CREATION .co.uk

Third Party Provider Developer Guide Production V1.3

Contents

Version History

Version	Date	Main changes
1.0	23-09-2019	Initial Version
1.1	27-11-2019	• Updated Content-Type for TPP Registration API. • Removed Client Certificate sections 3.1.1, 4.1.1 and 5.1.1.
1.2	10-02-2020	• Added information about the CBPII API. • Added information about the statements. • Added information about manual onboarding.
1.3	01-04-2020	• Added section 3 Mutual Authentication TLS. • Removed section 8 Sandbox Integration.

1. Preface

1.1 Purpose of the document

This document describes how to use the TPP Registration API, AISP API, and the Authorisation Server of Creation Financial Services Limited.

1.2 Audience

This document is intended for third party providers that provide account information services to payment service users. It is also intended for card based payment instrument issuer (CBPII) type of third party providers.

1.3 Abbreviations and Definitions

Abbreviation	**Meaning**
AISP	Account Information Service Provider
AS	Authorisation Server
ASPSP	Account Servicing Payment Service Provider
A&T	Account and Transaction
CA	Certificate Authority
CBPII	Card Based Payment Instrument Issuer
CREATION	Creation Financial Services Limited
ETSI	European Telecommunications Standards Institute
MATLS	Mutual Authentication TLS
OB	Open Banking
OBD	Open Banking Directory
OIDC	OpenID Connect
PSU	Payment Service User
OBWAC	Open Banking Website Certificate (ETSI Certificate)
OBSEAL	Open Banking Electronic Seal Certificate (ETSI Certificate)
SSA	Software Statement Assertion
TPP	Third Party Provider
XS2A	Access to Account

This document refers to various external Open Banking documents. They also use the generic term ASPSP which stands for Account Servicing Payment Service Provider. Creation Financial Services Limited is an ASPSP.

2. Introduction

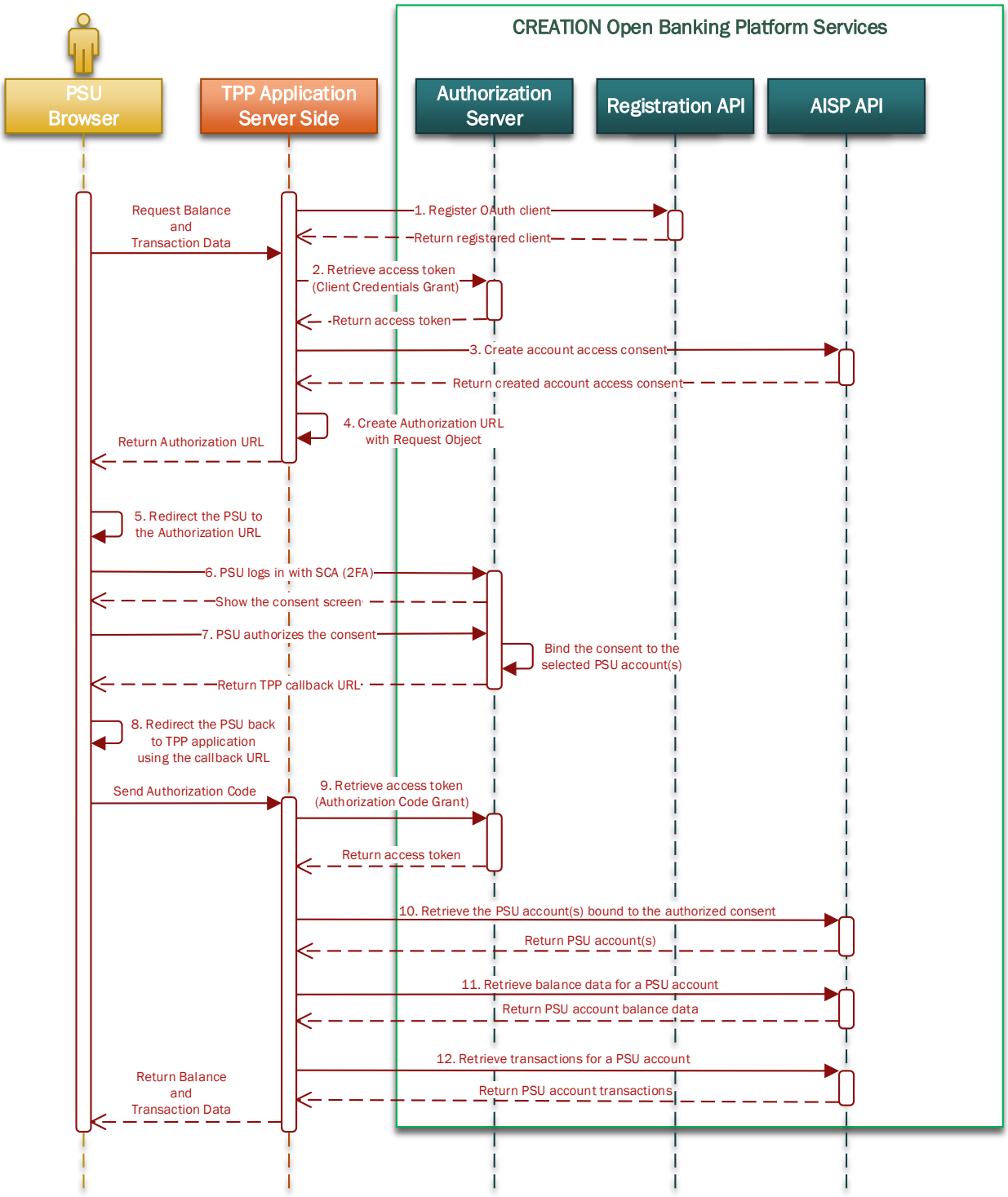
TPPs can use AISP API to retrieve information belonging to a PSU such as payment accounts, balances, and transactions. The API is secured and access tokens will need to be retrieved from the Authorisation Server. These access tokens are then used to access the resources provided by the AISP API.

