

Inland Revenue

Identity and Access Services build pack

Date: 05/02/2018

Version: 1.9

UNCLASSIFIED

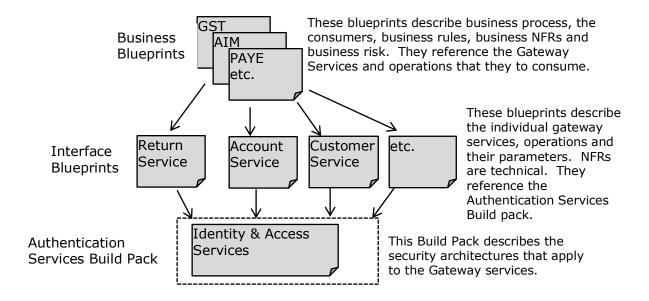


About this Document

This document is intended to provide Service Providers with the technical detail required to consume the Identity and Access services offered by Inland Revenue.

This is a technical document that supports the on boarding processes of an end to end solution. The associated on-boarding document(s) describe the end-to-end business level solution, of which this build pack is part. This document describes the architecture of the technical solution, the interaction with other build packs, schemas and endpoints. Also included are sample payloads to use in non-production environments.

It is part of a 3-tier hierarchy of documents that are depicted in the following diagram:





Contents

| 1 | Overvie | ew | 5 |
|---|-----------|--|------|
| | 1.1 This | s solution | 5 |
| | 1.1.1 | Organisational Authentication and Authorisation | 5 |
| | 1.1.2 | End-User Authentication and Authorisation | 6 |
| | 1.2 Inte | ended audience | 6 |
| | 1.3 Info | ormation IR will provide Service Providers | 6 |
| | 1.3.1 | Token Auth (Cloud or Native) | 6 |
| | 1.3.2 | SSH keys | 6 |
| | 1.4 Info | ormation Service Providers must provide IR | 7 |
| | 1.4.1 | Service Provider Information | 7 |
| | 1.4.2 | Token Auth (Cloud or Native) | 7 |
| | 1.4.3 | SSH keys | 7 |
| 2 | Descrip | tion of the IR Authentication Mechanisms | 8 |
| | 2.1 IR | Token Auth Implementation using OAuth 2.0 | 8 |
| | 2.1.1 | High Level View of OAuth 2.0 | 8 |
| | 2.1.2 | Authorisation Services | 9 |
| | 2.1.3 | Authorisation Service | 9 |
| | 2.1.4 | Refresh Token Service | |
| | 2.1.5 | Revoke Token Service | 13 |
| | 2.1.6 | Security Considerations | 13 |
| | 2.1.7 | Endpoints | 14 |
| 2 | .2 Native | Application Token Auth | . 14 |
| 2 | .3 SSH Ke | ys | . 15 |
| | | Signing Certificate | |
| 3 | Append | lix A – Sample payloads | . 16 |
| | 3.1 Rec | າuest Authorisation Code | 16 |
| | 3.1.1 | Request | 16 |
| | 3.2 Aut | horisation Code response | 16 |
| | 3.2.1 | Success Response – Authorisation Code sent | |
| | 3.3 Rec | ງuest Access token | |
| | 3.3.1 | Exchange Authorisation Code for oAuth Access Token | 16 |
| | 3.3.2 | Success Response – Access Token sent | |
| | 3.4 Rec | រុuest Refresh token | |
| | 3.4.1 | Refresh request | |
| | 3.4.2 | Refresh token reply | 18 |
| | 3.4.3 | Error Response. | |
| | 3.5 Rev | oke token request | 19 |
| | 3.5.1 | Revoke token request | 19 |
| | 3.5.2 | Revoke token reply | 19 |



| | 3.5.3 | Revoke token Error Response | . 19 |
|---|-------|-----------------------------|------|
| 4 | Appen | dix B - Glossary | 20 |
| 5 | Appen | dix C—Change log | 22 |



1 Overview

1.1 This solution

Inland Revenue (IR) is establishing a new set of Identity and Access services. These will provide Service Providers with authentication and authorisation mechanisms for accessing IR's new Gateway Services.

There are two distinct entities for which IR provides mechanisms for authentication and authorisation:

- 1. Organisations
- 2. End Users

The mechanisms are as per the diagram below:



Organisational Authentication & Authorisation

- SSH key
- Client Signing Certificate (future capability)



End User
Authentication &
Authorisation

- Token Authorisation (OAuth 2.0)
 - Cloud
 - Native Application (under development)

Figure 1 Entities and Authentication & Authorisation mechanisms

1.1.1 Organisational Authentication and Authorisation.

The table below details the current and future mechanisms IR provide to authenticate and authorise an organisation for Machine to Machine (M2M) communication.

| Authorisation Mechanism | Uses |
|----------------------------|---|
| SSH keys | This mechanism is used in SFTP file transfers to identify the organisations sending/receiving files. |
| | SSH keys need to be exchanged to authenticate both parties. |
| Client Signing Certificate | Used when the Service Provider server is anonymous. X.509 client certificates are used to sign messages in order |
| X.509 certificate based | to identify the service provider to IR. Service providers will |
| (Future capability) | be able to register their certs with IR through a self-help portal. |



This is not currently available for use.

Table 1 Organisational Authentication and Authorisation Methods

1.1.2 End-User Authentication and Authorisation.

The OAuth 2.0 process is used to authenticate end-users using their IR user ID and password and grant 3rd party software consent to access their information.

The OAuth 2.0 mechanism to be used by a service provider is based upon the nature of the client application the end-user will be using.

| Authorisation Mechanism | Uses |
|--|---|
| Cloud application Token Auth OAuth2.0 based | Use when the client application is a web-enabled cloud based application. It requires an online user to enter their myIR user ID and password to grant the application access to their IR information. |
| Native application Token Auth OAuth2.0 based | Use when the client application is a desktop or other native application. It also requires an online user to enter their myIR user ID and password to grant the application access to their IR information. |
| | See section 2.2 Native Application Token Auth below for details |

Table 2: End-User Authentication and Authorisation Methods

1.2 Intended audience

This build pack and the resources to which it refers are primarily focused on the needs of Software Developers' technical teams and development staff.

