XAuth

IETF 107 - TxAuth BoF
March 23, 2020

dick.hardt@gmail.com
XAuth Goals

- **Extensible** - interactions, authorizations, claims, client authN
- **Migration** - from OAuth 2.0 and OpenID Connect
- **Reuse** - build on what has come before
- **Scalable** - decomposable architecture, separation of concerns
- **Simple** - simple things are simple
- **Flexible** - hard things are possible
Parties
Key Terms

- **Claims** - information / assertions about the User provided by GS
- **Authorization** - Client access to a RS provided by GS.
- **Grant** - Collection of authorizations and/or claims. Issue by Grant Server (GS)
- **GS** - OAuth Authorization Server (**AS**) & OpenID Connect **OP** (OpenID Provider)
- **Interaction** - how User is directed to the GS to authorize a Grant
General Sequence

Client

(--1)--- Create Grant --------->

<--- Interaction Response ---(2)---

(--3)--- Redirect -------------

<--- Redirect -----------------(6)-

(--7)--- Read Grant ----------->

<------- Grant Response ---(8)--

GS

+-------+

User

---(4)--> authN

---(5)--> authZ

+-------+

+-------+
(1) Create Grant

HTTP POST to GS URI

POST /endpoint
Host: gs.example

{
   "iat" : 15790460234,
   "uri" : "https://gs.example/endpoint",
   "nonce" : "f6a60810-3d07-41ac-81e7-b958c0dd21e4",
   "client": {
      "id": "di3872h34dkJW"
   },
   "interaction": {
      "type" : "redirect",
      "completion_uri" : "https://web.example/return/36489b95-2cfc-4594"
   },
   "authorization": { ... },
   "claims" : { ... }
}
(2) Interaction Response

GS creates Grant URI and Redirect URI

```
{
  "iat" : 15790460234,
  "nonce" : "f6a60810-3d07-41ac-81e7-b958c0dd21e4",
  "uri" : "https://gs.example/endpoint/G/9006314c-fdc3-43a9-819a-5289e0b15901",
  "interaction" : {
    "type" : "redirect",
    "redirect_uri" : "https://gs.example/I/ebb55ecb-76ff-4305-93be-a5a49b71a1b8"
  }
}
```
General Sequence

Client

---(1)--- Create Grant --------------->

<--- Interaction Response ---(2)---

---(3)--- Redirect -------------------

<--- Redirect -------------------(6)-

---(7)--- Read Grant --------------->

<-------- Grant Response ---(8)---

GS

User

---(4)---

authN

<---(5)---

authZ

---(6)---
(7) Read Grant

HTTP GET to Grant URI

GET /endpoint/G/9006314c-fdc3-43a9-819a-5289e0b15901
Host: gs.example

(8) Grant Response

{
    "iat" : 15790460234,
    "uri" : "https://gs.example/endpoint/G/9006314c-fdc3-43a9-819a-5289e0b15901",
    "nonce" : "f6a60810-3d07-41ac-81e7-b958c0dd21e4",
    "authorization" : { ... },
    "claims" : { ... }
}
Request Client Object

- Registered Client

```
{
    "client": {
        "id": "di3872h34dkJW"
    }
}
```

- Dynamic Client

```
{
    "client": {
        "display": {
            "name": "SPA Display Name",
            "uri": "https://spa.example/about"
        }
    }
}
```
Request Interaction Object

- Redirect Interaction

  ```json
  {
    "interaction": {
      "type": "redirect",
      "completion_uri": "https://client.example/complete/3902f52"
    }
  }
  ```

- Indirect Interaction

  ```json
  {
    "interaction": {
      "type": "indirect",
      "information_uri": "https://client.example/info"
    }
  }
  ```

- Extension point for new interaction types
Request Authorization Object

- OAuth 2.0 Scope Type

```json
{
    "authorization": {
        "type": "oauth_scope",
        "scope": "read_calendar"
    }
}
```

- Other Authorization Types
  - "oauth_rar" - Rich Authorization Request
  - XYZ style - "actions", "locations", "datatypes"
  - Extension point for new authorization types
Multiple Authorizations

