IETF 107
~~Vancouver~~
Virtual Interim
May 2020

OAUTH WG

DPoP

OAuth 2.0 Demonstration of Proof-of-Possession at the Application Layer

Daniel Fett
Brian Campbell
John Bradley
Torsten Lodderstedt
Michael Jones
David Waite

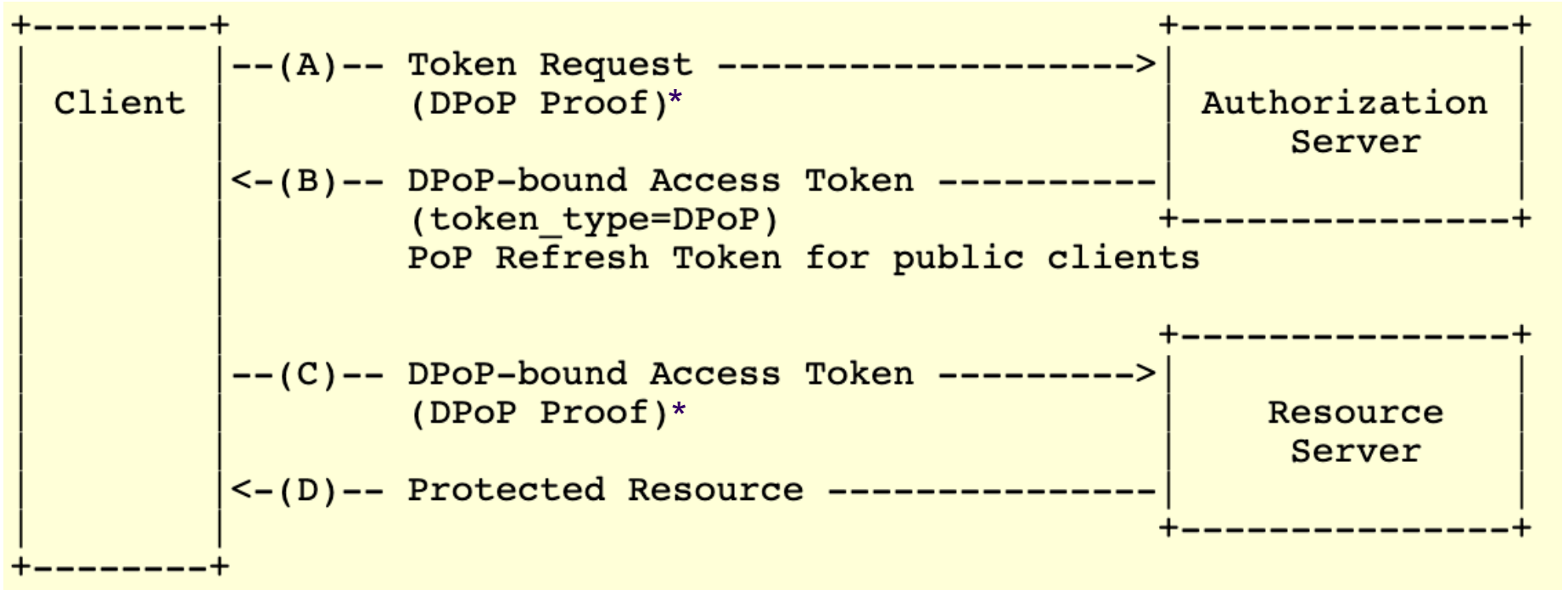
draft-ietf-oauth-dpop

DPoP Overview / Refresher

A new[ish] simple and concise approach to proof-of-possession for OAuth access and refresh tokens using application-level constructs and leveraging existing JWT library support

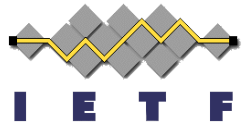
draft-fett-oauth-dpop-00 was published during IETF 105 in Prague thereby justifying the use of this photo

Basic DPoP flow (in ASCII)



