Appendix A

RFC

Internet Engineering Task Force (IETF)
Request for Comments: XXXX
Obsoletes: XXXX
Category: Standards Track
ISSN: XXXX-XXXX

Magendanz, Quinn
MIT Future of Data Initiative

February 2024

The OTrace Traceability Framework

A.0.0.1 Abstract

The OTrace Traceability Framework enables a User to track the sharing and usage of their personal data after it has been provided to, or collected by, an initial Data Provider that has explicitly received User consent. For the purpose of monitoring and auditing, the Data Provider and Data Recipient submit records to a Traceability Server to record initial User consent for data sharing as well as subsequent sharing and usage of the User's data. This specification introduces new standards for recording data sharing and usage as Traceability Records into a consent framework which builds off elements of the OAuth 2.0, PAR, PKCE, JWT, JWS, and TB protocols from RFCs 6749 [39], 9126 [41], 7636 [44], 7519 [45], 7515 [46], and 8471 [47] respectively, as well as the FAPI [42] and FDX [1] standards for financial data sharing.

A.0.0.2 Status of This Memo

This is a Massachusetts Institute of Technology (MIT) Computer Science and Artificial Intelligence Laboratory (CSAIL) document and is a product of the Future of Data Initiative (FDI).

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at `https://datatracker.ietf.org/drafts/current/`.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February XX, XXXX.

A.0.0.3 Copyright Notice

POSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.0.0.4 Table of Contents

A.1 Introduction

This document defines the OTrace Traceability Server and the accepted Traceability Records within an authorization framework that extends upon OAuth 2.0 and other existing frameworks. OTrace enables a User to monitor records of initial consent to data sharing alongside subsequent sharing and usage in a manner which

emphasizes the properties of informed consent, data traceability, and accountable use [15].

A.1.1 Consent

At the time of this document's publishing, organizations provide inadequate modes for a User to view all the data sharing and data usage that they have consented to via both implicit and explicit User Agreements. Both users and governing bodies require a more detailed breakdown of consents granted. Any solution to informed consent must share the following characteristics to serve as a viable solution:

- *Consistency across service providers.* Issuing consent preferences must be standardized so that data sharing across organizations can carry with it the consent metadata.
- *Granularity in consent choices.* A user should be able to express specific ways data can be used while still disallowing other types of use. This must be granular enough to capture the different types of use while not so fragmented as to confuse users.
- *Equal access to services regardless of consent choices.* Use of "dark patterns" to coerce, wheedle, and manipulate users to grant consent shall not be permitted.
- *Flexible update plan.* As new laws are ratified, the protocol may need to incorporate additional consent metadata.
- *Secure, scalable communication protocols.*

A.1.2 Traceability

Organizations need a medium to prove their compliance to both Users and governing bodies, especially as data is shared to third parties. Any solution to traceability must provide a method of both detecting misuse of data and of handling changes in consent preferences. Traceability monitoring must also be scalable for large amounts

of both automated and manual data processing and for sharing across many different, untrusted organizations. Existing cryptographic systems cannot be relied on to prove compliance as they can neither sufficiently scale nor be properly managed across different environments. Instead, the solution will likely depend on accumulated attestations of data use and sharing from multiple different organizations to verify compliance. To monitor these traceability records, a simple, unified platform should be available which is capable of processing incoming information from many organizations and displaying summaries and reports to users.

By defining a framework for reporting and viewing consent, OTrace provides a consistent, secure, scalable solution for specifying granular consent in a unified standard where updates can be applied as they arise. The mode of authorization extends the OAuth 2.0 [39] and OpenID Connect (OIDC) [43] frameworks in a manner similar to FAPI [42], aiming to provide enhanced security features tailored to the needs of high-stakes exchange of personal data.

OTrace relies on doubly attested Traceability Records to document and verify the history of data sharing as both parties publish records to a Traceability Server for each step of a data sharing transaction. However, OTrace provide cannot doubly attested verification of data use as the data user may be the only entity aware of data use. Instead, OTrace relyies on market pressures and enforcement of federal/state regulation to pressure organizations to accurately report data usage after sharing has occured.

A.1.3 Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in [RFC2119].

This specification uses the Augmented Backus-Naur Form (ABNF) notation of [RFC5234]. Additionally, the rule URI-reference is included from "Uniform Resource Identifier (URI): Generic Syntax" [RFC3986].

Certain security-related terms are to be understood in the sense defined in

[RFC4949]. These terms include, but are not limited to, "attack", "authentication", "authorization", "certificate", "confidentiality", "credential", "encryption", "identity", "sign", "signature", "trust", "validate", and "verify".

Unless otherwise noted, all the protocol parameter names and values are case sensitive.

A.2 Definitions

A.2.1 Roles

- User

  The end user whose personal information is held within the data. An example of the definition of personal information can be found in California Civil Code §1798.140(o)(1) - CCPA [12]. This can include, but it not limited to Personally Identifiable Information, bank account numbers, location, activity history, etc.

- Authentication Server (AS)

  The service that performs User authentication and receives initial, explicit consent to collect and store User data.

- Resource Server (RS)

  The server which contains the data/resources requested by the Data Recipient on behalf of the user. The Resource Server must confirm that the Data Recipient has sufficient consent to retrieve the requested data.

- Data Provider (DP)

  The entity that owns both the Authentication Server and Resource Server. Where differentiating between the AS and RS roles is not necessary, this role may be used to refer generally to both the AS and RS servers or a server that implements both rolls.

Users usually have an existing relationship, account, and direct interactions with Data Providers. However, if a Data Recipient is given consent to further share User data, it assumes the role of Data Provider with the secondary sharing, potentially without direct User interaction.

- Data Recipient (DR)

  The third party which seeks to act on behalf of the User and/or access their data.

- Traceability Server (TS)

  The server which receives and stores records of data sharing and usage for the purpose of User monitoring and legal auditing.

A.2.2 Record Definitions

- Traceability Key-pair

  A Traceability Key-pair is a key-pair used to sign all JWS's sent to the Traceability Server. Each of the Data Provider and Data Recipient MUST have a Traceability Key-pair that is unique to the actor. The public key of the key-pair will be used to identify each actor as they interact with the Traceability Server.

- Traceability Record

  Traceability Records are received by the Traceability Server from both the Data Provider and Data Recipient and logged for verification by the User.

  Each Traceability Record MUST be represented as a JSON Web Token (JWT) [45]. Parameter names and string values MUST be included as JSON strings. Since Traceability Records are handled across domains and potentially outside of a closed ecosystem, per Section 8.1 of RFC 8259 [54], these JSON strings MUST be encoded using UTF-8 RFC 3629 [55]. Numerical values MUST be included as JSON numbers. Traceability Records MAY include any extension parameters. This JSON object of parameters constitutes the JWT Claims Set defined in

JWT RFC 7519 [45]. The JWT Claims Set is then signed within a JSON Web Signature (JWS) RFC 7515 [46] by the Data Provider or Data Recipient Traceability Key-pair. The result is a JWS-signed JWT of the following format:

```
base64url-encoded(UTF8(JWS Protected Header)) || '.' ||
base64url-encoded(JWS Payload) || '.' ||
base64url-encoded(JWS Signature)
```

The Traceability Server MUST verify each JWS and check that the public key used for signing matches one of the Proof Key of Code Exchange (PKCE) challenges [44] provided in the Traceability Record Set's most recent attested Traceability Policy Record.

Each Traceability Record contains the trace_id field for identifying the Traceability Record Set and the time field to prevent replay of Traceability Records. The Traceability Server MUST not include duplicate copies of records generated at the same time.

