OpenID Connect

by Roland Hedberg

From OAuth2 to OpenID Connect

The visionaries

The vision

- ’OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol.’

- It enables Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner.

The differences

- Only authorization grant and implicit grant flows
- Dynamic provider discovery and client registration
- ID Token
- Additions/Clarifications/Constrictions
- UserInfo endpoint

Flows

- Authorization code
  - code
- Implicit
  - id_token
  - id_token token
- *Hybrid* (authorization code with a twist)
  - code id_token
  - code token
  - code id_token token

Dynamic provider discovery and client registration

Dynamic discovery and registration

A. Find the provider

B. Discover provider configuration

C. Register client information

1.Find the provider

- Webfinger (RFC 7033)
  - User identifier -> URL
  - carol@example.com ->

GET /.well-known/webfinger?

resource=acct:carol@example.com&

rel=http://openid.net/specs/connect/1.0/issuer

Host: example.com

RFC 5785

Defines a path prefix for "well-known locations”, "/.well-known/", in selected Uniform Resource Identifier (URI) schemes.

Created with the expectation that it will be used to make site-wide policy information and other metadata available directly (if sufficiently concise), or provide references to other URIs that provide such metadata.

Webfinger response

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/jrd+json


```
{
   "subject" : "acct:carol@example.com",
   "links" :
   [ {
      "rel" : "http://openid.net/specs/connect/1.0/issuer",
      "href" : "https://openid.example.com"
   } ]
}
```

2.Discover provider info - query

```
GET /.well-known/openid-configuration HTTP/1.1
Host: openid.example.com
```

2. Discover provider info - response

- issuer
- jwks_uri
- *endpoints*
- *functions supported*
- *support for signing/encrypting algorithms*
- *policy/tos*

Required information

- issuer
- jwks_uri
- authorization_endpoint
- token_endpoint (*)
- response_types_supported
- subject_types_supported
- id_token_signing_alg_supported

demo

3. Client registration

- uris
- application information
- support for signing/encrypting algorithms
- key material
- server behaviour
- client behaviour

required information

- redirect_uris

Client registration response

- client_id
- possibly client_secret *and if so* client_secret_experies_at
- and the Authorization servers view of things

An Authorization Server

- MAY add fields the client didn’t include.

- MAY reject or replace any of the Client's requested field values and substitute them with suitable values.

- MAY ignore values provided by the client, and MUST ignore any fields sent by the Client that it does not understand.

demo

A OIDC Client can **not**

- modify a registration
- delete a registration

ID Token

ID Token

- A security token that contains Claims about the **Authentication** of an End-User by an Authorization Server when using a Client, and potentially other requested Claims.

- Is represented as a JSON Web Token (JWT)