• "authorizations" vs "authorization"

```json
{
    "authorizations": [
        { <authorization object> },
        { <authorization object> }
    ]
}
```

or change to

```json
{
    "authorizations": {
        "token1": { <authorization object> },
        "token2": { <authorization object> }
    }
}
```

• or only "authorizations" with polymorphism - object, or array of objects
Request Claims Object

• OpenID Connect Claims

{
    "claims": {
        "oidc": {
            "id_token": {
                "email": { "essential": true },
                "email_verified": { "essential": true }
            },
            "userinfo": {
                "name": { "essential": true },
                "picture": null
            }
        }
    }
}

• Other Claim Types

• "oidc4ia" - OpenID Connect for Identity Attestation

• "vc" - W3C Verified Credentials

• Extension point for new claim types
Response Authorization Object

- Refreshable Authorization (AZ URI is returned)

```
{
    "authorization": {
        "uri": "https://gs.example/endpoint/A/644adacc-3183-4f02-84f3-9780a16a59a6"
    }
}
```

or Authorization JSON (without AZ URI) if single use or limited time

```
{
    "authorization": {
        "type": "oauth_scope",
        "scope": "read_contacts",
        "expires_in": 3600,
        "mechanism": "bearer",
        "token": "eyJ2D6.example.access.token.mZf9p"
    }
}
```

- Extension point for other RS access mechanisms eg: "jose"
Response Claims Object

• OpenID Connect Claims

```json
{
  "claims": {
    "oidc": {
      "id_token": "eyJhbUzI1N.example.id.token.YRw5DFdbW",
      "userinfo": {
        "name": "John Doe",
        "picture": "https://photos.example/p/eyJzdkiO"
      }
    }
  }
}
```

• Extension point for other claims eg: "oidc4ia" or "vc"
Read Access Token

HTTP GET to AZ URI

GET /endpoint/A/644adacc-3183-4f02-84f3-9780a16a59a6
Host: gs.example

returns

{
    "uri" : "https://gs.example/endpoint/A/644adacc-3183-4f02-84f3-9780a16a59a6",
    "type" : "oauth_scope",
    "scope" : "read_contacts",
    "expires_in" : 3600,
    "mechanism" : "bearer",
    "token" : "eyJJJ2D6.example.access.token.mZf9p"
}

Access Token Refresh is same call
HTTP OPTIONS to GS URI

OPTIONS /endpoint
Host: gs.example

returns

{
    "uri" : "https://gs.example/endpoint",
    "interactions": [
        "redirect",
        "registered",
    ],
    "clients": [
        "registered",
    ],
    "client_authentication": [
        "jose",
    ],
    "authorization": { ... },
    "claims" : { ... },
    "features": {
        "user_exists": true,
        "keep_interaction": true,
        "authorizations": true,
    }
}
# GS APIs

<table>
<thead>
<tr>
<th>request</th>
<th>http verb</th>
<th>URI</th>
<th>response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Grant</td>
<td>POST</td>
<td>GS URI</td>
<td>interaction, wait, or grant</td>
</tr>
<tr>
<td>Verify Grant</td>
<td>PATCH</td>
<td>Grant URI</td>
<td>grant</td>
</tr>
<tr>
<td>Read Grant</td>
<td>GET</td>
<td>Grant URI</td>
<td>wait or grant</td>
</tr>
<tr>
<td>Update Grant</td>
<td>PUT</td>
<td>Grant URI</td>
<td>interaction, wait, or grant</td>
</tr>
<tr>
<td>Delete Grant</td>
<td>DELETE</td>
<td>Grant URI</td>
<td>success</td>
</tr>
<tr>
<td>Read Authorization</td>
<td>GET</td>
<td>AZ URI</td>
<td>authorization</td>
</tr>
<tr>
<td>Update Authorization</td>
<td>PUT</td>
<td>AZ URI</td>
<td>authorization</td>
</tr>
<tr>
<td>Delete Authorization</td>
<td>DELETE</td>
<td>AZ URI</td>
<td>success</td>
</tr>
<tr>
<td>GS Options</td>
<td>OPTIONS</td>
<td>GS URI</td>
<td>metadata</td>
</tr>
<tr>
<td>Grant Options</td>
<td>OPTIONS</td>
<td>Grant URI</td>
<td>metadata</td>
</tr>
<tr>
<td>Authorization Options</td>
<td>OPTIONS</td>
<td>AZ URI</td>
<td>metadata</td>
</tr>
</tbody>
</table>
URI Summary

