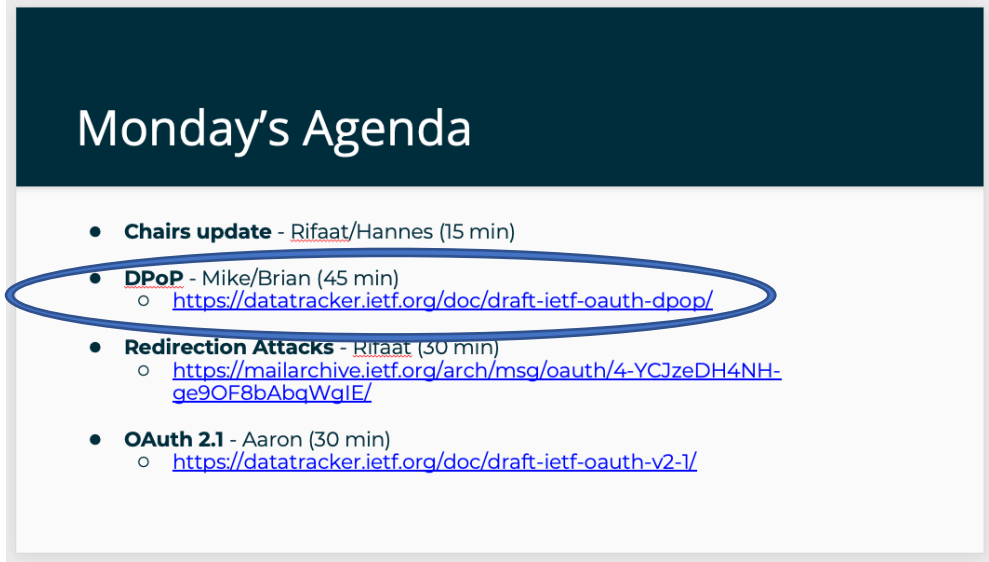

# OAuth 2.0 Demonstrating Proof-of-Possession at the Application Layer

Brian Campbell, Dr. Daniel Fett, Michael B. Jones, Torsten Lodderstedt, David Waite, John Bradley

https://datatracker.ietf.org/doc/draft-ietf-oauth-dpop/

# Expectation Setting

- 45 minutes (hopefully less)
- DPoP protocol overview
  - noting recent changes
- Wrap up



# DPoP Objectives

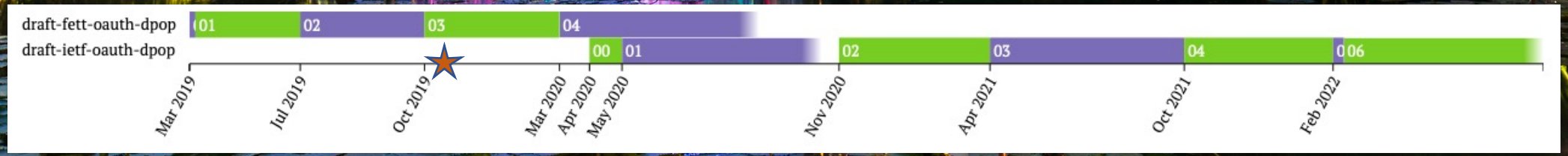

Pragmatic* application-level proof-of-possession mechanism for OAuth 2.0 access tokens and refresh tokens issued to public clients

