parameter	Mandatory or Not	Description
		<ul style="list-style-type: none"> - Device Type: If Process Method is set to Transfer, you need to configure the skill queue or IVR flow to which the call is transferred. <ul style="list-style-type: none"> ▪ Skill Queue ▪ IVR • Queuing and waiting configuration: When a customer needs to wait in a queue after making an inbound call, a voice can be played to optimize the customer waiting process. <ul style="list-style-type: none"> - Queuing Method <ul style="list-style-type: none"> ▪ Default Wait Tone ▪ Customizing the Wait Tone ▪ IVR • Keeping and waiting configuration: When a call needs to be held and the customer needs to wait, a voice can be played to optimize the customer waiting process. <ul style="list-style-type: none"> - Keeping Method <ul style="list-style-type: none"> ▪ Default Keeping Tone ▪ Customizing the Keeping Tone • Skill AnswerMode: After an agent answers a call from a customer, the employee ID of the agent can be played to the customer. <ul style="list-style-type: none"> - Answer Type <ul style="list-style-type: none"> ▪ Report employee ID ▪ Report no voice <p>NOTE Calls in voice, video, and click-to-dial skill queues can be transferred to IVRs or skill queues. Calls in multimedia skill queues can be transferred only to skill queues. The skill queue selected for call transfer must be of the same type as the skill queue to be created. The waiting tones can be set only for voice and video skill queues and are not displayed for multimedia and click-to-dial skill queues.</p>

3. Click **Complete**.

Step 3 Add a called route of the voice or video type.

1. Choose **Configuration Center > Access Configuration > Called Route**.
2. Click **New** to add parameter information for the VDN and click **Complete**, as shown in **Figure 4-3**. **Table 4-3** describes the parameters.

Figure 4-3 Page for configuring a called route

Table 4-3 Parameters for configuring a called route

Parameter	Mandatory or Not	Description
Access Code	Yes	Customer service hotline. Customers can dial the access code to connect to agents. Click to select an access code, for example, a multimedia access code, from the list in the dialog box that is displayed.
Extension Code	No	To set one access code for multiple destination devices, you can configure extension codes. For example, if the access code is 12345, you can add extension code 1 to route calls to skill queue A and extension code 2 to route calls to skill queue B. In this way, a customer can dial 123451 to directly access skill queue A.
Device Type	Yes	Select Skill Queue to configure a called route of the skill queue type.
Skill Queue	Yes	Associate the skill queue created in Step 2 . Skill Queue: Click to select a skill queue from the list in the dialog box that is displayed. The type of the skill queue is the same as that of the access code. For example, if you select a multimedia access code, all available skill queues are of the multimedia type.

Step 4 Configure a business account and skill queue.

1. Choose **Configuration Center > Employee Center > Agent Management**.
2. Select an agent ID and click **Configure** in the **Operation** column. The page for configuring agent information is displayed, as shown in **Figure 4-4**. **Table 4-4** describes the parameters.
3. Associate the business account and skill queue with the agent.

Figure 4-4 Page for configuring agent information

Table 4-4 Parameters for configuring agent information

Parameter	Mandatory or Not	Description
Platform Role	Yes	Agent role. This parameter is mandatory. <ul style="list-style-type: none"> - Common agent: This role can answer or transfer inbound calls from customers. - Quality checker: This role can intervene in calls between common agents and customers. For example, this role can perform operations, such as insertion, interception, and forcible busy state setting, to coach and supervise agents' handling of inbound calls. - Callout agent: This role can answer, transfer, or reject inbound calls from customers.
Agent Type	Yes	Type of businesses that can be processed by an agent. This parameter is mandatory. <ul style="list-style-type: none"> - Voice agent - Video agent - Multimedia agent - Versatile agent

Parameter	Mandatory or Not	Description
Agent Mobile/Fixed-Line Number	No	Mobile number or fixed-line phone number used by an agent.
Account	Yes	Employee account. For details about how to configure an employee, see User Guide > Tenant Administrator Guide > Managing Employees .
Intelligent Recognition	No	Whether an agent is an intelligent agent. By default, this switch is turned off. In addition to basic voice control functions, intelligent agents support real-time ASR and related intelligent recommendation functions. Before turning on this switch, ensure that the number of agents for which intelligent recognition is enabled does not exceed the number of intelligent agents allocated when the tenant is created.
SinglePhone Agent Recognition	No	After this switch is turned on, an agent can dial a specified access code to access an IVR flow, press a key as prompted to enter the employee ID and password to sign in, and answer calls on a mobile phone. When this switch is turned on, the system O&M personnel need to customize the single-phone agent process for the tenant based on the platform, and the tenant needs to provide number resources for accessing the single-phone agent process.
Agent Number Anonymization Flag	No	Flag for a third party to mark whether an agent has the anonymization feature. This is not a feature switch. The anonymization feature enables agents to customize the calling number displayed on the user side (the calling number displayed to the user) and the calling number displayed on the agent side (the calling number displayed to the customer manager).

Parameter	Mandatory or Not	Description
Select Skill Queue	Yes	Agent skill queue. If multiple skill queues need to be added, ensure that the media types of all the skill queues are the same, except for versatile agents. For example, the media types of all the skill queues are voice and video, or multimedia. NOTE <ul style="list-style-type: none"> - If Agent Type is set to Video agent, set the number of video agents allowed when applying for tenant resources. - If Agent Type is set to Multimedia agent, set the number of multimedia agents allowed when applying for tenant resources. - If Agent Type is set to Versatile agent, set the number of versatile agents allowed when applying for tenant resources. - To add more business accounts, choose Configuration Center > Employee Center > Employee.

4. Click **Submit**. The business account and skill queue are associated with the agent ID.
5. (Optional) Click **Batch Configure**. On the **Batch Agent Info Configuration** page, configure agent information in batches, as shown in [Figure 4-5](#).

Figure 4-5 Batch configuration

Batch Select

Batch Selection Mode
 By Employee ID By Segment

Selected Agents:
 1519 1520

Agent Info Configuration

Platform Role: Common agent (dropdown)
 Agent Type: Video agent (dropdown)

Enter a New Password:
 Confirm Password:

Current Account Password:

Intelligent Recognition: Please Select... (dropdown)
 SinglePhone Agent Recognition: Please Select... (dropdown)

Select Skill Queue:

Skill Queue: defaultVideoSkill *Agent Skill Weight: 1 *Agent Weight: 1

- **Batch Select:** Select agents to be configured by employee ID or employee ID segment.
- **Agent Info Configuration:** Set parameters by referring to [4-2](#).

