

Obtain Authorization

This page describes the authorization process of third-party Apps with VoipNow.

- [Overview](#)
 - [Access token validity & expiration](#)
- [Obtain authorization](#)
 - [Request user permission](#)
 - [Use trusted apps](#)
- [Access token management](#)
 - [Refresh the access token](#)
 - [App De-authorization](#)
- [Using an Access Token](#)

Overview

Voipnow APIs use the [OAuth 2.0 protocol](#) for the authentication and authorization of your App. Identification takes the form of an [OAuth 2.0](#) access token.

The App is required to use TLS for any of the requests described below.

Access token validity & expiration

Access tokens obtained from VoipNow become valid as soon as they are received and can be used in API requests. Access tokens are only valid for 2 hours. After 2 hours, the access token is expired and the App must request another token or [refresh it](#).

Obtain authorization

Access tokens are obtained using one of the flows described below:

Request user permission

Step 1

The App redirects the user to the server authorization endpoint. The App must make a HTTP POST or GET request to the authorization endpoint located at **<https://<hostname>/oauth/authorize.php>**. When making the request, the header Content-Type must be set to application/x-www-form-urlencoded and the body must contain the Request Parameters listed in the table below:

Name	Required	Description
response_type	Yes	Always set to code .
redirect_uri	Yes	URI where the system sends the response. Must be URL encoded if it's sent in the query parameters.
client_id	Yes	App Key as generated by the system at registration time .
state	Yes	Parameter to be used by the App to verify if the response received from the system is valid. Should be a random string of minimum 16 characters.

The following example shows how to make a POST request to the authorization endpoint:

HTTP Request
<pre>POST /oauth/authorize.php Host: <VoipNowHostname> Content-Type: application/x-www-form-urlencoded response_type=code&redirect_uri=https://<hostname>/app/redirecturi/&client_id=5~2wKMPg9h~GExN3s01-7wX2XmLI_Xbz&state=appstate</pre>

We strongly advise you to validate/invalidate the response received from the system, if the state does not correspond to the one initially sent.

Step 2

An App cannot obtain the authorization without the user's permission. Once the App makes the request, the user receives the following form:



Permission Request

The application is requesting permission to access your resources

Username *

Password *

[Forgot your password?](#)

[Deny](#) [Allow](#)

Step 3

The user permits or forbids the App to access their resources. To grant access to an App, the user must enter their credentials and click the **Allow** button. The user can also validate their credentials using the account of a third-party application (e.g Google).

To deny the authorization of the App, the user must click the **Deny** button.

Step 4

The App receives an authorization code. If the user has been granted access to the App, the system will redirect them to the URI specified in the `redirect_uri` parameter. The system uses the HTTP GET method to make the request to the App's endpoint:

```
GET /app/redirect/endpoint/?code=632848d4033835dba1232cb5983ac971e51a214925bcbcad2f601a2a2c62009f&state=appstate
Host: <AppHostname>
Content-Type: application/x-www-form-urlencoded
```

The authorization **code** received has a 10-minute lifetime.

Step 5

The App must request an access token. Using the authorization code received in the previous step, the App must make a HTTP POST request to the token endpoint located at <https://<hostname>/oauth/token.php>. When making the request, the header Content-Type must be set to `application/x-www-form-urlencoded` and the body must contain the Request Parameters listed in the table below:

Name	Required	Description
grant_type	Yes	Always set to authorization_code
code	Yes	The code received in STEP 4 .
redirect_uri	Yes	URI where the system returns the response.
client_id	Yes	App Key as generated by the system at registration time .
client_secret	Yes	App Secret as generated by the system at registration time . Can be missing if sent in the Authorization header.