The TPP application / client will need to be registered with CREATION. Please refer to section 4 Client Registration for detailed steps on how to register the TPP application / client with CREATION.

2.1 Main happy flow overview

Below is an overview of the process that takes place when a TPP application wants to access the balance and transaction information from a PSU for the first time. It serves to illustrate how the different components are used. The interaction with the components is explained in more detail in subsequent sections. You'll find below the textual and the visual overview of the process.

1) The TPP registers an OAuth client with CREATION using the Registration API.
2) The TPP application retrieves a Client Credentials Grant access token using the Authorisation Server.
3) The TPP application creates an account access consent with this access token using the AISP API.
4) The TPP application creates an Authorisation URL with Request Object that references the consent.
5) The TPP application in the PSU browser redirects the PSU to the Authorisation URL of the Authorisation Server.
6) The PSU logs in with SCA (2FA).
7) The PSU authorizes the consent, and the Authorisation Server binds the consent to the PSU's account(s).
8) The Authorisation server returns the TPP callback URL so that the PSU browser can redirect the PSU back to the TPP application.
9) The TPP application exchanges the authorisation code from the callback URL for an Authorisation Code Grant access token using the Authorisation Server.
10) The TPP application retrieves the PSU account(s) bound to the authorized consent using the AISP API.
11) The TPP application retrieves balance information of a bound account using the AISP API.
12) The TPP application retrieves transaction information of a bound account using the AISP API.


CREATION Open Banking Platform Services
PSU Browser
TPP Application Server Side
Authorization Server
Registration API
AISP API
Request Balance and Transaction Data
1. Register OAuth client
Return registered client
2. Retrieve access token (Client Credentials Grant)
Return access token
3. Create account access consent
Return created account access consent
4. Create Authorization URL with Request Object
Return Authorization URL
5. Redirect the PSU to the Authorization URL
6. PSU logs in with SCA (2FA)
Show the consent screen
7. PSU authorizes the consent
Bind the consent to the selected PSU account(s)
Return TPP callback URL
8. Redirect the PSU back to TPP application using the callback URL
Send Authorization Code
9. Retrieve access token (Authorization Code Grant)
Return access token
10. Retrieve the PSU account(s) bound to the authorized consent
Return PSU account(s)
11. Retrieve balance data for a PSU account
Return PSU account balance data
12. Retrieve transactions for a PSU account
Return PSU account transactions
Return Balance and Transaction Data

2.2 Variables

The following variables are used in this document.

Variable Name	Value
AISP_API_ENDPOINT	https://api-psd2.creation.co.uk/open-banking/v3.1/aisp
CBPII_API_ENDPOINT	https://api-psd2.creation.co.uk/open-banking/v3.1/cbpii
CLIENT_REGISTRATION_ENDPOINT	https://api-psd2.creation.co.uk/open-banking/v3.2/tpp
AUTH_WELL_KNOWN_ENDPOINT	https://api-auth-psd2.creation.co.uk/auth/realms/psd2-cards/.well-known/openid-configuration
AUTH_ENDPOINT	https://api-auth-psd2.creation.co.uk/auth/realms/psd2-cards/protocol/openid-connect/auth
TOKEN_ENDPOINT	https://token-psd2.creation.co.uk/auth/realms/psd2-cards/protocol/openid-connect/token
JWKS_ENDPOINT	https://api-auth-psd2.creation.co.uk/auth/realms/psd2-cards/protocol/openid-connect/certs

When the variables are used in this document, they are prefixed by **$** symbol.
For instance: $TOKEN_ENDPOINT

3. Mutual Authentication TLS

All our APIs are secured by means of MATLS. This means that when performing requests to our APIs, TPPs must include their transport certificate when setting up a connection [7]. On top of MATLS being required, the transport certificate will also be checked for available PSD2 roles when tokens are requested by a TPP. If the TPP is requesting OAuth scopes for which the TPP no longer has access to, the token will not be issued.

For details on how to set up MATLS connections please refer to the documentation of your specific technology stack.

4. Client Registration

The first step a TPP should take in order to start accessing the information of a CREATION PSU is make themselves known to CREATION. In order to do this they should register their application that will access PSU information as an OAuth client using the TPP Registration API.

4.1 Endpoints

Endpoint	URL	Access Token
Register	$CLIENT_REGISTRATION_ENDPOINT/register	
	$CLIENT_REGISTRATION_ENDPOINT/register/{ClientId}	Access token retrieved using the OAuth 2.0 Client Credentials Grant. See section 5.2.

4.2 Registering a client

A client can be registered in the Authorisation Server via the TPP Registration API. This API comes with a swagger 2.0 definition file.

In order to register a client do the following:

- Make a POST call to the Register endpoint (without ClientId in the path).
- Provide the following headers:
  - Content-Type: application/jose
- Include a Dynamic Client Registration Request in the body.

A Dynamic Client Registration Request is a JSON Web Token (JWT). The TPP will need to create a JWT and sign it with the TPP's private key used for signing. The TPP **must** use PS256 as the signing algorithm. The signature will be verified using the JSON Web Key Set (JWKS) specified in the **software_jwks_endpoint** claim of the JWT representing the Software Statement Assertion (part of the Dynamic Client Registration Request).

The TPP needs to be onboarded with Open Banking in order to dynamically register a client. Refer to the Open Banking documentation on how to enrol in Open Banking [1]. Refer to section 8 for steps on how TPPs that are not onboarded with Open Banking can register a client.

The JWT is composed of the following values.

Header

Field	Value
alg	PS256
kid	The id of the public key in the JWKS that should be used to verify the signature.

Body

Field	Required	Example Value	Description
iss	Y		Identifier for the TPP. This value must be unique for each TPP registered by the issuer of the SSA. For SSAs issued by the OB Directory, this must be the software_id.
iat	Y	1551987835	The time at which the request was issued by the TPP expressed as seconds since the epoch.
exp	Y	1551995035	The time at which the request expires expressed as seconds since the epoch. CREATION will reject requests where the current time is greater than the time specified in the claim.

