International FAIR Digital Objects Implementation Summit 2024

Extended Abstracts

https://doi.org/10.52825/ocp.v5i.1193

Published: 18 Mar. 2025

FDO Access Control Using FDO Operations

Christophe Blanchi

DONA Foundation, Genève, Switzerland

Correspondence: Christophe Blanchi, cblanchi@dona.net

Abstract. The FDO Requirement Specification Version 3.0 [1] general requirement 9 (**FDO-GR-9)** states that each FDO can have metadata of different types such as access permissions. FDO access permissions is a potentially complex aspect of FDOs: A simple text license could be sufficient for some FDOs, but others may need more restrictive and configurable forms of access controls. While FDOs should be ultimately findable, accessible, interoperable and re-usable, some may not want to be accessible by all nor for free. Moreover, some repository managers may only make their objects accessible as FDOs on the condition that they retain full control over the access control methodology, and can restrict who can access them and under which conditions.

The FDO Requirement Specification Version 3.0 specifications do not specify any particular access control operations but the FDO Specification General Requirement 6 **(FDO-GR6)** suggests that application extensible FDO Operations could be used for that purpose. We call an FDO Operation that enforce an FDO's access control an FDO Access Control Operation. Since it is an FDO operation, an FDO Access Control Operation is intrinsically FAIR and can offer a wide and extensible range of different types of access controls.

While the complete details of FDO Access Control Operations not explicitly specified in the FDO Requirements specifications, much of these concepts are described in the DOIPV2.0 [2] protocol and can be directly applied to the FDO space in an implementation neutral manner.

Keywords: FDO Operation, FDO Operation ID, FDO Access Control, FDO Operation Extensibility, Interoperability, CORDRA Implementation, DOIP Protocol

1. Using Extensible FDO Operations for FDO Access Control

The FDO specification states that the FDO interface protocol, in addition to the basic Create, Read, Update, and Delete operations, supports the ability to add applications and community defined new FDO operations to add the needed FDO functionality. It is this FDO operations extensibility that we propose to leverage to develop a set of global FDO access control operations that can be used by machines to authenticate and negotiate access to FDOs using a wide range of approaches.

An FDO Operation is simply used by invoking the FDO Operation's ID on a specific FDO-ID using the FDO Protocol. Each FDO Operation's FDO provides useful information to facilitate its use such by describing the nature of the operation to the benefit of human users, and more importantly, by including resources to assist services and clients acquire the necessary reference, software, and or services to implement and/or use the operation.

Each FDO Operation's definition specifies the nature, format and encoding of its input query in the FDO protocol request as well as the operation's results nature, format and encoding it returns in its protocol response.

If we express an FDO operation as a function **F**, the FDO's identifier as **FDO-ID**, and the function's associated specific parameter as **PARAMS**, we represent a client's request to perform an operation on an FDO as follow:

$$\mathbf{F(\ FDO\text{-}ID\ ,\ PARAMS\ ) = RESPONSE} \quad (1)$$

The RESPONSE is generated by the FDO service as a result of applying the operation on the targeted FDO. The RESPONSE is entirely specific to the function F.

There are no intrinsic limits to what FDO operations can do. Some operations will perform transformations of the FDO's data, other could be used as FDO indexing services or perform client authentication operations. The use of FDO Operations to authenticate and authorize clients is the focus of our use of FDO Operations as it provides an extensible set of solutions and a high level of FDO access control granularity since each FDO can have its own set of FDO access control operations.

As per the FDO Specification, each FDO will provide, at a minimum, the ListOperation operation that, as its name suggests, will list the operations that the FDO can perform. The full operations list could be restricted based on a client's authentication and authorization. Using the functional notation, we list the set of FDO protocol interactions between a FDO service and a client to authenticate and authorize it using the PKIChallenge FDO Access Control Operation. As the authentication succeeds, the list of operations available to the client changes.