----End

5 Integration Development

This section describes how to integrate the connection control in your system.

[5.1 Authentication Before Sign-In](#)

[5.2 Obtaining CEC Account Information](#)

You can invoke the `queryAgentInfo` API to obtain agent information that can be synchronized, including the registration server information, agent account and password, and CEC sign-in account. After the API is invoked, the CEC sign-in account is associated with the agent sign-in account in your business system. When the connection bar is requested on the enterprise page, the matching account information can be automatically obtained.

[5.3 Developing a Token Generation Mechanism and Verification API](#)

[5.4 Developing an Integration Page](#)

[5.5 Making Outbound Calls in One-Click Mode](#)

5.1 Authentication Before Sign-In

Before using the CEC in your system, the authentication and verification are performed by using the AK/SK-based authentication API (`tokenByAkSk`).

Step 1 Invoke the API Fabric verification API `tokenByAkSk` of the CEC to perform authentication and obtain an access token.

1. Use an API test tool to send a request to obtain the token returned by the CEC.
2. Obtain the URL in the following format and select the POST mode.

HTTPS method: POST

URL: `https://Domain address/apigovernance/api/oauth/tokenByAkSk`

NOTE

- Replace *Domain address* with the actual address or domain name of the CEC. For example, in the Huawei Cloud production environment, replace *Domain address* with **service.besclouds.com**. In this case, the URL is **https://service.besclouds.com/apigovernance/api/oauth/tokenByAkSk**.
- HTTP is an insecure protocol, which may bring risks to the system. Therefore, it is not recommended. HTTPS is recommended.

3. Enter the values of **app_key** and **app_secret** to the body based on the format of the API calling example to obtain the value of **AccessToken**, as shown in **Figure 5-1**. You can learn about requirements on request and response parameter attributes by referring to **Table 5-1** and **Table 5-2**.

Figure 5-1 Invoking AccessToken

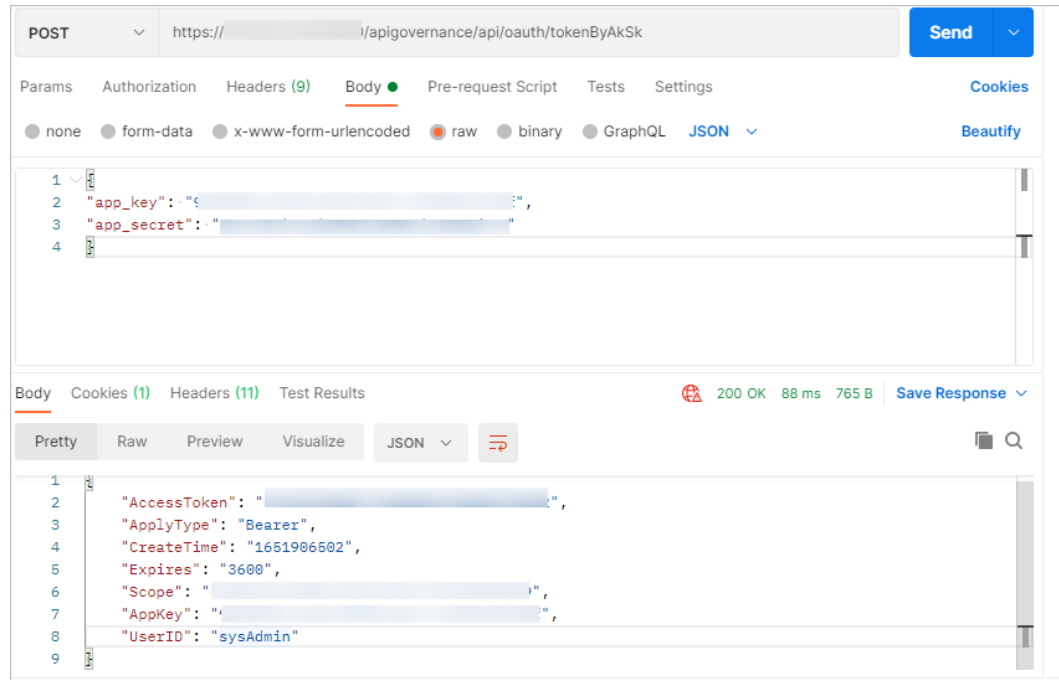


Table 5-1 Request body description

Parameter	Type	Position	Mandatory or Not	Description
app_key	String	Body	Yes	App key corresponding to the app created in the API Fabric (CEC API management center). After a tenant space is provisioned, the CEC operations administrator will send the key value at the same time. Keep the key value properly.

Parameter	Type	Position	Mandatory or Not	Description
app_secret	String	Body	Yes	App secret value corresponding to the app created in the API Fabric. After a tenant space is provisioned, the CEC operations administrator will send the secret value at the same time. Keep the secret value properly.

- Example

```
{
  "app_key": "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
  "app_secret": "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
}
```

Table 5-2 ResponseBody description

Parameter	Type	Position	Mandatory or Not	Description
AccessToken	String	Body	Yes	Obtain the output parameter token based on the app key and app secret created in the API Fabric. The token is used to invoke synchronization information.
ApplyType	String	Body	Yes	Token type. Currently, only Bearer is supported.
CreateTime	String	Body	Yes	Token creation time.
Expires	String	Body	Yes	Token expiration time.
Scope	String	Body	Yes	Scope of APIs that can be accessed using the token.
AppKey	String	Body	Yes	App key.
UserID	String	Body	Yes	User ID.

- Example

```
{
  "AccessToken": "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
  "ApplyType": "Bearer",
  "CreateTime": "1543395801",
  "Expires": "600",
}
```


Table 5-3 RequestHeader description

Parameter	Type	Position	Mandatory or Not	Description
x-app-key	String	Header	Yes	Identifier of an app. That is, app key.
Authorization	String	Header	Yes	Authentication information. The format is Bearer {Value of AccessToken obtained by invoking the authentication API}.