The following are the types of Traceability Records:

  - Traceability Policy Record
  - Traceability Share Record
  - Traceability Usage Record
  - Traceability Migration Record

- Traceability Record Set

  A Traceability Record Set is a collection of Traceability Records which all derive their consent from the same User consent for the sharing and/or use of data. They are initiated by a Traceability Policy Record from the Data Provider which documents the details of the User consent and is attested by a matching Traceability Policy Record from the Data Recipient. Further records will be

considered to be part of this Traceability Record Set if they contain the same trace_id (which is returned by the Traceability Server) and are signed by either the Data Provider or Data Recipient.

If a Data Recipient receives consent to further share User data, they MUST create and record records to a new Traceability Record Set. This new Traceability Record Set MUST use the same Traceability Server as the original and MUST use the parent_ids parameter of the new Traceability Policy Record to refer to the original Traceability Record Set.

A.3 Token Binding Format

Within OTrace, Token Bindings will be applied to Access Tokens, Authorization Codes, Refresh Tokens, JWT Authorization Grants, and JWT Data Recipient Authentication [48]. This cryptographically binds these tokens to a Data Recipient's Token Binding key pair, possession of which is proven on the TLS connections over which the tokens are intended to be used. This use of Token Binding protects these tokens from man-in-the-middle and token export and replay attacks. The Token Binding message format is defined using the TLS presentation language of RFC 8446 [56]:

```
enum {
   rsa2048_pkcs1.5(0), rsa2048_pss(1), ecdsap256(2), (255)
} TokenBindingKeyParameters;

struct {
   opaque modulus<1..2^16-1>;
   opaque publicexponent<1..2^8-1>;
} RSAPublicKey;

struct {
```

```
   opaque point <1..2^8-1>;
} TB_ECPoint;

struct {
   TokenBindingKeyParameters key_parameters;
   uint16 key_length;  /* Length (in bytes) of the following
                          TokenBindingID.TokenBindingPublicKey */
   select (key_parameters) {
       case rsa2048_pkcs1.5:
       case rsa2048_pss:
           RSAPublicKey rsapubkey;
       case ecdsap256:
           TB_ECPoint point;
   } TokenBindingPublicKey;
} TokenBindingID;

enum {
   (255)        /* No initial TB_ExtensionType registrations */
} TB_ExtensionType;

struct {
   TB_ExtensionType extension_type;
   opaque extension_data<0..2^16-1>;
} TB_Extension;

enum {
   provided_token_binding(0), referred_token_binding(1), (255)
} TokenBindingType;

struct {
```

```
   TokenBindingType tokenbinding_type;
   TokenBindingID tokenbindingid;
   opaque signature<64..2^16-1>; /* Signature over the concatenation
                                    of tokenbinding_type,
                                    key_parameters, and EKM */
   TB_Extension extensions<0..2^16-1>;
} TokenBinding;

struct {
   TokenBinding tokenbindings<132..2^16-1>;
} TokenBindingMessage;
```

A.4 Protocol Flow

A.4.1 Consent for Data Sharing

A.4.1.1 User Initiated

For the User to initiate data sharing with a Data Provider on behalf of a Data Recipient, the User MUST have independent relationships with both the Data Recipient and the Data Provider. The User MUST initiate the consent request from within the Data Recipient's experience/application. Before beginning the consent request, the Data Recipient determines the types of data access it intends to access from Data Provider and MUST disclose its intent to the User. During User Authentication to the Data Provider, the User MUST actively authorize the Data Provider to enable the Data Recipient's access to End User's data [1].

1. (1) PAR

   The Pushed Authorization Request (PAR) is an OAuth 2.0 JWT-Secured Authorization Request (JAR) Request Object as defined in RFC 9101 [57]. A Request Object is used to provide authorization request parameters for an OAuth

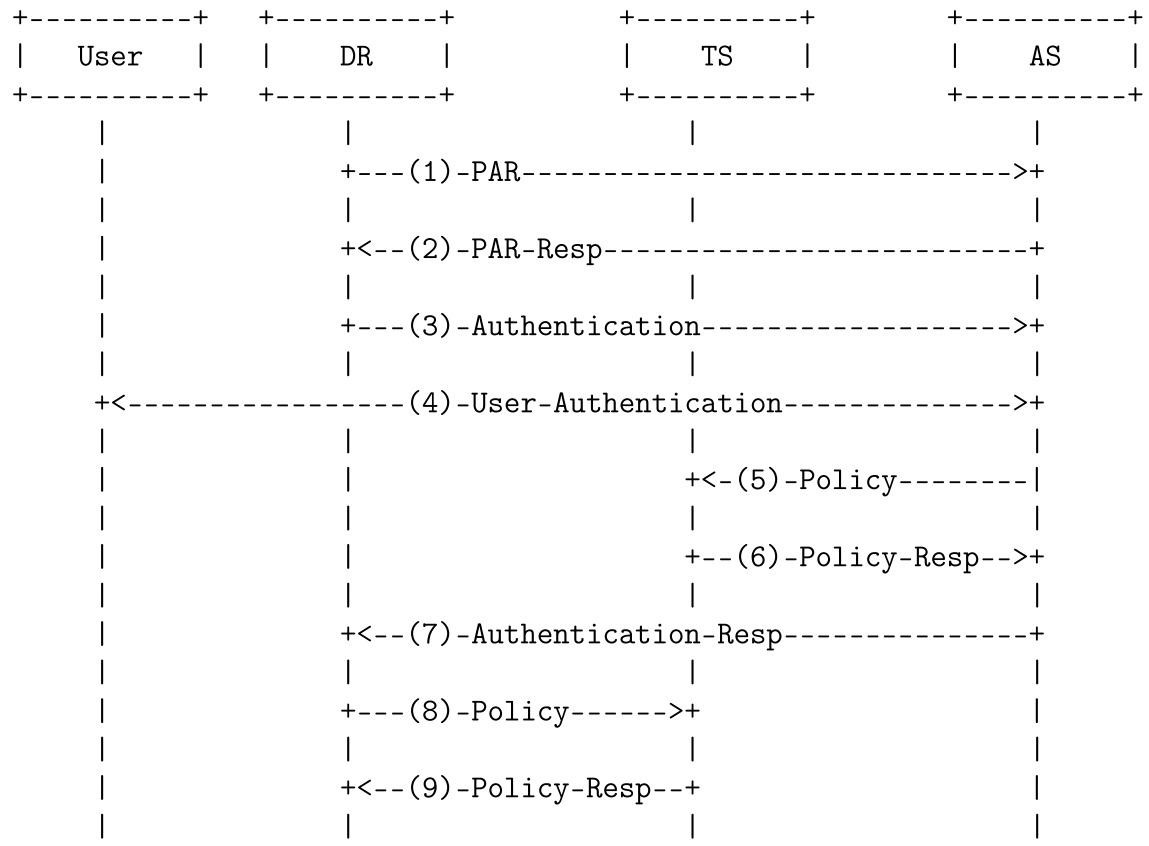

Figure A-1: User Initiated Consent

2.0 authorization request. The parameters are represented as the JWT Claims of the object. Parameter names and string values MUST be included as JSON strings. Since Request Objects are handled across domains and potentially outside of a closed ecosystem, per Section 8.1 of RFC 8259 [54], these JSON strings MUST be encoded using UTF-8 RFC 3629 [55]. Numerical values MUST be included as JSON numbers. The Request Object MAY include any extension parameters.