Field	Required	Example Value	Description
aud	Y		The audience for the request. This should be the unique identifier for CREATION as issued by the issuer of the software statement. CREATION will validate the value of the claim and reject software statements for which CREATION is not the audience.
jti	Y	91071e1f-16c5-4925-948a-dee0dcd50e7e	A unique identifier for the JWT. The value must be a UUID (v4) GUID.
client_id	N	xs2a-tpp-1	The desired client id. If not specified then it will be generated. The generated value a is a UUID (v4) GUID. The client_id must not already exist, otherwise a validation error will be returned.
redirect_uris	Y	[“https://www.example.com/callback”, “https://www.example.com/callback2”]	Registered URIs the TPP will use to interact with the APIs and Authorisation Server. If the software statement defines a master set of redirect URIs, this must match or be a subset of the redirect URIs in the SSA. Each of the URIs must adhere to the following rules: • The URI MUST use the https scheme. • The URI MUST NOT contain a host with a value of localhost. • If the redirect_uris metadata element is empty in the request, the entire contents of the software_redirect_uris element in the SSA are considered to be requested by the TPP.
token_endpoint_auth_method	Y	private_key_jwt	Authentication method used when calling the token endpoint for this client. This must be private_key_jwt. The JWKS URI will be extracted from the SSA.
grant_types	Y	[“client_credentials”, “authorisation_code”, ”refresh_token”]	The grant types enabled for the client. Possible options are • client_credentials • authorisation_code • refresh_token The grant_types must at least contain client_credentials and authorisation_code.
response_types	N	[“code id_token”]	A JSON array specifying what the TPP can request to be returned from the authorisation endpoint. The only allowed option is: • code id_token

Field	Required	Example Value	Description
			Defaults to code id_token if not specified.
software_id	N		If specified, the software_id in the request MUST match the software_id specified in the SSA.
scope	Y	accounts offline_access	Requested scopes for the client. Scope accounts must always be specified.
software_statement	Y	JWT representing SSA	JWT representation of a Software Statement Assertion. This can be obtained from the Open Banking Directory. The data model for the software statements issued by the Open Banking directory are documented as part of the Directory Specification. The SSA must be signed with PS256 algorithm.
application_type	Y	web	Application type of the client. Options are: • web • mobile
id_token_signed_response_alg	Y	PS256	The algorithm which the TPP expects AS to sign the id_token, if an id_token is returned. Also, the algorithm which the TPP expects AS to sign the access_token, if an access_token is returned. Only PS256 is allowed.
request_object_signing_alg	Y	PS256	The algorithm which the TPP will use to sign the request object which is part of the authorisation request sent to the Authorisation Server. Only PS256 is allowed.
token_endpoint_auth_signing_alg	Y	PS256	The signing algorithm used for the client_assertion when calling the token endpoint for this client. Only PS256 is allowed.

4.3 Retrieving a client

If a TPP wants to check what data they registered their client with, they should retrieve the client using the Register API.

In order to retrieve a registered TPP client do the following:

- Make a GET call to the Register endpoint.
- Provide the client_id in the request path.
- Provide the following headers:
    - Authorisation: Bearer <access token retrieved using OAuth 2.0 Client Credentials Grant>

The access token must be issued to the client with the client_id in the request path or the response will be an unauthorised error.

4.4 Updating a client

If a TPP wants to change the data that belongs to their registered client, they should update the client using the Register API.

In order to update a registered TPP client do the following:

- Make a PUT call to the Register endpoint.
- Provide the client_id in the request path.
- Provide the following headers:
    - Authorisation: Bearer <access token retrieved using OAuth 2.0 Client Credentials Grant>
    - Content-Type: application/jose
- Include a Dynamic Client Registration Request in the body.

The access token must be issued to the client with the client_id in the request path or the response will be an unauthorised error.

The JWT is composed exactly the same as when first registering a client. Note that the client_id cannot be updated.

4.5 Deleting a client

If a TPP wants to undo their registration with CREATION they should delete their client using the Register API.

In order to delete a registered TPP client do the following:

- Make a DELETE call to the Register endpoint.
- Provide the client_id in the request path.
- Provide the following headers:
    - Authorisation: Bearer <access token retrieved using OAuth 2.0 Client Credentials Grant>

The access token must be issued to the client with the client_id in the request path or the response will be an unauthorised error.

5. Authorisation Server

After registering the TPP will need to obtain access tokens from CREATION's Authorisation Server. There are two types of access tokens that can be retrieved.

- Access tokens retrieved using OAuth 2.0 Client Credentials Grant [5]. These tokens will give access to the TPP client itself as well as allow the creation and subsequent retrieval of consent resources that can be authorized by PSUs.
- Access tokens retrieved using OIDC Hybrid Flow [6]. This access token makes use of a PSU-authorized consent to give access to the PSU's account, balance and transaction information.

5.1 Endpoints

Endpoint	**URL**
OIDC .well-known	$AUTH_WELL_KNOWN_ENDPOINT
Authorisation	$AUTH_ENDPOINT
Token	$TOKEN_ENDPOINT
JWKs	$JWKS_ENDPOINT

Note: OIDC .well-known endpoint also includes the URL of the Token endpoint. If the URL does not match the Token URL mentioned in the table above, then the Token URL in the table above should be used.

Furthermore, the registration endpoint URL found within the OIDC .well-known endpoint should not be used. Instead use the Register endpoint as described in the section 4 Client Registration.

5.2 OAuth 2.0 Client Credentials Grant

The access token retrieved using the Client Credentials Grant is used for access to Account Access Consent resource which is mentioned in the AISP API section as well as retrieving, updating and deleting a TPP's registered client.

To retrieve the access token do the following.

Make a POST call to the following URL: $TOKEN_ENDPOINT

Provide the following headers:

- Content-Type: application/x-www-form-urlencoded

Provide the following fields in the body:

Field	Value
grant_type	client_credentials
scope	accounts
client_assertion_type	urn:ietf:params:oauth:client-assertion-type:jwt-bearer
client_assertion[1]	<this is a JWT signed with the TPP client's private key>

The access token will have a short lifetime.