- Example

```
{
  "x-app-key":XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  "Authorization":Bearer XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
}
```

Table 5-4 ResponseBody description

Parameter	Type	Position	Mandatory or Not	Description
sipServiceIp	String	Body	No	Softphone registration address, in IPv4 format.
sipServicePort	Integer	Body	No	Softphone registration port number. The value ranges from 1 to 65535.
agents	List	Body	Yes	Agent set.
workNo	Integer	agents	Yes	Agent ID.
sipAccount	String	agents	No	Softphone number.
sipPwd	String	agents	No	Softphone password.
accountCode	String	agents	No	CEC sign-in account.

- Example

```
{
  "sipServiceIp":"10.100.10.10",
  "sipServicePort":1000,
  "agents": [
    {
      "workNo": 1001,
      "accountCode": "11",
      "sipAccount": "1001",
      "sipPwd": "cti-1234"
    }
  ]
}
```

```
]
}
```

Step 2 After agent information is synchronized, you may need to perform the following operations in your system:

1. (Optional) Associate an account in your business system with a CEC agent. The associated account is required when you request the CEC in [5.4 Developing an Integration Page](#). If you want a user of your business system to select a CEC sign-in account during each sign-in to the CEC, skip this step.
2. (Optional) Provide the softphone number, softphone password, and softphone registration address and port number for an agent so that the agent can use the information to log in to the OpenEye client and make or answer calls. If you use other SIP softphone tools to handle call businesses, skip this step.

----End

5.3 Developing a Token Generation Mechanism and Verification API

You need to develop a token generation mechanism for generating verification tokens and develop a verification API for the CEC to perform authentication. When receiving a request from an agent (that is, a request sent during page integration), the CEC invokes the API to obtain the values of **access_token** (which is generated by the token generation mechanism and is different from **AccessToken** in [5.1 Authentication Before Sign-In](#)), **tenantSpaceld**, and **userAccount**, and sends them to your system for confirmation.

NOTE

A demo of the token generation mechanism and verification API is provided for your reference. You can download the demo from the Huawei Developer Forum(<https://bbs.huaweicloud.com/forum/thread-192048-1-1.html>). The demo provides the algorithm and verification logic. Use a secure algorithm in actual use. This demo cannot be directly used for production and is for demonstration only. Design the token generation mechanism and verification API based on your system security requirements.

Step 1 Develop a token generation mechanism. You need to develop the mechanism based on the features and security requirements of your system.

Step 2 Develop a verification API. The verification API must meet the following requirements. [Table 5-5](#) and [Table 5-6](#) describes the requirements for request parameters and response parameters.

HTTP method: POST

URL: `http(s)://IP address.Port number/rest/cc-management/v1/thirdparty/thirdPartyValidate` (which can be customized by the enterprise)

NOTE

HTTP is an insecure protocol, which may bring risks to the system. Therefore, it is not recommended. The secure HTTPS protocol is recommended. If the HTTPS protocol is used, you need to prepare a certificate in CER format to verify the validity of the HTTPS website.

Table 5-5 Request body description

Parameter	Type	Position	Mandatory or Not	Description
access_token	String	Body	Yes	Verification information generated after an enterprise develops a token generation mechanism.
tenantSpaceId	String	Body	Yes	Tenant space ID provided by the CEC. To obtain the tenant space ID, choose Configuration Center > System Management > Tenant Information .
userAccount	String	Body	Yes	CEC sign-in account. The value comes from the accountCode parameter in 5.2 Obtaining CEC Account Information .

- Example

```
{
  "access_token":"XXXXXXXXXXXXXXXXX",
  "tenantSpaceId":"XXXXXXXXXXXXXXXXX",
  "userAccount":"XXXXXXXXXXXXXXXXX"
}
```

Table 5-6 Response description

Parameter	Type	Position	Mandatory or Not	Description
retCode	Integer	Body	Yes	API result code. The value 0 indicates success, and other values indicate failure.
message	String	Body	Yes	Result.

- Example

```
{
  "retCode":0,
  "message":"validate success"
}
```

Step 3 Submit the URL and certificate file of the verification API to the CEC operations administrator. The operations administrator configures the URL and certificate file in the CEC.

----End

5.4 Developing an Integration Page

After configuring the tenant space and API, you can integrate the CEC functions into your customer service system.

5.4.1 Core Code Analysis

Procedure

- Step 1** Import the security certificate provided by the system O&M administrator to the current browser.
- Step 2** Introduce the required JavaScript framework. In this example, jQuery needs to be introduced. The reference version is jQuery-v1.8.0, as shown in [Figure 5-3](#).

Figure 5-3 JavaScript framework reference example

```
<script type="text/javascript" src="js/jquery-1.8.0.js"></script>
```

- Step 3** Add the following code to the page to be referenced.

If a cross-origin resource sharing (CORS) problem occurs during page integration, resolve the problem by referring to [How Can I Resolve the Reported Cross-domain Error When the XMLHttpRequest Requests the URL of the CEC?](#)

```
<script type="text/javascript">
  // The current demo needs to introduce jQuery.
  // Create the <script> tag and run the script.
  var importScript = (function (oHead) {

    function loadError(oError) {
      throw new URIError("The script " + oError.target.src + " is not accessible.");
    }

    return function (sSrc, fOnload) {
      var oScript = document.createElement("script");
      oScript.type = "text/javascript";
      oScript.onerror = loadError;
      if (fOnload) { oScript.onload = fOnload; }
      oHead.appendChild(oScript);
      oScript.innerHTML = sSrc;
    }

  })(document.head || document.getElementsByTagName("body")[0]);

  // Set request parameters. access_token indicates the authentication information, tenantSpaceld
indicates the tenant space ID, and userAccount indicates the sign-in account. For details, see Table 5-7.
  var param = {
    "access_token": "xx",
    "tenantSpaceld": "xxx",
    "userAccount": "xxx"
  }

  // Define the $aicc_ContextPath variable. The variable name must be $aicc_ContextPath, and the
value is https://IP address:Port number/service-cloud/ or https://Domain name/service-cloud/.
  const $aicc_ContextPath = "https://10.10.10.10:8080/service-cloud/";
  // Request the CEC and pass the data content (access_token, tenantSpaceld, and userAccount).
  $.ajax({
    type: "post",
    data: JSON.stringify(param),
    url: $aicc_ContextPath+"ccdesktop/pages/cc-bar/js/ccbarclient.js?t="+ Math.random(),
```

```

crossDomain: true,
xhrFields: {
  withCredentials: true
},
error: function (XMLHttpRequest, textStatus, errorThrown) {

},
success: function (data) {
  // After the request is successful, the JavaScript is executed. The CEC returns the JavaScript and
  dynamically generates a connection bar. The style and position of the connection bar cannot be modified.
  importScript(data)

  // Transfer an event (agent status) using the postMessage.
  var agentStatusParam = new Array;
  agentStatusParam.push("AgentState_Busy");
  agentStatusParam.push("AgentState_Work");
  agentStatusParam.push("AgentState_Ready");

  agentStatusParam.push("AgentEvent_TransOutResult");
  var json = { name: "eventpost", param: agentStatusParam }
  setTimeout(function(){ window.frames["ccbarclient"].postMessage(JSON.stringify(json),
  $aicc_ContextPath); }, 1000)
  // Check the latest agent status in the CEC every second.
  // If the value of window.frames["ccbarclient"] is incorrect, set the check period to a
  larger value, for example, 2000 or 3000.
  // frames: The value is ccbarclient, which is dynamically generated based on the data
  content returned by importScript.
  window.listeners["eventpost"] = callback;

}
});
</script>