- GS URI: https://gs.example/endpoint
- Grant URI: https://gs.example/endpoint/G/8e3a6354
- AZ URI: https://gs.example/endpoint/A/fad923b4
- Redirect URI: https://gs.example/R/f52d256a-4d19-4284
- Short URI: https://gs.example/S/5d2f63
- Completion URI: https://client.example/complete/3902f52
- Information URI: https://client.example/info
Other Features

• Wait Response
• Update, Verify, Delete Grant
• Update, Delete Authorization
• GS initiated Grant Creation
• Reciprocal Delegation
• user.exists
• interaction.keep
JOSE AS Client Authentication

• Header (GET, DELETE, OPTIONS)

```
"payload":{
    "iat" : 15790460234,
    "jti" : "f6d72254-4f23-417f-b55e-14ad323b1dc1",
    "uri" : "https://gs.example/endpoint/G/9006314c-fdc3-43a9-819a-5289e0b15901",
    "verb" : "GET"
}
```

GET /endpoint/G/9006314c-fdc3-43a9-819a-5289e0b15901
Host: gs.example
Authorization: jose eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c

• Body (POST, PUT, PATCH)

```
POST /endpoint
Host: as.example
Content-Type: application/jose
Content-Length: 155

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```
JOSE RS Client Authentication

- Authorization header used for all HTTP verbs

```json
"header":{
  "alg"   : "ES256",
  "typ"   : "JOSE",
  "x5u"   : "https://as.example/cert/example2"
}

"payload":{
  "iat"   : 15790460234,
  "jti"   : "f6d72254-4f23-417f-b55e-14ad323b1dc1",
  "uri"   : "https://calendar.example/calendar",
  "verb"  : "GET",
  "token" : "eyJhbG.example.access.token.mZf9pTSpA"
}
```

GET /calendar HTTP/2
Host: calendar.example
Authorization: jose eyJhbG.example.jose.token.adks
Review XAuth Goals

- **Extensible** - interactions, authorizations, claims, client authN
- **Migration** - from OAuth 2.0 and OpenID Connect
- **Reuse** - build on what has come before
- **Scalable** - decomposable architecture, separation of concerns
- **Simple** - simple things are simple
- **Flexible** - hard things are possible
Extensible

- Request
  - client information
  - interaction types
  - authorization types
  - claim schemas
  - new top level objects

- Client AS Authentication
  - JOSE, HTTP Sig, MTLS

- Response
  - interaction parameters
  - authorization types
  - RS access mechanisms
    - bearer, jose, HTTP sig, MTLS
  - claim schemas
  - new top level objects
OAuth/OIDC Migration

"client": {
   "id": <existing client id>
}
"authorization": {
   "type": "oauth_scope",
   "scope": <existing oauth scopes>
}
"claims": {
   "oidc": {
      "userinfo": { <OIDC claims> }
      "id_token": { <OIDC claims> }
   }
}
}
Reuse

- JSON, TLS, HTTP
- HTTP verbs for RESTful API
- OAuth 2.0 client id and scopes
- OpenID Connect client id and claims
- JOSE
Scalable

• **Grant Server**
  - Routing at HTTP method and URI path
  - Signed header and body allows independent verification
  - proof-of-possession authentication vs shared secret

• **Client**
  - certificate chain and proof-of-possession vs shared secret

• **Resource Server**
  - certificate from GS delegates Client identification with JOSE mechanism
Simple and Flexible?
Questions?