

# **Salesforce OAuth 2.0 JWT Bearer Flow for Server-to-Server Integration**

**Prepared By**

Florentino Aryo Widyantoro,

Technical Consultant



**PT METRODATA ELECTRONICS**

September 2021

## Table of Contents

1. Create Digital Signature .....	3
2. Upload the certificate file to Salesforce's Connected App .....	4
3. Generate Json Web Token (JWT) .....	5
4. Request access token.....	7
5. Using access token .....	9
Reference: .....	11

Sometimes you want to authorize servers to access data without interactively logging in each time the servers exchange information. For these cases, you can use the OAuth 2.0 JSON Web Token (JWT) bearer flow. This flow uses a certificate to sign the JWT request and doesn't require explicit user interaction. With the OAuth 2.0 JWT bearer token flow, the client posts a JWT to the Salesforce OAuth token endpoint. Salesforce processes the JWT, which includes a digital signature, and issues an access token based on prior approval of the app.

Let's go over each step of this authorization flow.

## 1. Create Digital Signature

We will use OpenSSL to generate digital signature.

- Download OpenSSL (e.g. from <https://www.softpedia.com/get/Programming/Components-Libraries/OpenSSL.shtml#download>)
- Install OpenSSL then run start.bat
- Generate RSA private key:  
`openssl genrsa -des3 -passout pass:Your_Password -out APICert.pass.key 2048`  
replace password using any string password.  
replace APICert using any string name you want.
- Create RSA key:  
`openssl rsa -passin pass:Your_Password -in APICert.pass.key -out APICert.key`
- Generate Certificate Signing Request  
`openssl req -new -key APICert.key -out APICert.csr`  
Enter some values in Distinguished Name (DN), you also can leave some field blank (some field has its own default values)





```
C:\Users\flores>openssl req -new -key APICert.key -out APICert.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:id
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:metrodata
```

- Create certificate file:

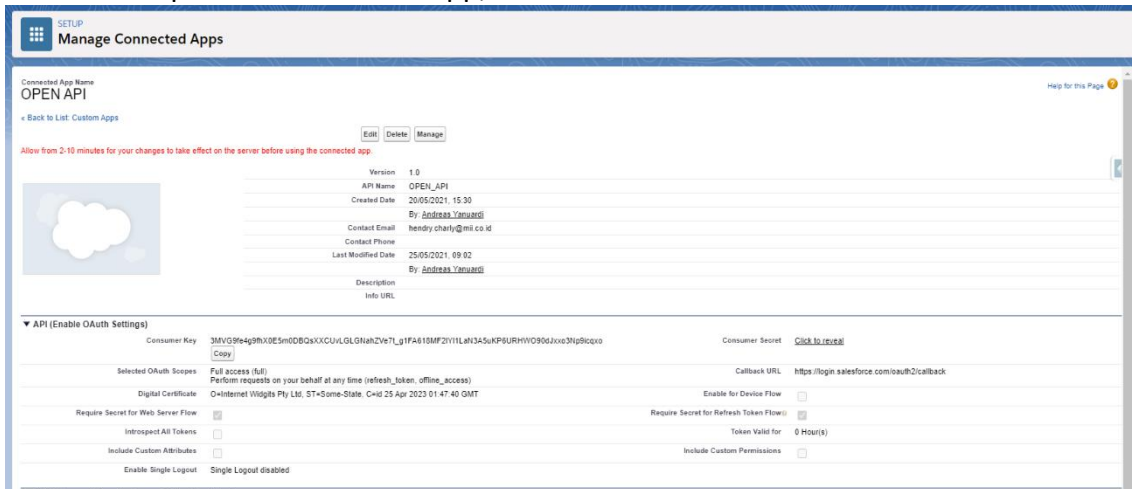
```
openssl x509 -req -sha256 -days 365 -in APICert.csr -signkey APICert.key -out APICert.crt
```

-day 365 means certificate will be expired after 356 days, define how long your certification valid by changing the number

 APICert	25/05/2021 8:47	Security Certificate	2 KB
 APICert.csr	25/05/2021 8:46	CSR File	2 KB
 APICert	25/05/2021 8:38	key	2 KB
 APICert.pass	25/05/2021 8:37	key	2 KB

## 2. Upload the certificate file to Salesforce Connected App

- Login to Salesforce then navigate Setup -> Apps -> App Manager.
- Find a prebuild connected app, click Edit.