This JSON object constitutes the JWT Claims Set defined in JWT [45]. The JWT Claims Set is then signed within a JSON Web Signature (JWS) [46]. The result is a JWS-signed JWT.

The following parameters are included as top-level members in the JSON message of the Request Object with any additional parameters necessary for Data Recipient authentication:

- client_id

  REQUIRED. The Data Recipient identifier issued to the Data Recipient during the registration process described by [39].

- response_type

  REQUIRED. Value MUST be set to "code" per [39].

- state

  REQUIRED. An opaque value used by the Data Recipient to maintain state between the request and callback. The Authorization Server includes this value when redirecting the User back to the Data Recipient. The parameter MUST be used for preventing cross-site request forgery as described in [39]

- code_challenge

  REQUIRED. The PKCE code challenge is derived from [44] and [48]. The value is the base64url encoding (per Section 5 of [RFC4648] with all trailing padding ('=') characters omitted and without the inclusion of any line

breaks or whitespace) of the SHA-256 hash of the Provided Token Binding ID that the Data Recipient will use when calling the authorization server's token endpoint. The Provided Token Binding ID MUST be the same that will be used for Token Binding during the Authentication and Token Request steps. See section A.3 for the Provided Token Binding ID format.

- code_challenge_method

  REQUIRED. Value MUST be set to "TB-S256" per [48]

- authorization_details

  REQUIRED. Value MUST contain a JSON-formatted object with two members in compliance with the RAR format specified by [40]. These two members are "type" with the value "fdx_v1.0" and "consentRequest" containing a valid JSON ConsentRequest entity as defined in [1]. Below is an example of authorization_details:

```
"authorization_details": {
    "type": "fdx_v1.0",
    "consentRequest": {
        "durationType": "ONE_TIME",
        "lookbackPeriod": 60,
        "resources": [
            {
                "resourceType": "ACCOUNT",
                "consents": [
                    {
                        "category": "user.contact",
                        "uses": "marketing.communications",
                    },
                    {
                        "category": "user.demographic",
                        "uses": "marketing.advertising",
```

```
                    },
                    {
                        "category": "user.location.impercise",
                        "uses": "personalize.content",
                    }
                ]
            }
        ]
    }
}
```

The Request Object may contain other parameters as recommended in [41] and [39] such as redirect_uri, client_secret, iss, aud, scope, etc.

2. PAR Response

   If the verification is successful, the server MUST generate a request URI and provide it in the response with a 201 HTTP status code. The following parameters are included as top-level members in the message body of the HTTP response using the "application/json" media type as defined by RFC 8259 [54]:

   (a) request_uri

   REQUIRED. The request URI corresponding to the authorization request posted. This URI is a single-use reference to the respective request data in the subsequent authorization request. The way the authorization process obtains the authorization request data is at the discretion of the authorization server and is out of scope of this specification. There is no need to make the authorization request data available to other parties via this URI.

   (b) expires_in

RECOMMENDED. A JSON number that represents the lifetime of the request URI in seconds as a positive integer. The request URI lifetime is at the discretion of the authorization server but will typically be relatively short (e.g., between 5 and 600 seconds).

3. Authorization Request

The Data Recipient requests an authorization code after successful User authentication by adding the following parameters to the query component of the authorization endpoint URI using the "application/x-www-form-urlencoded" format, per [39]:

(a) client_id

REQUIRED. The Data Recipient identifier issued to the Data Recipient during the registration process described by [39].

(b) request_uri

REQUIRED. The request_uri returned from the PAR response.

(c) code_challenge_TS

REQUIRED. The PKCE code challenge is derived from [44] and [48]. The value is the base64url encoding (per Section 5 of [RFC4648] with all trailing padding ('=') characters omitted and without the inclusion of any line breaks or whitespace) of the SHA-256 hash of the public key of the Data Recipient's Traceability Key-pair.

(d) code_challenge_TS_method

REQUIRED. Value MUST be set to "TB-S256" per [48]

The Authorization MUST also contain the "Sec-Token-Binding" header for the purpose of performing OAuth Token Binding [48]. See section A.3 for details.

The provided token binding public key MUST be the same public key used in the PKCE challenge of the PAR Request Object. The token binding will use the Exported Keying Material (EKM) of the current TLS connection through

which the Authentication Request is made per [48]. The value of "Sec-Token-Binding" MUST be the base64url-encoded concatenation of the provided TokenBindingMessage preceded by two bytes denoting the length of the bytes that follow.

```
base64url-encoded(uint16 following_bytes ||
    provided_TokenBindingMessage)
```

4. User Authentication

   Data Provider performs User authentication. User consents to specific types and granularity of data sharing and authorized actions to the Data Recipient.

   While the User is providing consent, Data Provider MUST provide an option to the User to either supply a custom Traceability Server URL or agree to utilize a User-accessible, default Traceability Server. The Data Provider SHOULD provide a mode for Users to upload the public key of a custom traceability server.

5. Traceability Policy Record

   The Data Provider submits a Traceability Policy Record of the User's consent to the Traceability Server. This record contains a specific description of the type of data that the consent concerns as well as granular permissions limiting the types of actions that the Data Recipient can take concerning the shared data. Per section A.2.2, all Traceability Records are signed JWTs. Traceability Policy Records contain the following parameters:

   - trace_id

     REQUIRED. This value is required to be "0" when the Data Provider initiates a new consent trace. The Traceability Server will reply with the assigned trace_id for use in all subsequent records submitted to the Traceability Server.

- time

  REQUIRED. Time at which the Traceability Record was generated by the sender. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.

- data_subject

  REQUIRED. The WebID HTTP URI representing the User who owns the data that this policy concerns [58]. This WebID Profile MUST be unique to the User relative to the Data Provider and SHOULD NOT provide any compromising information to those who view the WebID Profile. If the User does not host and provide their own unique WebID, it is expected for the Data Provider to host such a WebID that provides at minimum a unique identifier for the User. This follows the same standards as the W3C Solid Community Project [59].

- description

  REQUIRED. A human-readable summary for the User describing the relevant data categories and the purpose of consent, as phrased to the User during acknowledgment.

- consents

  REQUIRED. A JSON structure which describes the data categories that the User consent covers and what data uses are permitted upon each data category. The consents available SHOULD offer granularity in consent choices.

  The structure of this field is intentionally left open so that Data Providers may define the structure with proper granularity and specificity to match their use cases and to aid the User's understanding of how data will be used.

  In order to ensure consistency across service providers, it is recommended that Data Providers use an widely recognized structure such as the Fides Language Taxonomy Classification Groups [3]. An example would be a

JSON array of structures with the following fields:

  - data_categories

    Data Categories are labels to describe the type of data processed by your software. See section [3] for further details.

  - data_uses

    Data Uses are labels that describe how, or for what purpose(s) a component of your system is using data. See section [3] for further details.

- parent_ids

  OPTIONAL. A JSON array of trace_ids of other Traceability Policy Records that this Traceability Policy Record derives consent from.

- provider_challenge

  REQUIRED. This PKCE code challenge is derived from [44] and [48]. The value is the base64url encoding (per Section 5 of [RFC4648] with all trailing padding ('=') characters omitted and without the inclusion of any line breaks or whitespace) of the SHA-256 hash of the public key from the Data Provider's Traceability Key-pair.

- provider_challenge_method

  REQUIRED. Value MUST be set to "TB-S256" per [48]

- recipient_challenge

  REQUIRED. This PKCE code challenge is derived from [44] and [48]. The value is the base64url encoding (per Section 5 of [RFC4648] with all trailing padding ('=') characters omitted and without the inclusion of any line breaks or whitespace) of the SHA-256 hash of the public key from the Data Recipient's Traceability Key-pair.

