# Data Standards Body

## Information Security Technical Working Group (ISTWG)

### Decision Proposal 033 – TLS/MTLS and Access Tokens

*Contact: Seyit Camtepe*

*Publish Date: 09 October 2018*

*Feedback Conclusion Date: 16 October 2018*

## Context

The Information Security Working Group's starting point is the UK open banking security profile, based on Open ID's Financial-grade API (FAPI) Read/Write API Security Profile [11]. The FAPI profile builds upon a set of OAuth2.0 [2] [3] [4] [5], OIDC [7] [8] [9] and TLS [1] [6] standards, drafts and specifications. The elements of the FAPI profile are illustrated in Appendix A, Figure 3.

This decision proposal identifies Transport Layer Security models (TLS) and associated services used to secure interactions between CDR stakeholders. The interactions within scope of this proposal are illustrated in Appendix B, Figure 4.

## Decisions to be made

1. The adoption of TLS and MTLS to secure communications between CDR stakeholders.
2. Use of cryptographic primitives.
3. Support for Certificate Bound Access Tokens.
4. Certificate extensions.

**Note: at points, these decisions make presumptions that a certificate authority will be part of authorisation and authentication, and that a Register will act as certificate authority. A procurement process is currently underway regarding a Register solution, administered by the Australian Competition & Consumer Commission, which is separate to this process. The discussions in this working group do not determine or shape that process. A solution design for that Register has not been determined, and may impact the final standards.**

Certain assumptions are made regarding the role played by a certificate authority, and how certificate management and revocation will work. The use of certificate bound access tokens is recommended, for example. We have included these decisions because they are important in the context of making decisions about TLS/MTLS.

Note: while not within scope of these proposals, certificate revocation needs to be considered. Certificates may be revoked by the authority for several reasons. Whatever Register solution is adopted, methods will have to be adopted to inform stakeholders about revoked certificates. While certificate revocation is outside of scope for the Information

Security Working Group, this proposal considers that any Register solution would maintain a certificate revoke list (CRL) and OCSP (Online Certificate Status Protocol – RFC 2560) where stakeholders can check the (revocation) state of a certificate online.

Note: authentication and authorisation through redirect and decoupled flow options will be considered in later decision proposals.

***The adoption of MTLS and TLS***

This document proposes the adoption of TLS 1.3 and use of MTLS with certificates as follows:

(A) Accredited Receiver to Register communication is protected by **TLS** using a server side external certificate.
(B) Data Holder to Register communication is protected by **TLS** using a server side external certificate.
(C) Accredited Receiver to Registration Endpoint of Data Holder communication is protected by **MTLS** using client and server side certificates.
(D) Consumer to Accredited Receiver communication is protected by **TLS** using a server side external certificate.
(E) Consumer to Authorization Endpoint of Data Holder communication (directed or CIBA) is protected by **TLS** using a server side external certificate.
(F) Authorization Endpoint of Data Holder to Redirect Endpoint of Accredited Receiver communication is protected by **MTLS** using client and server side certificates.
(G) Accredited Receiver to Token Endpoint of Data Holder communication is protected by **MTLS** using client and server side certificates.
(H) Accredited Receiver to Resource Endpoint of Data Holder communication is protected by **MTLS** using client and server side certificates.

For an endpoint to verify a certificate, it should know the public key of the certificate authority.

***Cryptographic Primitives and Key Sizes***

This decision proposal recommends the use of cryptographic primitives supported by TLS 1.3, and the adoption of an at least 80-bit security rating, preferably over 128-bit security rating, following the NIST key size guidelines. TLS 1.3 removes certain cryptographic primitives, which are outlined in greater detail under Considerations.

***Support for Certificate Bound Access Tokens***

This decision proposal supports the adoption of certificate bound access tokens with MTLS, to prevent the replay of or use of stolen access tokens by any malicious parties.

