OAuth 2.0 Attestation-Based Client Authentication

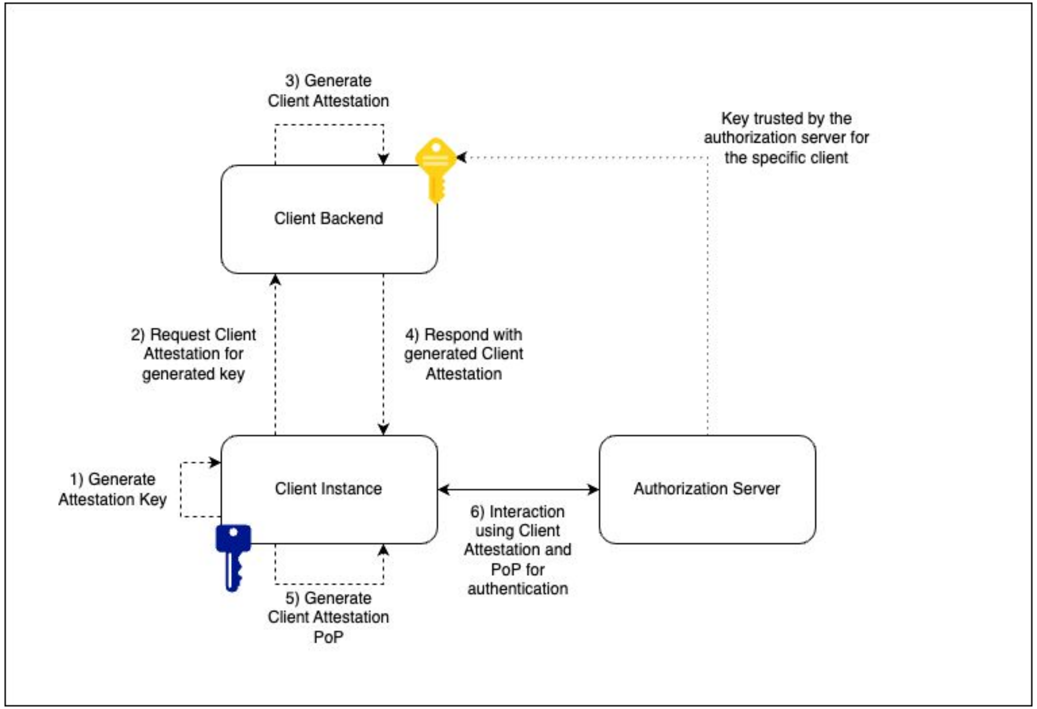
Tobias Looker, Paul Bastian

A Refresher - Motivation

- Environments that *public* clients are operating/deployed in increasingly have primitives that can be used for client authentication examples such as:
  - App Attest on iOS for Native iOS applications
  - Play Integrity on Android for Native Android applications
- The question is how to appropriately use these capabilities to allow clients to authenticate with an authorization server?

