FAPI updates: What's new in the OpenID and OAuth specs that drive Banking APIs

Nat Sakimura (@_nat_en)
Research Fellow, Nomura Research Institute
Chairman of the board, OpenID Foundation

https://nat.sakimura.org
https://nat.Sakimura.org/youtube.php
https://www.linkedin.com/in/natsakimura
Do you know "FAPI"?
FAPI stands for “Financial-grade API”, aiming at creating medium to high security OAuth and OpenID Connect profiles.

- **Closed circuit Factory application** (High Value, High Environment control level)
- **Financial-grade API** - Read & Write (High Value, Low Environment control level)
- **Financial-grade API** - Read only (Low Value, Low Environment control level)
- **Social sharing** (Low Value, High Environment control level)

Basic choices ok.

Basic choices NOT OK

No need to satisfy all the security requirements by OAuth.
While RFC6749 is not complete with source, destination, and message authentication,

<table>
<thead>
<tr>
<th>AuthZ Req</th>
<th>Sender AuthN</th>
<th>Receiver AuthN</th>
<th>Message AuthN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthZ Req</td>
<td>Indirect</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>AuthZ Res</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Token Req</td>
<td>Weak</td>
<td>Good</td>
<td>Good</td>
</tr>
<tr>
<td>Token Res</td>
<td>Good</td>
<td>Good</td>
<td>Good</td>
</tr>
</tbody>
</table>
FAPI Currently consists of:

4 Implementer’s Drafts:
- Part 1: Read Only Security Profile
- Part 2: Read and Write Security Profile
- (Part 3): Client Initiated Backchannel Authentication Profile
- JWT Secured Authorization Response Mode for OAuth 2.0 (JARM)

- And working on several other documents (in conjunction with other bodies):
  - Cross-Browser Payment Initiation Attack
  - Implementation and Deployment Advice
  - OAuth 2.0 Mutual-TLS Client Authentication and Certificate-Bound Access Tokens (MTLS)
  - OAuth 2.0: JWT Secured Authorization Request (JAR)
  - OAuth 2.0: Pushed Authorization Requests (PAR)
  - OpenID Connect for Identity Assurance 1.0
Part 1: Read Only Security Profile
Part 2: Read and Write Security Profile

Redirect Approach
Decoupled Approach
Embedded Approach
FAPI Part 2 is complete with source, destination, and message authentication.

- By using OpenID Connect’s Hybrid Flow and Request Object, you are pretty well covered.

<table>
<thead>
<tr>
<th></th>
<th>Sender AuthN</th>
<th>Receiver AuthN</th>
<th>Message AuthN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthZ Req</td>
<td>Request Object</td>
<td>Request Object</td>
<td>Request object</td>
</tr>
<tr>
<td>AuthZ Res</td>
<td>Hybrid Flow or JARM</td>
<td>Hybrid Flow or JARM</td>
<td>Hybrid Flow or JARM</td>
</tr>
<tr>
<td>Token Req</td>
<td>Good</td>
<td>Good</td>
<td>Good</td>
</tr>
<tr>
<td>Token Res</td>
<td>Good</td>
<td>Good</td>
<td>Good</td>
</tr>
</tbody>
</table>
Tokens are **Sender Constrained** instead of being **bearer**

<table>
<thead>
<tr>
<th>Security Levels</th>
<th>Token Types</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Sender Constrained Token</strong></td>
<td>Only the entity that was issued can used the token.</td>
</tr>
<tr>
<td></td>
<td><strong>Bearer Token</strong></td>
<td>Stolen tokens can also be used</td>
</tr>
</tbody>
</table>
And now working on the decoupled approach ...

- CIBA (client initiated backchannel authentication) profile = 1st Implementer’s Draft
  
  [Link](https://bitbucket.org/openid/fapi/src/master/Financial_API_WD_CIBA.md)

Redirect Approach

Decoupled Approach

Embedded Approach
O2O: Online Authentication for Offline Transaction
-- CIBA: Client Initiated Backchannel Authentication

- Use-case 1: Customer authentication @ Call centers
O2O: Online Authentication for Offline Transaction
-- CIBA: Client Initiated Backchannel Authentication

- Use-case 2: IoT including Smart Speakers
O2O: Online Authentication for Offline Transaction

-- CIBA: Client Initiated Backchannel Authentication

- Use-case 3: Payment at PoS
The system is asking for the permission. Approve?

1. Purchase ABC.
2. smart speaker
3. backend system
4. consumption device
5. asks for the permission
6. grants the permission
7. calls APIs
8. resource server that provides APIs

CIBA
These are in the form of check lists.

5.2 Read and Write API Security Provisions

5.2.1 Introduction
Read and Write access carries higher financial risk; therefore the protection level required is higher than Read-Only access.

As a profile of The OAuth 2.0 Authorization Framework, this document mandates the following for the Read and Write API of the FAPI.

5.2.2 Authorization Server
The authorization server shall support the provisions specified in clause 5.2.2 of Financial API - Part 1: Read-Only API Security Profile.

In addition, the authorization server, for the Write operation,

1. shall require the request or request_uri parameter to be passed as a JWS signed JWT as in clause 6 of OIDC;
2. shall require the response_type values code id_token or code id_token token;
3. shall return ID Token as a detached signature to the authorization response;
4. shall include state hash, s_hash, in the ID Token to protect the state value;
5. shall only issue holder of key authorization code, access token, and refresh token for write operations;
6. shall support OAuth or MTLS as a holder of key mechanism;
7. shall support user authentication at LoA 3 or greater as defined in X.1254;
8. shall support signed ID Tokens; and
9. should support signed and encrypted ID Token.