For more information on OAuth 2.0 grants please refer to [3].

5.3 OIDC Hybrid Flow

The access token retrieved using the Hybrid Flow is used for access to PSU's Account Information resources which are mentioned in the AISP API section.

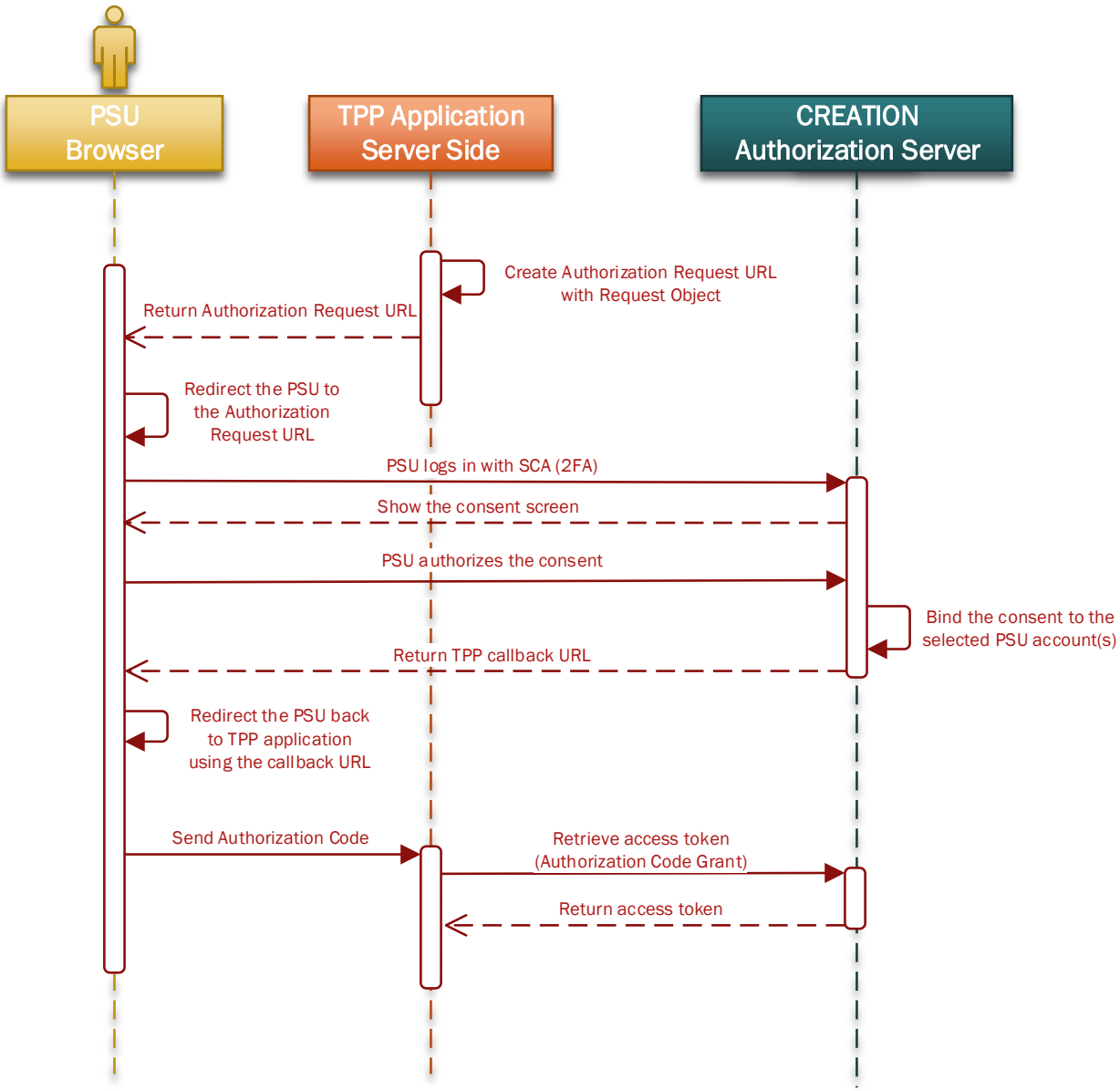
To retrieve the access token, the following steps need to be performed:

- Create an Authorisation Request
- Obtain Access Token using the Authorisation Code

In order to have full access to all account and transaction resources that the consent requests, this flow should be done at least once every 90 days. Additionally, for transactions there is a 5 minute window after the consent authorisation has been done in which the TPP can retrieve all transactions of the PSU that are known to CREATION if the account-access-consent grants the right permissions. After this 5 minute window only transactions up to 90 days in the past and up to 90 days in the future from the authorisation timestamp are available.

[1] See section 5.4 Generate Client Assertion

The below figure visualizes the steps.



5.3.1 Create an Authorisation Request

A TPP will need to create an Authorisation Request URL (using a signed JWT Request containing the Account Access Consent ID as a claim) to send the PSU to CREATION's Authorisation Server. There the PSU will authorize the consent to their account(s), or reject it. Once the consent is successful, the PSU will be redirected to the provided redirect URI of the TPP client. The TPP client will now possess the Authorisation Code and ID token. The TPP client will now introspect the ID token and use it as a detached signature to check:

- The hash of the Authorisation Code to prove it hasn't been tampered with during redirect (comparing the hash value against the c_hash attribute in ID token). See section 5.3.1.3 for more information.
- The hash of the State to prove it hasn't been tampered with during redirect (comparing the state hash value against the s_hash attribute in the ID token). See section 5.3.1.4 for more information.

Once the state and code validations have been confirmed as successful by use of the ID token, the TPP client will proceed to obtain an access token using the Authorisation Code it now possesses.