* if not as simple as it once was or terribly efficient

# It’s been a while…

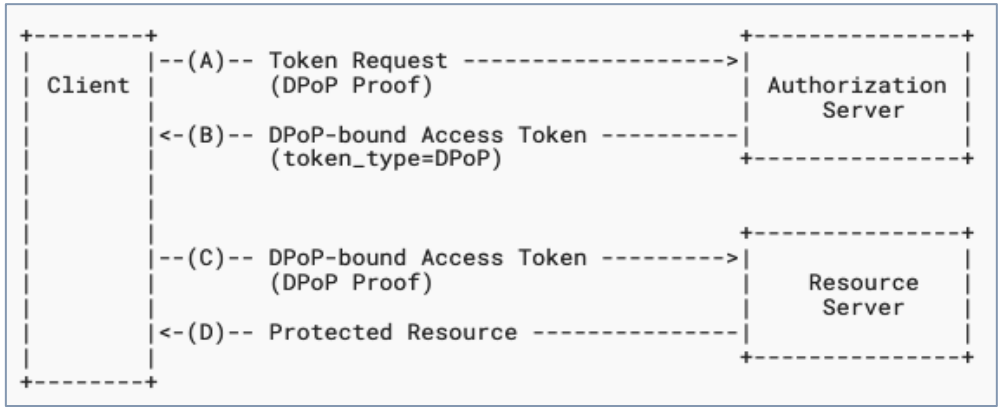
## Happenings since the last in-person meeting