The reader is assumed to have a suitable level of technical knowledge in order to comprehend the information provided. A range of technical terms and abbreviations are used throughout this document, and while most of these will be understood by the intended readers, a glossary is provided at the end.

This document is not intended for use by managerial staff or those with a purely business focus.

1.3 Information IR will provide Service Providers

1.3.1 Token Auth (Cloud or Native)

- 1. URLs and parameters for invoking the Authentication services.
- 2. Client ID (agreed with service consumer)
- 3. Client secret (used in step 2 in section 2.1.2)

1.3.2 SSH keys

- 1. SSH keys for SFTP.
- 2. PGP public keys if used for payload encryption and signing



1.4 Information Service Providers must provide IR

1.4.1 Service Provider Information

- 1. Full business name
- 2. Client ID (agreed with IR and used in requests to IR))
- 3. Key Contact
- 4. Email of key contact or delegate
- 5. Mobile Phone number (SMS may be used for some information)
- 6. IP addresses Service Providers will use for test instances for IR firewall whitelisting.

1.4.2 Token Auth (Cloud or Native)

1. Redirect URI for Authorisation code and Authentication token.

1.4.3 SSH keys

- 1. SSH public keys if using SFTP.
- 2. Their own Public Keys if using PGP.



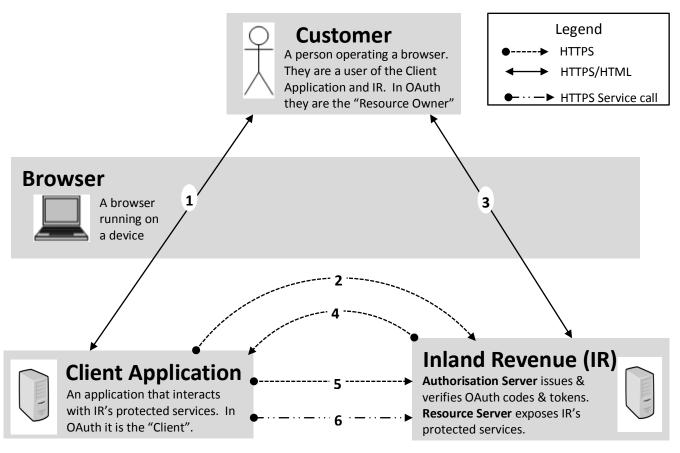
2 Description of the IR Authentication Mechanisms

2.1 IR Token Auth Implementation using OAuth 2.0

This section describes the IR OAuth 2.0 implementation. This high-level description covers the services offered by IR's implementation of OAuth Authorisation services including both Cloud and Native application usage.

2.1.1 High Level View of OAuth 2.0

For OAuth 2 the following diagram depicts the high level end-to-end view of the components and the interactions between them:



- 1. The User is interacting with the Client Application. They access a protected service provided by IR (e.g. to file a return, retrieve a balance etc.)
- 2. The Client Application invokes the Authorisation API to get an authorisation code, the user's browser is redirected to IR's logon page.
- 3. IR prompts the User to logon, they are authenticated. On first use the User must also supply their consent for the Client Application to access IR on their behalf. IR issues the Authorisation Code.
- 4. The Authorisation Code is returned to the Client Application.
- 5. The Client Application invokes IR's Token service to redeem the Authorisation Code for an OAuth Access Token. It has a finite time to live.
- 6. The Client Application can then invoke IR's protected services (e.g. to file a return etc.) supplying the OAuth Access Token in the header. The OAuth Access Token can be used for multiple invocations until it expires.



Inland Revenue's implementation of the OAuth 2 standard conforms to the Authorisation Code Grant flow described in section 4.1 of RFC 6749 (https://tools.ietf.org/html/rfc6749).

2.1.2 Authorisation Services

This section describes the services offered by IR through the Authorisation services, including:

- 1. Authorisation Service (via OAuth 2.0)
- 2. Refresh Token Service
- 3. Revoke Token Service.

The following sections will describe the steps and service calls required to integrate with the IR implementation.

2.1.3 Authorisation Service

This section describes the steps and service calls required when using the IR implementation of OAuth 2.0. These are the same for both Cloud and Native App usage.

2.1.3.1 Customer accesses the Client Application (Step1)

The Customer accesses the Client application and triggers the need for it to consume one of Inland Revenue's protected services (e.g. to retrieve an account balance, to file a return etc.).

2.1.3.2 Request Authorisation Code (Step 2)

The customer's browser is redirected to the IR Authorisation service to authenticate the user and confirm scope using the GET method described below (example is for a Test environment – see 2.1.4 below for all endpoints):

https://q.services.ird.govt.nz/ms oauth/oauth2/endpoints/oauthservice/authorize?

response_type=code

&client_id=IdOfCompanyUsingTheAPI

&redirect_uri=http://client.example.com/return

&scope=MYIR.Services

&state=xyz

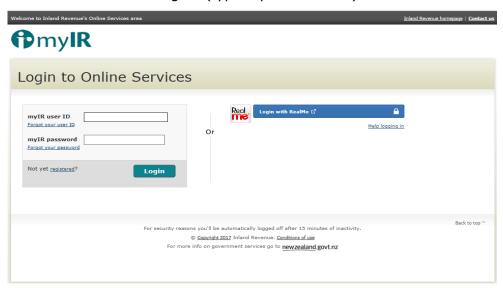
| Name | Description | Required | Valid Values |
|---------------|---|----------|-----------------------------|
| response_type | Response type requested | Required | "code" |
| client_id | The agreed Client identifier established at registration. Inland Revenue maintains this list of values. | Required | client_id |
| redirect_uri | The Client application's redirect URI to which the Authorisation Code is returned. | Required | Business Partner defined |
| scope | Use space-separated values. Define scope values in the configuration/scope registry. | Required | "MYIR.Services" |



| state | A value used by the Business Partner to maintain state between the request and callback. The parameter should be used to prevent cross-site forgery requests. | Recommended | Business Partner defined |
|-------|---|-------------|-----------------------------|
|-------|---|-------------|-----------------------------|

2.1.3.3 Login with myIR Credentials (step 3)

During this step the customer may be required to authenticate, and, if this is required, will be redirected to the following myIR logon screen. For OAuth 2.0 for Native Apps this authorisation request is in an external user-agent (typically the browser).



