Client

“App”

Authorization Server

“App Backend/API”
Client

“App”

Authorization Server

“App Backend/API”

Identity Provider or Security Monitoring Tools

Revoke Tokens!
Goal

Create a token revocation API that provides existing applications with the shortest path to implement for interoperability.
Existing Token Revocation / Logout Standards

- RFC 7009: Token Revocation
- OpenID Connect Front-Channel Logout
- OpenID Connect Back-Channel Logout
- OpenID Shared Signals Framework: CAEP / RISC
RFC 7009: Token Revocation?

- RFC 7009 is client-initiated
- Input to RFC 7009 is the access token itself
  - The OAuth client tells the Authorization Server to revoke a token
  - We want to be able to call this from parties other than an OAuth client
OpenID Connect Front-Channel Logout?

- Front-Channel Logout is client-initiated
OpenID Connect Back-Channel Logout?

- Mostly talks about revoking sessions, no mention of access tokens, barely a mention of refresh tokens

  "Refresh tokens issued with the offline_access property normally SHOULD NOT be revoked"

- Many of these authorization servers don't do OpenID Connect
  - They might be only OAuth authorization servers
  - They might integrate upstream with SAML providers, not OpenID

- Input to Back-Channel Logout is a JWT, more work to validate than other options if you don't already support OpenID Connect
Shared Signals Framework?

- CAEP (Continuous Access Evaluation Profile) is more of a hint/suggestion
- RISC (Risk Incident Sharing and Coordination) has somewhat stronger language than CAEP
- Both require significant infrastructure setup to receive these events
## Existing Token Revocation APIs

<table>
<thead>
<tr>
<th>App</th>
<th>Input</th>
<th>API</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Zoom</strong></td>
<td>User ID or email</td>
<td>DELETE /users/{userId}/token</td>
</tr>
</tbody>
</table>
| **Box** | User ID or email             | POST /2.0/users/terminate_sessions  
user_ids= ["1234"]  
user_logins= ["user@example.com"] |
| **Slack**| User ID                      | POST /api/admin.users.session.reset  
user_id=1234  
mobile_only=true, web_only=true |
| **Zendesk**| User ID & Session ID      | DELETE /api/v2/users/{user_id}/sessions/{session_id}  
Note: first use "list session" API to get session_id |
Global Token Revocation

Abstract

Global Token Revocation enables parties such as a security incident management tool or an external Identity Provider to send a request to an Authorization Server to indicate that it should revoke all of the user's existing tokens and require that the user re-authenticates before issuing new tokens.

Global Token Revocation

Input

- Security Event Token Subject Identifier (draft-ietf-secevent-subject-identifiers)

Authentication:

- Required, but out of scope, just like Token Introspection (RFC 7662)

Outcome:

- MUST revoke refresh tokens
- SHOULD revoke access tokens
- MUST prevent issuing new access tokens and refresh tokens without re-authenticating the user
Global Token Revocation


POST /global-token-revocation
Host: example.com
Content-Type: application/json
Authorization: Bearer f5641763544a7b24b08e4f74045

{
  "subject": {
    "format": "email",
    "email": "user@example.com"
  }
}
Global Token Revocation


POST /global-token-revocation
Host: example.com
Content-Type: application/json
Authorization: Bearer f5641763544a7b24b08e4f74045

{
  "subject": {
    "format": "opaque",
    "email": "U1234567890"
  }
}
Global Token Revocation


HTTP response code indicates success/failure

HTTP/1.1 204 No Content

HTTP/1.1 400 Bad Request

HTTP/1.1 404 Not Found

etc
Next Steps

- Please review the draft!