IETF 106 Singapore, Nov 2019



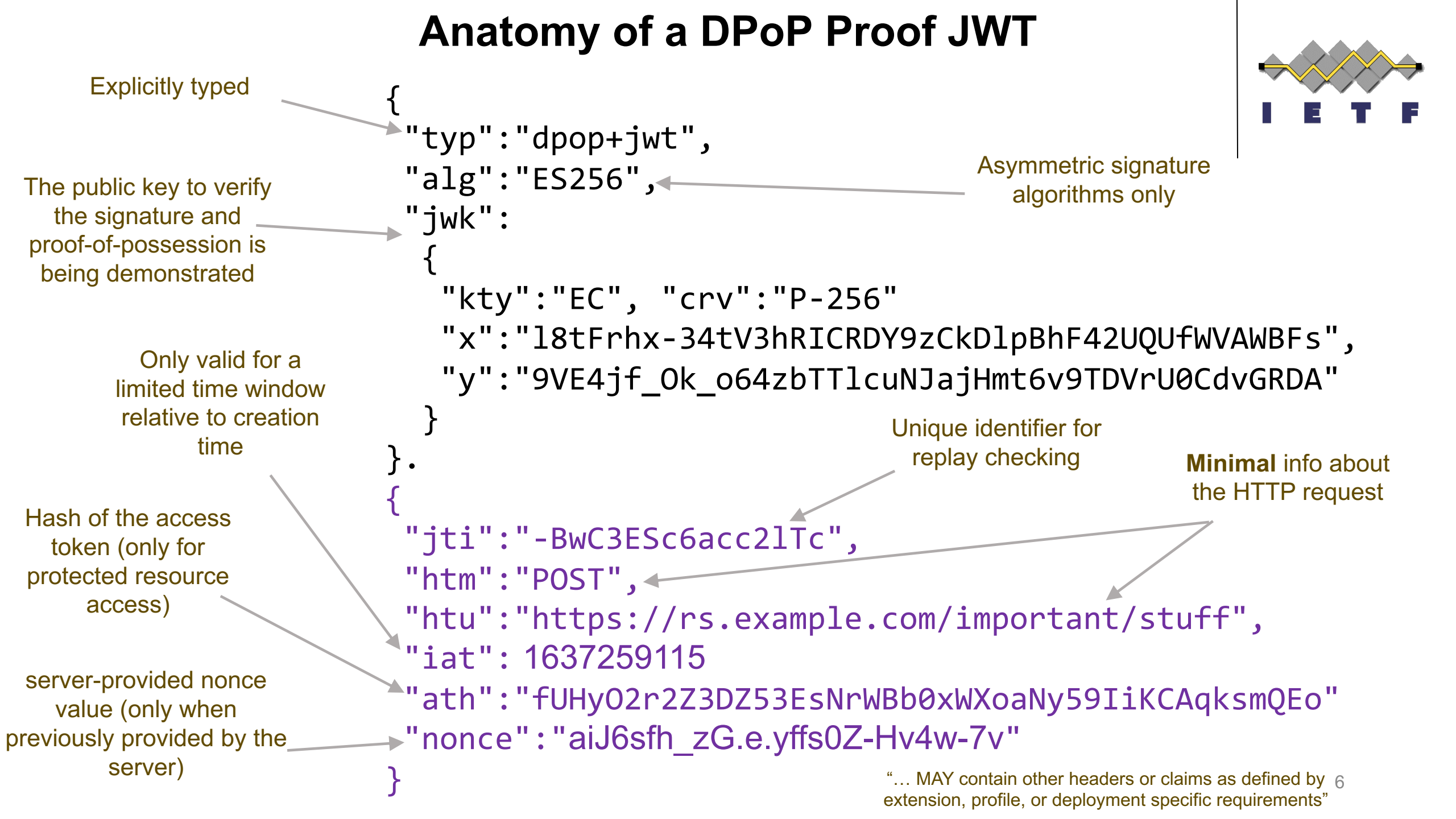
# DPoP Overview

- DPoP Proof JWT sent as an HTTP header
  - Demonstrates a reasonable level of proof-of-possession in the context of the request
  - Sent the same way with (mostly) the same syntax and semantics for both:
    - token requests to the AS's token endpoint
    - protected resource requests
  - AS uses the proof to bind tokens
  - RS uses the proof to verify bound tokens




```
DPoP: eyJ0eXAiOiJkcG9wK2p3dCIsImFsZyI6IkVTMjU2IiwiandrIjp7Imt0eSI6Ik
 VDIiwieCI6Imw4dEZyaHgtMzR0VjNoUklDUkRZOXpDa0RscEJoRjQyVVFVZldWQVdCR
 nMiLCJ5IjoiOVZFNGpmX09rX282NHpiVFRsY3VOSmFqSG10NnY5VERWclUwQ2R2R1JE
 QSIsImNydiI6IlAtMjU2In19.eyJqdGkiOiItQndDM0VTYzZhY2MybFRjIiwiaHRtIj
 oiUE9TVCIsImh0dSI6Imh0dHBzOi8vc2VydmVyLmV4YW1wbGUuY29tL3Rva2VuIiwia
 WF0IjoxNTYyMjYyNjE2fQ.2-GxA6T8lP4vfrg8v-FdWP0A0zdrj8igiMLvqRMUvwnQg
 4PtFLbdLXiOSsX0x7NVY-FNyJK70nfbV37xRZT3Lg
```

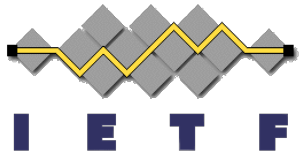
# Anatomy of a DPoP Proof JWT




```
{
 "typ":"dpop+jwt",
 "alg":"ES256",
 "jwk":
  {
   "kty":"EC", "crv":"P-256"
   "x":"l8tFrhx-34tV3hRICRDY9zCkDlpBhF42UQUfWVAWBFs",
   "y":"9VE4jf_Ok_o64zbTTlcuNJajHmt6v9TDVrU0CdvGRDA"
  }
}.
{
 "jti":"-BwC3ESc6acc2lTc",
 "htm":"POST",
 "htu":"https://rs.example.com/important/stuff",
 "iat": 1637259115
 "ath":"fUHyO2r2Z3DZ53EsNrWBb0xWXoaNy59IiKCAqksmQEo"
 "nonce":"aiJ6sfh_zG.e.yffs0Z-Hv4w-7v"
}
```

"… MAY contain other headers or claims as defined by extension, profile, or deployment specific requirements"

# (code) Access Token Request




```
POST /token HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded;charset=UTF-8
DPoP: eyJ0eXAiOiJkcG9wK2p3dCIsImFsZyI6IkVTMjU2IiwiandrIjp7Imt0eSI6Ik
 VDIiwieCI6Imw4dEZyaHgtMzR0VjNoUklDUkRZOXpDa0RscEJoRjQyVVFVZldWQVdCR
 nMiLCJ5IjoiOVZFNGpmX09rX282NHpiVFRsY3VOSmFqSG10NnY5VERWclUwQ2R2R1JE
 QSIsImNydiI6IlAtMjU2In19.eyJqdGkiOiItQndDM0VTYzZhY2MybFRjIiwiaHRtIj
 oiUE9TVCIsImh0dSI6Imh0dHBzOi8vc2VydmVyLmV4YW1wbGUuY29tL3Rva2VuIiwia
 WF0IjoxNTYyMjYyNjE2fQ.2-GxA6T8lP4vfrg8v-FdWP0A0zdrj8igiMLvqRMUvwnQg
 4PtFLbdLXiOSsX0x7NVY-FNyJK70nfbV37xRZT3Lg

grant_type=authorization_code
&code=SplxlOBeZQQYbYS6WxSbIA
&redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
&code_verifier=bEaL42izcC-o-xBk0K2vuJ6U-y1p9r_wW2dFWIWgjz-
```