***Certificate Extensions***

This decision proposal recommends the adoption of standard X.509 v3 extensions, and to define custom extensions for passing accreditation information and redirect-uri information from the accredited receiver to data holder. This will be used as an additional protection on top of the methods proposed by OAuth2.0 standards and provide a means for data holders to cross check uris before redirecting the customer along with authorization code, access token or ID token.

## Considerations informing decision proposals

***TLS/MTLS***

TLS 1.3 (RFC 8446) [1] was published in August 2018. It reduces the number of messages exchanged within the handshake protocol. Key exchange starts with the first message, meaning that application data encryption can start immediately after the second message, before the handshake is completed. Moreover, if shared key option is used, data encryption can start with the first message exchanged in the subsequent connections (a.k.a., 0-RTT) .This way, TLS 1.3 achieves a better performance. TLS 1.3 also removes cryptographic primitives which are proven to be insecure, such as SHA-1, MD5, RC4, DES, 3DES, CBC mode ciphers (AES-CBC) and RSA key exchange.

TLS use for confidentiality and authentication is recommended and enforced by the FAPI specifications, UK OB security profiles [12], OAuth2.0 standards and OIDC specifications. For example:

- **Shall** authenticate the confidential client at the token endpoint using methods including TLS [FAPI Read-Write]
- The authorization server **MUST** require the use of TLS [OAuth2.0, OIDC]
- Communication with the Token Endpoint **MUST** utilize TLS. [OIDC]
- The redirection endpoint SHOULD require the use of TLS [OAuth2.0]
- Access token credentials **MUST** only be transmitted using TLS [OAuth2.0]
- Refresh tokens **MUST** be kept confidential in transit and storage [OAuth2.0]

Though MTLS use brings security advantages, it has drawbacks as illustrated in Figure 1. If a data holder or accredited receiver is using a TLS termination proxy, information stored in the certificate (e.g., redirect url, subject name and accreditation codes) may not reach to the Endpoint. The information conveyed in the client certificate has to be extracted by the proxy and communicated to the endpoint (server).

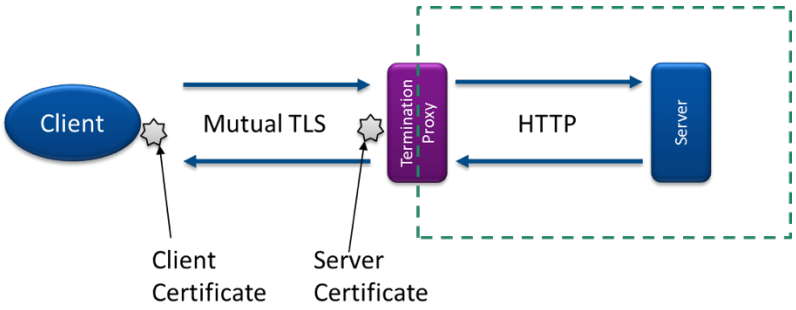


*Figure 1 TLS Termination Proxy*

### *Cryptographic primitives and key sizes*

TLS 1.3 standard (RFC 8446) [1] removes cryptographic primitives which are proven to be insecure such as SHA-1, MD5, RC4, DES, 3DES, CBC mode ciphers (AES-CBC) and RSA key exchange. Hence, TLS 1.3 standard can be used as the guideline for cryptographic primitive selection in key agreement, encryption, signatures and hashes.

Selection of key size is an important aspect of security. Any cryptographic scheme having less than an 80-bit security rating is considered as weak. 80-bit security means that a computer needs to search $2^{80}$ different keys in a brute force key attack to break the code. State-of-the-art computing systems can achieve a brute force key attack for the algorithms having less than 80-bit security in a reasonable time. AES-128 (AES with 128 bit key size) provides 128-bit security scale, which is assumed to be safe. Additionally, it has been shown that brute-force key search on quantum computers cannot be faster than $2^{n/2}$, which means that AES-256 (AES with 256 bit key size) provides 128-bit security scale against quantum brute force key attack. Table 1 by NIST compares the security scale of the symmetric and asymmetric key schemes.

