

Methods for Establishing Trust between OAuth 2.0 Proxies in Different Trust Domains

Publication Date [2024-11-25]
Authors: Diana Gudu (ed), Valeria Ardizzione, Marcus Hardt, Jens Jensen, Ivan Kanakarakis, Christos Kanellopoulos, Nicolas Liampotis, Jan Pavlíček, Mischa Sallé, Hannah Short, Michal Stava, Gabriel Zachmann
Document Code: AARC-I058
DOI: 10.5281/zenodo.17054048
Community: Architecture Area

Abstract

*This informational document elaborates about different methods for establishing trust between Entities in different Trust Domains. This is needed for the Entities to exchange authentication and authorisation information e.g. in OAuth2.0 flows.
The six approaches presented include their specific flows.*



1 Introduction	2
1.1 Notational Conventions	2
1.2 Terminology	2
2 Trust evaluation	3
3 Recommendations	8
4. Trust evaluation approaches	8
4.1 Establishing trust with shared list of trusted entity identifiers	8
4.2 Establishing trust with shared aggregated entity statements	10
4.2.1 API for retrieving entity statements	11
4.2.1.1 Resolve Entities Request	11
4.2.1.2 Resolve Entities Response	12
4.3 Establishing trust with shared aggregated entity statements: via OID-Fed's extended listing	12
4.4 Establishing trust with targeted entity statement resolution: Resolve by subject ID	13
4.5 Establishing trust with shared list of trusted entity identifiers with OpenID Federation Metadata	15
4.6 Federation entity discovery	17
References	22

1 Introduction

This document explores different approaches for enabling interaction and establishing trust among entities such as OAuth 2.0 Authorization Servers (AS) and Resource Servers (RS) residing in distinct domains. These interactions are facilitated through trusted third parties referred to as Trust Authorities, which are entities issuing authoritative statements about entities that participate in an identity federation. The federation may use OpenID Federation but it is not a requirement.

1.1 Notational Conventions

The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document are to be interpreted as described in [\[RFC2119\]](#).

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

1.2 Terminology

This section defines the terminology used by this specification. This section is a normative portion of this specification, imposing requirements upon implementations.

This specification uses the terms "access token", "authorization endpoint", "authorization grant", "authorization server" ("AS"), "client", "client identifier", "protected resource", "refresh

token", "resource owner", "resource server" ("RS"), and "token endpoint" defined by OAuth 2.0 [RFC6749], the terms "claim names" and "claim values" defined by JSON Web Token (JWT) [RFC7519], the terms "token introspection" and "introspection endpoint" defined by OAuth 2.0 Token Introspection [RFC7662], the terms "entity", "entity identifier", "trust anchor", "federation entity", "entity statement", "entity configuration", "subordinate statement", "entity type", "entity type identifier", "leaf entity", "subordinate entity", "federation entity discovery", "trust chain", "trust mark", "federation entity keys" defined by OpenID Federation [OID-Fed], and the terms "infrastructure proxy", "SP-IdP-Proxy" defined by the AARC Blueprint Architecture 2019 [AARC-G045].

This specification defines the following terms:

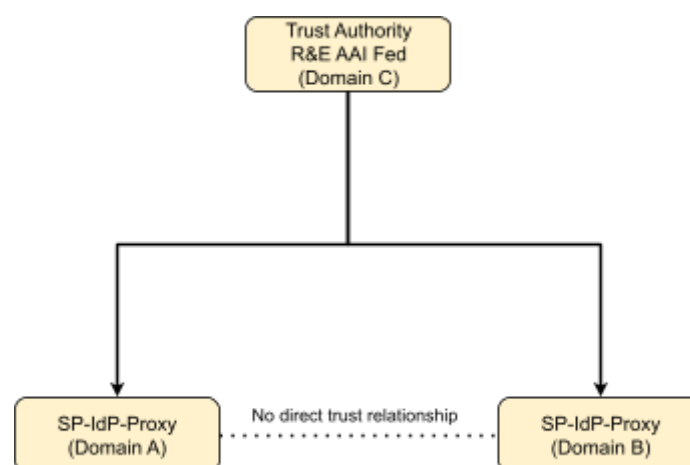
Trust Authority

A Trust Authority is a third party which is trusted to identify entities.

