



# OpenID Foundation Certification Program

**Joseph Heenan**  
**Certification Technical Lead**  
**OpenID Foundation**

# OpenID Certification Program Overview



- A light-weight, low-cost, certification program to serve members, drive adoption and promote high-quality implementations
  - Identity Providers launched in early 2015
  - Relying Parties launched in late 2016
  - Financial-grade profiles launched in 2019
- Each certification makes it easier for those that follow and helps make subsequent deployments more trustworthy, interoperable and secure
- All certified implementations are freely available at <https://openid.net/developers/certified/>
- OIDF certification pricing has been widely accepted

# Certification Program Success



616 certifications of 200 deployments

Total OP Certifications	436	Total RP Certifications	94
Total OP Deployments	125	Total RP Deployments	34
<hr/>			
Total FAPI Certifications	70	Total FAPI-CIBA Certifications	12
Total FAPI Deployments	36	Total FAPI-CIBA Deployments	3
Total FAPI RP Certifications	4		
Total FAPI Deployments	2		

# Open Banking Adoption of FAPI & FAPI Certification

- UK led the way with FAPI adoption and FAPI certification under the direction of the Open Banking Implementation Entity
  - Currently 15 UK banks have 31 FAPI certifications of 16 deployments
  - Most of the CMA9 have certified
  - OBIE require the largest 9 banks to recertify annually
- Additional jurisdictions adopting FAPI and FAPI certification
  - US – ODF anticipates the Financial Data Exchange formally adopting FAPI and requiring FAPI certification
  - AU – ODF coordinating with AU DSB team who has adopted FAPI as a normative standard and will be encouraging AU banks to FAPI certify
  - Brazil – Security Work Group in Brazil has adopted FAPI as part of Brazil’s open banking stack and will require banks to be FAPI certified. ODF collaborating with Security WG on Brazil-specific conformance tests
  - Other jurisdictions – ODF working with regulators and coordinators in Europe, Bahrain and other locals to encourage and support the adoption of FAPI and FAPI conformance

# Why use the OIDF's conformance program?

- OIDF tests are developed with close support of relevant working group
  - Tests are updated based on requests from working group
- Testers get direct support from the OIDF certification team
  - Domain experts familiar with all the specs
  - Team have access to OIDF/OAuth2 spec authors when necessary
- Internationally recognized, award winning
- Tests are maintained and updated by OIDF when:
  - new versions of underlying specs published
  - new potential security vulnerabilities are found
  - new interoperability problems are found
  - testers find failures difficult to interpret
- Issues found by testers are raised back to the relevant OIDF working groups
  - Specs can be improved / clarified / disambiguated as necessary

# OIDF FAPI Certification Program

- FAPI-RW ID1 OP testing (OBUK specific) started December 2017
- FAPI-RW ID2 OP testing launched April 2019
- FAPI-RW ID2 RP testing launched in June 2019
- FAPI-CIBA ID1 OP testing launched September 2019
- Optionally supports:
  - OpenBanking UK intent lodging
  - Australian Consumer Data Rights for OPs – launched January 2021
  - FAPI-RW ID2 OP using PAR (Pushed Authentication Requests – launched January 2021
  - App2app authentication/authorization
- Visit <https://openid.net/certification/instructions/> for details

# FAPI1 - Advanced Final

- Final FAPI 1.0 parts 1 and 2 published March 12, 2021
- Relatively few normative changes
- New names
  - FAPI-R -> FAPI Baseline
  - FAPI-RW -> FAPI Advanced
- Launch of tests for the new spec planned for May 2021
  - OP & RP
  - Implementers Draft 2 versions of the tests will be retained

# PAR (Pushed Authentication Requests)

- IETF Standard from OAuth2 Working Group
- Draft Status : <https://tools.ietf.org/html/draft-ietf-oauth-par>
- An evolution of FAPI-RW's request object endpoint
- Avoids passing authorization request details via the front channel
  - Better for privacy
  - Better for security (client authenticates before authentication begins)
  - Avoids any size limits on URLs
- Working Group Last Call was August 2020
- Australian CDR planning to go live with PAR from July 2021, wide vendor support
- Certification program for FAPI-RW with PAR launched January 2021



# Australian CDR

- Based on FAPI-RW
- 4 or 5 banks(OPs) live, 3 RPs live
  - Many of banks are now going through FAPI conformance testing
- Some extra restrictions compared to base FAPI-RW spec
  - `private_key_jwt` must be used
  - `x-v` header must be sent to resource server endpoint
  - Refresh tokens must be supported
  - Returned `id_tokens` must be encrypted
  - For ACR claims, a CDR specific value is used, “`urn:cds.au:cdr:2`”
- Development of CDR version of FAPI RP tests under discussion

# Brazil OpenBanking

- Based on FAPI1-Advanced
- 40 banks due to certify by July, more in September
- Some extra restrictions compared to FAPI Advanced spec
  - Encrypted request objects required
  - PS256 for signing
  - Intent pre-lodging (similar to UK OpenBanking)
  - Intent id passed in a structured scope
  - Brazil specific ACR claim values
- Development of Brazil version of FAPI OP/RP tests under discussion