DPoP proof JWT in HTTP header

# Access Token Response

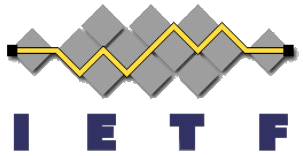
```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-cache, no-store

{
  "access_token":" Kz~8mXK1EalYznwH-LC-1fBAo.4Ljp~zsPE_NeO.gxU",
  "token_type":"DPoP",
  "expires_in":3626,
  "refresh_token":"Q..Zkm29lexi8VnWg2zPW1x-tgGad0Ibc3s3EwM_Ni4-g"
}
```

Token type indicates that the **access token** is bound to the DPoP public key

**refresh token** is bound to the DPoP public key for a public client

# (refresh) Access Token Request



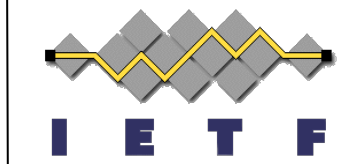

```
POST /token HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded;charset=UTF-8
DPoP: eyJ0eXAiOiJkcG9wK2p3dCIsImFsZyI6IkVTMjU2IiwiandrIjp7Imt0eSI6Ik
 VDIiwieCI6Imw4dEZyaHgtMzR0VjNoUklDUkRZOXpDa0RscEJoRjQyVVFVZldWQVdCR
 nMiLCJ5IjoiOVZFNGpmX09rX282NHpiVFRsY3VOSmFqSG10NnY5VERWclUwQ2R2R1JE
 QSIsImNydiI6IlAtMjU2In19.eyJqdGkiOiItQndDM0VTYzZhY2MybFRjIiwiaHRtIj
 oiUE9TVCIsImh0dSI6Imh0dHBzOi8vc2VydmVyLmV4YW1wbGUuY29tL3Rva2VuIiwia
 WF0IjoxNTYyMjY1Mjk2fQ.pAqut2IRDm_De6PR93SYmGBPXpwrAk90e8cP2hjiaG5Qs
 GSuKDYW7_X620BxqhvYC8ynrrvZLTk41mSRroapUA

grant_type=refresh_token
&refresh_token=Q..Zkm29lexi8VnWg2zPW1x-tgGad0Ibc3s3EwM_Ni4-g
```

DPoP proof JWT
in HTTP header

# DPoP Bound Access Token

## JWT & Introspection Response

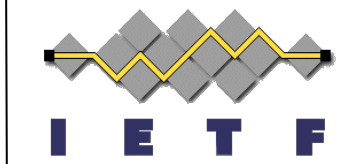



```
{
   ... other claims / members ...

   "cnf":
   {
     "jkt":"0ZcOCORZNYy-DWpqq30jZyJGHTN0d2HglBV3uiguA4I"
   }
 }
```

Confirmation claim carries the SHA-256 JWK Thumbprint of the DPoP public key to which the access token is bound