- recipient_challenge_method

  REQUIRED. Value MUST be set to "TB-S256" per [48]

- trace_uri

  REQUIRED. A URI identifying the Traceability Server for this consent.

- trace_cert (optional)

  RECOMMENDED. The Data Provider may submit a certificate by which the Traceability Server can identify itself. This certificate should be the base64url-encoding of the certificate format described in either section 4.4.2. of RFC 8446 [56] or RFC 8879 [60]

6. Traceability Policy Record Response

   If the Traceability Policy Record is successfully received and processed, the server MUST generate a response with a 201 HTTP status code. The following parameters are included as top-level members in the message body of the HTTP response using the "application/json" media type as defined by RFC 8259 [54]:

   - trace_id

     REQUIRED. The trace_id generated by the Traceability Server to identify the Traceability Policy Record and all subsequent records under the policy. This MUST be unique to this set of Traceability Record Set.

7. Authentication Response

   If the User grants the access request and the Authentication message Token Binding is verified, the Authorization Server issues an authorization code and delivers it to the Data Recipient by adding the following parameters to the query component of the redirection URI using the "application/x-www-form-urlencoded" format, per RFC 6749 [39].

   - code

     REQUIRED. Per RFC 6749 [39], the authorization code generated by the Authorization Server. The authorization code MUST expire shortly after it is issued to mitigate the risk of leaks. A maximum authorization code lifetime of 10 minutes is RECOMMENDED. The Data Recipient MUST NOT use the authorization code more than once. If an authorization code is used more than once, the authorization server MUST deny the request

and SHOULD revoke (when possible) all tokens previously issued based on that authorization code. The authorization code is bound to the Data Recipient identifier and redirection URI.

- state

  REQUIRED if the "state" parameter was present in the Data Recipient Authorization Request per RFC 6749 [39]. The exact value received from the Data Recipient.

- trace_policy

  The same Traceability Policy Record sent by the Data Provider to the Traceability Server, re-signed after the returned trace_id value has been applied.

- id_token

  As defined in OpenID Connect [43], the ID Token data structure enables Users to be authenticated. The ID Token is a security token that contains Claims about the Authentication of an User by an Authorization Server when using a Data Recipient, and potentially other requested Claims. The ID Token is represented as a JWT [45]. The JWT MUST be signed as a JWS [46].

  The following Claims are used within the ID Token within the Authentication Response of OTrace. ID Tokens MAY contain other Claims. Any Claims used that are not understood MUST be ignored.

  - iss

    REQUIRED. Issuer Identifier for the Issuer of the response. See OpenID Connect for further details [43]

  - sub

    REQUIRED. Subject Identifier. See OpenID Connect for further details [43]

  - aud

REQUIRED. Audience(s) that this ID Token is intended for. See OpenID Connect for further details [43]

- exp

  REQUIRED. Expiration time on or after which the ID Token MUST NOT be accepted for processing. See OpenID Connect for further details [43]

- iat

  REQUIRED. Time at which the JWT was issued. See OpenID Connect for further details [43]

- code_hash

  REQUIRED. The SHA256 hash of the code parameter.

- state_hash

  REQUIRED. The SHA256 hash of the state parameter.

- trace_policy_hash

  REQUIRED. The SHA256 hash of the trace_policy parameter.

8. Traceability Policy Record

   The Data Recipient submits a record of the User's consent to the Traceability Server. This message follows the same format as the Traceability Policy Record step from the Data Provider above and MUST contain the same parameter values as provided by the Data Provider in the Authorization Response. See section 5 for required fields.

9. Traceability Policy Record Response

   If the Traceability Policy Record is successfully received and processed, the server MUST generate a response with a 201 HTTP status code.

A.4.1.2 Data Provider Initiated

The Data Provider may initiate Consent for Data Sharing at any point within their relationship with the User while they are already authenticated. This often occurs