5.3.1.1 Generate the Authorisation Request URL

The URL: $AUTH_ENDPOINT?<query parameters>

Parameter	Required	Notes
response_type	Yes	The value is: **code id_token**
client_id	Yes	This is the client_id as presented during Client Registration. Example value: xs2a-tpp-1
redirect_uri	Yes	TPPs must provide the URI to which they want the resource owner's user agent to be redirected to after authorisation. This must be a valid, absolute URI that was registered during Client Registration. Example value: https://app.getpostman.com/oauth2/callback
scope	Yes	The value is: **openid accounts** It is also possible to provide value **openid accounts offline_access** This will make the refresh token in the response an offline token which is valid for 90 days and will not be refreshed itself.
state	No	TPPs may provide a state parameter. It is an opaque value used to maintain state between the request and the callback. Typically, Cross-Site Request Forgery (CSRF, XSRF) mitigation is done by cryptographically binding the value of this parameter with a browser cookie. If the parameter is provided, the Authorisation Server will play-back the value in the redirect to the TPP. s_hash will be included in the ID token if this parameter is provided. Example value: af0ifjsldkj
nonce	Yes	String value used to associate a Client session with an ID token, and to mitigate replay attacks. The value is passed through unmodified from the Authorisation Request to the ID token. Sufficient entropy must be present in the nonce values used to prevent attackers from guessing values. Example value: n-0S6_WzA2Mj

Parameter	Required	Notes
request	Yes	This is a JWT signed with the TPP client's private key. The JWT payload must consist of a JSON object containing a request object as per OIDC Core 6.1 [4]. The request object must contain a claims section that includes an ID Token as a minimum: • **openbanking_intent_id**: that identifies the intent id for which this authorisation is requested. The intent id represents the Account Access Consent ID. This resource is created using the AISP API. Refer to the AISP API section for more information. • **acr_values:** TPPs may provide a space-separated string that specifies the acr values that the Authorisation Server is being requested to use for processing this Authorisation Request. The only value present should be: - urn:openbanking:psd2:sca: To indicate that strong customer authentication must be carried out as mandated by the PSD2 RTS Value urn:openbanking:psd2:ca is permitted but ignored. Strong customer authentication will always take place. Refer to section 5.3.1.2 Generate the Request Object for more information.

5.3.1.2 Generate the Request Object

Request Object is a JSON Web Token (JWT). The TPP will need to create a JWT and sign it with the TPP's private key used for signing. Use PS256 as the signing algorithm.

The JWT is composed of the following values.

Header: Algorithm & Token Type

Field	Value
alg	PS256
kid	The id of the public key in the JWKS that should be used to verify the signature

Payload: Data

Field	Example Value	Description
iss	xs2a-tpp-1	Issuer (who created and signed this token). This is the client_id as presented during Client Registration.
aud	$TOKEN_ENDPOINT	Audience (who or what the token is intended for). This must be the URL of the Authorisation Server's Token Endpoint.
client_id	xs2a-tpp-1	The value of the same parameter in the Authorisation Request URL.
response_type	code id_token	
redirect_uri	https://app.getpostman.com/oauth2/callback	
scope	openid accounts	
max_age	3600	Maximum Authentication Age. This claim is optional. Specifies the allowable elapsed time in seconds since the last time the End-User was actively authenticated by the Authorisation Server. If the elapsed time is greater than this value, the Authorisation Server will attempt to actively re-authenticate the End-User. When max_age is used, the ID token returned will include an auth_time Claim Value.

claims	{
	"id_token": {
	"openbanking_intent_id": {
	"value": "1",
	"essential": true
	},
	"acr": {
	"essential": true,
	"values": [
	"urn:openbanking:psd2:sca"
	]
	}
	}
	}
		Includes information about the Account Access Consent ID and the PSU Authentication type.
		The openbanking_intent_id represents the Account Access Consent ID. This resource is created using the AISP API. Refer to the AISP API section for more information.

5.3.1.3 Validate Authorisation Code parameter using c_hash in ID token

c_hash represents the Code hash value. Its value is the base64url encoding of the left-most half of the hash of the octets of the ASCII representation of the code value, where the hash algorithm used is the hash algorithm used in the alg Header Parameter of the ID token's JOSE Header. For instance, if the alg is PS256, hash the code value with SHA-256, then take the left-most 128 bits and base64url encode them. The c_hash value is a case sensitive string.

5.3.1.4 Validate State parameter using s_hash in ID token

s_hash represents the State hash value. Its value is the base64url encoding of the left-most half of the hash of the octets of the ASCII representation of the state value, where the hash algorithm used is the hash algorithm used in the alg Header Parameter of the ID token's JOSE Header. For instance, if the alg is PS256, hash the code value with SHA-256, then take the left-most 128 bits and base64url encode them. The s_hash value is a case sensitive string.

5.3.2 Obtain Access Token using the Authorisation Code

The TPP client will present its Authorisation Code together with the private_key_jwt. The access token is required by the TPP client in order to access PSU's Account Information resources.

To retrieve the access token do the following.

Make a POST call to the following URL: $TOKEN_ENDPOINT

Provide the following headers:

- Content-Type: application/x-www-form-urlencoded

Provide the following fields in the body:

Field	**Value**
grant_type	authorisation_code
code	<the code value as issued to the authenticated TPP client by the Authorisation Server in the redirect / callback URI>
redirect_uri	<this value must be identical to the redirect_uri parameter value that was included in the initial Authorisation Request.>
client_assertion_type	urn:ietf:params:oauth:client-assertion-type:jwt-bearer
client_assertion[2]	<this is a JWT signed with the TPP client's private key>

The access token will have a short lifetime. In the access token response will also be a refresh token with a longer lifetime which can be used to gain a new access token. If the scope *offline_access* was provided the refresh token will be an offline token with a lifetime of 90 days.

[2] See section 5.4 Generate Client Assertion

5.4 Generate Client Assertion

Client assertion is a JSON Web Token (JWT). The TPP will need to create a JWT and sign it with the TPP's private key used for signing. Use PS256 as the signing algorithm. For more information refer to the private_key_jwt authentication method in section 9. Client Authentication of OpenID Connect Core [4].

The JWT is composed of the following values.