```

Table 5-7 Parameter description

Parameter	Mandator y or Not	Description
ip&port	Yes	Actual domain name. Replace https://10.10.10.1/ with the public network domain name of the AICC.
access_token	Yes	Verification information generated by the enterprise. The value is the same as the token in Table 5-5 .
tenantSpaceld	Yes	Tenant space ID.
userAccount	Yes	CEC sign-in account. The value comes from the accountCode parameter in 5.2 Obtaining CEC Account Information .

----End

5.4.2 JavaScript Code Example

Environment Requirement	--
--------------------------------	----

Reference Library	jquery.js
Download Link	index.html

NOTICE

- The demo described in this document may involve the use of personal data. You are advised to comply with relevant laws and regulations and take sufficient measures to ensure that personal data is fully protected.
- The demo described in this document is for demonstration only. Commercial use of the demo is prohibited.
- Information in this document is for reference only and does not constitute any offer or commitment.

index.html

```
<!DOCTYPE html>
<!--
  Tutorial.
  Check available devices.
-->
<html>

<head>
  <title>Tutorial. Check devices</title>
  <meta charset="UTF-8" />
  <!--link rel="stylesheet" href="phone.css"-->
  <!--link rel="icon" href="favicon.png"-->
</head>

<body>
  <!-- Check that browser is not IE -->
  <script>
    var ua = window.navigator.userAgent;
    if (ua.indexOf('MSIE ') > 0 || ua.indexOf('Trident/') > 0) {
      alert("Internet Explorer is not supported. Please use Chrome or Firefox");
    }
  </script>

  <!-- Phone script -->
  <script src="js/jquery-1.8.0.js"></script>
  <script>
    // The current demo needs to introduce jQuery.
    // Create the <script> tag and run the script.
    var importScript = (function (oHead) {

      function loadError(oError) {
        throw new URIError("The script " + oError.target.src + " is not accessible.");
      }

      return function (sSrc, fOnload) {
        var oScript = document.createElement("script");
        oScript.type = "text/javascript";
        oScript.onerror = loadError;
        if (fOnload) { oScript.onload = fOnload; }
        oHead.appendChild(oScript);
        oScript.innerHTML = sSrc;
      }
    })(document.head || document.getElementsByTagName("body")[0]);
```

```

// Set request parameters. access_token indicates the authentication information, tenantSpaceld
indicates the tenant space ID, and userAccount indicates the sign-in account. For details, see Table 5-7.
var param = {
  "access_token": "XXXXXXXXXXXX",
  "tenantSpaceld": "XXXXXXXXXXXX",
  "userAccount": "XXXXXXXXXXXX"
}

// Define the $aicc_ContextPath variable. The variable name must be $aicc_ContextPath, and the
value is https://IP address:Port number/service-cloud/ or https://Domain name/service-cloud/.
const $aicc_ContextPath = "https://ip:port/service-cloud/";
// Request the CEC and pass the data content (access_token, tenantSpaceld, and userAccount).
$.ajax({
  type: "post",
  data: JSON.stringify(param),
  url: $aicc_ContextPath+"ccdesktop/pages/cc-bar/js/ccbarclient.js?t=" + Math.random(),
  crossDomain: true,
  xhrFields: {
    withCredentials: true
  },
  error: function (XMLHttpRequest, textStatus, errorThrown) {

  },
  success: function (data) {
    // After the request is successful, the JavaScript is executed. The CEC returns the JavaScript and
    dynamically generates a connection bar. The style and position of the connection bar cannot be modified.
    importScript(data)

    // Transfer an event (agent status) using the postMessage.
    var param = new Array;
    param.push("AgentState_Busy");
    param.push("AgentState_Work");
    param.push("AgentState_Ready");

    param.push("AgentEvent_TransOutResult");
    var json = { name: "eventpost", param: param }
    setTimeout(function(){ window.frames["ccbarclient"].postMessage(JSON.stringify(json),
$aicc_ContextPath); }, 1000)
    // Check the latest agent status in the CEC every second.
    // If the value of window.frames["ccbarclient"] is incorrect, set the check period
    (1000) to a larger value, for example, 2000 or 3000.
    // frames: The value is ccbarclient, which is dynamically generated based on the data content
    returned by importScript.
    window.listeners["eventpost"] = callback;

  }
});

</script>

<!-- HTML components of simple GUI -->
<div id="status_line">
</div>
</body>
</html>

```

5.5 Making Outbound Calls in One-Click Mode

The one-click outbound call function enables an agent to click a customer's phone number in your business system or on a customer information page to a one-click callback. The agent does not need to manually enter the phone number. If you expect this function to be implemented in your system, go on.

The following examples are provided for you.