*Table 1 NIST Key Size Guideline*

| Security Scale | Symmetric Key Cryptography Key Sizes | RSA and Diffie-Hellman Key Sizes | Elliptic Curve Cryptography Key Sizes |
|---|---|---|---|
| **80-bit** | 80-bit | 1024-bit | 160-bit |
| **112-bit** | 112-bit | 2048-bit | 224-bit |
| **128-bit** | 128-bit | 3072-bit | 256-bit |
| **192-bit** | 192-bit | 7680-bit | 384-bit |
| **256-bit** | 256-bit | 15360-bit | 521-bit |

### *Certificate Bound Access tokens*

Use of MTLS with client (accredited receiver) and server (data holder) certificates brings two benefits: (i) mutual authentication between the accredited receiver and the end-points of data holder (Figure 4, Appendix B) and (ii) binding access tokens to the certificate. The OAuth 2.0 authorization framework defined in RFC6749 uses a shared secret method of client authentication and it allows the use of additional client authentication mechanisms when interacting directly with the data holder's end-points.

Authentication is achieved in two steps. In the first step, the TLS handshake protocol (Figure 5, Appendix C) is utilized to validate the accredited receiver's possession of the private key corresponding to the public key presented within the certificate. In the second step, Subject DN (distinguished name) within the validated certificate is matched to the registered information of the accredited receiver. An Accredited receiver selects a Subject DN while registering with a register, and receives a X.509 v3 certificate with this Subject DN. Later, an accredited receiver uses the same Subject DN while registering to the data holder. This method requires the accredited receiver to use the same Subject DN every time a new

certificate is received from the register. Changes in the Subject DN require change to the register and data holder registration.

With mutual TLS certificate bound access tokens [2], resource end-points of the data holder ensure that only the party who is in possession of the private key corresponding to the certificate can utilize the token to access the associated resources. Such a binding between access tokens and the MTLS client (accredited receiver) certificates prevents the use of stolen access tokens and the replay of access tokens by any malicious parties. Access token binding is achieved through use of certificate fingerprints (certificate hash). Following the abovementioned authentication process, the token end-point of data holder includes the certificate fingerprint within the access token issued to accredited receiver. When an accredited receiver contacts the resource end-point of data holder with the access token, following the abovementioned authentication process, the resource end-point matches the fingerprint stored in the token with the fingerprint of the certificate presented.

This scheme has two drawbacks though. First, the scheme cannot be used with implicit grant of OAuth2.0 (a.k.a. implicit flow of OIDC) since in the implicit flow, the accredited receiver receives the access token from the authorization end-point, not from the token end-point (Figure 4). Second, data holders using TLS termination proxy may require an additional method to communicate the certificate validity status and Subject DN information to the relevant end-points (Figure 1).

***Certificate Extensions***

There are two pieces of critical information that need to pass between stakeholders: redirect-uri information of OAuth2.0, and accreditation information. As illustrated in Figure 2, X.509 v3 provides extensions: there are number of standard extensions and the possibility to define custom extensions.