* DPoP Proof syntax and semantics don't change

Anatomy of a DPoP Proof JWT



Explicitly typed

Asymmetric signature algorithms only

The public key for which proof-of-possession is being demonstrated

Minimal info about the HTTP request

Unique identifier for replay checking

Only valid for a limited time window relative to creation time

Other stuff *could* go here

```
{
 "typ":"dpop+jwt",
 "alg":"ES256",
 "jwk":
  {
   "kty":"EC", "crv":"P-256"
    "x":"l8tFrhx-34tV3hRICRDY9zCkDlpBhF42UQUfWVAWBFs",
    "y":"9VE4jf_Ok_o64zbTTlcuNJajHmt6v9TDVrU0CdvGRDA"
  }
}
{
 "jti":"-BwC3ESc6acc2lTc",
 "htm":"POST",
 "htu":"https://server.example.com/token",
 "iat":1562262616
}
```

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

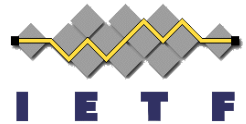

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-cache, no-store

{
   "access_token":"eyJhbGciOiJFUzI1NiIsImtpZCI6IkJlQUxrYiJ9.eyJzdWIiOi
     Jzb21lb25lQGV4YW1wbGUuY29tIiwiaXNzIjoiaHR0cHM6Ly9zZXJ2ZXIuZXhhbXB
     sZS5jb20iLCJhdWQiOiJodHRwczovL3Jlc291cmNlLmV4YW1wbGUub3JnIiwibmJm
     IjoxNTYyMjYyNjExLCJleHAiOjE1NjIyNjYyMTYsImNuZiI6eyJqa3QiOiIwWmNPQ
     09SWk5ZeS1EV3BxcTMwalp5SkdIVE4wZDJIZ2xCVjN1aWd1QTRJIn19.vsFiVqHCy
     IkBYu50c69bmPJsj8qYlsXfuC6nZcLl8YYRNOhqMuRXu6oSZHe2dGZY0ODNaGg1cg
     -kVigzYhF1MQ",
   "token_type":"DPoP",
   "expires_in":3600,
   "refresh_token":"4LTC8lb0acc6Oy4esc1Nk9BWC0imAwH7kic16BDC2"
}
```

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

DPoP Bound Access Token

JWT & Introspection Response