The screenshot shows the 'Manage Connected Apps' page in Salesforce Setup. The selected app is 'OPEN API'. The page displays various settings for the app, including its version (1.0), API name (OPEN\_API), and creation date (20/05/2021, 15:30). It also shows contact information for 'Andreas Yamasaki'. The 'API (Enable OAuth Settings)' section is expanded, showing the Consumer Key, Consumer Secret, and various OAuth settings like 'Selected OAuth Scopes' (Full access (Full)), 'Digital Certificate' (O=Internet Widgits Pty Ltd), and 'Require Secret for Web Server Flow' (checked). The 'Callback URL' is set to 'https://login.salesforce.com/oauth2/callback'. Other settings like 'Enable for Device Flow', 'Require Secret for Refresh Token Flow', 'Token Valid for', and 'Include Custom Permissions' are also visible.

- Below 'Use digital signature', click 'Choose File' then upload .crt file generated before.

### 3. Generate Json Web Token (JWT)

This step should be done programmatically, but we will simulate JWT generation using <https://jwt.io>.

- Construct a JWT header with this format: {"alg":"RS256"}
- Construct a JSON Claims Set for the JWT with the following parameters.

PARAMETER	DESCRIPTION
iss	The issuer must contain the OAuth <code>client_id</code> or the connected app (Consumer Key, see #2) for which you registered the certificate.
aud	The audience identifies the authorization server as an intended audience. The authorization server must verify that it's an intended audience for the token. Use the authorization server's URL for the audience value: <code>https://login.salesforce.com</code>
sub	The subject must contain the username of the user if implementing for an Experience Cloud site. For backward compatibility, you can use principal ( <code>prn</code> ) instead of subject ( <code>sub</code> ). If both are specified, <code>prn</code> is used.
exp	The validity must be the expiration time of the assertion within 3 minutes, expressed as the number of seconds from 1970-01-01T0:0:0Z measured in UTC.

Here's an example JSON Claim Set for the JWT.

```
{  
  "iss": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
xxxxxx5uKP6URHW090dJxxo3Np9icqxo",  
  "sub": "salesforce_username",  
  "aud": "https://login.salesforce.com",  
  "exp": "1621804579"  
}
```

- Paste .crt file to Certificate and .key file to private key in Verify Signature

eyJhbGciOiJSUzI1NiJ9.eyJpc3MiOiIzTVZHOWZlNGc5ZmhmYUe1bTBEQ1FzZWfhDVXZMR0xHTmFoWzlnN3RfZzFGQTYxOE1GMmxZbDFMYU4zQTV1S1A2VVJIV085MGRKEhVhM05wOWljcXhvIiwic3ViIjoiaYWRTaW5kc2FAdGVSa29tLmNvLmIkIiwiaXVkiJioiaHR0cHM6Ly9sb2dpci5zYWxlczZvcmlmNvbiSImV4cCI6IjE2MjE4MDQ1Nzkiqf.VNW-\_4v5htRtpBkgTXXS1CaD7lpADC0i9rcmXWbVfkdsADgWb8\_HymVghBwuCDrUguDz\_yzDnCojV3Y8Fa1CAyILvBDZTnrWbqbo6q5KutWMtJojbc7Im77HAZnuwc-hhQ1yjVBic01b2XAqIX19jZqsPu--rs2H8-V6mAXND\_\_6ivPDJDkHjwS8qd-\_uZi4hXWuf0BRCMDwjthgMZVDQJ0mVsNX8P5DC7\_Izsv3wVbgWuFVTRY50HAGbPgruI7AxKKEJafIYARBu5Op1RzpehDmBy\_7UyWhbu04Yrz4z444Fty6Cy3XDR112JPCqhsb3e5UaRa5v16Mv6\_14BIA

HEADER: ALGORITHM & TOKEN TYPE

{ "alg": "RS256" }

PAYLOAD: DATA

F21Y11LaN3A5uKP6URHW090dJxxo3Np9icqxo",  
"sub": "admins@telkom.co.id",  
"aud": "https://login.salesforce.com",  
"exp": "1621804579"  
}