In the following example, CLIENT-A represents the ID of the client. We assume that CLIENT'A's FDO contains that client's public key and matching key certificates which the FDO service will use to evaluate the client's authentication and authorization.

In the example below, the *italic* sections describe the client's and FDO service's actions. The indented *FDO Service Action* sections describe the actions of the FDO service as it processes the client's request and generates the appropriate response.

- ListOperation (FDO-ID) = [List, PKIChallenge]
    - *FDO Service Action: unknown client, return only public operations.*
- *Client Action: No useful data operation available. PKIChallenge is an FDO Access Control Operation that I know. Proceed with authentication request.*
- PKIChallenge (FDO-ID, CLIENT-A) = ChallengeToClientA
    - *FDO Service Action: CLIENT-A wants to authenticate. Create and send a challenge.*
- *Client Action: Sign ChallengeToClientA using my private key creating ChallengeToClientA-SignedByClientA.*
- PKIChallenge (FDO-ID, CLIENT-A, ChallengeToClientA-SignedByClientA) = Authentication_Information
    - *FDO Service Action: The FDO service resolves CLIENT_A's FDO. Get the public key and certificate. Verifies that ChallengeToClientA-SignedByClientA is signed by CLIENT-A by decrypting it using the CLIENT-A's public key fout in the CLIENT-A's FDO. If that validates, check the certificate. It is signed by an entity I trust and gives a certain level of admin right. The FDO Service creates an Authentication token for CLIENT-A that includes the authorization information and returns it along with other information within the Authentication_Information.*
- *Client Action: Parses Authentication_Information and get the Authentication_Token. Issues a ListOperations with the token to see if anything changed*

- *ListOperations (FDO-ID, Authentication_Token) = [List, PKIChallenge, Update, DeleteElement, Delete, Read, FourrierTransform]*
    - *FDO Service Action: Checks the Authentication_Token and determines the set of operations that CLIENT-A is authorized to perform on the FDO and sends the list.*
- *Client Action: DeleteElement is what I needed. Issue request.*
- *DeleteElement (FDO-ID , ElementID , Authentication_Token) = DeleteElementStatus*
    - *Server Action: Validate the AuthenticationToken for the request, verifies that A has the right to DeleteElement. Finds ElementID and deletes it. Returns a success status in the DeleteElementStatus.*
- *Client Action: DeleteElementStatus is successful. I am done.*

How each FDO service implements a given FDO Access Control Operation will typically be transparent to all clients. Specific implementation information could be requested by a client by querying the FDO service which itself accessible as an FDO and using appropriate FDO Operation request.

2. The benefits

The benefit of using FDO Access Control Operations are as follows:

- **Extensibility**. New FDO Access Control Operations can be created by anyone at any time to match their applications' requirements. New FDO access control operations such as zero knowledge proofs could be added as they become available.

- **Interoperability**. Each FDO Access Control Operation are FAIR. A validator service could be referenced by an FDO Access Control Operation to enable the independent testing and validation of client and server implementations.

- **Flexibility**. Each FDO can have more than one FDO access control operation to increase the chances of clients to access the FDO. Each FDO can have its own access control mechanism independently from the other FDOs in the FDO service.

- **Backward compatibility**. Existing repositories can use their existing repository access control methods by mapping them through existing or new FDO access control operation.

3. Flexibility of Implementations

The FDO operations approach to FDO access control is currently implemented in various deployments of CORDRA [3] which uses DOIP V2.0 [2] as the underlying protocol. This functionality could easily be implemented using web technologies as there are deployed implementations of DOIP over HTTP.

Data availability statement

n.a.

Underlying and related material

n.a.

Author contributions

Conceptualization: CB; Writing – original draft: CB; Writing – review & editing: CB

Competing interests

The authors declare that they have no competing interests.

Funding

n.a.

References

[1] FDO Forum FDO Requirements Specifications Version 3.0 https://zenodo.org/records/7782262
[2] DOIPV2.0 https://hdl.handle.net/0.100/DOIPV2.0
[3] CORDRA https://www.cordra.org/