

Understanding OIDC Settings and Security Terms

Overview

When configuring OpenID Connect, applications may ask for settings such as grant type, scopes, client authentication, PKCE, S256, or RS256.

These terms describe different parts of the authentication process. They are not interchangeable.

This guide explains what each setting means and how it relates to ID Everywhere.

Client ID

The Client ID identifies an application registered with ID Everywhere.

It is not a password.

Both public and confidential applications use a Client ID.

Client Secret

A Client Secret is a credential used by a confidential application to authenticate itself to the token endpoint.

A Client Secret must be stored securely on a trusted server.

Public applications do not use a Client Secret.

Never publish or embed a Client Secret in browser JavaScript, mobile applications, or desktop software distributed to users.

Authorization Code grant

A grant type describes how an application obtains tokens.

ID Everywhere's current OIDC integration uses the **Authorization Code** grant.

The process is:

1. The application redirects the user to ID Everywhere.
2. The user authenticates.
3. ID Everywhere redirects back with an authorization code.
4. The application exchanges the code for tokens.
5. The application validates the authentication result.

An authorization code is temporary and intended for a single exchange.

Other OAuth grant types should not be selected unless ID Everywhere explicitly supports them.

PKCE — Proof Key for Code Exchange

PKCE adds protection to the Authorization Code flow.

It helps prevent a stolen or intercepted authorization code from being exchanged by a party that does not possess the original code verifier.

PKCE uses two important values:

Code Verifier: A cryptographically random value generated by the client.

Code Challenge: A value derived from the verifier and included in the authorization request.

During the token exchange, the client sends the verifier. The authorization server checks that it matches the original challenge.

PKCE is required for public clients in the documented IDE configuration.

Confidential clients may also use PKCE when supported.

SHA-256 and S256

SHA-256 is a cryptographic hashing algorithm.

S256 is the PKCE code challenge method that uses SHA-256.

Conceptually:

```
code_challenge = BASE64URL(SHA256(code_verifier))
```

In an application's configuration screen, you might see:

- S256
- SHA-256
- PKCE SHA-256

These generally refer to the same PKCE challenge method.

SHA-256 is not an authentication protocol or OAuth grant type.

RS256

RS256 is a digital signature algorithm based on RSA and SHA-256.

ID Everywhere advertises RS256 for signing OIDC ID tokens.

Applications can retrieve the provider's public signing keys through the JWKS endpoint and use them to verify token signatures.

RS256 and S256 are different:

Setting	Purpose
S256	Protects the authorization-code exchange through PKCE
RS256	Signs and verifies identity tokens
SHA-256	Underlying cryptographic hash algorithm

Do not select RS256 as the PKCE challenge method.

Scopes

Scopes indicate which information or permissions the client requests.

ID Everywhere advertises:

`openid`

Required to request OIDC authentication.

`profile`

Requests standard profile claims, such as name information.

`email`

Requests email-related identity claims.

`groups`

Requests group-related information where available.

A typical sign-in request uses:

`openid profile email`

Additional claims depend on the granted scopes and the provider's implementation.

ID Token

An ID Token communicates authentication and identity information to the OIDC client.

ID tokens are commonly represented as JSON Web Tokens (JWTs).

Typical claims include:

- `iss` — Issuer
- `sub` — Unique user subject identifier
- `aud` — Intended audience, typically the Client ID
- `exp` — Expiration time
- `iat` — Issued-at time

An application must validate the ID token's signature, issuer, audience, expiration, and other applicable requirements, including nonce validation when used.

Access Token

An Access Token is used to access a protected endpoint.

For example, an application can send an access token to ID Everywhere's UserInfo endpoint to request authorized user information.

An access token is not the same as a Client Secret or an ID Token.

Do not assume access tokens and ID tokens can be used interchangeably.

Refresh Token

A Refresh Token can allow an application to request new access tokens without repeating interactive sign-in.

However, **refresh-token support is not currently advertised in ID Everywhere's documented OIDC discovery capabilities.**

Do not enable refresh-token-dependent behavior without confirming that the integration supports it.

Redirect URI

A Redirect URI is the registered destination that receives the browser after authorization.

Example:

```
https://portal.example.com/auth/callback
```

The redirect URI must match an authorized URL registered in ID Everywhere.

It is a security-sensitive setting.

Issuer

The issuer identifies the OIDC provider that created the identity token.

ID Everywhere's issuer is:

```
https://app.ideverywhere.com
```

The application should validate that the ID token's issuer matches the expected issuer.

Discovery Document

The discovery document provides standardized information about the identity provider.

ID Everywhere discovery URL:

```
https://app.ideverywhere.com/.well-known/openid-configuration
```

Applications supporting OIDC discovery can use this URL to learn the provider's supported endpoints and capabilities.

JWKS — JSON Web Key Set

JWKS publishes public keys that applications can use to verify token signatures.

ID Everywhere JWKS URL:

```
https://app.ideverywhere.com/.well-known/jwks.json
```

Public signing keys can be shared. Private signing keys must remain protected.

Client authentication methods

Client authentication describes how a confidential application proves its identity when calling the token endpoint.

ID Everywhere advertises:

- `client_secret_basic` — Sends client credentials using HTTP Basic authentication.
- `none` — Used by public clients that do not authenticate using a client secret.

These methods should not be confused with the user's sign-in method.

Frequently asked questions

Is PKCE required for all applications?

It is required for public applications in the documented IDE integration. It can also provide additional protection for confidential clients.

Should I select S256 or RS256 in Postman?

For the **PKCE Code Challenge Method**, select **S256 / SHA-256**.

RS256 relates to ID-token signatures, not the PKCE challenge setting.

Is OAuth 2.0 the same as OIDC?

No. OIDC adds authentication and identity information to OAuth 2.0.

Is the Client ID secret?

No. The Client ID identifies the application. The Client Secret is confidential.

Related articles

- [Understanding Authentication Methods in ID Everywhere](#)
- [Choosing the Right OIDC Application Type](#)
- [Testing ID Everywhere OIDC with Postman](#)

Revision #1

Created 2026-10-10 22:11:30 UTC by Administrator

Updated 2026-10-10 22:11:42 UTC by Administrator