# FAPI-RW Certification: Core goals

- Interoperability
- Security
- Correct deployment of certified software

However:

- FAPI tests do not test all of OpenID Connect Core or OAuth
  - 'Pretty good' coverage of relevant parts though
  - Vendors should run OpenID Connect Core tests as well (if they support non-FAPI)

# FAPI-RW Certification: Reasons to Test

- Reduced support costs
  - If your implementation is interoperable it will “just work” for third parties
- Evidence of compliance to show government regulators
- Evidence of compliance may reduce insurance costs, chances of security breach, etc.
- It can be embarrassing if other people test your server & you fail
  - Anyone can test a server

# FAPI Certification: First, FAPI compliance

- First, become FAPI compliant
- Ideally upgrade to a FAPI certified version of your vendor's product
- Software that is not FAPI certified is likely to be missing:
  - Important configuration controls
  - “Recent” required standards like MTLS sender constrained access tokens
  - Well established but higher security OAuth2 options  
e.g. client authentication using replay-proof asymmetric cryptography
  - Tamper proof (JWT Secured) OAuth2 authorization requests
- Check any HSMs (Hardware Security Modules) in use
  - Older ones may only support RSASSA-PKCS1-v1\_5, which has known weaknesses

# FAPI Certification: Pre-testing steps

- Two registered OAuth2 clients are required
- Tester needs to be able to create & register client credentials
  - Or be provided with them in the correct format
- Recommended that tester has existing domain knowledge
  - TLS certificates, JWKS manipulation, OAuth2, FAPI
  - For first run, a developer or highly-technical tester is desirable

# Wrap up

- Conformance Suite source code etc publicly available on gitlab:  
<https://gitlab.com/openid/conformance-suite>
- Instructions for testing/certifying:  
<https://openid.net/certification/instructions/>
- Production deployment:  
<https://www.certification.openid.net/>  
(Login with any google/gitlab/openid account)
- Contact me if you'd like some help:
  - [joseph.heenan@oidf.org](mailto:joseph.heenan@oidf.org) or [certification@oidf.org](mailto:certification@oidf.org)
  - <https://twitter.com/josephheenan>
  - <https://www.linkedin.com/in/josephheenan>

Additional slides



# Who Am I?

- OpenID Certification Team lead developer
- Software engineer & architect with over 25 years' experience
- Active contributor to the OpenID Connect FAPI/CIBA/FAPI-CIBA/eKYC specifications
- 20+ years of mobile app experience
- Assisted 30+ UK banks with achieving compliance to the OpenID/FAPI specifications

<https://www.linkedin.com/in/josephheenan/>

# Conformance Suite Architecture

- Multi-party protocol testing
- Structured configuration
- Structured logging and results
- Deterministic, modular execution units
  - Small pieces of java code
  - Easily unit testable
- Protect sensitive configuration and results data
- Transparent process
- Usable as part of CI

# Architecture - continued

- Loosely bound backend, frontend and test modules
  - Clear interfaces
  - Heavy use of JSON
- Consistent logging of all inputs and outputs
- Easily extensible to new protocols
  - E.g. CIBA added without requiring any changes to frontend/backend
- Does not use existing OAuth2/OpenID Connect libraries
  - Easier to introduce negative tests
  - Easier to show the user exactly what happened and why

# Security Checks - Issuer

[RFC 8414](#)

OAuth 2.0 Authorization Server Metadata

June 2018

## 3.3. Authorization Server Metadata Validation

The "issuer" value returned MUST be identical to the authorization server's issuer identifier value into which the well-known URI string was inserted to create the URL used to retrieve the metadata. If these values are not identical, the data contained in the response MUST NOT be used.

---

15:05:56	<b>SUCCESS</b>	CheckDiscEndpointDiscoveryUrl
<b>1</b> More ^		discoveryUrl
	<b>actual</b>	<a href="https://fapidev-as.authlete.net/.well-known/openid-configuration">https://fapidev-as.authlete.net/.well-known/openid-configuration</a>
15:05:56	<b>SUCCESS</b>	CheckDiscEndpointIssuer
	<b>OIDCD-4.3</b> ↗	issuer is consistent with the discovery endpoint
	<b>OIDCD-7.2</b> ↗	