The following example shows how to make a POST request to the token endpoint:

HTTP Request

```
POST /oauth/token.php
Host: <VoipNowHostname>
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&code=632848d4033835dba1232cb5983ac971e51a214925bcbcad2f601a2a2c62009&redirect_uri=
https://<hostname>/app/redirecturi/&client_id=5~2wKMPg9h~GExN3s01~7wX2XmLI_Xbz&client_secret=Q-
jxXg900X_mCpXvLfw.V12X3NQv-nc5&state=appstate
```

Client authentication is performed as described in the OAuth standard: <https://tools.ietf.org/html/rfc6749#section-2.3>. This means that the `client_id` and `client_secret` parameters can be sent in the Authorization header or in the body of the request.

Step 6

Assuming that the code is still valid and the operation is successful, the App receives the following response:

HTTP Response

```
HTTP 1.1 200 OK
...

{
  "access_token": "1|5~2wKMPg9h~GExN3s01~7wX2XmLI_Xbz|1|1345716093|O_XQYdHR0P-xMvqbVsh_OwRH7GT4.FtR",
  "expires_in": 7200,
  "token_type": "bearer",
  "refresh_token": "9_s2TBCQ1y.PPzVNXkT-Gff6tB9z_bqr"
}
```

The response parameters are:

- **access_token**: Value of the access token. This value is used when making requests to APIs.
- **expires_in**: Period of time, in seconds, during which the token is valid.
- **token_type**: The type of the token. Only Bearer is possible.
- **refresh_token**: Refresh token that can be used to [regenerate the token](#) once expired.
- **device_id**: The ID of the device associated with the token.

When the token expires, the App can use the [refresh token](#) to generate another `access_token` or repeat the steps described above.

Use trusted apps

We advise you to use this flow only if you trust the App requesting authorization.

Step 1

The App requests an `access_token`. It makes a HTTP POST request to the token endpoint located at <https://<hostname>/oauth/token.php>. The request URI is made using the following parameters in the body and it uses the `application/x-www-form-urlencoded` format. The Request Parameters are listed in the table below:

Name	Required	Description
grant_type	Yes	Always set to client_credentials
client_id	Yes	App Key as generated by the system at registration time .
client_secret	Yes	App Secret as generated by the system at registration time . Can be missing if it's sent in the Authorization header.

Client authentication is performed as described in the OAuth standard: <https://tools.ietf.org/html/rfc6749#section-2.3>. This means that the `client_id` and `client_secret` parameters can be sent in the Authorization header or in the body of the request.

Step 2

App receives the access token. The response is similar to the [response received in the previous flow](#). The only difference is that no `refresh_token` is generated. When the token expires, the App must request to authorize with the system again, by repeating the step above.

Access token management

Refresh the access token

Once a token has expired, the App must generate a new one in order to access the system's resources. If the token was initially generated using the [User Permission](#) flow, you can refresh it using the refresh_token obtained. To do this you must follow the steps below:

Step 1

The App requests an access_token. It makes a HTTP POST request to the token endpoint located at **https://<hostname>/oauth/token.php**. The request URI is made using the following parameters in the body and it uses the application/x-www-form-urlencoded format.

Name	Required	Description
grant_type	Yes	Always set to refresh_token
client_id	Yes	App Key as generated by the system at registration time .
client_secret	Yes	App Secret as generated by the system at registration time . Can be missing if it's sent in the Authorization header.
refresh_token	Yes	The refresh token received in STEP 6 (The Request User Permission Section).

Client authentication is performed as described in the OAuth standard: <https://tools.ietf.org/html/rfc6749#section-2.3>. This means that the client_id and client_secret parameters can be sent in the Authorization header or in the body of the request.

Step 2

The App receives the access_token. The response is similar to the one received when using the [User Permission](#) flow. When this request is sent, the system invalidates the current refresh_token and returns a new one. The new refresh_token must be saved by the App.

App De-authorization

At any time, the user can remove the authorization granted to the App following the recommendations in the [Apps Management](#) section.

Using an Access Token

When making requests using one of the APIs, you can use access_tokens. For more details, check out the authentication section of the [UnifiedAPI](#) or [SystemAPI](#) documentation.