If the software provider chooses (not generally recommended as this page may change from time to time) to present this page within a frame the minimum recommended size in pixels is $600w \times 500h$.

Note the customer must already have an IR Online Services credential.

Invalid User ID or password will return a HTTP:200

2.1.3.4 Respond with Authorisation Token (Step 4)

If successful, the authorisation service will respond with the Authorisation Code to the Business Partner redirect_uri as described below:

https://client.example.com/return?code=eyJhbG...rWWk8hbs o6uY&state=xyz

| Name | Description | Valid Values |
|-------|---|--------------------------------------|
| Code | Authorisation code value - Includes the following: Expiry Client_id Redirect_uri | Encrypted string ~1000 characters |
| State | Business Partner defined state | Business Partner defined (as passed) |



If not successful an error is sent with a HTTP code and a JSON response containing the error code and description.

Errors are:

| HTTP code | Error Type | Description | |
|----------------|----------------------|---|--|
| 400 | invalid_redirect_uri | Redirect URI mismatch with Business Partner | |
| | | арр | |
| | Invalid client ID | API Key contains invalid information | |
| invalid_client | | Business partner identifier invalid | |
| | invalid_scope | Requested scope is invalid, unknown, or malformed | |
| | server_error | Authentication - Runtime processing error | |
| | access_denied | End-user denied authorisation | |
| 500 | InternalError | An internal and unexpected error occurred | |
| 504 | · | | |
| | | the upstream server | |

2.1.3.5 Request Authorisation Token (Step 5)

Once an Authorisation Code has been returned to the Client application it must be exchanged for an OAuth Access Token by doing an HTTPS Post to the Token Service as follows:

```
https://q.services.ird.govt.nz/ms_oauth/oauth2/endpoints/oauthservice/tokens

<form>

redirect_uri=https://client.example.com:17001/return

&grant_type=authorization_code

&code=eyJhbG...rWWk8hbs_o6uY

</form>
```

With the values of:

| Name | Description | Required | Valid Values |
|--------------|---|----------|-----------------------------------|
| redirect_uri | The Client application's redirect URI for the Authorisation Token. | Required | Business Partner defined |
| grant_type | The grant type is authorization_code | Required | authorization_code |
| code | Authorisation Code as supplied by the authorisation service in step 4 | Required | Encrypted string ~1000 characters |

The header fields must contain the Client ID and Client secret and content type:

| Authorization: Basic | NTQzMjFpZZWxjb211MQ== |
|--------------------------------|----------------------------------|
| Content-Type: application/x-ww | vw-form-urlencoded;charset=UTF-8 |



With the values of:

| Name | Description | Required | Valid Values |
|---------------|--|----------|---|
| Authorization | "Basic " + Base64 encoded (ClientID + ":" + Client Secret) | Required | Base64 encoded string |
| Content-Type | Content type | Required | application/x-www-form- urlencoded;charset=UTF-8 |

The response contains the OAuth Access Token – this must be passed on subsequent service calls.

If the client is registered for Refresh Tokens, this will also be given at the point, see Section 2 Refresh Token Service.

If not successful an error is sent with a HTTP code and a JSON response containing the error code and description.

Errors:

| HTTP code | Error Type | Description |
|---|---------------------------------------|--|
| 400 | invalid_redirect_uri | Redirect URI mismatch with Business Partner app |
| | Invalid client ID | API Key contains invalid information |
| | Invalid client_id or client_secret | API Secret contains invalid information |
| | invalid_client | Business partner identifier invalid |
| invalid_scope Requested scope is invalid, malformed | | Requested scope is invalid, unknown, or malformed |
| | server_error | Authentication - Runtime processing error |
| | access_denied | End-user denied authorisation |
| 500 | InternalError | An internal and unexpected error occurred |
| 504 | GatewayError | Gateway did not receive a timely response from the upstream server |

2.1.4 Refresh Token Service

In normal use it is expected that the Access Token will be re-used while it is active and the client will make several calls using this Access Token. If the Access Token has expired, the client can request another Access Token using the Refresh Token and client credentials.

The Refresh Token is granted as part of the Cloud Authorisation Service, the client will need to hold this and when required can be exchanged for a new Access and Refresh Token granting longer term use.

The Rest call to exchange a Refresh Token for an Access token takes the form:

 $https://q.services.ird.govt.nz/ms_oauth/oauth2/endpoints/oauthservice/tokens$



<form>

grant_type=refresh_token

&refresh_token=<refresh-token-value>

</form>

With the values of:

| Name | Description | Required | Valid Values |
|---|--|----------|------------------------------|
| grant_type | The grant type is required and takes the value refresh_token | Required | refresh_token |
| refresh_token The refresh_token field contains the Refresh Token | | Required | <refresh token=""></refresh> |

The header fields must contain the Client ID and Client secret and content type:

Authorization: Basic NTQzMjFpZ...ZWxjb21lMQ==

Content-Type: application/x-www-form-urlencoded;charset=UTF-8

With the values of:

| Name | Description | Required | Valid Values |
|---------------|--|----------|---|
| Authorization | "Basic " + Base64 encoded (ClientID + ":" + Client Secret) | Required | Base64 encoded string |
| Content-Type | Content type | Required | application/x-www-form- urlencoded;charset=UTF-8 |

The normal expected response if the Access Token is valid is a new Access and Refresh Token.

Note that the refresh token does not expire. A new Access Token can be requested using the Refresh Token at any time after it was obtained as long as the Customer consent is still valid.

2.1.5 Revoke Token Service

A token revoke process is not currently supported for this release. Customers may revoke their consent through a function in IR Online Services.

2.1.6 Security Considerations

Protecting the integrity of the Client Secret is an important requirement for providers, the exact implementation is left to the provider but it <u>must not be stored in plain text</u> either in the web, mobile, or desktop application. Our preference is for this to be stored on a back-end server and made available to the Business Partner application.