---

# Security Checks - Keys

12:34:03

**FAILURE**

EnsureServerJwksDoesNotContainPrivateOrSymmetricKeys

2 More ^

Jwks contains private and/or symmetric keys

private\_keys

```
[
  {
    "p": "uKADG9h1fv0aWcdBArKbIuMwlsWta_3vWMGymWaA0McIFrmoYi0_MNQAqos3hKEu1TltpzBWXBooDjz2oqptD464SGonWDK3oDawcSyH1T0mTgePlffVfn7u8",
    "kty": "RSA",
    "q": "uFhhMgTXP9u_Upv6i1C7T-YHk_jJ2e3P09RxF74gfkPoP35N6K0RVELZgaAC0q3xr6TikTYyRL_B3PYH4KWxiW9uErV3yNGDFGxp0mhxNR6zTPxGec1gUk2mU",
    "d": "FSd7Am9oKHWmabvsV0r_aAXH0Rr22AQwJgFR0gAbAiTYC8bJSDXK1CjzHzzQB5-U5hsLTDNtvEpZy_LFnPEsxn0qLE8BLWFQcaFUczA8AKPIS5NHZ_rywXixwa5y1KeIWXr_dyMGeiNtP6_mABXTWFagvgVwwSMT8Ufd-Evw8PKb46yR0cIub-1F9h0Ainqqaq7FovHIQDa5MuKWB",
    "e": "AQAB",
    "use": "sig",
    "kid": "sig-2020-07-21T11:27:04Z",
    "qi": "jkzvNCY02KW9Bky833DCNJApkXjc4PHd5J98bAqZzLP3o3smbLWqvdl92acP0a-PxSuRkt6MUFitlCpgeN1n69L6326kkMfM_aT00rhMM0gZembd4rJKgI6k",
    "dp": "lvJMWGHbfp3VA34DSv9YE2gIe9zW8ypEnB6RtRW3T_rKRDo6zzoLJhLPEKC0HazzwQ2iWnFDK6rZ_9AAJLemFDWk0hhA0Zsngk97i10T_MXLvD3DjFkvwg2GoU",
    "alg": "PS256",
    "dq": "Dm99TPlsEagXl1R3jilIQb11onS8-b_RlpHQ0Ve-G6UdrrspRqpoWvzRI4FwNyEwSdzTkSN5VEDf4XmyrDjNakG7k0N8-dd0Pu8uXlCHb012hPTMYAVhIZDLE",
    "n": "hPK_VckSwJtFaGRpbBlnjTyRsnpaN9m1CCZHVfSJi3IPh8cregl0HVsC2jFG6LgVzesHvTRi-dDRgtAFGwc_U_go2W_7MqH4zkHw_RIliGP814hIWmi-zrEH5-5Yrvo8H_f80hx2rWF89BknLeeDIPDaaXHzZY0khaP7cc03W7EzkUud9y64TEMxGY_AeMDCbDr-maycRHy54AgZk"
  }
]
```

symmetric\_keys

```
[]
```



# Security Checks – objects ‘signed’ with alg ‘none’

08:45:40

SUCCESS

SerializeRequestObjectWithNullAlgorithm

1 More ^

Serialized the request object

**request\_object**

eyJhbGciOiJub25lIn0.eyJhdWQiOiJodHRwczpcL1wvcmV2aWV3LWFzLmF1dGhsZXRLM5ldFwvliwic2NvcGUiOiJvcGVuaWQgcGF5bWVudHMiLCJjbGFpbXMiOnsiaWRfdG9rZW4iOnsImVzc2VudGlhbCI6dHJ1ZX19fSwiaXNzIjoNTI0ODA3NTQwNTMiLCJyZXNwb25zZV90eXBlljoiY29kZSBpZF90b2tlibiIsInJlZGlyZWN0X3VyaSI6Imh0dHBzOlwvXC93d3cuY2VydGZXdcL2NhbGxiYWNRliwic3RhdGUiOiI3R2J5VW4zTTNmliwiZXhwIjoxNTkzNzYyNjQwLCJub25jZSI6IjJjPV3VmUHpITmQiLCJjbGllbnRfaWQiOiI1MjQ4MDc1NDA1MyJ9.

# Further Security Checks – Request Object

- 'exp' already expired
- Incorrect 'aud'
- Correctly signed, but with non-permitted alg
- With a syntactically valid, but incorrect, signature
- Valid signature but from a different client
- With nonce only outside request object
- With non-registered redirect uri



# Security Checks – Token Endpoint

- Calling token endpoint
  - Without client authentication
  - With expired client authentication assertion
  - With client authentication assertion intended for different server ('aud')
  - Valid client authentication, but passing client\_id for target client
  - With already-used authorization code
  - With authorization code issued to another client
  - No MTLS client cert supplied for binding access token to

# Security Checks – continued

- JWKS
  - Keys too short
- Authorization code
  - Too short
  - Not enough entropy
- Calling resource server
  - With valid mtls client cert, but not the one bound to access token
- TLS 1.0/1.1 not allowed
- Insecure ciphers not allowed
- And many more...

# Interoperability Checks – Time Stamps

“Seconds since 1<sup>st</sup> Jan 1970” has been a well-known standard for years... but:

09:37:08	<b>FAILURE</b>	EnsureUserInfoUpdatedAtValid
<a href="#">2 More ^</a>	<a href="#">OIDCC-5.1</a>	updated_at appears to be in the future
<b>updated_at</b>	May 31, 52521, 1:30:00 AM	
<b>now</b>	Jul 21, 2020, 8:37:08 AM	

# Interoperability checks - continued

- The standard 'happy' flow
- Variants on Accept: headers
  - With/without charset
  - With q parameters
  - With multiple options
- With optional fields
  - All present
  - All missing
- Where case insensitive, testing both cases
- With allowed variants
  - 'aud' is an array
- Discovery document
  - Reflects what's supported
  - Syntactically valid