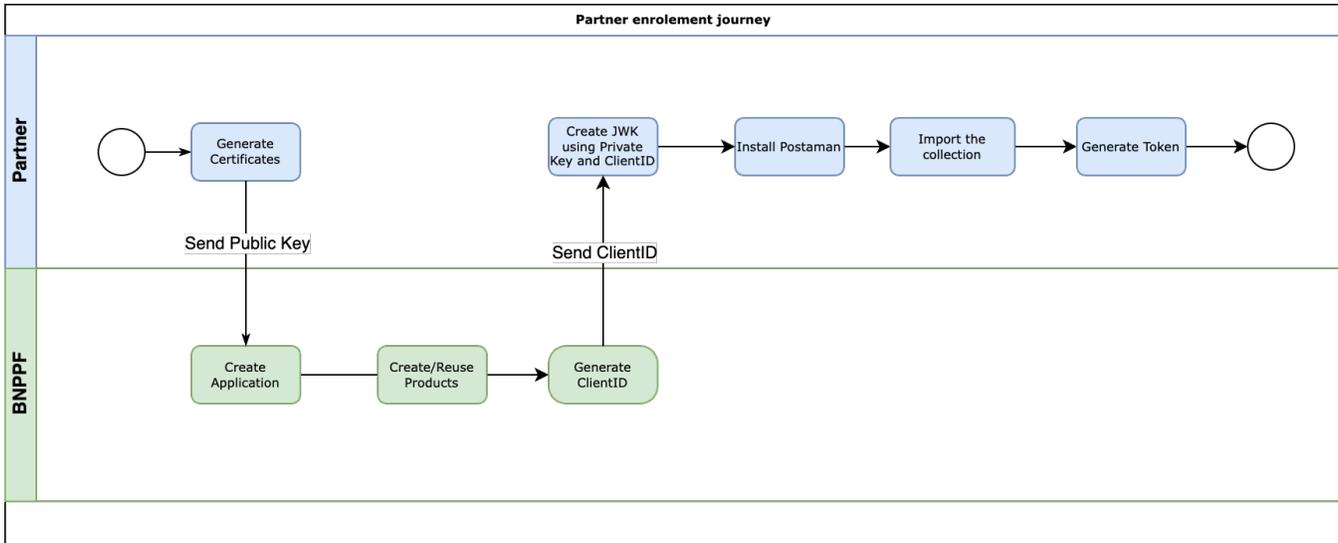


Apigee partner enrolment journey OAuth2



- How to create the client Id & transmit the certificate
 - Step 1 : Use Openssl to create the private key and the certificate
 - Step 2 : Transform your private key to JWK
 - Option 1
 - Option 2
- How to request an access token
 - Step 1 : Install Postman
 - Step 2 : Create a postman global variable
 - Step 3 : Create the postman Body request
 - Step 4 : Store the access_token as a variable to use it as authorization in the APIs call



• How to create the client Id & transmit the certificate

Environement	APIGEE URLs
Sandbox	apis-ce.staging.bnpparibas-pf.com
Staging	apis-ce.staging.bnpparibas-pf.com
Pre-Production	api-ppce.bnpparibas-pf.com
Production	api-ce.bnpparibas-pf.com

Step 1 : Use Openssl to create the private key and the certificate

Open your cmd and execute the next command

```
openssl req -x509 -newkey rsa:2048 -keyout private.key -out public.cert -days 365
```

ou should have now a certificate with this format:

```
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
```

And a private key you will use to sign the certificate with the next format

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
...
-----END ENCRYPTED PRIVATE KEY-----
```

Step 2 : Transform your private key to JWK

For the Postman collection test, you will have to transform your private key to JWK:

Option 1

Use the Python script provided below in the same directory of your certificate.

Prerequisites

Install Python if needed : <https://www.python.org/>

Add crypto lib using PIP: **pip3 install cryptography==41.0.5**

Launch the script : **python3 convert\jwk.py**

Transform private key to JWKS