Header: Algorithm & Token Type

Field	Value
typ	JWT
alg	PS256
kid	The id of the public key in the JWKS that should be used to verify the signature

Payload: Data

Field	Example Value	Description
iss	xs2a-tpp-1	Issuer (who created and signed this token). This is the client_id as presented during Client Registration.
sub	xs2a-tpp-1	Subject (whom the token refers to). This is the client_id as presented during Client Registration.
aud	$TOKEN_ENDPOINT	Audience (who or what the token is intended for). This must be the URL of the Authorisation Server's Token Endpoint.
jti	91071e1f-16c5-4925-948a-dee0dcd50e7e	JWT ID (unique identifier for this token, which can be used to prevent reuse of the token)
exp	1551995035	Expiration time on or after which the token must not be accepted for processing (seconds since Unix epoch)
iat	1551987835	Time at which the JWT was issued (seconds since Unix epoch)

6. AISP API

After an access token has been acquired by the TPP they can access the AISP API to either create a new consent for a PSU to authorize or access a PSUs account, balance and transaction information.

This API comes with a swagger 2.0 definition file.

6.1 Endpoints

All AISP API resource endpoints need to be appended to: $AISP_API_ENDPOINT

Resource	**Endpoint**	**Access Token**
Account Access Consents	/account-access-consents /account-access-consents/{ConsentId}	Access token retrieved using the OAuth 2.0 Client Credentials Grant. See section 5.2.
Accounts	/accounts /accounts/{AccountId}	Access token retrieved using the OIDC Hybrid Flow. See section 5.3.
Balances	/accounts/{AccountId}/balances	
Transactions	/accounts/{AccountId}/transactions	
Statements	/accounts/{AccountId}/statements /accounts/{AccountId}/statements/{StatementId}/file	

6.2 Request Headers

The following headers can be used to identify a request.

- x-fapi-interaction-id
- x-fapi-customer-last-logged-time
- x-fapi-customer-ip-address
- x-customer-user-agent

The x-fapi-interaction-id will be sent back as a header in the response. If x-fapi-interaction-id is not provided in the request then the API will generate a new UUID for this header in the response. Furthermore the x-fapi-financial-id header should be present on all requests and should contain the identifier of CREATION as issued by Open Banking.

6.3 Creating Account Access Consents

In order to create an account access consent do the following:

- Make a POST call to the Account Access Consents endpoint (without ConsentId in the path).
- Provide at least the following headers:
  - Content-Type: application/json
  - x-fapi-financial-id: <the identifier of CREATION as issued by Open Banking>
  - Authorisation: Bearer <access token retrieved using OIDC Hybrid Flow>
- Include a request body.

Below is an example request body to create an Account Access Consent along with an explanation of each field.

```
{
  "Data": {
    "Permissions": [
      "ReadAccountsDetail",
      "ReadBalances",
      "ReadTransactionsDetail",
      "ReadTransactionsCredits"
    ],
    "TransactionFromDateTime": "2016-01-25T00:00:00.000Z",
    "TransactionToDateTime": "2025-12-31T23:59:59.999Z"
  },
  "Risk": {}
}
```

Field name	Description	Example Values
Data	The consent data	
Data.Permissions	The permissions the consent will give to the TPP once it has been authorized.	ReadAccountsBasic – Gives access to a PSU's accounts' basic information. ReadAccountsDetail – Gives access to a PSU's accounts' detailed information. ReadBalances – Gives access to the balances of PSU's accounts'. ReadTransactionsBasic – Gives access to basic information about a PSU's transactions. ReadTransactionsCredits – Gives access to credit transactions of a PSU's accounts. ReadTransactionsDebits – Gives access to debit transactions of a PSU's accounts. ReadTransactionsDetail – Gives access to detailed information about a PSU's transactions. ReadStatementsBasic – Gives access to basic information of a PSU's statements. ReadStatementsDetail – Gives access to detailed information about a
Data.TransactionFromDateTime	Specified start date and time for the transaction query period. If this is not populated, the start date will be open	2017-04-05T10:43:07+00:00 2017-04-05T10:43:07.0000Z

Field name	Description	Example Values
	ended, and data will be returned from the earliest available transaction.	2017-04-05T10:43:07Z
Data.TransactionToDateTime	Specified end date and time for the transaction query period. If this is not populated, the end date will be open ended, and data will be returned to the latest available transaction.	2017-04-05T10:43:07+00:00 2017-04-05T10:43:07.0000Z 2017-04-05T10:43:07Z
Risk	Should always be empty.	

6.4 Request Validations

This section includes some significant validations that are done by the API.

6.4.1 Generic Validation

This validation takes place for all requests to the AISP API.

Validation	Description	HTTP Status Code
Authorisation Header	Validate that the authorisation header is not blank and starts with prefix Bearer.	401 Unauthorized with WWW-Authenticate header.
Access Token Signature	Validate the access token and its signature using the public key of the authorisation server.	
Accounts Scope in Token	Validate that the access token includes the accounts scope.	403 Forbidden

6.4.2 Account Access Consent Resource

This validation takes place only for requests to the Account Access Consent endpoints.

Validation	Description	HTTP Status Code
Resource with ConsentId belongs to TPP	Validates that the TPP identified by the access token is owner of the requested account-access-consent.	403 Forbidden
Resource with ConsentId is found for GET or DELETE methods	Validate that an account-access-consent resource with ConsentId is found for GET or DELETE methods.	400 Bad Request with an error response according to OB specification in the payload.
ReadAccountsBasic or ReadAccountsDetail is available	Validate that either ReadAccountsBasic or ReadAccountsDetail is available in the permissions list when creating an account-access-consent resource.	
ReadTransactionsBasic or	If ReadTransactionsBasic or ReadTransactionsDetail is available in the	

Validation	Description	HTTP Status Code
ReadTransactionsDetail is available	permissions list then either ReadTransactionsCredits or ReadTransactionsDebits must be available in the permissions list.	
ReadTransactionsCredits or ReadTransactionsDebits is available	If ReadTransactionsCredits or ReadTransactionsDebits is available in the permissions list then either ReadTransactionsBasic or ReadTransactionsDetail must be available in the permissions list.	

