

JWT Auth Playground

[View JWT Auth Guides](#) 

APEX Cloud supports **JWT Auth** to secure APIs transactions. Dive into each step of the Authorization flow and learn how to handle each component.

1

Generate JWKS and create application

2

Subscribe your application to an API and generate API keys for your application

3

Construct JWT payload and generate JWT token

Checklist

⏪ Start from the beginning

- Generate JWKS
- Create application with generated JWKS
- Subscribed** my application to the API I want to transact with
- Generated** an API key to allow my application to be recognized by the API Gateway
- Generated** header for the JWT token
- Populated** `iat` of the JWT payload

Change all actions in the checklist to present tense. First two are correct.

Might help to emphasise these parameters/claims (.e.g **bold** or with `code font`)

- Populated `exp` of the JWT payload
- Populated `iti` of the JWT payload
- Populated `iss` of the JWT payload
- Populated `aud` of the JWT payload
- Populated `sub` of the JWT payload
- Populated `data` of the JWT payload
- Generated the JWT token

Change all actions in the checklist to present tense.

Broke up the paragraphs for readability:

To begin **the flow**, you'll need to generate a JWKS first.

The JSON Web Key Set (JWKS) is a set of keys containing the public keys used to verify any JSON Web Token (JWT) issued by the Authorization Server.

For this demo, you may generate a random JWKS below, and we will store the corresponding private key (that will be used in the later steps) in your browser's local storage.

In the actual client registration, instead of specifying the JWKS in JSON form, you are expected to host the JWKS in an endpoint of your choice.

1. Generate JWKS and create application

To begin with the flow, you'll need to generate a JWKS first.

The JSON Web Key Set (JWKS) is a set of keys containing the public keys used to verify any JSON Web Token (JWT) issued by the Authorization Server. For this demo, you may generate a random JWKS below, and we will store the corresponding private key (that will be used in the later steps) in your browser's local storage.


In the actual client registration, instead of specifying the JWKS in JSON form, you are expected to host the JWKS in an endpoint of your choice.

JWKS

-

 Generate Random JWKS

You may refer to [our guide on creating an application with a JWKS](#).

With the JWKS generated, you need to create an application and link it to the generated JWKS. You may refer to our guides on [creating application with JWKS](#) .

For the purposes of this demo, we don't require that you create an actual application. Instead, you can use the demo application below that will be stored in your browser's local storage.

Application Name

✓ Register Demo Application

2. Subscribe your application to an API and generate API keys for your application

Now that you have an application with JWKS attached, you may continue the JWT Auth development process by subscribing the application to an JWT Auth protected API.

This will be the APIs in the APEX API gateway that will require JWT token generated before sending requests.

Application Name

My JWT Auth Playground App

Subscribe to Sample API

Now that you have an application with a JWKS attached, you may continue the JWT Auth process by subscribing your application to a JWT Auth-protected API.

These JWT Auth-protected APIs are the APEX Gateway APIs that require generating a JWT token before sending requests.

3. Construct JWT payload and generate JWT token

With the setup so far, now you are ready to generate JWT token that is required when making API calls.

These are the important values we generated previously, which will be used when generating the JWT token.

API Key


0969cf40-e558-4574-b462-42eafc85b906

JWKS

```
1  {
2    "keys": [
3      {
4        "kty": "EC",
5        "kid": "2y7fhJozT3M3SH2upJ6HYd-QI6u463VPAQZhb591hY",
6        "use": "sig",
7        "alg": "ES256",
8        "crv": "P-256",
9        "x": "a_ywnDnGWQGMrioKfzjbaXlTgQVjjN0Ig20J9kL1Yas",
10       "y": "ceeWwVUndaxTRDa4uwfaa04sRmHXu03lbwTwWlU3KjQ"
11     }
12   ]
13 }
```

With the steps completed so far, you are now all set to generate a required JWT token for making API calls.

The API Key and JWKS values from the previous steps will now be used to generate the JWT Token.

JWT stands for JSON Web Token and generally consists of 3 sections: **Header**, **Payload** and **Signature**. You can refer to our guide for details of the [required JWT claims](#) .

Header of JWT used by APEX contains 3 claims: `alg`, `typ` and `kid`

`alg` is fixed with value "ES256" and `typ` is fixed with value "JWT".

`kid` needs to be filled with value of the exact `kid` field from your JWKS key.

JWT Header

-

✓ Generate JWT Header

JWT stands for JSON Web Token It generally consists of 3 sections: Header, Payload, and Signature. You can refer to our guide for more details on the required JWT claims.

The APEX JWT Header contains 3 claims: `alg`, `typ`, and `kid`

- `alg` is a fixed value of "ES256"
- `typ` is a fixed value of "JWT".
- `kid` should have the same value as the `kid` from your JWKS key.

Next you will need to fill up fields of the JWT Payload with values. We will fill them one by one in this demo, and you can preview the generated payload below after each segment by selecting the "Inspect JWT Payload" button.

iat means **issue at**, which represents the time (Unix epoch time in seconds) when the JWT token is generated.

iat

✓ Populate iat value

🔍 Inspect the JWT Payload

exp means **expiry**, which represents the time (Unix epoch time in seconds) when the JWT token is expired. The maximum value of the field is 180s from **iat**.

exp

✓ Populate exp value

🔍 Inspect the JWT Payload

Next, you need to fill out the JWT Payload fields step by step in this demo. You can preview the generated payload after each segment by selecting the "Inspect JWT Payload" button.

iat means **issue at**, which represents the time Unix epoch time in seconds when the JWT token was generated.

exp means **expiry**, which represents the Unix epoch time in seconds when the JWT token expires. The maximum value of the field is equal to 180 seconds from the iat value.

iss means **issuer**, which represents the issuer of this JWT token. For APEX Cloud, the value of this field will be the API key(s) of your application (comma-separated if you have multiple API keys).

iss

✓ Populate iss value

🔍 Inspect the JWT Payload

aud means **audience**, which represents the recipients of this JWT token intended for. For APEX Cloud, the value of this field will be the API endpoint you are calling.

aud

✓ Populate aud value

🔍 Inspect the JWT Payload

iss means **issuer**, which represents the issuer of the JWT token. For APEX Cloud, the value of this field will be the API key(s) generated for your application. If you have multiple API keys, they should be comma-separated.


aud means **audience**, which represents the recipients of the JWT token. For APEX Cloud, the value of this field will be the API endpoint that you are calling.

sub means **subject**. For APEX Cloud, the value of this field will be the method of the API.

sub

✓ Populate sub value

🔍 Inspect the JWT Payload

data is the SHA-256 hash of your **request body**. Do note that the payload has to be standardized before computing the hash. For detailed explanation, please refer to our guide for [API payload hash](#) .

For the purpose of this demo, please try with JSON for the request body.

Request Body **JSON**

```
1 {"payload": "data"}
```

sub means **subject**. For APEX Cloud, the value of this field will be the API method.

data is the SHA-256 hash of your **request body**. Do note that the payload has to be standardized before computing the hash. For a more detailed explanation, please refer to our [API payload hash guide](#).

For the purpose of this demo, try adding a request body with JSON.

Now you can use your favorite JWT library to generate the JWT token with the prepared header and payload

Now you can use your favorite JWT library to generate the JWT token using the header and payload values.



Congratulations!

You have successfully completed the entire Authorization flow! With your newly generated JWT token, you can try calling the demo API that you have subscribed earlier in the tutorial to conclude this tutorial.

[▶ Try out with Subscribed API](#)

You've successfully completed the entire authorization flow. With your newly generated JWT token, you can make a call to the demo API you subscribed to earlier in this tutorial.