OAuth 2.0 Mutual TLS Client Authentication and Certificate Bound Access Tokens

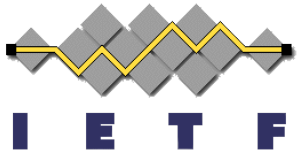
Brian Campbell
John Bradley
Nat Sakimura
Torsten Lodderstedt

IETF 101
London
March 2018

draft-ietf-oauth-mtls

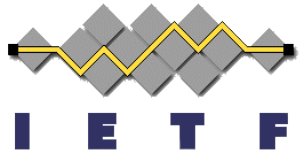
https://tools.ietf.org/html/draft-ietf-oauth-mtls-07

Context: What is it?



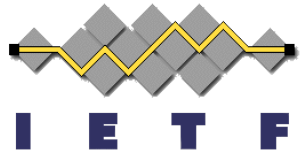
- Mutual TLS client authentication to the AS
  - Two methods:
    - PKI based
    - Self-signed certificate based mode
- Mutual TLS sender constrained access tokens for protected resources access
  - Certificate bound access tokens

Context: Why?



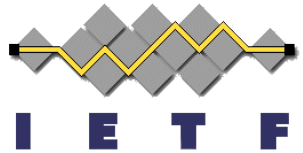
- Mutual TLS client authentication, which provides better security characteristics than shared secrets, is something that's been done in practice for OAuth but we've never had a spec for it
- Mutual TLS sender constrained resources access binds access tokens to the client certificate so they can't be (re)played or used by any other entity without proof-of-possession
- Banks "need" these for API use cases being driven by new open banking regulations
- Referenced by the OpenID Foundation's Financial API (FAPI) WG's "Read and Write API Security Profile" as a suitable holder of key mechanism
- Referenced by the UK Open Banking API Security Profile

Context: How Mutual TLS Client Authentication Works



- MTLS client authentication to the authorization server
  - TLS connection from client to token endpoint is established with mutual X509 certificate authentication
  - Client includes the "client_id" HTTP request parameter in all requests to the token endpoint
  - AS verifies that the MTLS certificate is the ‘right’ one for the client (based on configuration and method)
    - Metadata supporting the PKI method
      - Authentication Method Name: "tls_client_auth”
      - Client Metadata: "tls_client_auth_subject_dn” specifies the expected subject distinguished name of the client certificate
    - Metadata supporting the Self-signed Certificate method
      - Authentication Method Name: "self_signed_tls_client_auth”
      - The existing "jwks_uri" or "jwks" RFC7591 metadata parameters used to convey a client's certificate(s)

Context: How Mutual TLS Sender Constrained Access Works



- AS associates a hash of the certificate with the access token
  - certificate bound access token
- TLS connection from client to resource is mutually authenticated TLS
  - The protected resource matches certificate from TLS connection to the certificate hash in the access token
- JWT Confirmation Method
  - X.509 Certificate SHA-256 Thumbprint Confirmation Method: x5t#S256
- Confirmation Method for Token Introspection
  - Same data as JWT x5t#S256 confirmation returned in the introspection response and checked by the protected resource
- Doesn't vary based on client authentication method

JWT Confirmation

```
{
  "iss": "https://server.example.com",
  "sub": "ty.webb@example.com",
  "exp": 1493726400,
  "nbf": 1493722800,
  "cnf":{
    "x5t#S256": "bwcK0esc3ACC3DB2Y5_lESsXE8o9ltc05O89jdN-dg2"
  }
}
```

Token Introspection

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "active": true,
  "iss": "https://server.example.com",
  "sub": "ty.webb@example.com",
  "exp": 1493726400,
  "nbf": 1493722800,
  "cnf":{
    "x5t#S256": "bwcK0esc3ACC3DB2Y5_lESsXE8o9ltc05O89jdN-dg2"
  }
}
```

Changes since Singapore

- Drafts -06 & -07
- Use RFC 8174 boilerplate
- Reference update to AS Metadata
- Move the Security Considerations section to before the IANA Considerations
- Elaborated on certificate bound access tokens a bit more in the Abstract
- Changed the title to be more descriptive
- A bit more text on certificate spoofing and CAs in the Security Considerations
- Add an explicit note that the implicit flow is not supported for obtaining certificate bound access tokens
- Add appendix describing the relationship of OAuth MTLS to OAuth Token Binding

Next Steps... WGLC?

"WGLC will be issued in december after clarification."

– Singapore WG meeting minutes