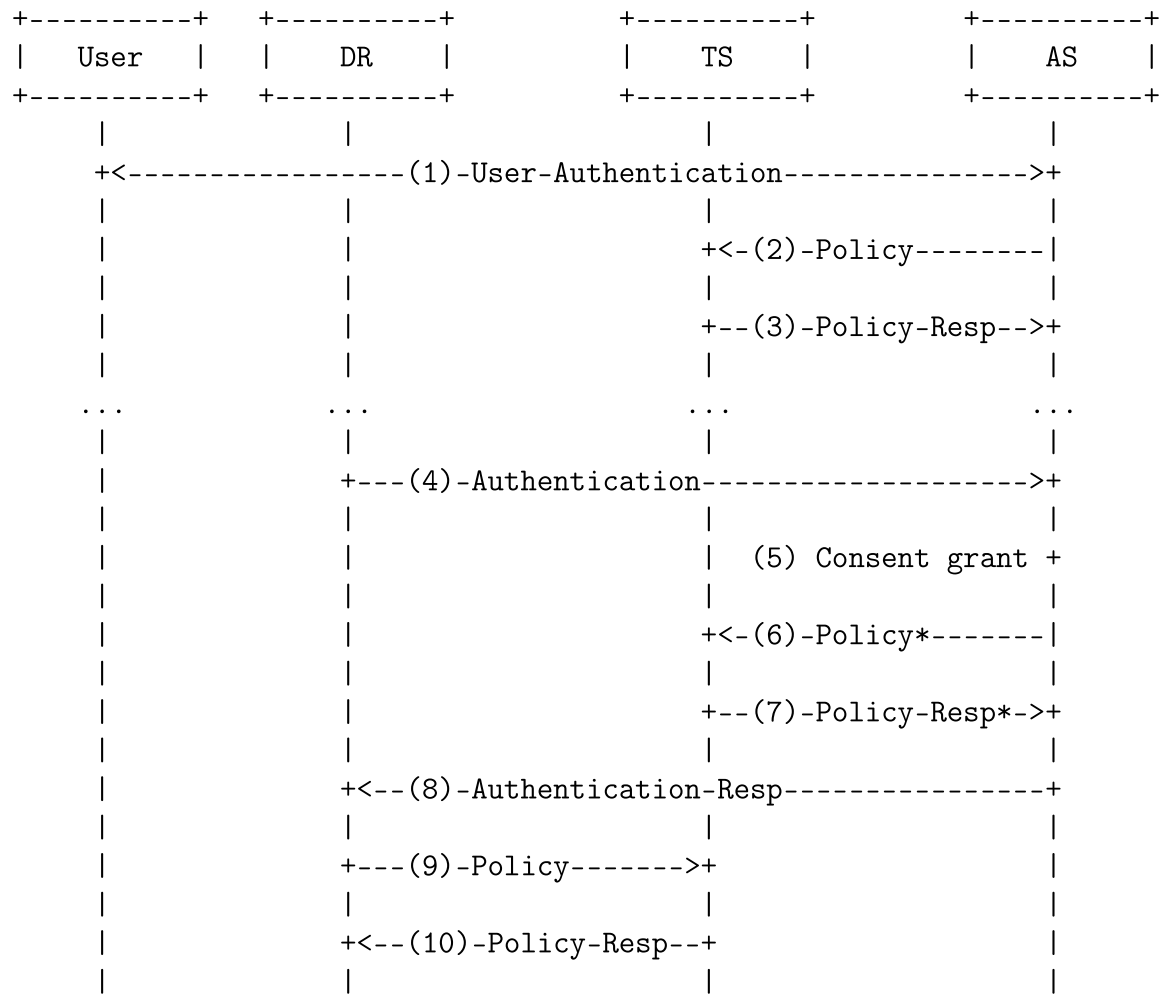

Figure A-2: Data Provider Initiated Consent

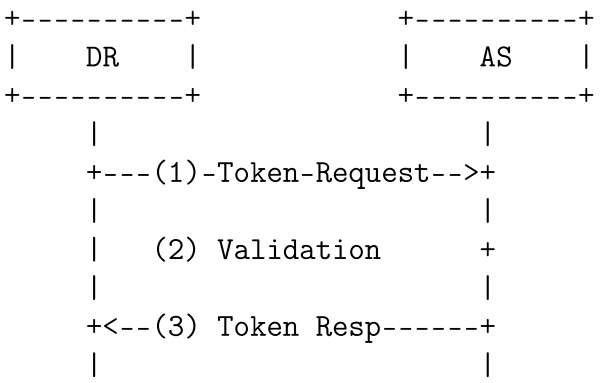

Figure A-3: Token Acquisition

when Data Providers regularly utilize the services of a third-party that requires User data to perform these services. Users often agree to this type of bulk, categorical data sharing in initial User agreements.

Consent for data sharing in these scenarios occurs with the same steps as when Data Recipients initiate data sharing but in a slightly different order. See Figure X for details.

However, only when the Traceability Policy Record from step (2) specifies a broad sharing consent do steps (6) and (7) occur. This Traceability Policy Record will reference the trace_id returned from step (3) in the list of parent_ids and must contain the same provider_challenge. Additionally in this case, steps (4) through (10) must occur every time a new Data Recipient requests access to bulk consented data from step (2).

A.4.1.3 Token Acquisition

1. Token Request

   The Data Recipient makes a request to the token endpoint by sending the following parameters using the "application/x-www-form-urlencoded" format per Appendix B with a character encoding of UTF-8 in the HTTP request entity-body per RFC 6749 [39]:

   - grant_type

REQUIRED. Value MUST be set to "authorization_code".

- code

  REQUIRED. The authorization code received from the authorization server.

- redirect_uri

  REQUIRED, if the "redirect_uri" parameter was included in the authorization request, and their values MUST be identical.

- client_id

  REQUIRED, if the client is not authenticating with the authorization server.

The Token Request MUST also contain the "Sec-Token-Binding" header for the purpose of performing OAuth Token Binding [48]. This header will contain both a provided and referred token according to the structure laid out in [47]. The provided token binding public key MUST be the same public key used in the PKCE challenge of the Request Object. The referred token binding public key will be used to sign a provided token binding when later communicating to the Resource Server. Both token bindings will use the EKM of the current TLS connection through which the Authentication Request is made per [48]. The value of "Sec-Token-Binding" MUST be the base64url-encoded concatenation of the provided and the referred TokenBindingMessage preceded by two bytes denoting the length of the bytes that follow.

```
base64url-encoded(uint16 following_bytes ||
    provided_TokenBindingMessage || referred_TokenBindingMessage)
```

2. Validation

   Follow validation steps required by RFC 6749 [39] authentication code grant workflow, specifically, ensuring that the authorization code was issued to the client, the authorization code is valid, and the "redirect_uri" parameter is

present if it was included in the initial authorization request. Additionally, ensure that the token binding is valid and matches the original PKCE per [48].

3. Token Response

   Per RFC 6749 [39], if the access token request is valid and authorized, the authorization server issues an access token and optional refresh token. The following parameters are included as top-level members in the message body of the HTTP response using the "application/json" media type as defined by [RFC8259] with a 200 (OK) status code:

   - access_token

     REQUIRED. The access token issued by the authorization server as described in RFC 6749 [39].

   - token_type

     REQUIRED. The type of the token issued as described in RFC 6749 [39].

   - grant_id

     REQUIRED. Contains the ConsentID per [1].

   - id_token

     As defined in OpenID Connect [43], the ID Token data structure enables Users to be authenticated. The ID Token is a security token that contains Claims about the Authentication of an User by an Authorization Server when using a Data Recipient, and potentially other requested Claims. The ID Token is represented as a JWT [45]. The JWT MUST be signed as a JWS [46].

     The following Claims are used within the ID Token within the Authentication Response of OTrace. ID Tokens MAY contain other Claims. Any Claims used that are not understood MUST be ignored.

     – iss

       REQUIRED. Issuer Identifier for the Issuer of the response. See OpenID Connect for further details [43]

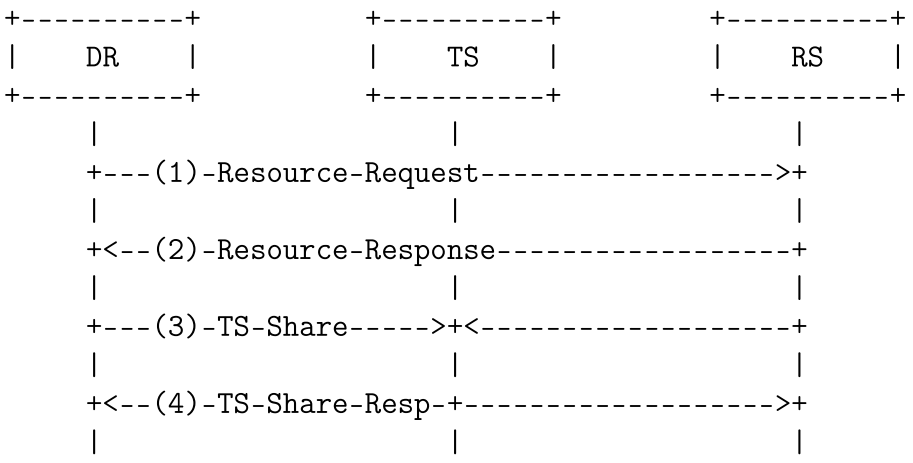

Figure A-4: Data Sharing

- sub

  REQUIRED. Subject Identifier. See OpenID Connect for further details [43]

- aud

  REQUIRED. Audience(s) that this ID Token is intended for. See OpenID Connect for further details [43]

- exp

  REQUIRED. Expiration time on or after which the ID Token MUST NOT be accepted for processing. See OpenID Connect for further details [43]

- iat

  REQUIRED. Time at which the JWT was issued. See OpenID Connect for further details [43]

- access_token_hash

  REQUIRED. The SHA256 hash of the access_token parameter per [2].

A.4.2 Data Sharing

1. Resource Request

The client accesses protected resources by presenting the access token to the resource server. The resource server MUST validate the access token and ensure that it has not expired and that its scope covers the requested resource. The methods used by the resource server to validate the access token (as well as any error responses) are beyond the scope of this specification but generally involve an interaction or coordination between the resource server and the authorization server.

The method in which the client utilizes the access token to authenticate with the resource server depends on the type of access token issued by the authorization server. Typically, it involves using the HTTP "Authorization" request header field [RFC2617] with an authentication scheme defined by the specification of the access token type used, such as [RFC6750]. See RFC 6749 [39] for further details on access token types.

In addition to access token, the Resource Request should include the following parameters as top-level members in the message body of the HTTP response using the "application/json" media type as defined by [RFC8259]:

- id_token

  The id_token parameter received during the Token Request step.

The Token Request MUST also contain the "Sec-Token-Binding" header for the purpose of performing OAuth Token Binding [48]. This header will contain a provided token according to the structure laid out in [47]. The provided token binding public key MUST match the referred token binding previously issued to the Authentication Server. The token binding will use the EKM of the current TLS connection through which the Resource Request is made per [48]. The value of "Sec-Token-Binding" MUST be the base64url-encoded concatenation of the provided TokenBindingMessage preceded by two bytes denoting the length of the bytes that follow.