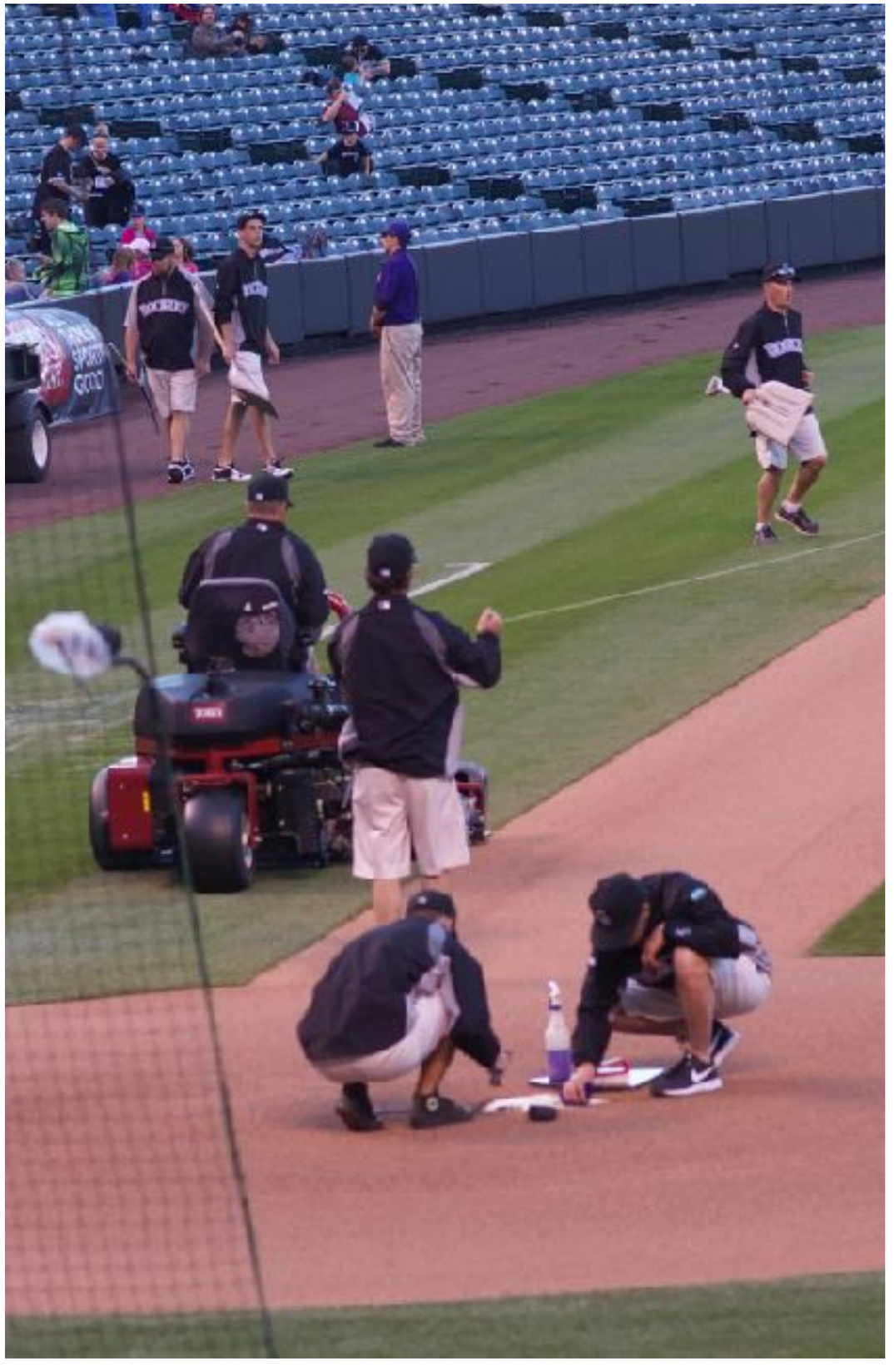
ID Token claims -required

- iss - Issuer Identifier for the Issuer of the response
- sub - Subject Identifier
- aud - Intended audience
- exp - Expiration time
- iat - Issued at
- auth_time - Authentication time
- nonce