6.4.3 Account Information Resources

6.4.3.1 Generic

This validation takes place for all requests to the Accounts, Balances and Transaction endpoints.

Validation	Description	HTTP Status Code
ConsentId in Token exists	Validate that ConsentId is provided in access token.	403 Forbidden
Resource for ConsentId exists	Validate that an account access consent resource for the provided ConsentId exists.	
Resource for ConsentId not expired	Validate that an account access consent resource for the provided ConsentId is not expired when ExpirationDateTime is available in the resource.	
Resource for ConsentId belongs to TPP	Validates that the TPP identified by the access token is owner of the account-access-consent identified by ConsentId in the access token.	
Resource for ConsentId was re-authenticated recently	Validates that an account access consent resource for the provided ConsentId was re-authenticated by the PSU with strong customer authentication within the last 90 days.	

6.4.3.2 Accounts

This validation takes place only for requests to the Accounts endpoints.

Validation	Description	HTTP Status Code
AccountId is linked to account access consent	Validate that the account access consent identified by the ConsentId in the access token provides access to the account identified by AccountId.	403 Forbidden

Validation	Description	HTTP Status Code
Resource with AccountId is found for GET method	Validate that an account resource with AccountId is found for GET method.	400 Bad Request with an error response according to OB specification in the payload.
ReadAccountsBasic or ReadAccountsDetail exists	To view the account information either ReadAccountsBasic or ReadAccountsDetail must exist in the permissions list of the account access consent resource.	403 Forbidden

6.4.3.3 Balances

This validation takes place only for requests to the Balances endpoint.

Validation	Description	HTTP Status Code
AccountId is linked to account access consent	Validate that the account access consent identified by the ConsentId in the access token provides access to the account identified by AccountId.	403 Forbidden
ReadBalances	To view the balances ReadBalances must exist in the permissions list of the account access consent resource.	

6.4.3.4 Transactions

This validation takes place only for requests to the Transactions endpoint.

Validation	Description	HTTP Status Code
AccountId is linked to account access consent	Validate that the account access consent identified by the ConsentId in the access token provides access to the account identified by AccountId.	403 Forbidden
ReadTransactionsBasic or ReadTransactionsDetail exists	To view transactions ReadTransactionsBasic or ReadTransactionsDetail must exist in the permissions list of the account access consent resource.	
ReadTransactionsCredits or ReadTransactionsDebits exists	To view transactions ReadTransactionsCredits or ReadTransactionsDebits must exist in the permissions list of the account access consent resource.	
The (optional) FromBookingDateTime and ToBookingDateTime query	If the PSU last authorized the consent more than 5 minutes ago, only transactions up to 90 days in the past and up to 90 days in the	

Validation	Description	HTTP Status Code
parameters are not too far in the past or future.	future from this authorisation timestamp are available. If the FromBookingDateTime and/or ToBookingDateTime specify a timestamp that does not comply with this 180 day window, the validation will fail.	

6.4.3.5 Statements

This validation takes place only for requests to the Statements endpoints.

Validation	Description	HTTP Status Code
AccountId is linked to account access consent	Validate that the account access consent identified by the ConsentId in the access token provides access to the account identified by AccountId.	403 Forbidden
ReadStatementsBasic or ReadStatementsDetail exists	To view statements ReadStatementsBasic or ReadStatementsDetail must exist in the permissions list of the account access consent resource.	

6.5 Pagination

The following endpoints are paginated:

- /accounts
- /accounts/{AccountId}/transactions
- /accounts/{AccountId}/statements

A TPP can provide a *Page* query parameter to retrieve the accounts or transactions of a specific page. When not provided the page will default to 1. If an invalid value is provided the AISP API will respond with a 'bad request' error. The number of accounts or transactions per page cannot be changed by the TPP.

6.6 Account Information Resource Responses

The responses of the accounts, balances and transactions endpoints contain the data as specified in the Swager specification, but with additional notes.

6.6.1 Accounts

For accounts 'Description', 'Nickname' and 'Servicer' are never present. Furthermore the 'AccountType' and 'AccountSubType' will always be *Personal* and *CreditCard* respectively. Consequently the 'Account.SchemeName' will always be *UK.OBIE.PAN* but 'Account.Identification' will be masked.

6.6.2 Balances

For balances 'CreditLine' information will never be returned and the 'Type' will always be *OpeningAvailable.*

6.6.3 Transactions

For transactions only the mandatory fields as well as 'TransactionInformation', if available, will be returned.

6.6.4 Statements

For 'GET /accounts/{AccountId}/statements' only the mandatory fields as well as StatementId will be returned.

7. CBPII API

After an access token has been acquired by the TPP, the TPP can access the CBPII API to either create a new Funds Confirmation Consent for a PSU to authorize, or confirm the availability of funds on a PSU's account for which consent has already been obtained.

7.1 Endpoints

All CBPII API resource endpoints need to be appended to: $CBPII_API_ENDPOINT

Resource	Endpoint	Access Token
Funds Confirmation Consents	/funds-confirmation-consents /funds-confirmation-consents/{ConsentId}	Access token retrieved using the OAuth 2.0 Client Credentials Grant. See section 5.2.
Funds Confirmations	/funds-confirmations	Access token retrieved using the OIDC Hybrid Flow. See section 5.3.

7.2 Request Headers

The following headers can be used to identify a request

- x-fapi-interaction-id
- x-fapi-customer-last-logged-time
- x-fapi-customer-ip-address
- x-customer-user-agent

The x-fapi-interaction-id will be sent back as a header in the response. If x-fapi-interaction-id is not provided in the request then the API will generate a new UUID for this header in the response.

Furthermore the x-fapi-financial-id header should be present on all requests and should contain the identifier of CREATION as issued by Open Banking.

7.3 Creating Funds Confirmation Consents

In order to create a funds confirmation consent do the following:

- Perform a POST call to the Funds Confirmation Consents endpoint.
- Provide at least the following headers:
    - Content-Type: application/json
    - x-fapi-financial-id: <the identifier of CREATION as issued by Open Banking>
    - Authorization: Bearer <access token retrieved using OIDC Hybrid Flow>
- Include a valid request body.