2. Resource Response

The Data Provider provides the requested resource to the Data Recipient.

3. Traceability Share Record

   The Data Provider and Data Recipient submit a Traceability Share Record of the sharing of User data with the Data Recipient within the bounds of the consent detailed in the Traceability Policy Record. Per section A.2.2, all Traceability Records are signed JWTs. Traceability Share Records contain the following parameters:

   - trace_id

     REQUIRED. The trace_id for this Traceability Record Set.

   - time

     REQUIRED. Time at which the Traceability Record was generated by the sender. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.

   - data_shared

     REQUIRED. A JSON structure which describes the data categories that the shared data falls within and what data uses are permitted upon each data category. This should follow the format used for the consents parameter of the Traceability Policy Record.

   - description

     REQUIRED. A human-readable summary for the User describing the relevant data being shared and the purpose of share.

4. Traceability Share Record Response

   If the Traceability Share Record is successfully received and processed, the server MUST generate a response with a 201 HTTP status code.

A.4.3 Data Usage

1. Data Usage

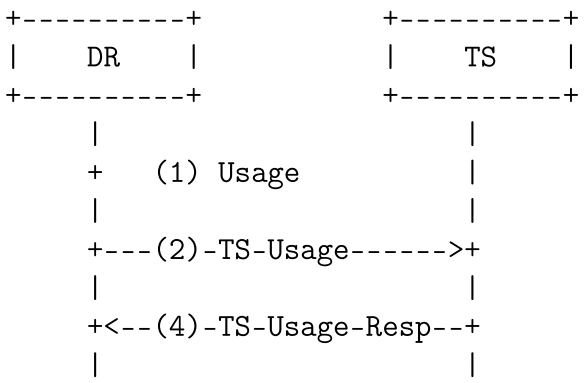

Figure A-5: Data Usage

The Data Recipient uses the shared User data within the bounds of the consent detailed in the Traceability Policy Record.

2. Traceability Usage Record

   The Data Recipient submits a Traceability Usage Record when the shared User data is used. The use MUST be within the bounds of the consent detailed in the Traceability Policy Record. Per section A.2.2, all Traceability Records are signed JWTs. Traceability Usage Records contain the following parameters:

   - trace_id

     REQUIRED. The trace_id for this Traceability Record Set.

   - time

     REQUIRED. Time at which the Traceability Record was generated by the sender. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.

   - data_used

     REQUIRED. A JSON structure which describes the data categories that the used data falls within and what data uses are taking place upon each data category. This should follow the format used for the consents parameter of the Traceability Policy Record.

   - description

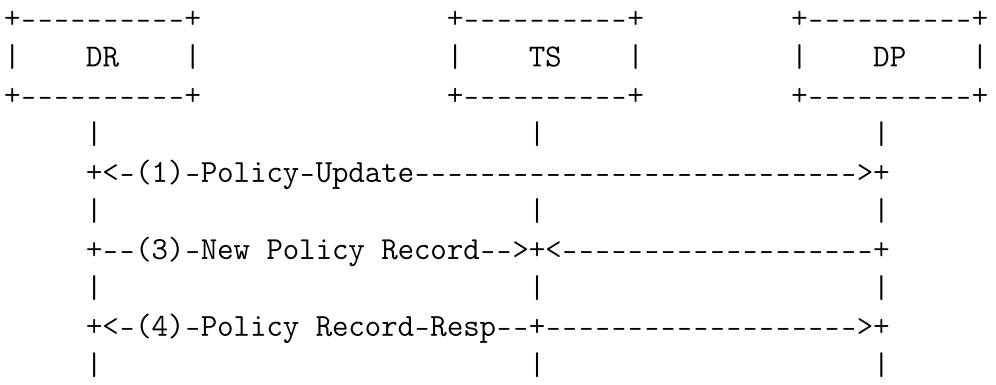

Figure A-6: Policy Update

REQUIRED. A human-readable summary for the User describing the relevant data being used and the purpose for use.

3. Traceability Usage Record Response

   If the Traceability Usage Record is successfully received and processed, the server MUST generate a response with a 201 HTTP status code.

A.4.4 Policy Update

1. Policy Update

   The Data Provider and Data Recipient exchange a new Traceability Policy Record if there is a change in consent or another field of the Traceability Policy Set's most recent Traceability Policy Record. This may result from changes in, for example, User data sharing preferences, Data Provider policies, or Data Recipient data use.

   Any new permissions beyond the original Traceability Policy Record MUST receive User consent as in the initial Consent process.

   A removal of all permissions is equivalent to a request for data deletion.

   The new Traceability Policy Record MUST NOT change the User, Data Provider, Data Recipient, and Traceability Server. As such, the new Traceability Policy Record SHOULD NOT contain changes to any field other than

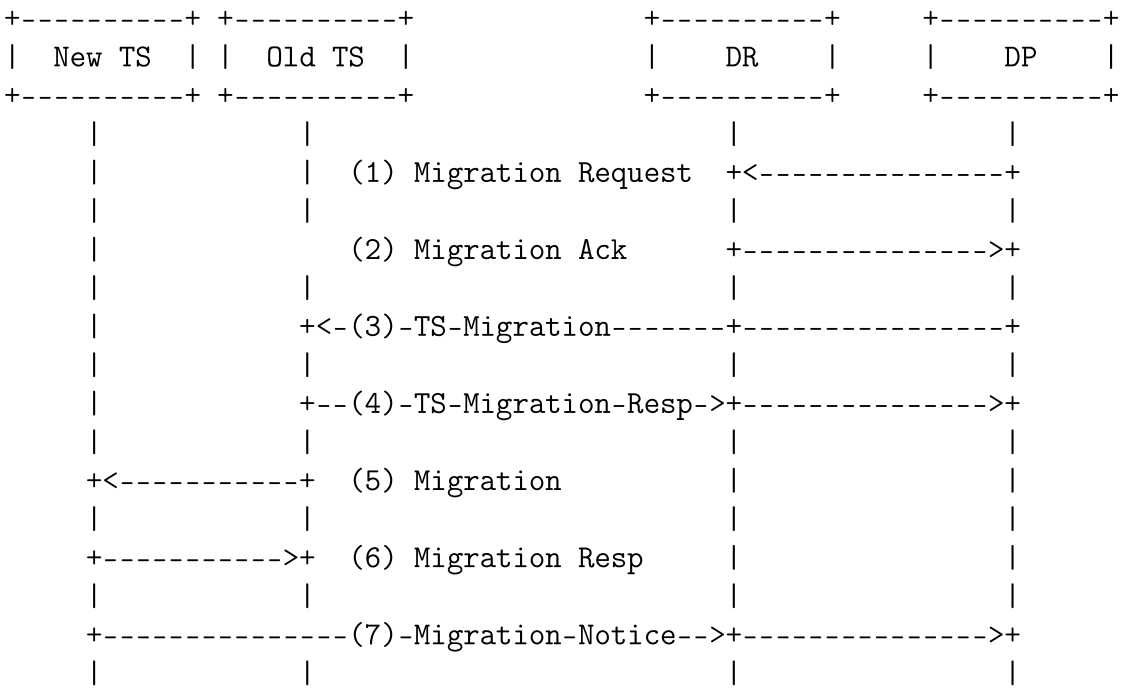

Figure A-7: Data Migration

description, consents, and parent_ids unless required to update keys or URIs.