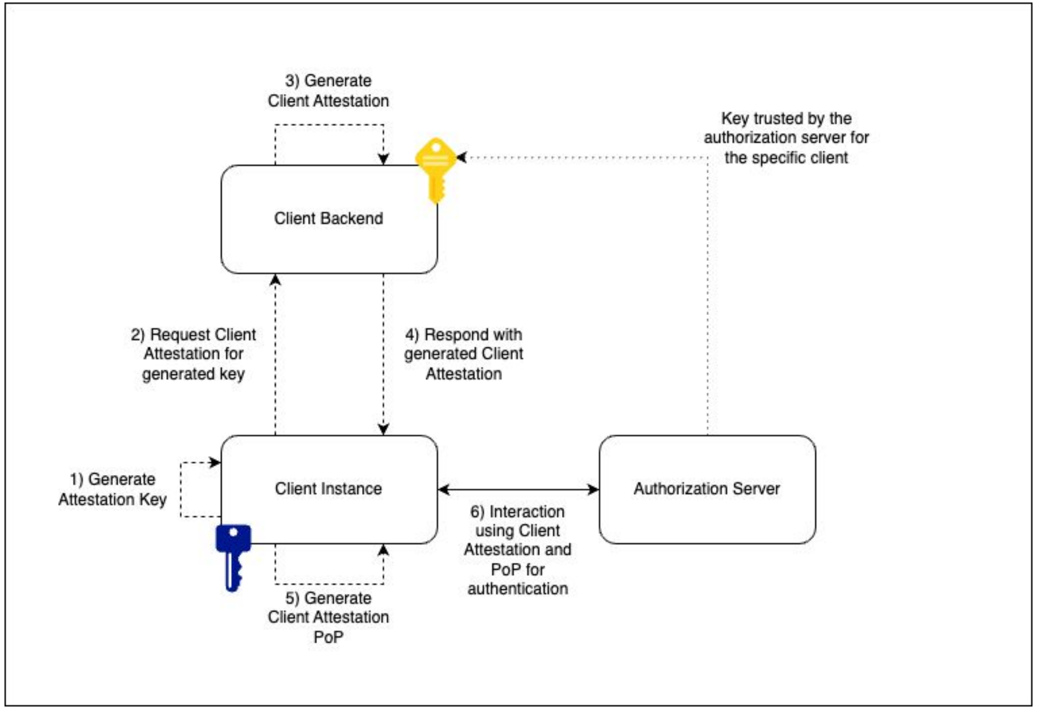
Proposed Solution

- Extends the established framework of RFC7521 for a new form of client authentication
- Client instance obtains an attestation from client backend
- Client backend may perform any number of security checks before issuing a key-bound attestation JWT to the client instance
- Client instance authenticates towards Authorization server during a token or PAR request
- **Note -** how the client communicates with the client backend in steps 2&4 are out of scope

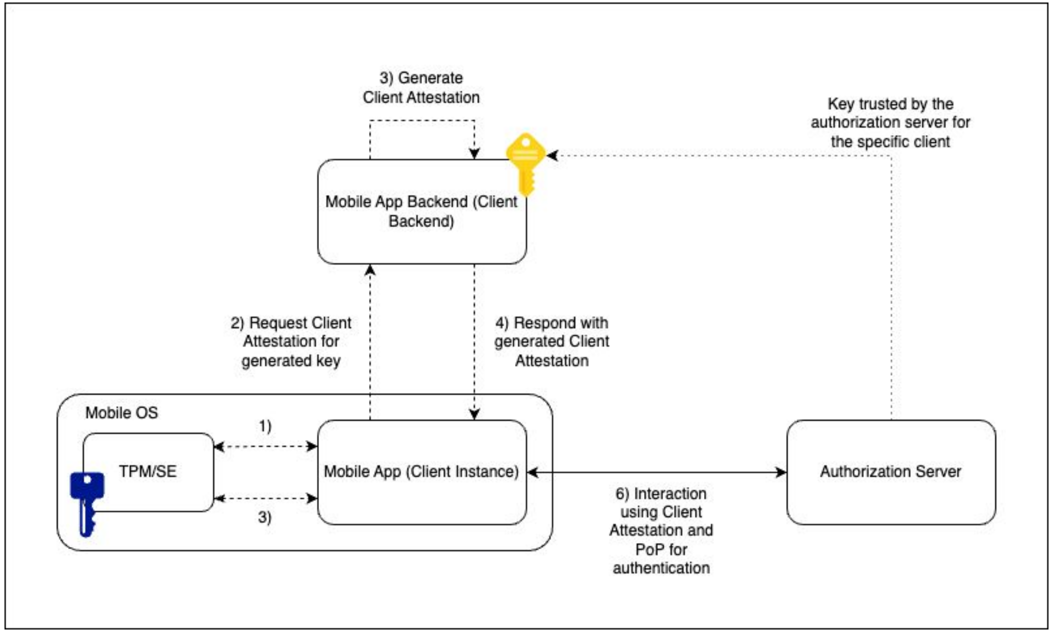

3) Generate Client Attestation
Key trusted by the authorization server for the specific client
Client Backend
2) Request Client Attestation for generated key
4) Respond with generated Client Attestation
1) Generate Attestation Key
Client Instance
Authorization Server
6) Interaction using Client Attestation and PoP for authentication
5) Generate Client Attestation PoP