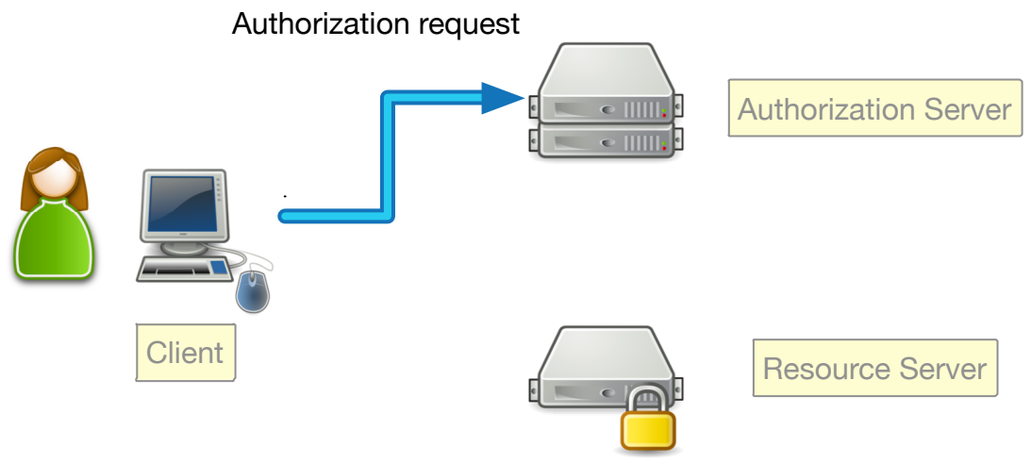
ID Token claims - optional

- acr - Authentication Context Class Reference
- amr - Authentication Method References
- azp - Authorized party

Additions/ Clarifications/ Constrictions



OAuth2 Authorization Request - details



Parameters

client_id

redirect_uri

response_type

scope

state

GET
http://example.com/authorization?state=15216719803168020 35&
redirect_uri=https://example.org/authz_cb&
response_type=code&
client_id=SFEBuhC7sp3a

Authentication Request - OpenID Connect extensions

- response_mode - The mechanism to use for returning parameters
- nonce - Associates client session with ID Token
- *Signed/encrypted Authentication Request*
- *End-user interactions*
- *Response details*

Signed/encrypted Authentication Request

- request - by value
- request_uri - by reference

- Single self-contained parameter
- Signed and/or encrypted (JWT)

End-user interactions

- display - How to display pages to End-User
- prompt - If the End-User should be prompted for re-authentication/ consent
- max_age - allowed max time since last authentication
- ui_locales - End-User’s preferred languages and scripts
- id_token_hint - ID Token previously issued
- login_hint - login identifier the End-User might want to use
- acr_values - requested Authentication Context Class Reference values

Response details

- claims
  - user_info
  - id_token

- claims specification
  - null
  - essential
  - value
  - values

Requested Claims example


```
{
    ”userinfo”: {
        "given_name": {"essential": true},
        "nickname": null,
        "email": {"essential": true},
        "email_verified": {"essential": true},
        "picture": null,
        "http://example.info/claims/groups": null
    },
    "id_token": {
        "auth_time": {"essential": true},
        "acr": {"values": ["urn:mace:incommon:iap:silver"] }
    }}
```

UserInfo endpoint

User info

Set of standard claims

- sub
- name
- given_name
- family_name
- middle_name
- nickname
- preferred_username
- profile
- picture
- website
- email
- email_verified
- gender
- birthdate
- zoneinfo
- locale
- phone_number
- phone_number_verfied
- address
- updated_at

claims types

- Normal
- Aggregated
- Distributed

Aggregated claims - example


```
{
   "name": "Jane Doe",
   "given_name": "Jane",
   "family_name": "Doe",
   "birthdate": "0000-03-22",
   "eye_color": "blue",
   "email": "janedoe@example.com",
   "_claim_names": {
      "address": "src1",
      "phone_number": "src1"
   },
   "_claim_sources": {
      "src1": {"JWT": "jwt_header.jwt_part2.jwt_part3"}
   }
}
```

Distributes Claims - example


```
{
   "name": "Jane Doe",
   "given_name": "Jane",
   "family_name": "Doe",
   "email": "janedoe@example.com",
   "birthdate": "0000-03-22",
   "eye_color": "blue",
"_claim_names": {
   "payment_info": "src1",
   "shipping_address": "src1",
   "credit_score": "src2"
},
"_claim_sources": {
   "src1": {"endpoint":
              "https://bank.example.com/claim_source"},
    "src2": {"endpoint":
              "https://creditagency.example.com/claims_here",
              "access_token": "ksj3n283dke"}
}}
```

Links to documents

- OpenID Connect Core 1.0 incorporating errata set 1
- OpenID Connect Discovery 1.0 incorporating errata set 1
- OpenID Connect Dynamic Client Registration 1.0 incorporating errata set 1