A page integrates the lightweight connection bar and needs to implement one-click outbound call. In this case, you need to add request code in the **Developing an Integration Page** step. The reference code is as follows:

```
//Saicc_ContextPath indicates the IP address and port number for loading ccbarclient.js, for example, 'https://10.101.95.209:28090/'
<script>
  callout = function ()
  {
    var data={"name":"callout","param":["88889007","audio"]};
    window.frames["ccbarclient"].postMessage(data, $saicc_ContextPath);
  }
</script>
<button onclick="callout();">Outbound call</button>
```

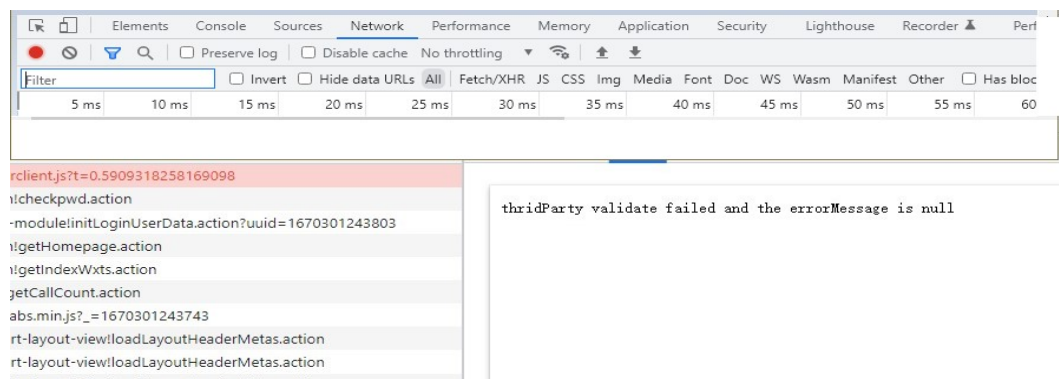
NOTE

A third-party system can invoke **data** in either of the following formats:

```
let data={
  name: 'callout',
  param: {
    number:'88880523',//Number for making an outbound call
    mode:'audio'//The options are audio (voice call) and video (video call).
  }
}
data={
  name: 'callout',
  param: [
    '88880813',
    'video'
  ]
}
```

5.5.1 What Do I Do If Error Code 500 Is Displayed for a CORS Request and Third-Party Authentication Fails

Symptom



The following error message is displayed: thridParty validate failed and the errorMessage is null

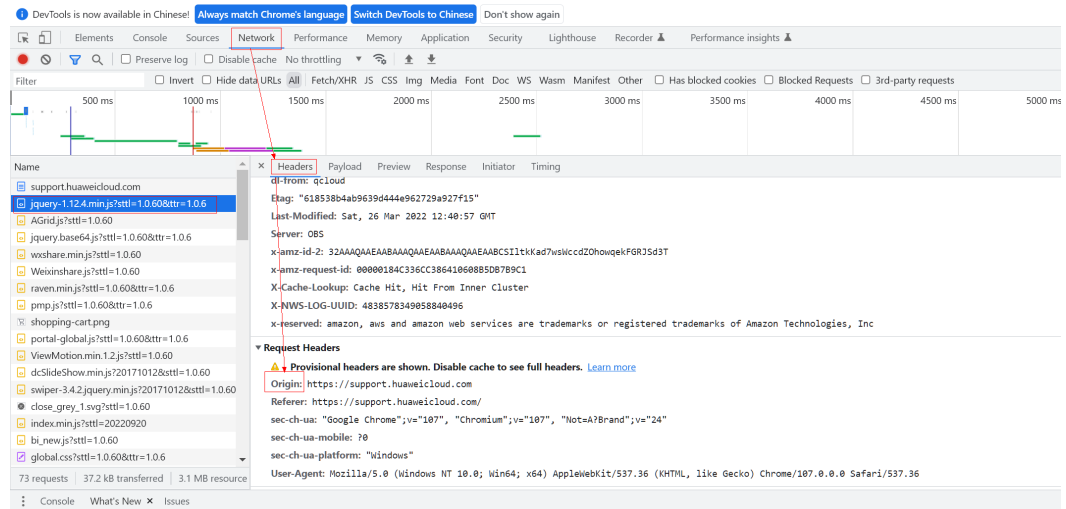
Solution

Generally, this error occurs because a security plugin is installed in the browser. Press **F12** to check header information.

Step 1 After the browser reports the error, press **F12**.

- Step 2** Refresh the page and click any line under **Name** (request name) on the left.
- Step 3** On the **Network** page, click the **Headers** tab page, expand **Request Headers**, and check the **Origin** parameter. This parameter indicates the request source. The value is the protocol and domain name of the page where the requested resource is located. If this parameter exists, the CORS request is normal.
- Step 4** If the **Origin** parameter does not exist, check the plugins installed in the browser.

Figure 5-4 Checking the request header parameter



----End

6 (Optional) Developing Other Functions

6.1 Listening to Connection Events

6.2 Implementing the One-Click Outbound Call Function

6.1 Listening to Connection Events

You can listen to call events that you pay attention to, such as agent status events and voice and video call events, in your enterprise system. If you want to use these events to develop functions such as logging and pop-up screen display in your system, continue to read this section. [Table 6-1](#) lists the main events that can be listened to.

Table 6-1 Event description

Event	Event Identifier	Function
Busy	AgentState_Busy	This event indicates that an agent is handling a call.
Working	AgentState_Work	This event indicates that an agent enters the wrap-up state.
Idle	AgentState_Ready	This event indicates that an agent enters the idle state.
Entering the Busy State Successfully	AgentState_SetNotReady_Success	This event indicates that an agent enters the busy state successfully.
Quitting the Busy State Successfully	AgentState_CancelNotReady_Success	This event indicates that an agent exists the busy state successfully.
Requesting Rest Successfully	AgentState_SetRest_Success	This event indicates that an agent successfully requests a rest.

Event	Event Identifier	Function
Quitting the Rest State Successfully	AgentState_CancelRest_Success	This event indicates that an agent successfully cancels the rest request.
Rest Timeout Reminder	AgentState_Rest_Timeout	This event indicates that the rest request of an agent times out, reminding the agent of the timeout. This event does not indicate that the agent exits the rest state.
Turning to the Working State	AgentState_SetWork_Success	This event indicates that an agent enters the working state.
Quitting the Working State	AgentState_CancelWork_Success	This event indicates that an agent exits the working state.
Entering the Talking State	AgentEvent_Talking	This event indicates that the call of an agent enters the talking state.
Holding Success	AgentEvent_Customer_Alerting	This event indicates that the current agent is in the holding state.
Making an Outgoing Call Unsuccessfully	AgentEvent_Call_Out_Fail	This event indicates that an agent fails to make an outbound call. The possible cause is that the phone number is incorrect.
Agent Exiting a Call	AgentEvent_Call_Release	This event indicates that an agent releases a call with a specified call ID.
User Exiting a Call	AgentEvent_Customer_Release	This event indicates that a customer releases a call with a specified call ID.
Phone in Ringing State	AgentOther_PhoneAlerting	This event indicates that the phone of an agent is ringing.