Key Callouts

- Proof of possession enabled client authentication method
- Can be used to authenticate the key used to bind to an access token via DPoP
- Direct mode of authentication between the client instance and the authorization server rather than a backend for front end pattern
- Avoids the client instance from having to register with the AS via DCR


3) Generate Client Attestation
Key trusted by the authorization server for the specific client
Client Backend
2) Request Client Attestation for generated key
4) Respond with generated Client Attestation
1) Generate Attestation Key
Client Instance
Authorization Server
6) Interaction using Client Attestation and PoP for authentication
5) Generate Client Attestation PoP

Native App Example


3) Generate Client Attestation
Key trusted by the authorization server for the specific client
Mobile App Backend (Client Backend)
2) Request Client Attestation for generated key
4) Respond with generated Client Attestation
Mobile OS
1)
TPM/SE
3)
Mobile App (Client Instance)
Authorization Server
6) Interaction using Client Attestation and PoP for authentication

Example - Token Request

```
POST /token HTTP/1.1
Host: as.example.com
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&
code=n0esc3NRze7LTCu7iYzS6a5acc3f0ogp4&
client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3A
client-assertion-type%3Ajwt-client-attestation&
client_assertion=eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
eyJpc3Mi[...omitted for brevity...].
cC4hiUPo[...omitted for brevity...]~eyJzI1NiIsImtphbGciOimtpZCI6IjIyIn0.
IjIyIn0[...omitted for brevity...].
iOiJSUzI1[...omitted for brevity...]
```

Example - Token Request

```
POST /token HTTP/1.1
Host: as.example.com
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&
code=n0esc3NRze7LTCu7iYzS6a5acc3f0ogp4&
client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3A
client-assertion-type%3Ajwt-client-attestation&
client_assertion=eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
eyJpc3Mi[...omitted for brevity...].
cC4hiUPo[...omitted for brevity...]~eyJzI1NiIsImtphbGciOimtpZCI6IjIyIn0.
IjIyIn0[...omitted for brevity...].
iOiJSUzI1[...omitted for brevity...]
```

New assertion type

Example - Token Request

```
POST /token HTTP/1.1
Host: as.example.com
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&
code=n0esc3NRze7LTCu7iYzS6a5acc3f0ogp4&
client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3A
client-assertion-type%3Ajwt-client-attestation&
client_assertion=eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
eyJpc3Mi[...omitted for brevity...].
cC4hiUPo[...omitted for brevity...]~eyJzI1NiIsImtphbGciOimtpZCI6IjIyIn0.
IjIyIn0[...omitted for brevity...].
iOiJSUzI1[...omitted for brevity...]
```

Two JWTs concatenated via a '~' character

- Client Attestation
- Client Attestation PoP

Example - Client Assertion

Client Attestation

Client Attestation PoP

```
eyJhbGciOiAiRVMyNTYiLCJraWQiOiAiMTEifQ.eyJpc3MiOiJodHRwcz
ovL2NsaWVudC5leGFtcGxlLmNvbSIsInN1YiI6Imh0dHBzOi8vY2xpZW5
0LmV4YW1wbGUuY29tIiwibmJmIjoxMzAwODE1NzgwLCJleHAiOjEzMDA4
MTkzODAsImNuZiI6eyJqd2siOnsia3R5IjoiRUMiLCJ1c2UiOiJzaWciL
CJjcnYiOiJQLTI1NiIsIngiOiIxOHdITGVJZ1c5d1ZONlZEMVR4Z3BxeT
JMc3pZa01mNko4bmpWQWlidmhNIiwieSI6Ii1WNGRTNFVhTE1nUF80Zlk
0ajhpcjdjbDFUWGxGZEFnY3g1NW83VGtjU0EifX19.SflKxwRJSMeKKF2
QT4fwpMeJf36POk6yJV_adQssw5c~eyJhbGciOiJFUzI1NiJ9.eyJpc3M
iOiJodHRwczovL2NsaWVudC5leGFtcGxlLmNvbSIsImF1ZCI6Imh0dHBz
Oi8vYXMuZXhhbXBsZS5jb20iLCJuYmYiOjEzMDA4MTU3ODAsImV4cCI6M
TMwMDgxOTM4MH0.coB_mtdXwvi9RxSMzbIey8GVVQLv9qQrBUqmc1qj9B
s
```