If a Client Secret is compromised it shall be invalidated and a new secret issued.



The OAuth Authorisation Code has a time to live of 15 minutes.

The OAuth Access Token has a time to live of 8 hours.

2.1.7 Endpoints

Endpoints for the token based Authentication and Authorisation Service are as follows:

Test Environments:

Code: https://q.services.ird.govt.nz/ms oauth/oauth2/endpoints/oauthservice/authorize

Token: https://q.services.ird.govt.nz/ms oauth/oauth2/endpoints/oauthservice/tokens

Production Environment

Code: https://services.ird.govt.nz/ms oauth/oauth2/endpoints/oauthservice/authorize

Token: https://services.ird.govt.nz/ms oauth/oauth2/endpoints/oauthservice/tokens

2.2 Native Application Token Auth

The OAuth 2.0 transaction flow described above will not change for Service Providers running Native Applications. A Native app is an application that is installed by the user to their device or a desktop application, as distinct from a web app that runs in the browser context only.

IR has a duty to protect customer information in transactions. In order to meet this obligation the security and traffic controls for Native endpoints will be more stringent. This is due to the more public nature of these endpoints and the inability to identify the caller by an x.509 certificate as per the cloud connection.

To further this end, IR is adopting RFC8252 OAuth 2.0 for Native Apps

Your attention is drawn to the best practice description in section 1 of using an external browser for OAuth by native apps.

IR will be providing separate endpoints where using gateway services with Native Auth applications. Mutual TLS will not be used for these endpoints so no x509 client cert will be required but server side TLS will still be used.

Please refer to the Build Pack for each respective service for further details.

Note the endpoints for the Native App OAuth calls will not be changing from those outlined in Section 2.1.7 above.

A Client ID and secret will still be issued, but for Native Applications the Client ID will need to show both the vendor and product e.g. SmartSoftware_payroll or SmartSoftware_tax.

At registration IR will note the OAuth user ID is type 'Native App' and not 'Normal'. The refresh token capability will not be offered for Native App OAuth tokens.

Special redirect will be allowed and the three redirection schemes in section 7 of RFC 8252 will be supported. IR does not limit or put a preference on any of these.

If considering 7.3 then attention is drawn to recommendations available on the internet regarding the use of 127.0.0.1 rather than local host. See ietf.org.

RFC 8252 section 6 describes Proof Key for Code Exchange (PKCE). Section 8.1 describes this is a proof-of-possession extension to OAuth 2.0.

Service Providers should note IR intends to implement the PKCE protocol at some time in the near future.



2.3 SSH Keys

This authentication and authorisation mechanism is used in SFTP file transfers to identify the respective organisations sending/receiving files, in this case IR and the Service Provider.

SSH Key authentication and authorisation will be used for file transfers in which SFTP 3.0 is used. This version of SFTP requires the use of SSH version 2.0.

The public key algorithm for SSH authentication keys must be ECDSA with a minimum field/key size of at least 160 bits.

Certain FTP file transfers will also require payload encryption and signing to ensure that once a file is transferred to an endpoint only an authorised party can interpret it. This is optional and the need for this will be identified in the respective On-boarding pack for a file transfer.

The need will be based upon the NZISM Information classification privacy rating based upon the sensitivity of the customer data along with considerations such as the volumes being transferred.

For files from IRD to partners that have PGP the PGP encryption will use Advanced Encryption Standard (AES) with a 256-bit key and the PGP hashing will use Secure Hash Algorithm (SHA) SHA-256.

Currently IR has the ability to push and pull files being exchanged, but the Service Provider always hosts the FTP server.

2.4 Client Signing Certificate

This is Inland Revenue's future M2M authentication mechanism to allow easy on-boarding of clients though a self-help portal.

A X.509 client certificate is used to provide IR with the identity of the Organisation when connecting to the mass market M2M interfaces.

As stated previously in this document, this mechanism is not yet available for use.

UNCLASSIFIED Page 15 of 22



3 Appendix A - Sample payloads

3.1 Request Authorisation Code

Service consumer to IR. Refer to step 2 in section 2.1.1.

3.1.1 Request

Service provider will send a HTTP GET request to Auth Server, url format as below:

https://q.services.ird.govt.nz/ms oauth/oauth2/endpoints/oauthservice/authorize? client id=Test30206492&redirect uri=https://jenkins.kiwisaver.ird.digitalpartner .services/&scope=MYIR.Services&response type=code

3.2 Authorisation Code response

IR to Service consumer. Refer to step 4 in section 2.1.1.

3.2.1 Success Response - Authorisation Code sent

https://jenkins.kiwisaver.ird.digitalpartner.services/?code=eyJhbGciOiJSUzUxMiIs InR5cCI6IkpXVCIsIng1dCI6Ino5VFR1R2pHSVJNd01pQjRaUGxjbzhDZ3J10CIsImtpZCI6Im9yYWt1 eSJ9.eyJvcmFjbGUub2F1dGgucmVkaXJlY3QtdXJpIjoiaHR0cHM6Ly9qZW5raW5zLmtpd2lzYXZlci5 pcmQuZGlnaXRhbHBhcnRuZXIuc2VydmljZXMvIiwic3ViIjpudWxsLCJvcmFjbGUub2F1dGgudXNlc19 vcmlnaW5faWRfdHlwZSI6IkxEQVBfVU1EIiwib3JhY2x1Lm9hdXRoLnVzZXJfb3JpZ21uX21kIjoic2F tbXkzOTAiLCJpc3MiOiJJbmxhbmRSZXZlbnVlIiwib3JhY2xlLm9hdXRoLnN2Y19wX24iOiJPQXV0aFN lcnZpY2VQcm9maWxlIiwiaWF0IjoxNTAyMzIxNDI3LCJvcmFjbGUub2F1dGgudGtfY29udGV4dCI6ImF 6YyIsImV4cCI6MTUwMjM1MDIyNywicHJuIjpudWxsLCJqdGkiOiJhNDg2ZTU1Ny0zZTc1LTQ3ZmYtODk ONC1hNTcxZWNlYzgzNmYiLCJvcmFjbGUub2F1dGguc2NvcGUiOiJNWUlSLlNlcnZpY2VzIiwib3JhY2x lLm9hdXRoLmNsaWVudF9vcmlnaW5faWQiOiJUZXN0MzAyMDY00TIiLCJ1c2VyLnRlbmFudC5uYW11Ijo iRGVmYXVsdERvbWFpbiIsIm9yYWNsZS5vYXV0aC5pZF9kX2lkIjoiMTIzNDU2NzgtMTIzNC0xMjM0LTE yMzQtMTIzNDU2Nzg5MDEyIn0.NTBu3R-JwaaOWfvMdWAHqY7Ji3YI3IbSTXqx6jauqEUhswLmAG6cbpGaSky50ECbHNv2skU8WVZ0RYv67KPgITGXJz0ZKSjqOgiZ0R4kFCZ7as