Below is an example request body to create a Funds Confirmation Consent along with an explanation of each field.

```
{
    "Data": {
        "ExpirationDateTime”: “2021-01-25T00:00:00.000Z”,
        "DebtorAccount": {
            "SchemeName": "UK.OBIE.PAN",
            "Identification": "5299321805019634",
            "Name": "John Doe"
        }
    }
}
```

Field name	Description	Example Values
Data	The consent data	
Data.ExpirationDateTime	The datetime on which the consent expires. Not mandatory. Not providing an expiration datetime means the consent is ongoing.	
Data.DebtorAccount	The debtor account information.	
Data.DebtorAccount.SchemeName	The identification scheme used.	UK.OBIE.PAN
Data.DebtorAccount.Identification	The account number.	<any CREATION card number>
Data.DebtorAccount.Name	Name of the account holder.	

7.4 Request Validations

This section includes some significant validations that are done by the API.

7.4.1 Generic Validation

This validation takes place for all requests to the CBPII API.

Validation	Description	HTTP Status Code
Authorization Header	Validate that the authorization header is not blank and starts with prefix Bearer.	401 Unauthorized with WWW-Authenticate header.
Access Token Signature	Validate the access token and its signature using the public key of the authorization server.	
Fundsconfirmations Scope in Token	Validate that the access token includes the fundsconfirmations scope.	403 Forbidden

7.4.2 Funds Confirmation Consent Resource

This validation takes place only for the requests to the Funds Confirmation Consent endpoints.

Validation	Description	HTTP Status Code
Resource with ConsentId belongs to TPP	Validates that the TPP identified by the access token is owner of the requested account-access-consent.	403 Forbidden
Resource with ConsentId is found for GET or DELETE methods	Validate that an account-access-consent resource with ConsentId is found for GET or DELETE methods.	400 Bad Request with an error response according to OB specification in the payload.
Identification scheme is valid	Validate that DebtorAccount.SchemeName is UK.OBIE.PAN	
Account exists	Validate that the account with the given identification exists.	

7.4.3 Funds Confirmations Resource

This validation takes place for all requests to Funds Confirmations endpoint.

Validation	Description	HTTP Status Code
ConsentId in Token exists	Validate that ConsentId is provided in access token.	403 Forbidden
Resource for ConsentId exists	Validate that a funds confirmation consent resource for the provided ConsentId exists.	
Resource for ConsentId not expired	Validate that a funds confirmation consent resource for the provided ConsentId is not expired when ExpirationDateTime is available in the resource.	
Resource for ConsentId belongs to TPP	Validates that the TPP identified by the access token is owner of the funds-confirmation-consent identified by ConsentId in the access token.	
Resource for ConsentId was re-authenticated recently	Validates that a funds confirmation consent resource for the provided ConsentId was re-authenticated by the PSU with strong customer authentication within the last 90 days.	
Request currency matches account currency	Validates that the currency in the funds confirmation request matches the currency of the DebtorAccount of the consent.	400 Bad Request with an error response according to OB specification in the payload.

8. Onboarding as non Open Banking enrolled TPP

Since TPPs can only dynamically register with us when they are enrolled with Open Banking, we've setup an alternative process to register a client with us for TPPs that are not enrolled with Open Banking.

In order to register a client with us when you are not enrolled with Open Banking, the following information should be provided in an email to OBdeveloper@creation.co.uk.

Information	**Description**
Signing Certificate (PEM)	The signing certificate in PEM format to be used to verify the signatures of client assertions and request objects during the various flows to obtain access tokens.
Transport Certificate (PEM)	The transport certificate to be used during MATLS in PEM format. This must be a valid eIDAS compliant certificate and will also be used to determine the PSD2 roles of your organization, which will affect what API's can be accessed.
Redirect URIs	The allowed redirect URIs used during the OIDC hybrid flow.
Software Name	The name of the software that is being registered. This will be used to identify your client.
Environment	The environment in which you want to register a client. Either sandbox or production.

The information will then be verified and a client will be created. Once the client has been created, the client id that can be used to obtain access tokens will be emailed back to you.

9. Incidents

If an incident or issue is discovered whilst trying to use the service for either environment (Sandbox or Production) then there are two routes that we advise:

1. **Open Banking Incident Form** – This should be completed for all incidents and issues that arise. This is located on the Creation Finance Open Banking Portal.
2. **Email us directly** – You will need to email OBdeveloper@creation.co.uk and this should only be used for any questions and queries.

Please note that if you have raised an incident, the ticket updates will be from BNP Paribas Personal Finance which is a trading style of Creation Financial Services Limited.

10. Related Documents and References

10.1 Related Documents

Title	Rev No
Read / Write Data API Specification – v3.1	3.1
Account and Transaction API Specification – v3.1	3.1
Account Access Consents v3.1	3.1
Accounts v3.1	3.1
Balances v3.1	3.1
Transactions v3.1	3.1
Statements v3.1	3.1
Confirmation of Funds API Specification - v3.1	3.1
Funds Confirmation Consent v3.1	3.1
Funds Confirmation v3.1	3.1
Open Banking Security Profile – Implementer's Draft v1.1.2	1.1.2
Dynamic Client Registration - v3.2	3.2

10.2 References

[1] Enroll as TPP onto Open Banking, https://www.openbanking.org.uk/providers/third-party-providers/

[2] JSON Web Token, https://jwt.io/

[3] OAuth 2.0, https://oauth.net/2/

[4] OpenID Connect Core, https://openid.net/specs/openid-connect-core-1_0.html

[5] Client Credentials Grant, https://tools.ietf.org/html/rfc6749#section-4.4

[6] Hybrid Flow, https://openid.net/specs/openid-connect-core-1_0.html#HybridFlowAuth

[7] Open Banking Directory Usage - eIDAS release (Production) - DRAFT V1.1