```
{
   "sub":"someone@example.com",
   "iss":"https://server.example.com",
   "aud":"https://resource.example.org",
   "nbf":1562262611,
   "exp":1562266216,
   "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 eyJhbGciOiJFUzI1NiIsImtpZCI6IkJlQUxrYiJ9.eyJzdWI
 iOiJzb21lb25lQGV4YW1wbGUuY29tIiwiaXNzIjoiaHR0cHM6Ly9zZXJ2ZXIuZXhhbX
 BsZS5jb20iLCJhdWQiOiJodHRwczovL3Jlc291cmNlLmV4YW1wbGUub3JnIiwibmJmI
 joxNTYyMjYyNjExLCJleHAiOjE1NjIyNjYyMTYsImNuZiI6eyJqa3QiOiIwWmNPQ09S
 Wk5ZeS1EV3BxcTMwalp5SkdIVE4wZDJIZ2xCVjN1aWd1QTRJIn19.vsFiVqHCyIkBYu
 50c69bmPJsj8qYlsXfuC6nZcLl8YYRNOhqMuRXu6oSZHe2dGZY0ODNaGg1cg-kVigzY
 hF1MQ
DPoP: eyJ0eXAiOiJkcG9wK2p3dCIsImFsZyI6IkVTMjU2IiwiandrIjp7Imt0eSI6Ik
 VDIiwieCI6Imw4dEZyaHgtMzR0VjNoUklDUkRZOXpDa0RscEJoRjQyVVFVZldWQVdCR
 nMiLCJ5IjoiOVZFNGpmX09rX282NHpiVFRsY3VOSmFqSG10NnY5VERWclUwQ2R2R1JE
 QSIsImNydiI6IlAtMjU2In19.eyJqdGkiOiJlMWozVl9iS2ljOC1MQUVCIiwiaHRtIj
 oiR0VUIiwiaHR1IjoiaHR0cHM6Ly9yZXNvdXJjZS5leGFtcGxlLm9yZy9wcm90ZWN0Z
 WRyZXNvdXJjZSIsImlhdCI6MTU2MjI2MjYxOH0.lNhmpAX1WwmpBvwhok4E74kWCiGB
 NdavjLAeevGy32H3dbF0Jbri69Nm2ukkwb-uyUI4AUg1JSskfWIyo4UCbQ
```

DPoP public key bound access token

DPoP proof

Current Status and Updates

Traveled through Frankfurt retuning from the 4th OAuth Security Workshop where DPoP was largely conceived thereby justifying the use of this photo

draft-ietf-oauth-dpop

- -00 WG draft published on April 1st (no joke)
- -01 published on May 1st
  - (not insignificant) Editorial updates
  - More formally define the DPoP Authorization header scheme
  - Define the 401/WWW-Authenticate challenge
    - With an algs param