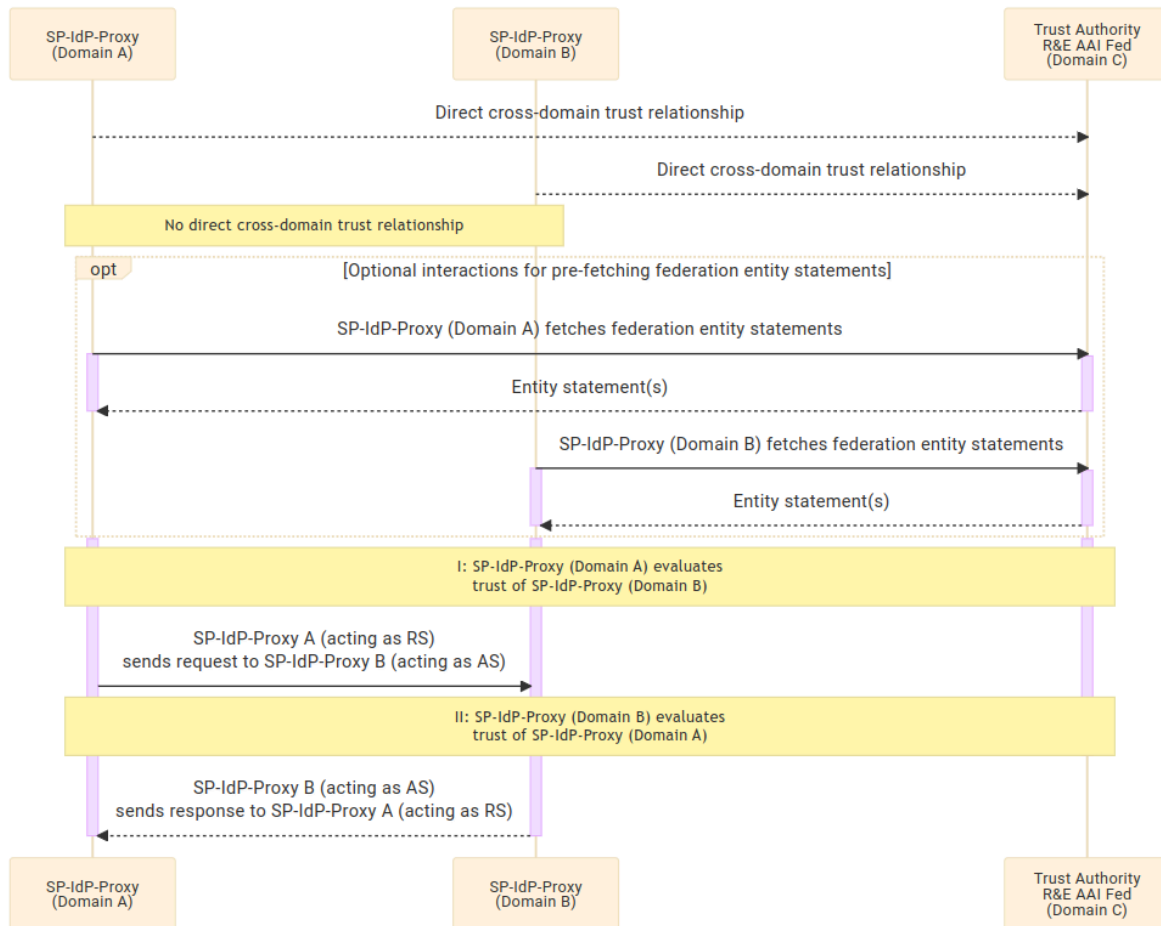
- In OpenID Federation, a Trust Authority is a third party represented by one or more [Trust Anchors](#).
- In IGTF, a Trust Authority is a Certification Authority; the Trust Anchors are its root or intermediate certificates in the IGTF distribution.
- A Trust Authority could simply maintain and publish a list of identifiers of entities, provided the association between each entity and its identifier can be checked by other trusted means.