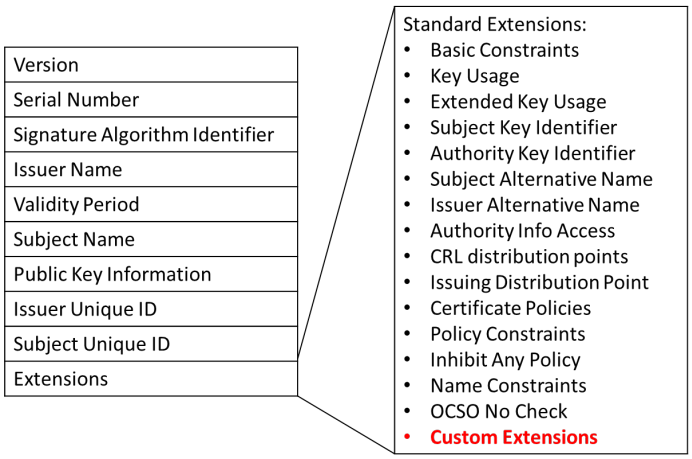


*Figure 2 X.509 v3 Certificates*

In OAuth 2.0, after completing its interaction with the customer, the authorization end-point of a data holder directs the customer's user-agent back to the accredited receiver using the redirect-uri. A redirect-uri is established with the data holder during accredited receiver to data holder registration (interaction (C) of Figure 4, Appendix B) or when an accredited receiver makes the authorization request. Redirect-uri is critical information and

can lead to variety of attacks including Phishing, Cross-Site Request Forgery and Code Injection (OAuth2.0 – RFC6749) if not protected properly. MTLS and accredited receiver certificates provide a safe means to pass this information along with accreditation status to data holder, through the use of two additional custom extensions.

## References


[1]. E. Rescorla, "*The Transport Layer Security (TLS) Protocol Version 1.3*", Internet Engineering Task Force (IETF), Request for Comments 8446, August 2018.

[2]. D. Hardt, "*The OAuth 2.0 Authorization Framework", Internet Engineering Task Force (IETF)*, Request for Comments 6749, October 2012.

[3]. M. Jones, D. Hardt, "*The OAuth 2.0 Authorization Framework: Bearer Token Usage*", Internet Engineering Task Force (IETF), Request for Comments 6750, October 2012.

[4]. N. Sakimura, J. Bradley, N. Agarwal, "*Proof Key for Code Exchange by OAuth Public Clients*", Internet Engineering Task Force (IETF), Request for Comments 7636, September 2015.

[5]. P. Hunt, T. Nadalin, "*OAuth 2.0 Software Statement*", OAuth Working Group Internet Draft, 27 September 2013 (expired on 31 March 2014).

[6]. B. Campbell, J. Bradley, N. Sakimura, T. Lodderstedt, *"OAuth 2.0 Mutual TLS Client Authentication and Certificate Bound Access Tokens"*, OAuth Working Group Internet Draft, 30 August 2018 (expires on 3 March 2019).

[7]. N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, C. Mortimore, "*OpenID Connect Core 1.0 incorporating errata set 1*", Open ID, https://openid.net/specs/openid-connect-core-1_0.html, November 2014.

[8]. G. Fernandez, F. Walter, A. Nennker, "*OpenID Connect MODRNA Client initiated Backchannel Authentication Flow 1.0*", Open ID, https://openid.net/specs/openid-connect-modrna-client-initiated-backchannel-authentication-1_0.html, March 2017.

[9]. N. Sakimura, J. Bradley, M. Jones, "*OpenID Connect Dynamic Client Registration 1.0 incorporating errata set 1*", Open ID, https://openid.net/specs/openid-connect-registration-1_0.html, November 2014.

[10]. N. Sakimura, J. Bradley, E. Jay, "*Financial-grade API - Part 1: Read-Only API Security Profile*", Open ID, https://openid.net/specs/openid-financial-api-part-1.html, August 2018.

[11]. N. Sakimura, J. Bradley, E. Jay, "*Financial-grade API - Part 2: Read and Write API Security Profile*", Open ID, https://openid.net/specs/openid-financial-api-part-2.html, August 2018.

[12]. Open Banking Limited 2018, "*Open Banking Developer Zone – Specifications: API, Read/Write, Security Profile, Directory*" , https://www.openbanking.org.uk/, 2018.


## Appendix A: High level overview FAPI profile

- FAPI builds upon a REST/JSON data model protected by an OAuth profile.
- OAuth2.0 is an authorization layer separating the role of the client from that of the resource owner: the client (accredited receiver) requests access to resources (financial data) controlled by the resource owner (customer) and hosted by the resource server (data holder).