(source) https://bitbucket.org/openid/fapi/src/master/Financial_API_WD_002.md
Crypto Requirements are tightened for interoperability and security

8.5 TLS Considerations

As confidential information is being exchanged, all interactions shall be encrypted with TLS (HTTPS).

Section 7.1 of Financial API - Part 1: Read Only API Security Profile shall apply, with the following additional requirements:

1. Only the following 4 cipher suites shall be permitted:
   - `TLS_DHE_RSA_WITH_AES_128_GCM_SHA256`
   - `TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256`
   - `TLS_DHE_RSA_WITH_AES_256_GCM_SHA384`
   - `TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384`

8.6 JWS Algorithm Considerations

JWS signatures shall use the `PS256` or `ES256` algorithms for signing.

(source) https://bitbucket.org/openid/fapi/src/master/Financial_API_WD_002.md
They are now 2nd Implementer’s Draft Heading towards the final

- ToDo
  - Wait till OAuth JAR becomes RFC and refer it.
  - Tighten the integration with JARM
  - Split out the Clause 7 to make it an independent drafts
    - draft-lodderstedt-oauth-par-00
    - draft-lodderstedt-oauth-rar-00
  - Perhaps split out Token Binding - not enough implementations
  - ... and Change the names of Part 1 & 2!
... because there are Read-only cases that deal with very sensitive data, and RW cases where write operation is not-so-high risk.

- Part 1: Read Only Security Profile
  - High Security Profile ?? ??

- Part 2: Read and Write Security Profile
  - Substantial/Medium Security Profile ?? ??

(We are now in the process of explicitly writing the security properties in each drafts.)
Other Potential Works

- Client Registration
  - E.g., UK Open Banking, Australian CDS

- Data Schema
  - E.g., FDX, UK Open Banking, etc.

- Bringing RW Profile to ISO/IEC JTC 1
  - Rather than TC68 as now it is not only “Financial” but “general” in nature.
Oh, by the way
How can we tell that the implementation conforms to the specification?
OpenID Foundation offers FAPI Part 2 and CIBA test suite.

- For more information, see:
- https://openid.net/certification/
- https://openid.net/certification/fapi_op_testing/
Once it passes the test, the implementer can self-certify and publish.

- That gets the implementers under the premise of the article 5 of the FTC Act.
- The log will be openly available so others can also find out false claims.

See [http://openid.net/certification/](http://openid.net/certification/) for details
FAPI Certification

- Each Test Numbers are linked to the clause number of the FAPI Specification
- Semi-Automated Testing

### Certified Financial-grade API (FAPI) OpenID Providers

These deployments have been granted certifications for these Financial-grade API (FAPI) conformance profiles:

<table>
<thead>
<tr>
<th>Organization</th>
<th>Implementation</th>
<th>FAPI R/W OP w/ MTLS</th>
<th>FAPI R/W OP w/ Private Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authlete</td>
<td>Authlete 2.1</td>
<td>1-Apr-2019 [view]</td>
<td>1-Apr-2019 [view]</td>
</tr>
<tr>
<td>Cater Allen</td>
<td>CA Open Banking v1.3.0</td>
<td>4-Sep-2019 [view]</td>
<td></td>
</tr>
<tr>
<td>Coutts &amp; company</td>
<td>F23</td>
<td>27-Sep-2019 [view]</td>
<td></td>
</tr>
<tr>
<td>Curity</td>
<td>Curity Identity Server 4.3.0</td>
<td>20-Sep-2019 [view]</td>
<td>20-Sep-2019 [view]</td>
</tr>
<tr>
<td>Filip Skokan</td>
<td>node oido-provider *6.5.0</td>
<td>20-Aug-2019 [view]</td>
<td>20-Aug-2019 [view]</td>
</tr>
<tr>
<td>ForgeRock</td>
<td>ForgeRock Financial 3.1.0-credence</td>
<td>1-Apr-2019 [view]</td>
<td></td>
</tr>
<tr>
<td>Ping Identity</td>
<td>PingFederate 9.2.3</td>
<td>29-May-2019 [view]</td>
<td></td>
</tr>
</tbody>
</table>

### Certified Financial-grade API Client Initiated Backchannel Authentication Profile (FAPI-CIBA) OpenID Providers

These deployments have been granted certifications for these Financial-grade API Client Initiated Backchannel Authentication Profile (FAPI-CIBA) conformance profiles:

<table>
<thead>
<tr>
<th>Organization</th>
<th>Implementation</th>
<th>FAPI-CIBA OP poll w/ MTLS</th>
<th>FAPI-CIBA OP poll w/ Private Key</th>
<th>FAPI-CIBA OP Ping w/ MTLS</th>
<th>FAPI-CIBA OP Ping w/ Private Key</th>
</tr>
</thead>
</table>
External Liaisons
Liaison relationships are integral to FAPI WG’s activity.

- Open Banking Implementation Entity (Ralph Bragg)
- ISO/TC68 (Dave Tonge)
- ISO/IEC JTC 1/SC 27/WG 5 (Anthony Nadalin)
- Financial Data Exchange (FDX) (Anoop Saxana)
- FDATA (Dave Tonge)
- … and potentially
  - W3C Web Payment WG
  - Australian CDS
  - Japanese Fintech Association etc.
Come join us.

https://openid.net/wg/fapi/