  - Added "invalid_dpop_proof" error code for DPoP errors in a token request
  - Fixed up and added to the IANA section
  - Added "dpop_signing_alg_values_supported" authorization server metadata
  - Moved the Acknowledgements into an Appendix and added a bunch of names (best effort looking back at emails)

[some] Open Questions

Currently pandemic fighting by self-isolating at home in Denver thereby justifying the use of this photo

Threat Model & Objectives

- Are not entirely clear
- But sometimes also maybe overly specific
- It’s a bit of a Rorschach test
- Honestly, I’m hoping Dr. Daniel Fett can help here

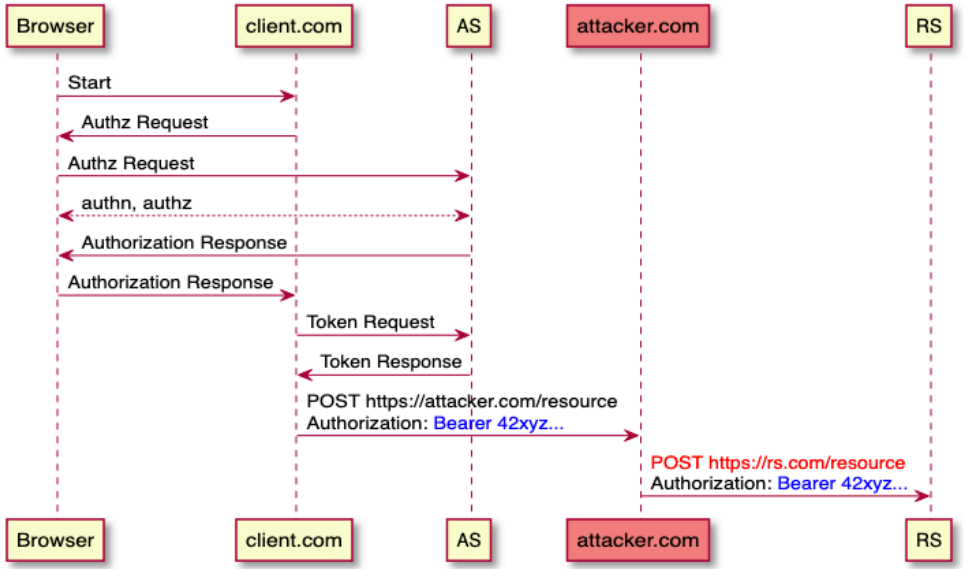
Attacker Model

Misconfigured Resource Endpoint

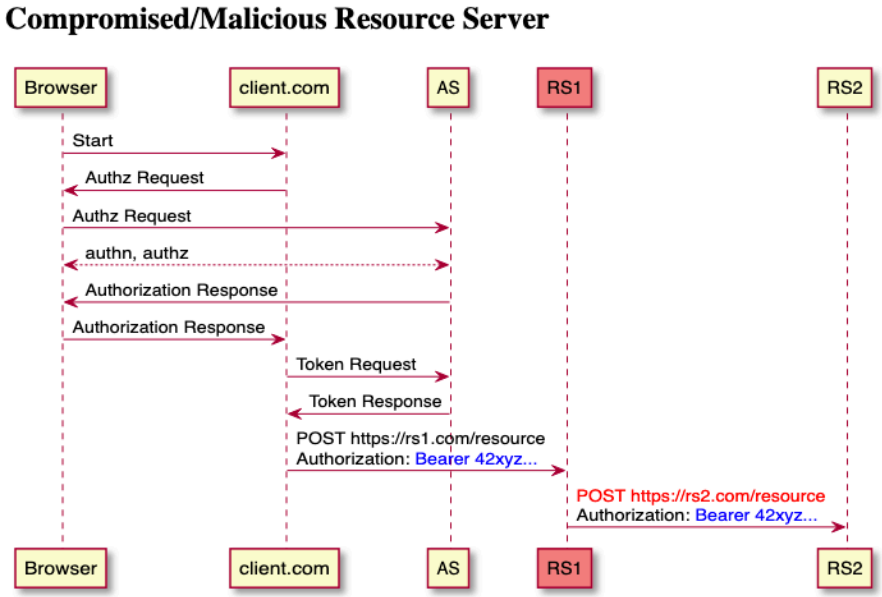
E.g.:

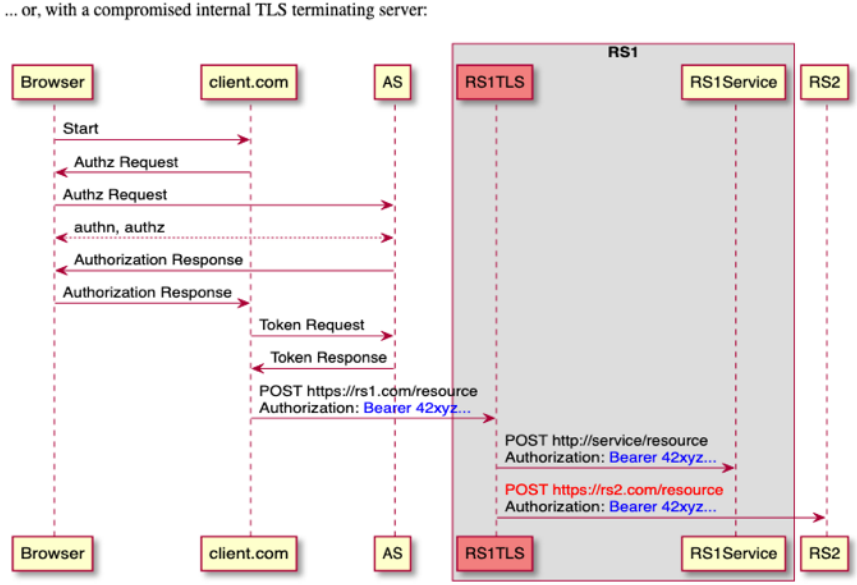
```
{
    "issuer": "https://attacker.com",
    "authorization_endpoint": "https://honest.com/authorize",
    "token_endpoint": "https://honest.com/token",
    "userinfo_endpoint": "https://attacker.com/userinfo"  #  ← attacker
}
```

Attack:



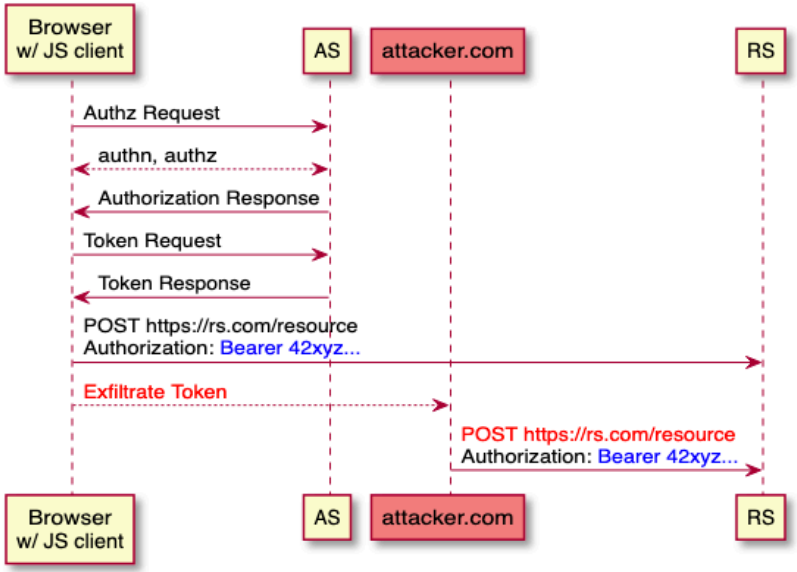
Attacker Model Cont.



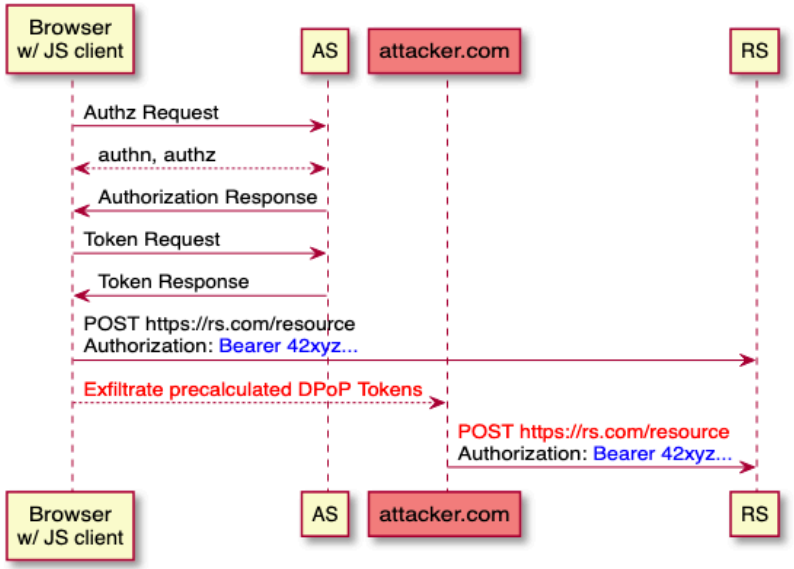


Attacker Model Cont.

Stolen Token (Offline-XSS)



Online-XSS (out of Scope)



Symmetric crypto is significantly more efficient than asymmetric

- True but there are other costs/complexities
- Real world implications unquantified
- A couple different potential approaches (at least)
  - Key distribution
  - Key agreement
- Consider this closed (for now anyway) coming out of the pre #107 interim meeting and WG adoption

Difficulties with `jti`

- Issues:
  - Detecting/preventing replay via `jti` can be very problematic for large-scale deployments (also exacerbating inefficiencies asymmetric crypto)
  - Can interfere with idempotence and retry
- Current situation:
  - `iat` can also limit replay window
  - replay check on `jti` is only a SHOULD and also qualified “within a reasonable consideration of accuracy and resource utilization, a JWT with the same jti value has not been received previously”
- Some options/ideas … ?
  - Explicitly mention that the replay space is qualified by the URI and method thus reducing the scope of data replication needed
    - There was a mention of splitting path out from htu
  - Further loosen/qualify (like perhaps a MAY)
  - Drop the tracking requirement all together
  - Something else

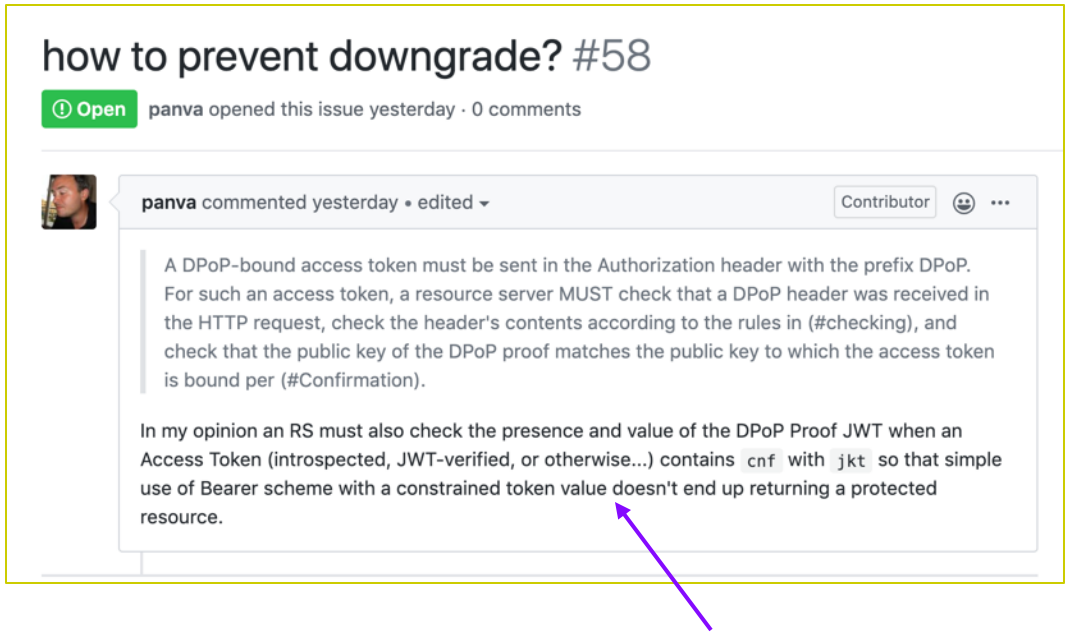
Signal that the RT is bound?

- Issue:
  - “useful to be able to have DPoP refresh tokens and Bearer access tokens as a transition step” but “It seems like the spec requires the same token_type for both access tokens and refresh tokens” - IIW summary
    - Note that token_type applies to the access token per RFC 6749
- Current situation:
  - **RTs are only bound for public clients** (this needs apparently needs better treatment in the draft)
  - DPoP access tokens are (most likely) useable as Bearer access tokens
  - Does the client need a signal?
- An option/idea … ?
  - A new token endpoint response parameter could be introduced
    - i.e. “the_refresh_token_in_this_here_response_is_dpopped”: true

Client Metadata?