Note signatures are invalid

Example - Client Attestation

```
{
    "alg": "ES256",
    "kid": "11"
}
.
{
  "iss": "https://client.example.com",
  "sub": "https://client.example.com",
  "nbf": 1300815780,
  "exp": 1300819380,
  … //other claims
  "cnf": {
    "jwk": {
      "kty": "EC",
      "crv": "P-256",
      "x": "18wHLeIgW9wVN6VD1Txgpqy2LszYkMf6J8njVAibvhM",
      "y": "-V4dS4UaLMgP_4fY4j8ir7cl1TXlFdAgcx55o7TkcSA"
    }
  }
}
```

Example - Client Attestation

```
{
    "alg": "ES256",
    "kid": "11"
}
.
{
  "iss": "https://client.example.com",
  "sub": "https://client.example.com",
  "nbf": 1300815780,
  "exp": 1300819380,
  … //other claims
  "cnf": {
    "jwk": {
      "kty": "EC",
      "crv": "P-256",
      "x": "18wHLeIgW9wVN6VD1Txgpqy2LszYkMf6J8njVAibvhM",
      "y": "-V4dS4UaLMgP_4fY4j8ir7cl1TXlFdAgcx55o7TkcSA"
    }
  }
}
```

Key used to verify the Client Attestation PoP

Use Cases

- Potentially applicable to any OAuth 2.0 Flow that uses Client Authentication
- Concrete applications include:
  - eIDAS 2.0 usage of OpenID for Verifiable Credentials
  - Software workload authorization - enabling ephemeral software workload instances to authenticate with authorization servers without having to register with the authorization server first

Progress Update

- Simplified and generalized introduction, updated the diagram
- Added guidance around replay attack detection
- Added explanations that client authentication mechanism is compatible to extensions like PAR
- Clarified usage of attestations with refresh tokens
- Fixed text around jti claim usage
- Fixed text around cnf claim
- Added examples matching to eIDAS

Work in Progress

- Authenticator Assurance Level (aal)
- Renaming Client Backend to Client Attester
  - Iss of Client Attestation must not be equal to its sub
- Discussion on mandatory JWT typ values / media types for Client Attestation
- Discussion and further defining the nonce mechanism for replay attack detection
  - jti vs nonce endpoint vs nonce error
- Discussions on generalizing the mechanism for other use cases but client authentication
  - should this mechanism be used to authenticate a wallet (AS) towards a RP (client) ? -> OpenID “Advanced Flow”
- Discussions on how to improve Client Attestations with Refresh Tokens

Links

Current Editors Copy -> https://vcstuff.github.io/draft-looker-oauth-attestation-based-client-auth/draft-looker-oauth-attestation-based-client-auth.html

Git Repository -> https://github.com/vcstuff/draft-looker-oauth-jwt-cwt-status-list

Questions?

Backup

Why not other approaches

- **Private key JWT Based Client Authentication**
  - Is a bearer based authentication mechanism so vulnerable to certain modes of token thief
  - A client instance would likely have to obtain a new attestation from a client backend for every AS interaction involving client authentication
- **Backend for Front Style Client Authentication**
  - Confidentiality and privacy issues for certain use-cases such as verifiable credential issuance where requests/responses would proxy through the client backend