Per section A.2.2, all Traceability Records are signed JWTs. See section 5 for required fields.

If the new Traceability Policy Record is successfully received, it MUST be acknowledged with a 200 HTTP status code.

2. Traceability Policy Record

   Both the Data Provider and Recipient submit their individually signed new Traceability Policy Records to the Traceability Server.

3. Traceability Policy Record Response

   If the new Traceability Policy Record is successfully received and processed, the server MUST generate a response with a 201 HTTP status code.

A.4.5 Migrate Traceability Server

1. Migration Request

The Data Provider submits a Traceability Migration Record to the Data Recipient if there is a change in the Traceability Server. This may result from changes in, for example, User requested change in Traceability Server or Data Provider change in default Traceability Server. Per section A.2.2, all Traceability Records are signed JWTs. Traceability Migration Records contain the following parameters:

- trace_id

  REQUIRED. The trace_id for this Traceability Record Set.

- time

  REQUIRED. Time at which the Traceability Record was generated by the sender. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.

- force

  OPTIONAL. "true" if the Data Provider and Data Recipient should start using the new Traceability Server immediately, submitting this Traceability Migration Record to the new Traceability Server.

- migrated_uri

  REQUIRED. A URI identifying the Data Provider's callback for when Traceability Server migration is complete.

- trace_uri

  REQUIRED. A URI identifying the new Traceability Server for this Traceability Record Set.

- trace_cert

  RECOMMENDED. The Data Provider may submit a certificate by which the Traceability Server can identify itself. This certificate should be the base64url-encoding of the certificate format described in either section 4.4.2. of RFC 8446 [56] or RFC 8879 [60].

2. Migration Acknowledgement

   If the Traceability Migration Record is successfully received and processed, the Data Recipient MUST generate a response with a 202 HTTP status code.

3. Traceability Migration Record

   Both the Data Provider and Recipient submit their individually signed Traceability Migration Records to the Traceability Server. The Data Recipient should replace the Data Provider migrated_uri with its own before sending its Traceability Migration Record.

4. Migration

   The migration of a Traceability Record Set from one Traceability Server to another uses the "application/json" media type as defined by RFC 8259 [54]. All Traceability Records within the Traceability Record Set MUST be wrapped within the following structure and sent as a JSON array:

   - type

     REQUIRED. Indicates the type of the Traceability Record. Value is either "policy", "share", "use", "change", or "migration".

   - time

     REQUIRED. Time at which the Traceability Record was received by the Traceability Server. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.

   - trace

     REQUIRED. The Traceability Record represented as a JWS signed JWT, as originally received by the Traceability Server.

5. Migration Response

   If the Traceability Migration Record is successfully received and processed, the new Traceability Server MUST generate a response with a 201 HTTP status

code. The new Traceability Server MUST verify the signatures from each Traceability Record in the migrated Traceability Record Set. The new Traceability Server MUST assign a new trace_id to the Traceability Record Set for future Traceability Record submission if the old trace_id is not unique within the new Traceability Server.

6. Migration Notice x2

   If the migration to the new Traceability Server is successfully received and processed, the new Traceability Server MUST send a PUT request to both the Data Provider and Data Recipient's migrate_uri conveyed within the last Traceability Migration Records. The following parameters are included as top-level members in the message body of the HTTP response using the "application/json" media type as defined by RFC 8259 [54]:

   - old_trace_id

     REQUIRED. The trace_id on the Traceability Migration Request.

   - new_trace_id

     REQUIRED. A new trace_id to uniquely represent the migrated Traceability Record Set on the new Traceability Server. This MAY be the same as old_trace_id if it is unique within the new Traceability Server.

     If trace_cert from the Traceability Migration Request was provided, these fields should instead be provided as a JWS that can be verified with trace_cert.

For a Traceability Server to commence migration, all Traceability Record Sets connected by the parent_ids field of the Traceability Policy Record must contain Traceability Migrate Records from the Data Provider and the Data Recipient (unless not present) detailing the same migration information [1]. While waiting on a response

[1]This prerequisite for migration is significant and potentially impractical and inconvenient for Users. This decision was made to reduce the complexity of implementation for v1.0 of this protocol. It is recommend that subsequent versions create a more flexible migration framework that builds on principles of distributed computing.

to indicating completion of migration which will contain a new trace_id, no additional Traceability Records should be submitted to the Traceability Server.

Bibliography

[1] Financial-Data-Exchange, "Financial data exchange api specification," May 2022. v5.1.

[2] D. Fett, P. Hosseyni, and R. Küsters, "An extensive formal security analysis of the openid financial-grade api." https://arxiv.org/pdf/1901.11520.pdf, Jan. 2019. arXivLabs.

[3] ethyca, "Fides language." https://ethyca.github.io/fideslang/.

[4] Apple, "ios and ipados software license agreement." https://www.apple.com/legal/sla/docs/iOS16_iPadOS16.pdf, 2022. v16.

[5] J. M. C. Bank, "Chase u.s. consumer privacy notice." https://www.chase.com/digital/resources/privacy-security/privacy/consumer-privacy-notice, apr 2022.

[6] B. Auxier, L. Raine, M. Anderson, A. Perrin, M. Kumar, and E. Turner, "Americans' attitudes and experiences with privacy policies and laws." https://www.pewresearch.org/internet/2019/11/15/americans-attitudes-and-experiences-with-privacy-policies-and-laws/, Nov. 2019. Pew Research Center.

[7] K. Liao, E. Radler, and D. Weitzner, *Building Accountable Systems Through Data Traceability (Extended Abstract)*. MIT Internet Policy Research Initiative.

[8] L. Brodsky and L. Oakes, "Data sharing and open banking," *McKinsey & Company*, vol. 1105, 2017.

[9] "Payment services directive (psd2)." https://finance.ec.europa.eu/regulation-and-supervision/financial-services-legislation/implementing-and-delegated-acts/payment-services-directive_en, 2015.

[10] "General data protection regulation (gdpr)." https://gdpr-info.eu, 2016.

[11] "Section 1033 of the dodd-frank wall street reform and consumer protection act." https://www.congress.gov/bill/111th-congress/house-bill/4173, 2010.

[12] "California consumer privacy act (ccpa)." https://oag.ca.gov/privacy/ccpa, 2018.

[13] "Notice of proposed rulemaking - required rulemaking on personal financial data rights." https://files.consumerfinance.gov/f/documents/cfpb-1033-nprm-fr-notice_2023-10.pdf, 2023.

[14] "Mit future of data." https://futureofdata.mit.edu/.

[15] "Accountability and traceability white paper and research roadmap." https://futureofdata.mit.edu/tr/2023/fod-account-trace-20230418.pdf, Apr. 2023. Future of Data Initiative, Massachusetts Institute of Technology.

[16] B. Glock, "Unlocking the opportunities of open banking." https://navigate.visa.com/na/money-movement/unlocking-the-opportunities-of-open-banking/, July 2022. Visa.

[17] "Visa open banking consumer survey," Apr. 2022. n=1,500, representative sample of U.S. population based on Census Bureau. Administered digitally.

[18] N. Lawford, "Literature review of user trust in online and open banking technologies (draft)." MIT Future of Data Initiative, 2023.

[19] B. Vatanasombut, M. Igbaria, A. C. Stylianou, and W. Rodgers, "Information systems continuance intention of web-based applications customers: The case of online banking," *Information & management*, vol. 45, no. 7, pp. 419–428, 2008.