- “were supportive of defining … [Client] Registration Metadata to declare support for DPoP ... [which] might [be] supported token_type values.” – IIW summary
- But the utility of client metadata isn’t entirely clear

Downgrades, Transitional Rollout & Mixed Token Type Deployments



In my opinion, we don’t want to do this.
And in reality, I don’t think we really can.

- JWT: “in the absence of [application specific] requirements, all claims that are not understood by implementations MUST be ignored.”
- Introspection: “implementations MAY extend this structure with their own service-specific response names”
- RFC 6750 is silent on it
- Ergo, DPoP bound access tokens are (most likely) useable as Bearer access tokens at existing RFC 6750 protected resources
- New policy and implementation can be introduced

Freshness & Scope of Signature

- Issue:
  - “[no] guarantees that the DPoP signature was freshly generated, as there is no nonce from the server incorporated into the signature.”
- Current Situation:
  - `iat` doesn’t keep it fresh with respect to pre-computation by an adversary who somehow (XSS?!) can use the private key but not steal it
  - No challenge/response was intentional
- Some options/ideas … ?
  - It’s sufficiently okay as is
  - “People agreed that having a server nonce would add additional security” and “[someone is] already… providing the nonce as a WWW-Authenticate challenge” value– IIW summary
  - Incorporate a hash of the authorization code, refresh token, access token, other artifact into the DPoP proof
  - Other…

Gratuitous closing slide featuring the city where will meet together next *

* *Maybe* Bangkok in the fall