# Protected Resource Request



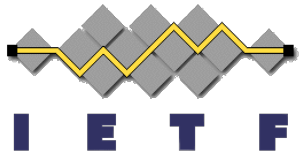

```
GET /protectedresource HTTP/1.1
Host: resource.example.org
Authorization: DPoP Kz~8mXK1EalYznwH-LC-1fBAo.4Ljp~zsPE_NeO.gxU
DPoP: eyJ0eXAiOiJkcG9wK2p3dCIsImFsZyI6IkVTMjU2IiwiandrIjp7Imt0eSI6Ik
 VDIiwieCI6Imw4dEZyaHgtMzR0VjNoUklDUkRZOXpDa0RscEJoRjQyVVFVZldWQVdCR
 nMiLCJ5IjoiOVZFNGpmX09rX282NHpiVFRsY3VOSmFqSG10NnY5VERWclUwQ2R2R1JE
 QSIsImNydiI6IlAtMjU2In19.eyJqdGkiOiJlMWozVl9iS2ljOC1MQUVCIiwiaHRtIj
 oiR0VUIiwiaHR1IjoiaHR0cHM6Ly9yZXNvdXJjZS5leGFtcGxlLm9yZy9wcm90ZWN0Z
 WRyZXNvdXJjZSIsImlhdCI6MTU2MjI2MjYxOH0.lNhmpAX1WwmpBvwhok4E74kWCiGB
 NdavjLAeevGy32H3dbF0Jbri69Nm2ukkwb-uyUI4AUg1JSskfWIyo4UCbQ
```

DPoP-bound (reference style) access token

Token is bound to the key in the proof

DPoP proof

# 401 with WWW-Authenticate Challenge




Example Response to a Protected Resource Request Without a Token

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: DPoP algs="ES256 PS256”
```

Example Response to a Protected Resource Request With an Invalid Token

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: error="invalid_token",
   error_description="Invalid DPoP key binding", algs="ES256”
```

# So you think you can Nonce?

- -04 added the option for a server-provided nonce in the DPoP proof while -05/-06 refined and clarified
  - "… DPoP-Nonce HTTP header in the response supplying a nonce value to be used when sending the subsequent request."

Authorization Server nonce challenge

```
HTTP/1.1 400 Bad Request
DPoP-Nonce: 7c1P2Sczb-3yBH0-Z.Gv4w4xY

{
  "error": "use_dpop_nonce"
  "error_description": "AS needs nonce in DPoP proof"
}
```

Protected Resource nonce challenge

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: DPoP error="use_dpop_nonce",
   error_description="plz use nonce in next DPoP proof"
DPoP-Nonce: ab3d6eqG96ye0i0999Z_HX42t77x
```

Next nonce provided with successful response (no challenge)