N8yjIzXgxwWk4mPXL5E02u24-VMbr hrNZYDZbakOpz4uY6UlSSNECmw0ac8

3.3 Request Access token

Service consumer to IR. Refer to step 5 in section 2.1.1.

3.3.1 Exchange Authorisation Code for oAuth Access Token.

At this step as a service provider you need to send a HTTP POST request to Auth Server with Authorization Code and your client credentials. Url format:

https://q.services.ird.qovt.nz/ms_oauth/oauth2/endpoints/oauthservice/tokens

Request headers: {Authorization=[Basic VGVzdDMwMjA2NDkyOk9hdXRoMklSU2VjcmV0dA==], Content-Type=[application/x-www-formurlencoded], Content-Length=[1127]}

Request body:

redirect uri=https://jenkins.kiwisaver.ird.digitalpartner.services/&grant type=a uthorization code&code=eyJhbGciOiJSUzUxMiIsInR5cCI6IkpXVCIsIng1dCI6Ino5VFR1R2pHS VJNd01pQjRaUGxjbzhDZ3J10CIsImtpZCI6Im9yYWtleSJ9.eyJvcmFjbGUub2F1dGgucmVkaXJ1Y3Qt dXJpIjoiaHR0cHM6Ly9qZW5raW5zLmtpd2lzYXZlci5pcmQuZGlnaXRhbHBhcnRuZXIuc2VydmljZXMv Iiwic3ViIjpudWxsLCJvcmFjbGUub2F1dGqudXNlcl9vcmlnaW5faWRfdHlwZSI6IkxEQVBfVUlEIiwi b3JhY2x1Lm9hdXRoLnVzZXJfb3JpZ21uX21kIjoic2FtbXkzOTAiLCJpc3MiOiJJbmxhbmRSZXZlbnV1 Iiwib3JhY2x1Lm9hdXRoLnN2Y19wX24i0iJPQXV0aFNlcnZpY2VQcm9maWx1IiwiaWF0IjoxNTAyMzIx NDI3LCJvcmFjbGUub2F1dGqudGtfY29udGV4dCI6ImF6YyIsImV4cCI6MTUwMjM1MDIyNywicHJuIjpu dWxsLCJqdGkiOiJhNDq2ZTU1Ny0zZTc1LTQ3ZmYtODk0NC1hNTcxZWN1YzqzNmYiLCJvcmFjbGUub2F1 dGquc2NvcGUiOiJNWUlSLlNlcnZpY2VzIiwib3JhY2x1Lm9hdXRoLmNsaWVudF9vcmlnaW5faWQiOiJU

UNCLASSIFIED Page 16 of 22



ZXNOMzAyMDY00TIiLCJ1c2VyLnRlbmFudC5uYW11IjoiRGVmYXVsdERvbWFpbiIsIm9yYWNsZS5vYXV0 aC5pZF9kX2lkIjoiMTIzNDU2NzgtMTIzNC0xMjM0LTEyMzQtMTIzNDU2Nzg5MDEyIn0.NTBu3R-JwaaOWfvMdWAHqY7Ji3YI3I-

3.3.2 Success Response – Access Token sent

The response contains the Access Token and expiry time. If refresh tokens are used this is also returned:

```
"expires_in": 28800,
"token_type": "Bearer",
"access_token":
```

"eyJhbGciOiJSUzUxMiIsInR5cCI6IkpXVCIsIng1dCI6Ino5VFR1R2pHSVJNd01pQjRaUGxjbzhDZ3J
10CIsImtpZCI6Im9yYWtleSJ9.eyJzdWIiOiJUUjI0NTczNzczIiwib3JhY2xlLm9hdXRoLnVzZXJfb3
JpZ2luX21kX3R5cGUiOiJMREFQX1VJRCIsIm9yYWNsZS5vYXV0aC51c2VyX29yaWdpb19pZCI6IlRSMj
Q1NzM3NzMiLCJpc3MiOiJJbmxhbmRSZXZlbnVlIiwib3JhY2xlLm9hdXRoLnN2Y19wX24iOiJPQXV0aF
NlcnZpY2VQcm9maWxlIiwiaWF0IjoxNTEzMTEzNDk4LCJvcmFjbGUub2F1dGgucHJuLmlkX3R5cGUiOi
JMREFQX1VJRCIsIm9yYWNsZS5vYXV0aC50a19jb250ZXh0IjoicmVzb3VyY2VfYWNjZXNzX3RrIiwiZX
hwIjoxNTEzMTQyMjk4LCJwcm4iOiJJUJjI0NTczNzczIiwianRpIjoiZDRlMDdkMzYtNDF1Mi00OTg5LT
kxNGMtNDYwZmUzNmYwZGQ0Iiwib3JhY2xlLm9hdXRoLmNsaWVudF9vcmlnaW5faWQiOiJNWU9CMTI3NT
QyMSIsIm9yYWNsZS5vYXV0aC5zY29wZSI6Ik1ZSVIuU2VydmljZXMiLCJ1c2VyLnRlbmFudC5uYW1lIj
oiRGVmYXVsdERvbWFpbiIsIm9yYWNsZS5vYXV0aC5pZF9kX21kIjoiMTIzNDU2NzgtMTIzNC0xMjM0LT
EyMzQtMTIzNDU2Nzg5MDEyIn0.P4c0twW-MFrGFuKa8gzuEC0AwxhMJDHSge5L7fUkWKF9k3WCmIwu5sKSxq2MNiLxs1XBDboqS8RA4-prv85-

sWQLN0patssDzK2qLCgWa9lKyTExIpmP0CvV_Yrnf5KCTNjyNzRl5Ba30T7bDaz5gn5iivrJn1HNghhb horY1I"

