

Troubleshooting OIDC Connections

Overview

This guide covers common problems when connecting an application to ID Everywhere using OpenID Connect.

Many connection issues are caused by incorrect redirect URLs, missing scopes, mismatched client credentials, or unsupported OAuth settings.

Start by identifying the exact error returned by the application or identity provider.

1. Error: `invalid_request`

Possible causes

- The authorization request is missing required parameters.
- The `openid` scope was omitted.
- The application is using an unsupported authorization flow.
- Required PKCE parameters are missing or malformed.

Recommended steps

1. Confirm the application uses the Authorization Code flow.
2. Include `openid` in the requested scopes.
3. Verify the Client ID.
4. Confirm the redirect URI is registered.
5. For public clients, confirm PKCE S256 is enabled.
6. Retry with a new authorization request.

A common working scope configuration is:

2. Redirect URI mismatch

Symptoms

Authentication is rejected before completion, or the application reports an invalid callback URL.

Possible causes

The callback URL supplied by the application does not match the registered redirect URL.

Recommended steps

Compare the URLs character by character.

Check:

- HTTP versus HTTPS.
- Hostname and subdomain.
- Port number.
- URL path.
- Trailing slash.
- Query parameters.

For example, these are different redirect URLs:

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

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

Do not add wildcard redirect URLs to work around mismatches.

3. Error: invalid_client

Possible causes

- Incorrect Client ID.
- Incorrect or outdated Client Secret.
- Wrong application type.
- Unsupported client authentication method.
- Credentials belonging to another application registration.

Recommended steps

1. Verify the Client ID in ID Everywhere.
2. Confirm whether the client is Public or Confidential.
3. For confidential clients, verify the Client Secret.
4. Confirm the application uses an appropriate token endpoint authentication method.
5. For public clients, remove any unnecessary Client Secret.
6. If the secret may have been exposed, rotate it and update the connected application.

4. Error: invalid_grant

Possible causes

- The authorization code has expired.
- The code has already been used.
- The PKCE verifier does not match the original challenge.
- The redirect URI differs between authorization and token exchange.
- The code was issued to a different client.

Recommended steps

1. Start a completely new sign-in attempt.
2. Do not reuse an authorization code.
3. Confirm the same Client ID is used throughout the flow.
4. Verify the redirect URI remains consistent.
5. Confirm PKCE S256 is configured correctly.
6. Ensure the application and server clocks are accurate.

5. Error: invalid_scope

Possible causes

The application requests a scope that is not supported or permitted.

Recommended steps

Start with:

openid profile email

ID Everywhere currently advertises:

- openid
- profile
- email
- groups

Remove unsupported scopes and retry.

Only request `groups` if the application needs group-related information and the integration supports the required claim behavior.

6. Error: Token signature validation failed

Possible causes

- The application is using the wrong issuer.
- It has outdated signing keys.
- The expected audience does not match.
- The token is expired.
- The application is configured with the wrong signing algorithm.

Recommended steps

Verify the expected issuer:

<https://app.ideverywhere.com>

Verify the JWKS endpoint:

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

Confirm that the application:

1. Retrieves the correct public signing keys.
2. Validates the token signature.
3. Validates the issuer (`iss`).
4. Validates the audience (`aud`).
5. Checks expiration (`exp`).
6. Performs any other required OIDC validation.

ID Everywhere advertises RS256 for ID-token signatures.

Never disable signature verification as a troubleshooting workaround.

7. Error: Unauthorized UserInfo request

Possible causes

- Missing Bearer token.
- Expired or invalid access token.
- Sending the wrong token type.
- User or client access restrictions.

Recommended steps

Make a GET request to:

```
https://app.ideverywhere.com/oauth2/userinfo
```

Include:

```
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Replace the placeholder with a valid access token.

Do not use the Client Secret or ID Token in place of the access token.

8. User cannot sign in

Possible causes

- Incorrect credentials.
- Disabled or otherwise ineligible account.
- Application or tenant access restrictions.
- Incomplete authentication requirements.

Recommended steps

1. Confirm the user can sign in to ID Everywhere normally.
2. Verify the user's account is active and authorized.
3. Confirm the correct tenant and application registration.
4. Review relevant authentication and audit information available to administrators.
5. Retry after resolving the underlying account issue.

Do not repeatedly retry a locked or restricted account without determining the cause.

9. Application asks for unsupported settings

Some applications offer multiple authentication protocols and grant types.

SAML metadata URL

SAML is not interchangeable with OIDC. The documented IDE integration does not currently provide SAML configuration.

OAuth 1.0 credentials

OAuth 1.0 is not supported by the documented integration.

Client Credentials grant

The documented OIDC integration uses Authorization Code, not Client Credentials.

Implicit or Password grant

These are not supported by the current documented IDE integration.

Refresh Token required

Refresh-token support is not currently advertised. Confirm the connected application's requirements before proceeding.

10. Third-party testing tool reports an error

Some online OIDC testing tools use their own backend services to exchange authorization codes.

An error from the testing tool does not necessarily indicate an error in ID Everywhere.

For example, a tool may reject a URL through its own security validation before contacting the provider.

If this happens:

1. Identify which service generated the error.
2. Verify ID Everywhere's discovery document.
3. Retry using Postman or another trusted OIDC-compatible client.
4. Compare the authorization and token requests.
5. Avoid exposing credentials in publicly accessible testing tools.

Information to provide when requesting support

To help diagnose a connection issue, provide:

- Application name.
- Client ID (not the Client Secret).

- Application type (Public or Confidential).
- Requested scopes.
- Grant type.
- Redirect URL.
- Exact error code and message.
- Approximate time of failure, including timezone.
- Whether the failure occurs before sign-in, after sign-in, or during token exchange.

Never send passwords, Client Secrets, authorization codes, access tokens, ID tokens, or private signing keys to support.

Related articles

- [Understanding Authentication Methods in ID Everywhere](#)
- [Choosing the Right OIDC Application Type](#)
- [Connecting an Application Using OIDC](#)
- [Understanding OIDC Settings and Security Terms](#)
- [Testing ID Everywhere OIDC with Postman](#)

Revision #1

Created 2026-10-10 22:12:18 UTC by Administrator

Updated 2026-10-10 22:12:29 UTC by Administrator