```
HTTP/1.1 200 OK
DPoP-Nonce: I1m3n0peJ7S45zGPeYJbYuciQmJxj26
```

# Metadata

- Authorization Server Metadata
  - dpop_signing_alg_values_supported: signal support for DPoP with the JWS alg values the AS supports for DPoP proofs

- -05/-06 added Client Registration Metadata
  - dpop_bound_access_tokens: indicate that the client always uses DPoP when requesting tokens from the authorization server

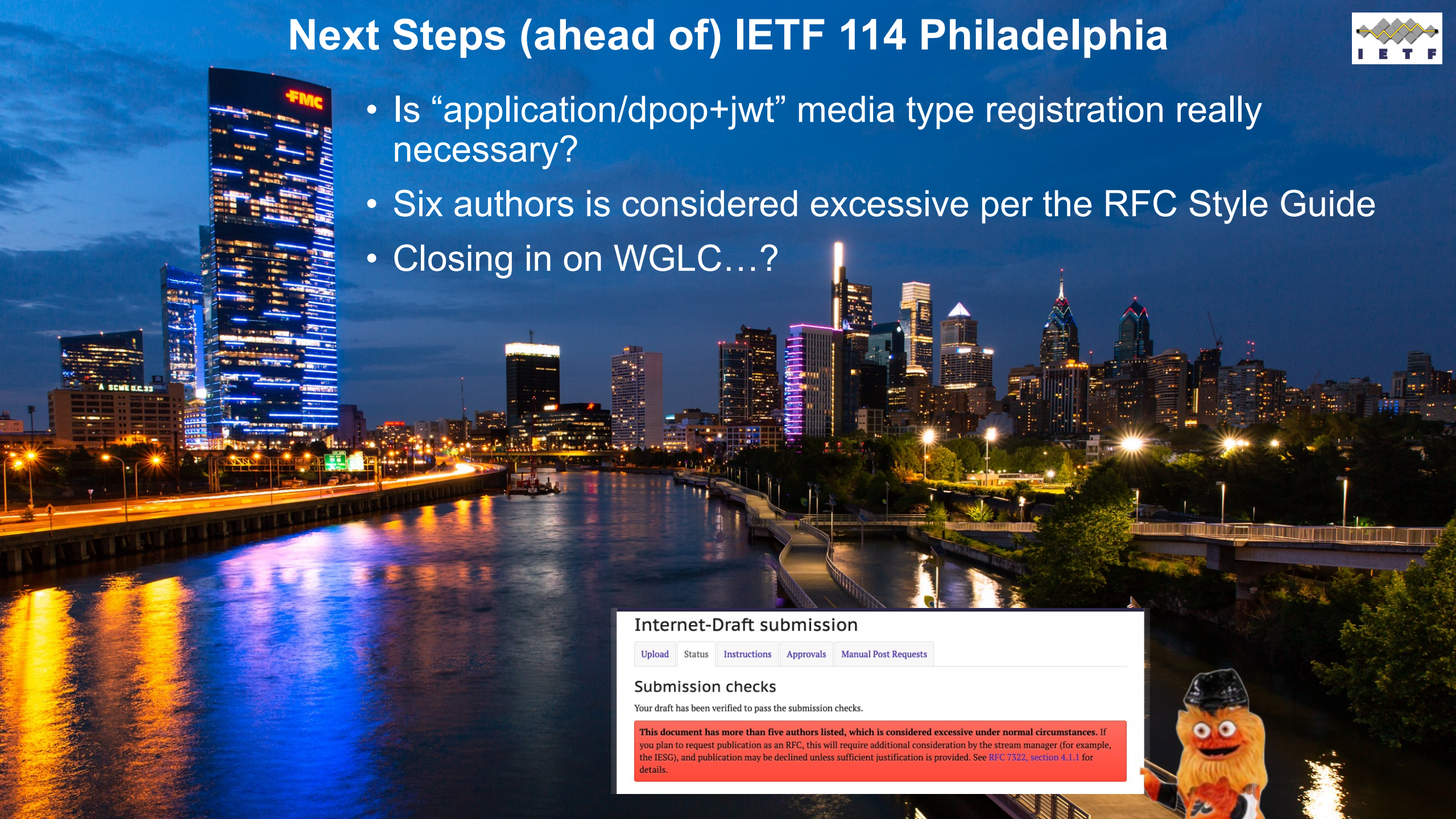
# Binding an Authorization Code to a DPoP Key

- -05 added optional dpop_jkt authorization request parameter
  - SHA-256 JWK Thumbprint of the proof-of-possession public key
  - AS binds the issued authorization code to the thumbprint
  - AS checks the binding against the DPoP proof on code redemption
  - Enables end-to-end binding of the whole authorization flow
  - Can be used in conjunction with PKCE as-is
  - PAR can bind from the DPoP proof or dpop_jkt (need to be the same)

```
GET /authorize?response_type=code&client_id=s6BhdRkqt3&state=xyz
  &redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
  &code_challenge=E9Melhoa2OwvFrEMTJguCHaoeK1t8URWbuGJSstw-cM
  &code_challenge_method=S256
  &dpop_jkt=NzbLsXh8uDCcd-6MNwXF4W_7noWXFZAfHkxZsRGC9Xs HTTP/1.1
Host: server.example.com
```

# Next Steps (ahead of) IETF 114 Philadelphia

- Is “application/dpop+jwt” media type registration really necessary?
- Six authors is considered excessive per the RFC Style Guide
- Closing in on WGLC…?




Internet-Draft submission

Upload | Status | Instructions | Approvals | Manual Post Requests

Submission checks

Your draft has been verified to pass the submission checks.

**This document has more than five authors listed, which is considered excessive under normal circumstances.** If you plan to request publication as an RFC, this will require additional consideration by the stream manager (for example, the IESG), and publication may be declined unless sufficient justification is provided. See RFC 7322, section 4.1.1 for details.