The response may contain the Refresh Token

```
"refresh token":
```

"eyJhbGcioiJSUzUxMiIsInR5cCI6IkpXVCIsIng1dCI6Ino5VFR1R2pHSVJNd01pQjRaUGxjbzhDZ3J loCIsImtpZCI6Im9yYWtleSJ9.eyJzdWIiOm51bGwsIm9yYWNsZS5vYXV0aC51c2VyX29yaWdpbl9pZF 90eXBlIjoiTERBUF9VSUQiLCJvcmFjbGUub2F1dGgudXNlcl9vcmlnaW5faWQiOiJUUjIONTczNzczIi wiaXNzIjoiSW5sYW5kUmV2ZW51ZSIsIm9yYWNsZS5vYXV0aC5ydC50dGMiOiJyZXNvdXJjZV9hY2Nlc3 NfdGsiLCJvcmFjbGUub2F1dGguc3ZjX3BfbiI6Ik9BdXRoU2VydmljZVByb2ZpbGUiLCJpYXQiOjE1MT MxMTM0OTgsIm9yYWNsZS5vYXV0aC50a19jb250ZXh0IjoicmVmcmVzaF90b2tlbiIsImV4cCI6MTUxMz E0MjI5OCwicHJuIjpudWxsLCJqdGkiOiJmNDZmMTU1YS1jNGU4LTRkYTktOTFkMy0zMjUxZTIyNDNhNj giLCJvcmFjbGUub2F1dGguc2NvcGUiOiJNWU1SLlNlcnZpY2VzIiwib3JhY2x1Lm9hdXRoLmNsaWVudF 9vcmlnaW5faWQiOiJNWU9CMTI3NTQyMSIsInVzZXIudGVuYW50Lm5hbWUiOiJEZWZhdWx0RG9tYWluIi wib3JhY2x1Lm9hdXRoLmlkX2RfaWQiOiIxMjM0NTY3OC0xMjM0LTEyMzQtMTIzNCOxMjM0NTY3ODkwMT IifQ.bsqzeC1BWEAireUWCLq3ZGhbSjcb6jrkfbV7F8lLuujVMF7xlGkOMoSLx9DSN_ePPE4z7PeDbz3igF269siiuplkitbA6JZiR6DRs3pl9ED3MCjwzp4_DVxXuip0K0FOamwOwzyROzSupjYsEEkJhkG0ur2 SGdTnjHFyXFoleoU",

3.4 Request Refresh token

Service consumer to IR. Refer to section 2.1.4.

3.4.1 Refresh request

}

At this step as a service provider you need to send a HTTP POST request to the Auth Server with the Refresh Token and your client credentials.

UNCLASSIFIED Page 17 of 22



```
Request headers: {Authorization=[Basic
VGVzdDMwMiA2NDkyOk9hdXRoMklSU2VicmV0dA==], Content-Type=[application/x-www-form-
urlencoded], Content-Length=[1127]}
```

Request body:

grant type=refresh token&refresh token=eyJhbGciOiJSUzUxMiIsInR5cCI6IkpXVCIsIng1d CI6Ino5VFR1R2pHSVJNd01pQjRaUGxjbzhDZ3J1OCIsImtpZCI6Im9yYWtleSJ9.eyJzdWIiOm51bGws $\verb|Im9yYWNsZS5vYXV0aC51c2VyX29yaWdpbl9pZF90eXBlIjoiTERBUF9VSUQiLCJvcmFjbGUub2F1dGgu| \\$ dXNlcl9vcmlnaW5faWQiOiJUUjIONTczNzczIiwiaXNzIjoiSW5sYW5kUmV2ZW51ZSIsIm9yYWNsZS5v YXV0aC5ydC50dGMiOiJyZXNvdXJjZV9hY2Nlc3NfdGsiLCJvcmFjbGUub2F1dGguc3ZjX3BfbiI6Ik9B dXRoU2VydmljZVByb2ZpbGUiLCJpYXQiOjE1MTMxMTM0OTqsIm9yYWNsZS5vYXV0aC50a19jb250ZXh0 IjoicmVmcmVzaF90b2tlbiIsImV4cCI6MTUxMzE0MjI5OCwicHJuIjpudWxsLCJqdGkiOiJmNDZmMTU1 YS1jNGU4LTRkYTktOTFkMy0zMjUxZTIyNDNhNjgiLCJvcmFjbGUub2F1dGguc2NvcGUiOiJNWUlSLlNl cnZpY2VzIiwib3JhY2x1Lm9hdXRoLmNsaWVudF9vcmlnaW5faWQiOiJNWU9CMTI3NTQyMSIsInVzZXIu dGVuYW50Lm5hbWUiOiJEZWZhdWx0RG9tYWluIiwib3JhY2x1Lm9hdXRoLmlkX2RfaWQiOiIxMjM0NTY3 OC0xMjM0LTEyMzQtMTIzNC0xMjM0NTY3ODkwMTIifQ.bsqzeC1BWEAireUWCLq3ZGhbSjcb6jrkfbV7F 81LuujVMF7xlGkOMoSLx9DSN ePPE4z7PeDbz3igF269siiuplkitbA6JZiR6DRs3p19ED3MCjwzp4 D VxXuip0K0FOamwOwzyROzSupjYsEEkJhkG0ur2SGdTnjHFyXFoleoU

3.4.2 Refresh token reply

The Refresh token reply is the same as an authorisation code exchange for an Access token

```
{
   "expires in": 28800,
   "token type": "Bearer",
  "refresh token":
```