You can use the following example to listen to required events on your page. The following uses the AgentOther_PhoneAlerting event as an example.

A lightweight connection bar is integrated into a page. When an agent receives an inbound call, the inbound notification is ringing on the page. You can set the callback method to capture the AgentOther_PhoneAlerting event to implement the inbound call notification function. The following are reference codes:

```
callback = function(data){
    alert("Hi, a new call is incoming!")
}

window.onload = function(){
    var param = new Array;
    param.push("AgentOther_PhoneAlerting");
    var json={name:"eventpost",param:param}

    window.frames["ccbarclient"].postMessage(JSON.stringify(json),$saicc_ContextPath);
}
if (window.addEventListener) {
    window.addEventListener('message', callback); }
else {
    window.attachEvent('onmessage', callback);
}
```

NOTE

callback: callback method, which implements specific business functions.

\$saicc_ContextPath: domain name, which is generated on the page after the integration JavaScript is successfully loaded.

ccbarclient: iframe name of the integrated connection bar.

eventpost: name of the method for obtaining connection events. The name is registered in the integrated connection bar.

addEventListener: adding a connection event listener.

6.2 Implementing the One-Click Outbound Call Function

The one-click outbound call function enables an agent to click a customer's phone number in your business system or on a customer information page to a one-click callback. The agent does not need to manually enter the phone number. If you expect this function to be implemented in your system, go on.

The following examples are provided for you.

A page integrates the lightweight connection bar and needs to implement one-click outbound call. In this case, you need to add request code in the [Developing an Integration Page](#) step. The reference code is as follows:

```
//$saicc_ContextPath indicates the IP address and port number for loading ccbarclient.js, for example,
'https://10.101.95.209:28090/'
<script>
    callout = function ()
    {
        var data={"name":"callout","param":["88889007","audio"]};
        window.frames["ccbarclient"].postMessage(data, $saicc_ContextPath);
    }
</script>
<button onclick="callout();">Outbound call</button>
```

 NOTE

A third-party system can invoke **data** in either of the following formats:

```
let data={
  name: 'callout',
  param: {
    number:'88880523',//Number for making an outbound call
    mode:'audio'//The options are audio (voice call) and video (video call).
  }
}data={
  name: 'callout',
  param: [
    '88880813',
    'video'
  ]
}
```

7 Test and Verification

After integration development is complete, perform the following steps to verify that the lightweight connection bar is successfully integrated to your page. The following uses the Google Chrome as an example.

Procedure

Step 1 On the page after integration development, press **F12** to open the console, click the **Network** tab, and refresh the page.

Click the **cckbarclient.js** request on the console and click **Response** on the right. If a response is returned, the invoking is successful.



```
1 //Domain name loaded by TODO JavaScript. During actual release, the domain name is that externally released by service-cloud
2 var $ContextPath = "https://100.101.114.153/service-cloud";
3
4 //Write JavaScript to the third-party web portal
5 document.write("<script src=" + $ContextPath + "/ccdesktop/pages/cc-
6 document.write("<script src=" + $ContextPath + "/ccdesktop/pages/cc-
7 document.write("<script src=" + $ContextPath + "/ccdesktop/pages/cc-
8 document.write("<link rel='stylesheet' type='text/css' href=" + $Con
9
10
11
12 var ele = document.createElement("div");
13 ele.id = "cckbar_popup";
14 document.body.appendChild(ele);
```

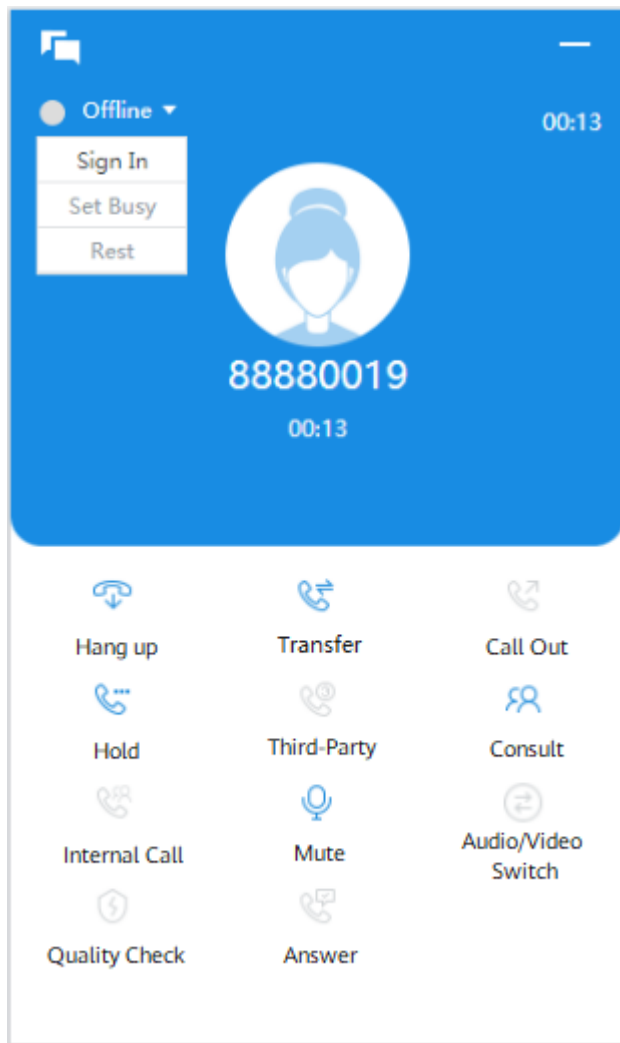
Step 2 Open the Google Chrome, choose **Application**, choose **Storage > Cookies > Customer domain name** on the left, and check whether the following parameters are written into cookies:

- sum_atime
- sum_refresh_token
- sum_sid
- sum_token

Name	Value	Domain	Path	Expires
bes-site-param	%7B%22siteVersion%22%3A%22c10%22%2C%22skinP...	10 .101.114.153	/service-cloud/	Ses...
com.huawei.boss.CURRENT_MENUID	6010100020004	10 .101.114.153	/	Ses...
com.huawei.boss.CURRENT_TAB	6010100020004	10 .101.114.153	/	Ses...
sum_atime	1543407621218	10 .101.114.153	/	Ses...
sum_gvcode_key	287a320ce94e45e0bafdec12e3c1a349	10 .101.114.153	/	Ses...
sum_refresh_token	eyJhbGciOiJIUzI1Ni9iOiJpYXQlOiE1NDM0MDc2MTYzI...	10 .101.114.153	/	Ses...
sum_sid	1543407616880038533	10 .101.114.153	/	Ses...
sum_token	eyJhbGciOiJIUzI1Ni9iOiJpYXQlOiE1NDM0MDc2MTYzI...	10 .101.114.153	/	Ses...
u-ct	/service-cloud	10 .101.114.153	/service-cloud	Ses...
u-i18n	99abd56465e17499d72fb37336cb1d31	10 .101.114.153	/service-cloud	Ses...
u-locale	zh_CN	10 .101.114.153	/	Ses...



Step 3 Click  in the lower right corner of the page to display the lightweight connection bar. Click  next to the **Offline** state and select **Sign In**.



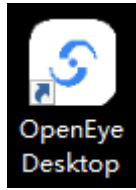
----End

To answer inbound calls after sign-in, perform the following steps to log in to the OpenEye.

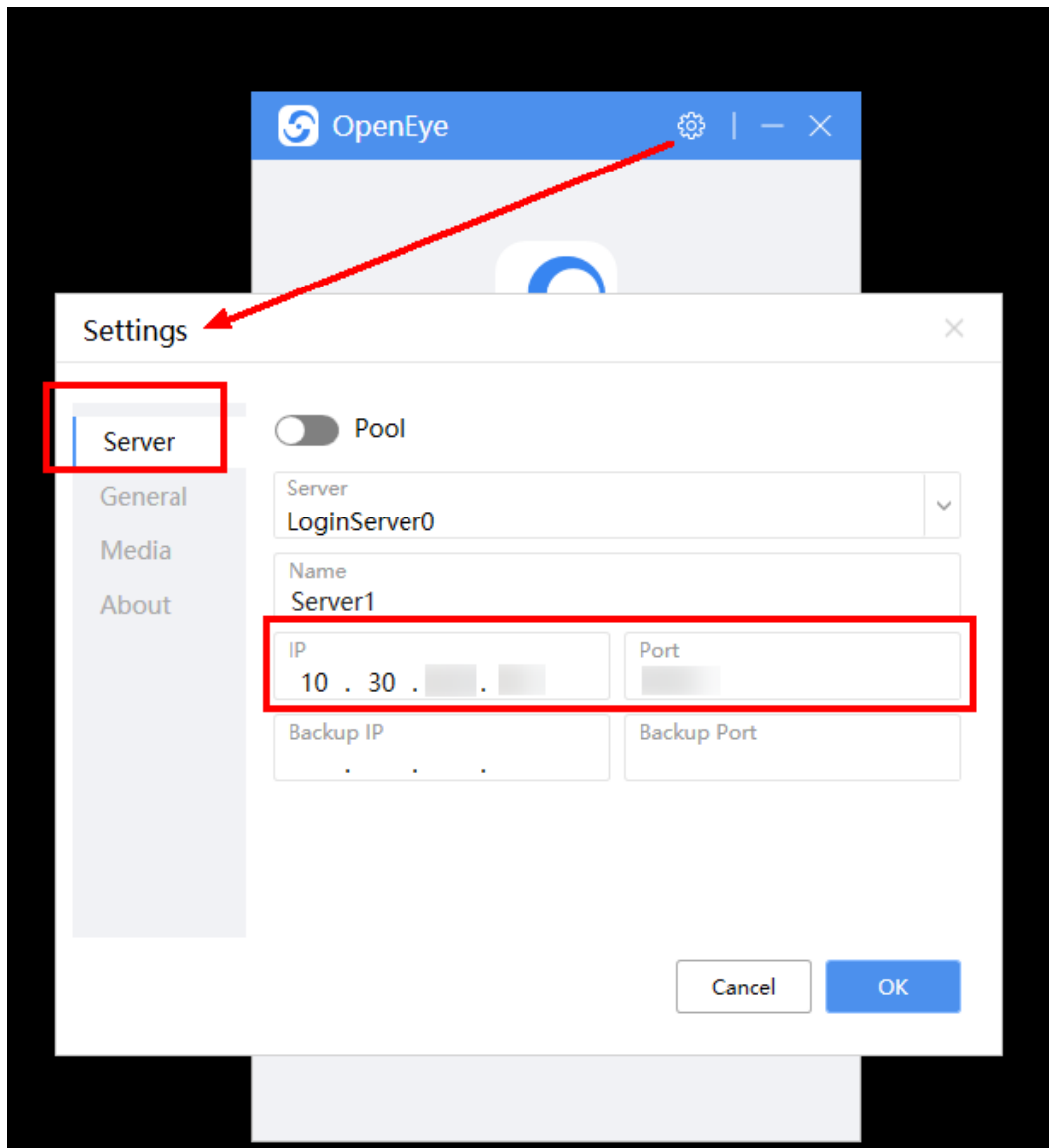
Step 1 Install the OpenEye on the local PC.

Obtain the OpenEye installation program and key value from the administrator, double-click **OpenEye_***.exe**, and use the default settings.