```
import json
import base64
from cryptography.hazmat.primitives import serialization
from cryptography.hazmat.primitives.asymmetric import rsa
with open('private.decrypted.key', 'rb') as f:
    private_key_pem = f.read()
private_key = serialization.load_pem_private_key(private_key_pem, password=None)
n = base64.urlsafe_b64encode(int(private_key.private_numbers().public_numbers.n).to_bytes(256, byteorder='big')).rstrip(b'=').decode('utf-8')
#e = base64.urlsafe_b64encode(int(private_key.private_numbers().public_numbers.e).to_bytes(4, byteorder='big')).rstrip(b'=').decode('utf-8')
d = base64.urlsafe_b64encode(int(private_key.private_numbers().d).to_bytes(256, byteorder='big')).rstrip(b'=').decode('utf-8')
p = base64.urlsafe_b64encode(int(private_key.private_numbers().p).to_bytes(128, byteorder='big')).rstrip(b'=').decode('utf-8')
q = base64.urlsafe_b64encode(int(private_key.private_numbers().q).to_bytes(128, byteorder='big')).rstrip(b'=').decode('utf-8')
dp = base64.urlsafe_b64encode(int(private_key.private_numbers().d % (private_key.private_numbers().p - 1)).to_bytes(128, byteorder='big')).rstrip(b'=').decode('utf-8')
dq = base64.urlsafe_b64encode(int(private_key.private_numbers().d % (private_key.private_numbers().q - 1)).to_bytes(128, byteorder='big')).rstrip(b'=').decode('utf-8')
qi = base64.urlsafe_b64encode(pow(private_key.private_numbers().q, -1, private_key.private_numbers().p).to_bytes(128, byteorder='big')).rstrip(b'=').decode('utf-8')
jwk = { "kty": "RSA", "n": n, "e": "AQAB", "d": d, "p": p, "q": q, "dp": dp, "dq": dq, "qi": qi }
with open('jwk.json', 'w') as fg:
    json.dump(jwk, fg, indent=2)
print('your decrypted private key was converted to jwk and writed in jwk.json file to check your assertion')
```

Option 2

[PEM to JWK converter - Tribestream](#)

How to request an access token

Step 1 : Install Postman

Step 2: Create a postman global variable

VARIABLE	TYPE [ⓘ]	INITIAL VALUE [ⓘ]	CURRENT VALUE [ⓘ]	...	Persist All	Reset All
<input checked="" type="checkbox"/> pmlib_code	default ▼	!function(t){if("object"===typeof exports&&"unde	!function(t){if("object"===typeof exports&&"undefined"!==typeof module)module.expor			
Add a new variable						

Step 3: Create the postman Body request

POST `https://apis-ce-d.pf.staging.echonet/auth/oauth2/v2/token` Send

Params Authorization Headers (12) **Body** Pre-request Script Tests Settings Cookies

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

KEY	VALUE	DESCRIPTION	***	Bulk Edit
<input checked="" type="checkbox"/> grant_type	urn:ietf:params:oauth:grant-type:jwt-bearer			
<input checked="" type="checkbox"/> assertion	{{jwt_apigee}}			
<input checked="" type="checkbox"/> scope	read write delete			
Key	Value	Description		

Body Cookies (2) Headers (23) Test Results (1/1) Status: 200 OK Time: 121 ms Size: 1.71 KB Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "access_token": "fIvUgR9M7QsS8b2omTuTFwML35vn",
3   "token_type": "Bearer",
4   "expires_in": 599,
5   "issued": 1732291802056,
6   "scope": "read write delete",

```

```

1 curl --location --request POST 'https://
apis-ce-d.pf.staging.echonet/auth/
oauth2/v2/token' \
2 --header 'Content-Type: application/
x-www-form-urlencoded' \
3 --header 'Cookie:
BIGipServer-AP24659xPA1017-AP24659xPA1
017_apis_ce_d_pf_http_tcp_443-P_apis_c
e_d_pf_staging_echonet_http_443=23628
93322.47873.0000;
TS01b510ce=0152f841c881648b77547b03c0b
163e0d78092042f376340ba31afc90f3f675df
9578941b6f15ec7d5dbcba48d685aae990081b
85b' \
4 --data-urlencode
'grant_type=urn:ietf:params:oauth:gran
t-type:jwt-bearer' \
5 --data-urlencode
'assertion=eyJ0eXAiOiJKV1QiLCJhbGciOiJI
UzI1NiJ9.
eyJpYXQiOiJlMzIyOTc0c3MzIiwiaWF0Ijoi
15MTc5Nyw1ZjZxMzIyOTc0c3MzIiwiaXNjaW
0iOiJZcVhNEWl1XzR09LejI1Mm04bGZ018vY
Xp3c3MzIiwiaWF0IjoiMTUyOTc0c3MzIiwiaX
NjaW0iOiJZcVhNEWl1XzR09LejI1Mm04bGZ0
18vYXp3c3MzIiwiaWF0IjoiMTUyOTc0c3MzI
jZS1kLnBmLnN0YXp3c3MzIiwiaWF0IjoiMTU

```

Step 4 : Store the `access_token` as a variable to use it as authorization in the APIs call

POST `https://apis-ce-d.pf.staging.echonet/auth/oauth2/v2/token` Send

Params Authorization Headers (12) **Body** Pre-request Script Tests Settings Cookies

```

1
2 pm.test("Status code is 200", function () {
3   pm.response.to.have.status(200);
4 });
5
6 var res = pm.response.json();
7 pm.environment.set('access_token', res.access_token);
8

```