"eyJhbGciOiJSUzUxMiIsInR5cCI6IkpXVCIsIng1dCI6Ino5VFR1R2pHSVJNd01pQjRaUGxjbzhDZ3J locisImtpZCI6Im9yYWtleSJ9.eyJzdWIiOm51bGwsIm9yYWNsZS5vYXV0aC51c2VyX29yaWdpbl9pZF 90eXBlIjoiTERBUF9VSUQiLCJvcmFjbGUub2F1dGgudXNlcl9vcmlnaW5faWQiOiJUUjI0NTczNzczIi wiaXNzIjoiSW5sYW5kUmV2ZW51ZSIsIm9yYWNsZS5vYXV0aC5ydC50dGMiOiJyZXNvdXJjZV9hY2Nlc3 NfdGsiLCJvcmFjbGUub2F1dGguc3ZjX3BfbiI6Ik9BdXRoU2VydmljZVByb2ZpbGUiLCJpYXQiOjE1MT MxMTQ3NjgsIm9yYWNsZS5vYXV0aC50a19jb250ZXh0IjoicmVmcmVzaF90b2tlbiIsImV4cCI6MTUxMz E0MzU2OCwicHJuIjpudWxsLCJqdGkiOiI2ZTQwOTIwNC05ZDkzLTRmM2ItYmMzNi1iMjZhMmMxZjA1Yj YiLCJvcmFjbGUub2F1dGguc2NvcGUiOiJNWUlSLlNlcnZpY2VzIiwib3JhY2xlLm9hdXRoLmNsaWVudF 9vcmlnaW5faWQiOiJNWU9CMTI3NTQyMSIsInVzZXIudGVuYW50Lm5hbWUiOiJEZWZhdWx0RG9tYWluIi wib3JhY2x1Lm9hdXRoLm1kX2RfaWQiOiIxMjM0NTY3OC0xMjM0LTEyMzQtMTIzNC0xMjM0NTY3ODkwMT IifQ.imUbjnAHZ4w2oENBMktN -tsMf71Uzn0RIKI7Ef9mK3ezkcTdaCc8LNPplYVB8oGnjV87GjEdLU0gE6rr7Vsu0pnPxJJrxQTlKAeiX2w6H-BTwux9718oniV-

E6VXb5eALWcOM33Htc9SHHOjglMNzyTl76-yg1F4TU3eueAG8",

"access token":

"eyJhbGci01JSUzUxMiIsInR5cCI6IkpXVCIsIng1dCI6Ino5VFR1R2pHSVJNd01pQjRaUGxjbzhDZ3J lOCIsImtpZCI6Im9yYWtleSJ9.eyJzdWIiOiJUUjI0NTczNzczIiwib3JhY2xlLm9hdXRoLnVzZXJfb3 JpZ2luX2lkX3R5cGUiOiJMREFQX1VJRCIsIm9yYWNsZS5vYXV0aC51c2VyX29yaWdpbl9pZCI6IlRSMj Q1NzM3NzMiLCJpc3MiOiJJbmxhbmRSZXZlbnVlIiwib3JhY2xlLm9hdXRoLnN2Y19wX24iOiJPQXV0aF NlcnZpY2VQcm9maWxlIiwiaWF0IjoxNTEzMTE0NzY4LCJvcmFjbGUub2F1dGgucHJuLmlkX3R5cGUiOi JMREFQX1VJRCIsIm9yYWNsZS5vYXV0aC50a19jb250ZXh0IjoicmVzb3VyY2VfYWNjZXNzX3RrIiwiZX hwIjoxNTEzMTQzNTY4LCJwcm4iOiJUUjI0NTczNzczIiwianRpIjoiMTkzNWEyZjktZGYwNi00NzA1LW I5MDMtYmQ1ZTY5NzIwZjE2Iiwib3JhY2xlLm9hdXRoLmNsaWVudF9vcmlnaW5faWQi0iJNWU9CMTI3NT ${\tt QyMSIsIm9yYWNsZS5vYXV0aC5zY29wZSI6Ik1ZSVIuU2VydmljZXMiLCJ1c2VyLnRlbmFudC5uYW11Ij}$ oiRGVmYXVsdERvbWFpbiIsIm9yYWNsZS5vYXV0aC5pZF9kX2lkIjoiMTIzNDU2NzgtMTIzNC0xMjM0LT EyMzQtMTIzNDU2Nzg5MDEyIn0.vMo0b H8ZeC09xcfDHoyRl50eqk0GeYdake99NrIvzcxUCsRtHbwMm gnBvzy45yL6T07-jfBlgRx3Tzt0MP-Gc70Kx-

ma s2SCD39m q1by9nX x7BWKFNGMM9gJl7jYTF6sWP6nNJCJWXfilfHJVuRchenyW4Lv8sVTKGPbQQQ

UNCLASSIFIED Page 18 of 22



3.4.3 Error Response.

See section for a more detailed list of error responses, a typical JSON response:

```
"error": "invalid_grant",
    "error_description": "Invalid Grant: grant_type=refresh_token"
```

3.5 Revoke token request

Service consumer to IR. Refer to section 2.1.5.

3.5.1 Revoke token request

At this step as a service provider you need to send a HTTP POST request to Auth Server with Access Token to revoke and your client credentials.