- OIDC (OpenID Connect) is a simple identity layer on top of the OAuth 2.0 protocol: the client (accredited receiver) verifies the identity of the End-User (customer) based on the authentication performed by an Authorization Server (data holder).
- MTLS (Mutual Transport Layer Security) is a protocol that provides encrypted communications and endpoint authentication.

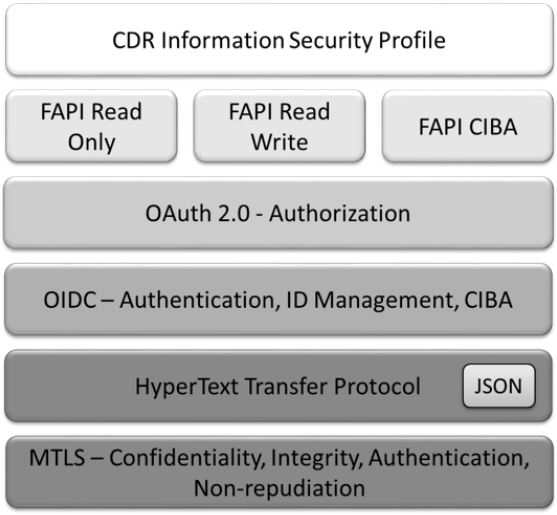


*Figure 3 CDR Information Security Stack*

## Appendix B: CDR stakeholder interactions

The interactions within scope of TLS related proposals include (Figure 4):

(A) Receiver gets accredited, registers and obtains SSA by contacting Register, becomes Accredited Receiver.
(B) Data Holder registers by contacting Register.
(C) Accredited Receiver contacts to Registration Endpoint of Data Holder to register.
(D) Consumer contacts Accredited Receiver.
(E) Consumer is directed (or CIBA) to Authorization Endpoint of Data Holder.
(F) Authorization Endpoint of Data Holder contacts the Redirect Endpoint of Accredited Receiver.
(G) Accredited Receiver contacts Token Endpoint of Data Holder.
(H) Accredited Receiver contacts Resource Endpoint of Data Holder.