2 Trust evaluation

This section describes different approaches for enabling Authorization Servers (AS) in different domains to interact and establish trust between them through a trusted third party called a Trust Authority. A Trust Authority is an entity whose primary purpose is to issue statements about entities, such as Authorization Servers (AS) and Resource Servers (RS) that participate in a federation. The diagram below depicts a single Trust Authority for simplicity. However, the trust evaluation mechanisms allow for multiple authorities.



The diagram below illustrates the general flow for how an SP-IdP-Proxy in Domain A can establish trust, acting as a Resource Server, with another SP-IdP-Proxy in Domain B acting as an OAuth 2.0 Authorization Server, both relying on a shared Trust Authority in Domain C.



In [Section 4](#), we explore different approaches to the trust evaluation mechanism employed by the AS (see Steps I & II in the diagram).

1. **Shared list of entity identifiers of trusted entities**: Proxies retrieve a list of entity identifiers of all entities (including the proxies acting as AS, RS above) from their Trust Authority(ies), either proactively, or when interaction is required. Upon interaction with another proxy, proxies rely on OIDC Discovery or OAuth 2.0 Protected Resource Metadata to obtain required information such as endpoints and keys. The list of entity identifiers, whether retrieved proactively or on-demand, MAY be cached. The proxies need to periodically update their entity identifier lists through calls to the relevant Trust Authority(ies). The approach relies on the Automatic Client registration mechanism defined in OID-Fed to establish trust and configuration between the proxies.
2. **Shared aggregated entity statements**: Proxies proactively retrieve metadata and Trust Marks for all federation members via the resolve endpoint of their Trust

Authority(ies)¹ as defined in Section 4.2. When interaction is required, proxies rely on cached information and periodically refresh the resolved metadata through calls to the Trust Authority's resolve endpoint.

3. **Shared aggregated entity statements via OID-Fed extended listing**: This approach is similar to the Shared aggregated entity statements approach, except it relies on the newly adopted [OpenID Federation Extended Subordinate Listing](#) specification to proactively retrieve metadata and Trust Marks for all federation members from the extended_list endpoint of their Trust Authority.
4. **Targeted entity statement resolution**: Proxies retrieve metadata and Trust Marks for the Entity they need to interact with via the resolve endpoint as defined in [OpenID Federation](#) of their Trust Authority(ies). The resolved entity metadata MAY be cached.
5. **Shared list of entity identifiers of trusted entities with OpenID Federation Metadata**: This approach is similar to the Shared list of entity identifiers of trusted entities but requires Proxies to implement publishing and obtaining entity configurations as defined by OID-Fed.
6. **Federation entity discovery**: Proxies retrieve sufficient entity statements to construct a trust chain from the proxy they intend to interact with to the Trust Anchors using the process defined in [Section 9 of OpenID Federation](#).

The table below provides an overview of the approaches and a comparison across a number of relevant metrics:

- **Interaction overhead**: any overhead that the method introduces that might affect performance
- **Support for multiple federation policies**: whether and how support for federation policies (such as SIRTFI, RAF) is expressed by federation members
- **Standards and specifications**: whether the approach relies on existing standards, and if yes, which ones, and where are they used

¹ This approach is similar to the current eduGAIN interfederation model.

Approach	Interaction overhead	Support for multiple Federation Policies	Standards and specifications
1. Shared list of entity identifiers of trusted entities (see Section 4.1)	Requires retrieval of list(s) of entity identifiers from Trust Authority(ies), on demand or periodically	Can be supported by the Trust Authorities distributing Trust Mark specific lists of entity identifiers	- Metadata: OpenID Connect Discovery [OIDC-Discovery] OAuth 2.0 Protected Resource Metadata [OAuth2-PRM] - Lack of a standard method for obtaining entity metadata, including security contact information in the event of security incidents. This would require an extension of the OpenID Discovery /OAuth Server metadata specification - No available standard for distributing list of entity identifiers
2. Shared aggregated entity statements (see Section 4.2)	Requires retrieval of entity statements from Trust Authority(ies), periodically, based on statement expiration or local policy	Can be supported using Trust Marks	- Standard format available for defining the entity statements as per OpenID Federation [OID-Fed], including security contact information - No standard protocol for publishing/updating the entity statements: - “resolve” requires extension to the OpenID Fed resolve entity request (unlikely to be accepted upstream)
3. Shared aggregated entity statements via OID-Fed extended listing (see Section 4.3)	Requires retrieval of entity statements from Trust Authority(ies), periodically, based on statement expiration or local policy	Can be supported using Trust Marks	- Standard format available for defining the entity statements as per OpenID Federation [OID-Fed], including security contact information - Standard format available for obtaining the entity statements as per OpenID Federation Extended Subordinate Listing [OID-Fed-ExtList] - No standard protocol for publishing/updating