Url format:

https://q.services.ird.govt.nz/ms_oauth/oauth2/endpoints/oauthservice/tokens

```
Request headers: {Authorization=[Basic
VGVzdDMwMjA2NDkyOk9hdXRoMklSU2VjcmV0dA==], Content-Type=[application/x-www-form-urlencoded], Content-Length=[1127]}
```

Request body: grant_type= oracle-idm:/oauth/grant-type/resource-access-token/jwt&oracle_token_action=delete&assertion=eyJhbGciOiJSUzUxMiIsInR5cCI6IkpXVCIsIng1dCI6Ino5VFR1R2pHSVJNd01pQjRaUGxjbzhDZ3JlOCIsImtpZCI6Im9yYWtleSJ9.eyJzdWIiOm51bGwsIm9yYWNsZS5vYXV0aC51c2VyX29yaWdpbl9pZF90eXBlIjoiTERBUF9VSUQiLCJvcmFjbGUub2F1dGgudXNlcl9vcmlnaW5faWQiOiJUUjIONTczNzczIiwiaXNzIjoiSW5sYW5kUmV2ZW51ZSIsIm9yYWNsZS5vYXV0aC5ydC50dGMiOiJyZXNvdXJjZV9hY2Nlc3NfdGsiLCJvcmFjbGUub2F1dGguc3ZjX3BfbiI6Ik9BdXRoU2VydmljZVByb2ZpbGUiLCJpYXQiOjE1MTMxMTM0OTgsIm9yYWNsZS5vYXV0aC50a19jb250ZXh0IjoicmVmcmVzaF90b2tlbiIsImV4cCI6MTUxMzE0MjI5OCwicHJuIjpudWxsLCJqdGkiOiJmNDZmMTU1YS1jNGU4LTRkYTktOTFkMy0zMjUxZTIyNDNhNjgiLCJvcmFjbGUub2F1dGguc2NvcGUiOiJNWU1SLlNlcnZpY2VzIiwib3JhY2xlLm9hdXRoLmNsaWVudF9vcmlnaW5faWQiOiJNWU9CMTI3NTQyMSIsInVzZXIudGVuYW50Lm5hbWUiOiJEZWZhdWx0RG9tYWluIiwib3JhY2xlLm9hdXRoLmlkX2RfaWQiOiIxMjMONTY3OC0xMjMOLTEyMzQtMTIzNC0xMjMONTY3ODkwMTIifQ.bsqzeC1BWEAireUWCLq3ZGhbSjcb6jrkfbV7F8lLuujVMF7xlGkOMoSLx9DSN_ePPE4z7PeDbz3igF269siiuplkitbA6JZiR6DRs3p19ED3MCjwzp4DVxXuipOK0FOamwOwzyROzSupjYsEEkJhkGOur2SGdTnjHFyXFoleoU

3.5.2 Revoke token reply

A sample response payload:

```
{
"successful":true
}
```

3.5.3 Revoke token Error Response.

Sample payloads when there is an error:

```
"error": "invalid_grant",
    "error_description":
"oracle.security.jps.internal.trust.token.TokenProviderException: Validate
operation failed."
}
```

UNCLASSIFIED Page 19 of 22



4 Appendix B - Glossary

<Terminology used in this document>

| Term | Meaning | |
|-----------------------|---|--|
| Abbreviation/Term | Description | |
| Client Application | A Client Application is an operating instance of Software that is deployed in one or more sites. | |
| | A number of deployment patterns are possible: | |
| | A single cloud based instance with multiple tenants and online users, An on-premise instance (e.g. an organisation's payroll system) A desktop application with an online user. | |
| Customer | A Customer is the party who is a tax payer or a participant in the social policy products that are operated by Inland Revenue. The Customer might be a person (an "individual") or a non-individual entity such as a company, trust, society etc. | |
| | Practically all of the service interactions with Inland Revenue are about a Customer (e.g. their returns, accounts, entitlements etc) even though these interactions might be undertaken by an Intermediary on their behalf. | |
| Intermediary | A party who interacts with Inland Revenue on behalf of a Customer. Inland revenue's Customer is a Client of the Intermediary. There are several types of Intermediary including Tax Agents, PTSIs, PAYE Intermediaries etc. | |
| Mutual authentication | refers to two parties authenticating each other at the same time, being a default mode of authentication in some protocols (e.g. SSH) and optional in other (TLS) | |
| OAuth 2.0 | OAuth 2.0 is an industry-standard protocol for authorization | |
| Native app | An application that is installed by the user to their device, as distinct from a web app that runs in the browser | |
| Protected Service | A general term for the business related web services that are accessed once authentication has occurred (e.g. the Return Service, the Intermediation Service, the Correspondence Service). This document describes the mechanisms that are used to authenticate access to Protected Services. | |
| SFTP | Secure File Transport Protocol | |
| Software | This is the computer software that contains interfaces to (consume) the services that Inland Revenue exposes. Software is developed and maintained by a Software Developer and subsequently deployed as one or more Client Applications. | |

UNCLASSIFIED Page 20 of 22



| Software Developer | The person or people who design, implement and test Software. This build pack and the resources to which it refers are primarily focused on the needs of Software Developers. They might be commercial vendors of software or an in-house developer of software. |
|--------------------|--|
| TLS 1.2 | A cryptographic protocol that provides communications security over a computer network. Version 1.2 is mandated in most cases. |
| WS-Security | An extension to SOAP to apply security to Web Services. An OASIS Web service specification |
| X.509 Certificate | A digital certificate that uses the widely accepted international X.509 public key infrastructure (PKI) standard to verify that a public key belongs to the user, computer or service identity contained within the certificate. |
| | |

UNCLASSIFIED Page 21 of 22



5 Appendix C—Change log

This table lists all changes that have been made to this build pack document since v $1.5\,04/09/2017$.

| Version | Date of Change | Document Section | Description |
|---------|-------------------|--|--|
| 1.6 | 15/09/2017 | Error! Reference source not found. 2.1.4 | Remove reference to Refresh and Revoke token until further discussions and agreement with Service Providers Amend Incorrect Scope value from "GWS" to "MYIR.Services" |
| 1.7 | 27/09/2017 | Multiple Appendix A – Sample payloads | Token time-out updated to 8 hours. Minor wording changes for ease of reading. Corrections to sample payloads where symbols had not correctly translated to word format e.g. https:// |
| 1.8 | 11/12/2017 | 2.1.4 2.1.5 3.4 3.5 | Added Refresh and Revoke token details and examples in the appendix. Minor update to Service structure. Highlighted in yellow |
| 1.8.1 | 23/01/2018 | 2.1.4 | Commentary added for refresh token validity Highlighted in yellow |
| 1.9 | 05/02/2018 | 2.2 | Further detail for Service Providers offering desktop solutions regarding OAuth for Native apps. Highlighted in blue |

UNCLASSIFIED Page 22 of 22