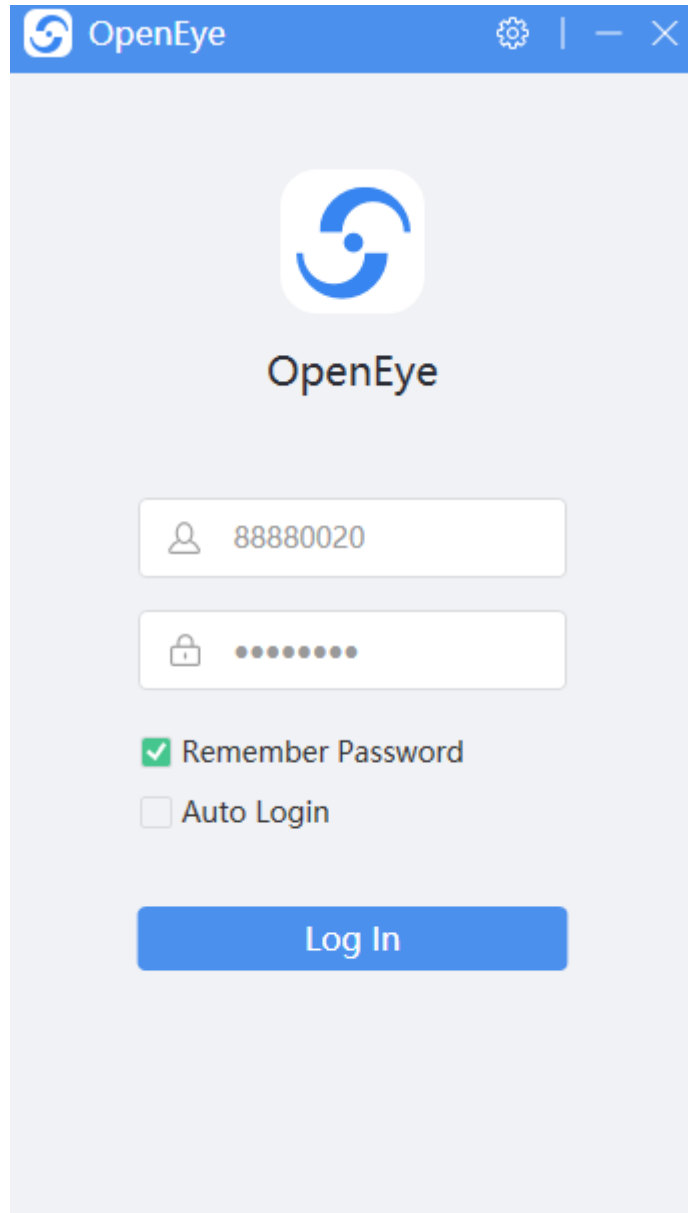
After the installation is complete, the OpenEye icon is displayed on the desktop, as shown in the following figure.



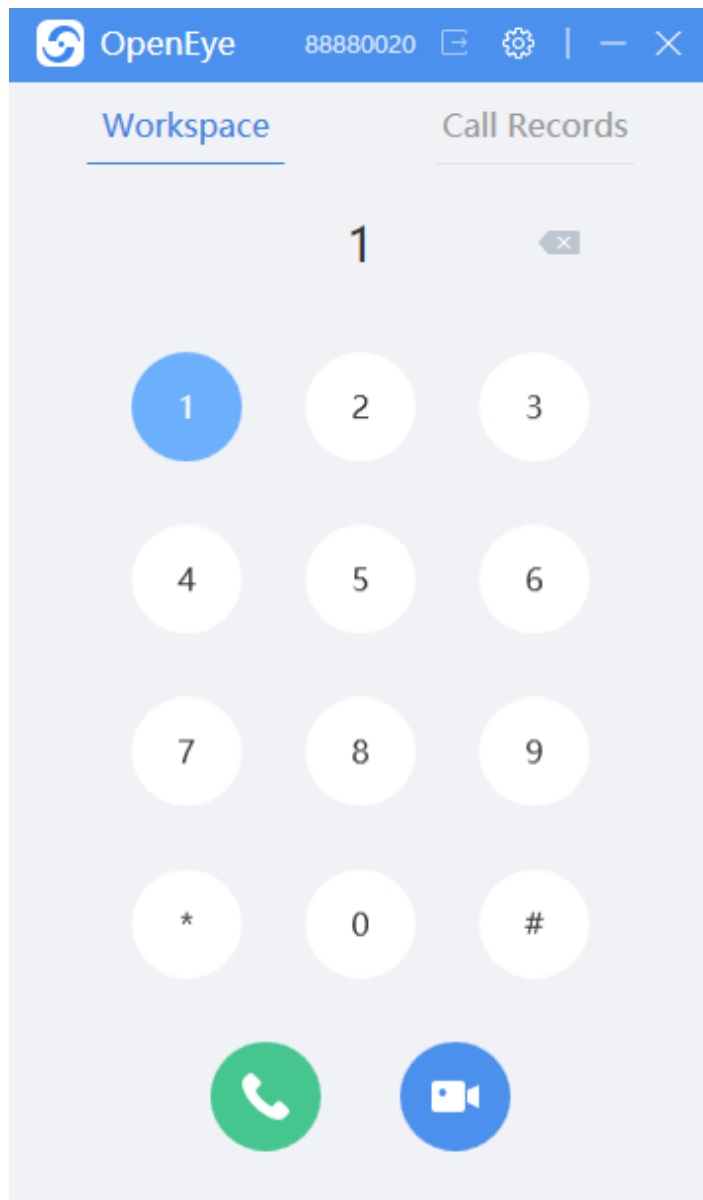
Step 2 Double-click the OpenEye icon and click . The **Settings** dialog box is displayed. In **Server**, select a server such as **LoginServer 0**, set the SIP server IP address and port number, and click **OK**.



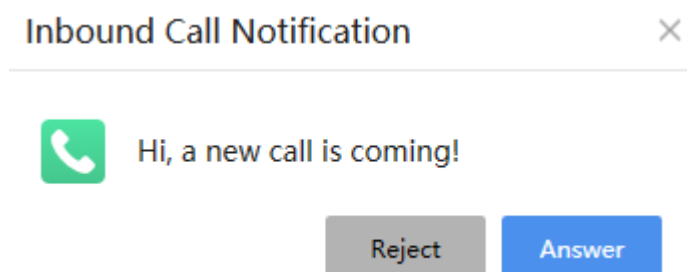
Step 3 Return to the OpenEye window, enter the softphone number and password obtained in [Table 5-4](#), and click **Log In**.



After the login is successful, the following page is displayed. In this case, you can use the lightweight connection bar to initiate calls.



In daily cases, the OpenEye is displayed in the task tray in the lower right corner of the task bar. When a new call is connected, a message is displayed.



After the call is connected, the integration is successful.

----End

 **NOTE**

Thank you for choosing our products. If you encounter any problem that cannot be solved, contact Huawei operations administrators. We will find a solution for you.