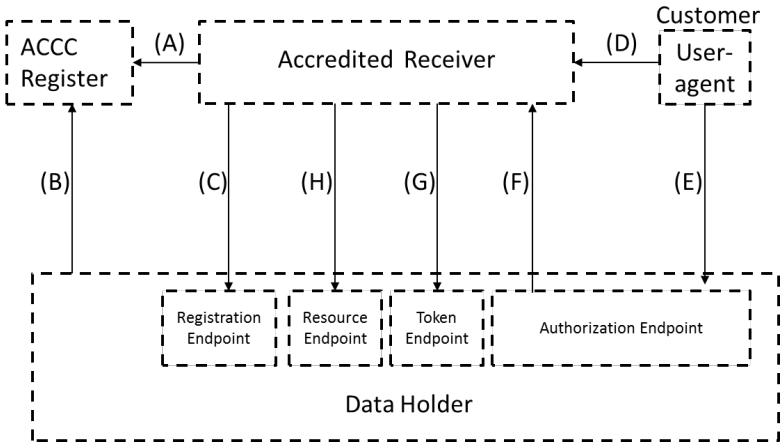


*Figure 4 CDR Stakeholder Interactions*

## Appendix C: Transport Layer Security (TLS)

Transport Layer Security (TLS) is a cryptographic protocol designed to provide secure communications on networks. As illustrated in Figure 5, TLS starts with a handshake protocol where a client and a server authenticate each other, exchange the cryptographic primitives that they support and establish a symmetric key to encrypt the application data for confidentiality. TLS is quite commonly used as HTTPS (i.e., HHTP over TLS) by major user-agents (e.g., browsers). Digital certificate (i.e., X.509 v3) offers an effective way to authenticate TLS parties to each other. While sever side digital certificate is commonly used, client side certificates are less common due to reasons including (i) the difficulty of managing certificates within a large client base, and (ii) difficulty of moving a certificate across a set of devices that a client may like to use, e.g., desktop, tablet, smarphone, etc. A digital certificate includes information (Figure 2) such as Issuer Name, Validity period, Subject name, Subject Public Key Info, Public Key Algorithm, Subject Public Key, Issuer Unique Identifier (optional), Subject Unique Identifier (optional), and Extensions (optional). These information are signed by the issuer using public key cryptography. Trusted certificate issuers are organized within a tree structures where an issuer at the root can certify another issuer to distribute certificate or certify other issues. Accordingly, certificates often carry a chain of signatures. Common issuers are known to the most user-agents (e.g., their public key is burned-in the user-agent), hence any certificate issued by them or by issuers certified by them can be validated by the user-agent.

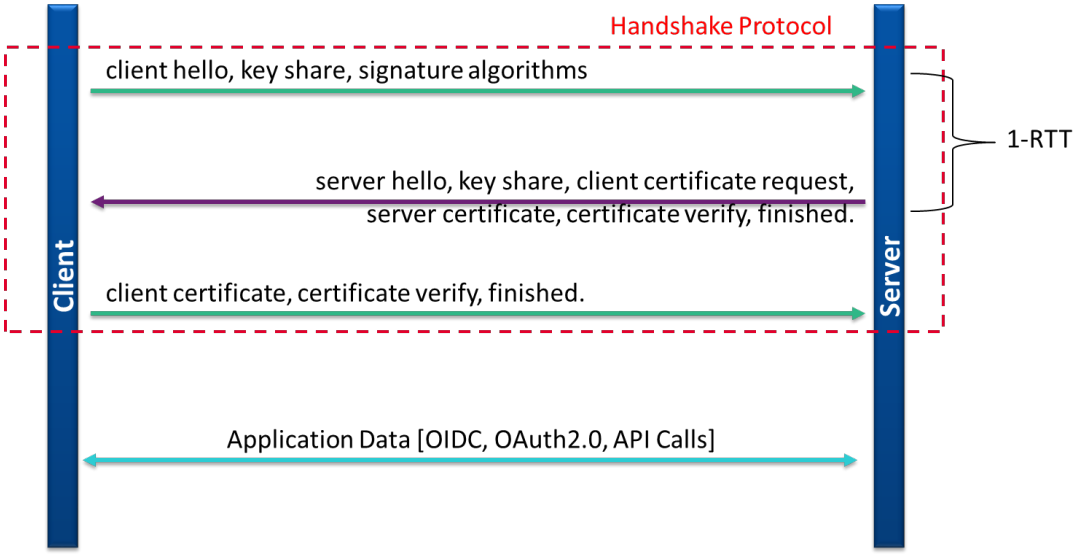


*Figure 5 TLS 1.3 Handshake*

## Appendix D: Stakeholders

*Table 2 Consumer Data Rights stakeholder mapping - stakeholder names in different documentation.*

| | Stakeholder 1 | Stakeholder 2 | Stakeholder 3 | Stakeholder 4 |
|---|---|---|---|---|
| **AU CDR** | Customer | Data Holder | Accredited Receiver | Register |
| **TLS 0[6]** | Client to data holders and accredited receivers | Client to register, server to accredited receiver and customer | Client to register and data holder, server to customer | Server to customers, data holders and accredited receivers |
| **OIDC [7][8][9]** | End-user, Consumption Device, Authentication Device | OpenID Provider | Relying Party and Client | - |
| **OAuth2.0 [2][3][4][5]** | Resource Owner | Authorization and Resource Servers | Client | - |
| **FAPI [10][11]** | End-user | Authorization and Resource Servers | Client | - |
| **UK OB [12]** | PSU: Payment Service User | ASPSP: Account Servicing Payment Service Provider | TPP: Third Party Providers, PISP: Payment Initiation Service Provider, AISP: Account Information Service Provider | OB Directory |
| **AU Finance** | Bank Customers | Banks | Fintechs | ACCC |