[20] P. L. Yu, M. Balaji, and K. W. Khong, "Building trust in internet banking: a trustworthiness perspective," *Industrial Management & Data Systems*, vol. 115, no. 2, pp. 235–252, 2015.

[21] R. F. Hasandoust and M. M. Saravi, "Identifying the effect of successful e-banking on customers' satisfaction, trust, commitment and loyalty," *QUID: Investigación, Ciencia y Tecnología*, no. 1, pp. 1716–1726, 2017.

[22] A. Mukherjee and P. Nath, "A model of trust in online relationship banking," *International journal of bank marketing*, vol. 21, no. 1, pp. 5–15, 2003.

[23] R. Kumra, R. Mittal, and L. Gunupudi, "Trust and its determinants in internet banking: A study of private sector banks in india 1," in *Information Systems*, pp. 141–158, Routledge India, 2019.

[24] C. Cruijsen, J. de Haan, and R. Roerink, "Trust in financial institutions: A survey," tech. rep., Netherlands Central Bank, Research Department, 2020.

[25] P. Palos-Sanchez, J. R. Saura, and F. Martin-Velicia, "A study of the effects of programmatic advertising on users' concerns about privacy overtime," *Journal of Business Research*, vol. 96, pp. 61–72, 2019.

[26] A. Sanayei and A. Noroozi, "Security of internet banking services and its linkage with users' trust: A case study of parsian bank of iran and cimb bank of malaysia," in *2009 International Conference on Information Management and Engineering*, pp. 3–7, IEEE, 2009.

[27] R. Chan, I. Troshani, S. Rao Hill, and A. Hoffmann, "Towards an understanding of consumers' fintech adoption: The case of open banking," *International Journal of Bank Marketing*, vol. 40, no. 4, pp. 886–917, 2022.

[28] O. Armantier, S. Doerr, J. Frost, A. Fuster, and K. Shue, "Whom do consumers trust with their data? us survey evidence," tech. rep., Bank for International Settlements, 2021.

[29] I. Van Zeeland and J. Pierson, "In banks we trust: Banks as custodians of personal data in open banking ecosystems," in *TPRC49: The 49th Research Conference on Communication, Information and Internet Policy*, 2021.

[30] S. Rotchanakitumnuai and M. Speece, "Corporate customer perspectives on business value of thai internet banking services," *Journal of electronic commerce research*, vol. 5, no. 4, pp. 270–286, 2004.

[31] A. Ali, A. Hameed, M. F. Moin, and N. A. Khan, "Exploring factors affecting mobile-banking app adoption: a perspective from adaptive structuration theory," *Aslib Journal of Information Management*, vol. 75, no. 4, pp. 773–795, 2023.

[32] E. Masoud and H. AbuTaqa, "Factors affecting customers' adoption of e-banking services in jordan," *Information Resources Management Journal (IRMJ)*, vol. 30, no. 2, pp. 44–60, 2017.

[33] R. Al-Dmour, M. Alnafouri, and A. Al-Alwan, "The mediating role of e-satisfaction in the relationship between e-service quality and customer e-loyalty in internet banking," *Jordan Journal of Business Administration*, vol. 15, no. 2, 2019.

[34] G. Briones de Araluze and N. Cassinello Plaza, "The relevance of initial trust and social influence in the intention to use open banking-based services: An empirical study," *SAGE Open*, vol. 13, no. 3, p. 21582440231187607, 2023.

[35] H. A. Zadha and G. Suparna, "The role of brand trust mediates the effect of perceived risk and brand image on intention to use digital banking service,"

[36] M. Bijlsma, C. van der Cruijsen, and N. Jonker, "Consumer propensity to adopt psd2 services: trust for sale?," 2020.

[37] R. J. Nam, "Open banking and customer data sharing: Implications for fintech borrowers," 2022.

[38] Z. He, J. Huang, and J. Zhou, "Open banking: Credit market competition when borrowers own the data," *Journal of financial economics*, vol. 147, no. 2, pp. 449–474, 2023.

[39] "The oauth 2.0 authorization framework." https://datatracker.ietf.org/doc/html/rfc6749, Oct. 2012. Internet Engineering Task Force (IETF).

[40] "Oauth 2.0 rich authorization requests." https://datatracker.ietf.org/doc/html/rfc9396, May 2023. Internet Engineering Task Force (IETF).

[41] "Oauth 2.0 pushed authorization requests." https://datatracker.ietf.org/doc/html/rfc9126, Sept. 2021. Internet Engineering Task Force (IETF).

[42] C. Michael, F. Gyara, J. Heenan, T. Lodderstedt, D. Postnikov, and D. Tonge, "Financial-grade api (fapi) profiles." https://openid.net/wg/fapi/, July 2022. OpenID.

[43] "Openid connect core 1.0." https://openid.net/specs/openid-connect-core-1_0.html, Nov. 2014. OpenID.

[44] "Proof key for code exchange by oauth public clients." https://datatracker.ietf.org/doc/html/rfc7636, Sept. 2023. Internet Engineering Task Force (IETF).

[45] "Json web token." https://datatracker.ietf.org/doc/html/rfc7519, May 2015. Internet Engineering Task Force (IETF).

[46] "Json web signatures." https://datatracker.ietf.org/doc/html/rfc7515, May 2015.

[47] "The token binding protocol version 1.0." https://datatracker.ietf.org/doc/html/rfc8471, Oct. 2018. Internet Engineering Task Force (IETF).

[48] "Oauth 2.0 token binding." https://datatracker.ietf.org/doc/html/draft-ietf-oauth-token-binding-08, Oct. 2018. Internet Engineering Task Force (IETF).

[49] "Data rights protocol." https://github.com/consumer-reports-innovation-lab/data-rights-protocol/. Consumer Reports Innovation Lab.

[50] S. Zimmeck, P. Snyder, J. Brookman, and A. Zucker-Scharff, "Global privacy control." https://privacycg.github.io/gpc-spec/, July 2023.

[51] "Solid specification." https://solidproject.org/TR/protocol, Dec. 2022. W3C Solid Community Group.

[52] K. Liao, "Traceability protocol for open banking (draft)." MIT Future of Data Initiative, 2023.

[53] "Guidelines 01/2022 on data subject rights - right of access." https://edpb.europa.eu/system/files/2022-01/edpb_guidelines_012022_right-of-access_0.pdf, Jan. 2022. European Data Protection Board.

[54] "The javascript object notation (json) data interchange format." https://datatracker.ietf.org/doc/html/rfc8259, Dec. 2017. Internet Engineering Task Force (IETF).

[55] "Utf-8, a transformation format of iso 10646." https://datatracker.ietf.org/doc/html/rfc3629, Nov. 2003. Internet Engineering Task Force (IETF).

[56] "The transport layer security (tls) protocol version 1.3." https://datatracker.ietf.org/doc/html/rfc8446, Aug. 2018.

[57] "The oauth 2.0 authorization framework: Jwt-secured authorization request (jar)." https://datatracker.ietf.org/doc/html/rfc9101, Aug. 2021. Internet Engineering Task Force (IETF).

[58] A. Sambra, H. Story, and T. Berners-Lee, "Webid 1.0." https://www.w3.org/2005/Incubator/webid/spec/identity/, 2014. W3C.

[59] S. Capadisli and T. Berners-Lee, "Solid webid profile." https://solid.github.io/webid-profile/#solid-profile, 2023. W3C Solid Community Group.

[60] "Tls certificate compression." https://datatracker.ietf.org/doc/html/rfc8879, Dec. 2020. Internet Engineering Task Force (IETF).