VERIFY SIGNATURE

RSASHA256(  
base64UrlEncode(header) + "." +  
base64UrlEncode(payload),  
-----BEGIN CERTIFICATE-----  
MIDHDTCCAgUCFC5EK2HXVUoTK//  
LRVKS9GSzXkuMA0GCSqGSIb3DQEBA  
CwUAMEsx  
-----BEGIN RSA PRIVATE KEY--  
---  
MIIEpAIBAAKCAQEAxUPbXHv1uonj  
8hChC3AHQUY7jGmzzRwH7nESiH2y  
M5qYi669

- JWT generated

This Java code is a simple example of constructing a JWT bearer token.

```
import org.apache.commons.codec.binary.Base64;
import java.io.*;
import java.security.*;
import java.text.MessageFormat;

public class JWTEExample {

    public static void main(String[] args) {

        String header = "{\"alg\":\"RS256\"}";
        String claimTemplate = "'{'iss\": \"{0}\", \"sub\": \"{1}\", \"aud\": \"{2}\", \"exp\": \"{3}\", \"jti\": \"{4}\"}'";

        try {
            StringBuffer token = new StringBuffer();

            //Encode the JWT Header and add it to our string to sign
            token.append(Base64.encodeBase64URLSafeString(header.getBytes("UTF-8")));

            //Separate with a period
            token.append(".");

            //Create the JWT Claims Object
            String[] claimArray = new String[4];
            claimArray[0] = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxJxox3Np9icqxo";
```

```

        claimArray[1] = "admin@mii.co.id";
        claimArray[2] = "https://login.salesforce.com";
        claimArray[3] = Long.toString( ( System.currentTimeMillis()/1000 ) + 300);
        claimArray[4]=<JTI>
        MessageFormat claims;
        claims = new MessageFormat(claimTemplate);
        String payload = claims.format(claimArray);

        //Add the encoded claims object
        token.append(Base64.encodeBase64URLSafeString(payload.getBytes("UTF-8")));

        //Load the private key from a keystore
        KeyStore keystore = KeyStore.getInstance("JKS");
        keystore.load(new FileInputStream("./path/to/keystore.jks"), "keystorepassword".toCharArray());
        PrivateKey privateKey = (PrivateKey) keystore.getKey("certalias", "privatekeypassword".toCharArray());

        //Sign the JWT Header + "." + JWT Claims Object
        Signature signature = Signature.getInstance("SHA256withRSA");
        signature.initSign(privateKey);
        signature.update(token.toString().getBytes("UTF-8"));
        String signedPayload = Base64.encodeBase64URLSafeString(signature.sign());

        //Separate with a period
        token.append(".");

        //Add the encoded signature
        token.append(signedPayload);

        System.out.println(token.toString());

    } catch (Exception e) {
        e.printStackTrace();
    }
}
}

```

## 4. Request access token

To request an access token, the connected app posts a token request to the Salesforce instance's token endpoint. It includes the JWT in the POST request.

```

POST /services/oauth2/token HTTP/1.1
Host: login.salesforce.com
Header:
Content-Type: application/x-www-form-urlencoded

Body:
grant_type= urn:ietf:params:oauth:grant-type:jwt-bearer&

```

assertion=JWT\_Bearer

## Result example using postman:

POST ▼ https://login.salesforce.com/services/oauth2/token ... Send ▼

Params Authorization Headers (10) **Body** ● Pre-request Script Tests ● Settings Cookies

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL

<input checked="" type="checkbox"/>	grant_type	urn:ietf:params:oauth:grant-type:jwt-bearer	
<input checked="" type="checkbox"/>	assertion	eyJhbGciOiJSUzI1Ni9.eyJpc3MiOiIzTVZHOWZINGc5ZS...	
	Key	Value	Description

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 1079 ms Size: 710 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ 🔍

```
1 {
2   "access_token": "00D5g000004DkS9!ARIAQOy77lHw7PeXJ13XpXtY2nbfcz8Jtk1rNpZS45HIAVpez.t4wMlyYf9Xcjc38W60ZS7wKDDVmw_ZLVzKrj5HK2amfee",
3   "scope": "full",
4   "instance_url": "https://telkom.my.salesforce.com",
5   "id": "https://login.salesforce.com/id/00D5g000004DkS9EAK/0055g000002p203AAI",
6   "token_type": "Bearer"
7 }
```



## 5. Using access token

Let's try fetching Case records

endpoint:	https://<Instance_Url>/services/data/v48.0/query/?q=<SOQL_Query>
For SOQL query replace space(' ') with plus ('+') then encode to Url format	
SOQL Query:	<div>to:</div> <div>SELECT id,field1_API_Name,field2_API_Name from Object_Name WHERE field_API_Name = 'Keyword'</div> <div>SELECT+id,field1_API_Name,field2_API_Name+from+Object_Name+WHERE+fieldparameters_API_Name+=+ 'Keyword'</div>
method:	GET
header:	Authorization: Bearer <access_token>
	Content-Type: application/json
Example for Case object:	
endpoint:	https://<Instance_Url>/services/data/v48.0/query/?q=select+id,+CaseNumber,+subject,+category__c,+status,+origin,+CreateDate,+ClosedDate,+Sub_Category__c,+Product__c,+contact.firstname,+contact.lastname,+owner.firstname+from+case+where+Product__c='Pedulilindungi'

Postman result example (first 2000 records)

The screenshot shows a Postman interface with a GET request to the endpoint: `{{urlprefix}}/services/data/v48.0/query/?q=select+id,+CaseNumber,+subject,+category__c,+status,+origin,+CreateDate,+ClosedDate,+Sub_Category__c,+Product__c,+contact.firstname,+contact.lastname,+owner.firstname+from+case+where+Product__c='Pedulilindungi'`. The authorization is set to Bearer Token with the token `{{access_token}}`. The response status is 200 OK, and the body is displayed in JSON format. The JSON response includes a `totalSize` of 11718, a `done` flag set to false, and a `nextRecordsUrl` for fetching the next 2000 records. The first record in the `records` array is a Case object with the following details:

- `attributes`:
  - `type`: "Case"
  - `url`: `"/services/data/v48.0/objects/Case/5005g000002a50yAAA"`
- `Id`: "5005g000002a50yAAA"
- `CaseNumber`: "00001450"
- `Subject`: "Sertifikat vaksin"
- `Category__c`: "Complaint"
- `Status`: "Closed"

Use nextRecordsUrl to fetch next 2000 records

endpoint:	https://<Instance_Url><nextRecordsUrl>
method:	GET
header:	Authorization: Bearer <access_token>
	Content-Type: application/json

Telkom Demo / nextRecordsUrl

GET

{{urlprefix}}/services/data/v48.0/query/01g5g0000032koRAAQ-2000

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Type

Bearer Token

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [Learn more about variables](#)

Token

{{access\_token}}

Body

Cookies (1)

Headers (11)

Test Results

Status: 200 OK

Time: 956 ms

Size: 1.32 MB

Save Response

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

"totalSize": 11718,

"done": false,

"nextRecordsUrl": "/services/data/v48.0/query/01g5g0000032koRAAQ-4000",

"records": [

{

"attributes": {

"type": "Case",

"url": "/services/data/v48.0/objects/Case/5005g000003IhbrAAC"

},

"Id": "5005g000003IhbrAAC",

"CaseNumber": "00029670",

"Subject": "New message received at pedulilindungi@mail.kominfo.go.id",

"Category\_\_c": "Non Produk",

"Status": "Closed",

"Origin": "Email",

"CreateDate": "2021-03-23T15:46:15.000+0000",

"ClosedDate": "2021-03-29T09:07:32.000+0000",

"Sub\_Category\_\_c": "Notifikasi Email",

}

**Reference:**

[https://developer.salesforce.com/docs/atlas.en-us.soql\\_sosl.meta/soql\\_sosl/sforce\\_api\\_calls\\_soql\\_select.htm](https://developer.salesforce.com/docs/atlas.en-us.soql_sosl.meta/soql_sosl/sforce_api_calls_soql_select.htm)

<https://www.openssl.org/docs/man1.1.1/man1/>

[https://help.salesforce.com/articleView?id=sf.remoteaccess\\_oauth\\_jwt\\_flow.htm&type=5](https://help.salesforce.com/articleView?id=sf.remoteaccess_oauth_jwt_flow.htm&type=5)

[https://developer.salesforce.com/docs/atlas.en-us.api\\_rest.meta/api\\_rest/resources\\_list.htm](https://developer.salesforce.com/docs/atlas.en-us.api_rest.meta/api_rest/resources_list.htm)