			the entity statements
4. Targeted entity statement resolution (see Section 4.4)	Requires retrieval of entity statement from Trust Authority(ies), on-demand (response can be cached based on statement expiration or local policy)	Can be supported using Trust Marks	- Standard format available for defining and obtaining the entity statements as per OpenID Federation [OID-Fed], including security contact information
5. Shared list of entity identifiers of trusted entities with OpenID Federation Metadata (see Section 4.5)	Requires periodic retrieval of list(s) of entity identifiers from Trust Authority(ies), on demand or periodically	Can be supported by 1. the Trust Authorities distributing Trust Mark specific lists of entity identifiers; or 2. The federation members publish Trust Marks signed by the trust list providers	- Standard method based on OpenID Federation [OID-Fed] for exposing/consuming metadata - Shared list trusted issuers is not standard
6. Federation entity discovery (see Section 4.6)	Requires retrieval of entity statements from one or more superior entities, on demand (response(s) can be cached based on statement expiration or local policy)	Flexibility in defining required federation policies in terms of technical and policy requirements	- Standard method based on OpenID Federation [OID-Fed] for exposing/consuming entity metadata, including security contact information in the event of security incidents - OpenID Federation is still draft

3 Recommendations

Deciding which approach to use depends on the importance assigned to each of the comparison metrics defined in the table above. However, the following approaches SHOULD be favoured:

6. Federation entity discovery (see [Section 4.6](#)): it fully relies on standards and provides the most flexibility, making it future-proof. However, the interaction overhead and implementation costs on the proxies might make it restrictive in the short term.

4. Targeted entity statement resolution (see [Section 4.4](#)): good alternative approach, as it also relies on existing standards, but in comparison to 6., it has reduced overhead and implementation costs on the proxies, offloading this to the Trust Authorities. However, it cannot handle complex hierarchical federations or dynamically updated metadata. In fact, it does not have a standard mechanism for publishing and updating metadata.

3. Shared aggregated entity statements via OID-Fed extended listing (see [Section 4.3](#)): similar to 4, it relies on existing standards and reduces overhead on proxies, but cannot handle complex hierarchical federations or dynamically updated metadata. Compared to 4, it is more suitable for small federations or densely connected federations (i.e. where each entity interacts with many other entities), and as such retrieving aggregated information about all entities is more efficient than targeted entity statement resolution. However, it does incur higher costs on the proxies compared to 4 when it comes to maintaining the cached aggregated entity statements.

All the other approaches MAY be considered, but only as compromise solutions or intermediate steps to reaching one of the two recommended approaches.

4. Trust evaluation approaches

4.1 Establishing trust with shared list of trusted entity identifiers

In this approach, the Trust Authority's role is to distribute a list of trusted entity identifiers. The SP-IdP-Proxies retrieve this list from their Trust Authority(ies), either proactively, or on-demand, when interaction is required.

To establish trust between two SP-IdP-Proxies that have no direct cross-domain trust relationship, each proxy must check if the shared list of trusted identifiers contains the other proxy. The approach then relies on the Automatic Client registration mechanism defined in the OpenId Federation specification.

In order to support multiple federation policies, the Trust Authority must maintain a list of trusted entity identifiers for each policy.

There is no standard available for the Trust Authorities to distribute these lists to the proxies, nor a standard method for obtaining entity metadata.

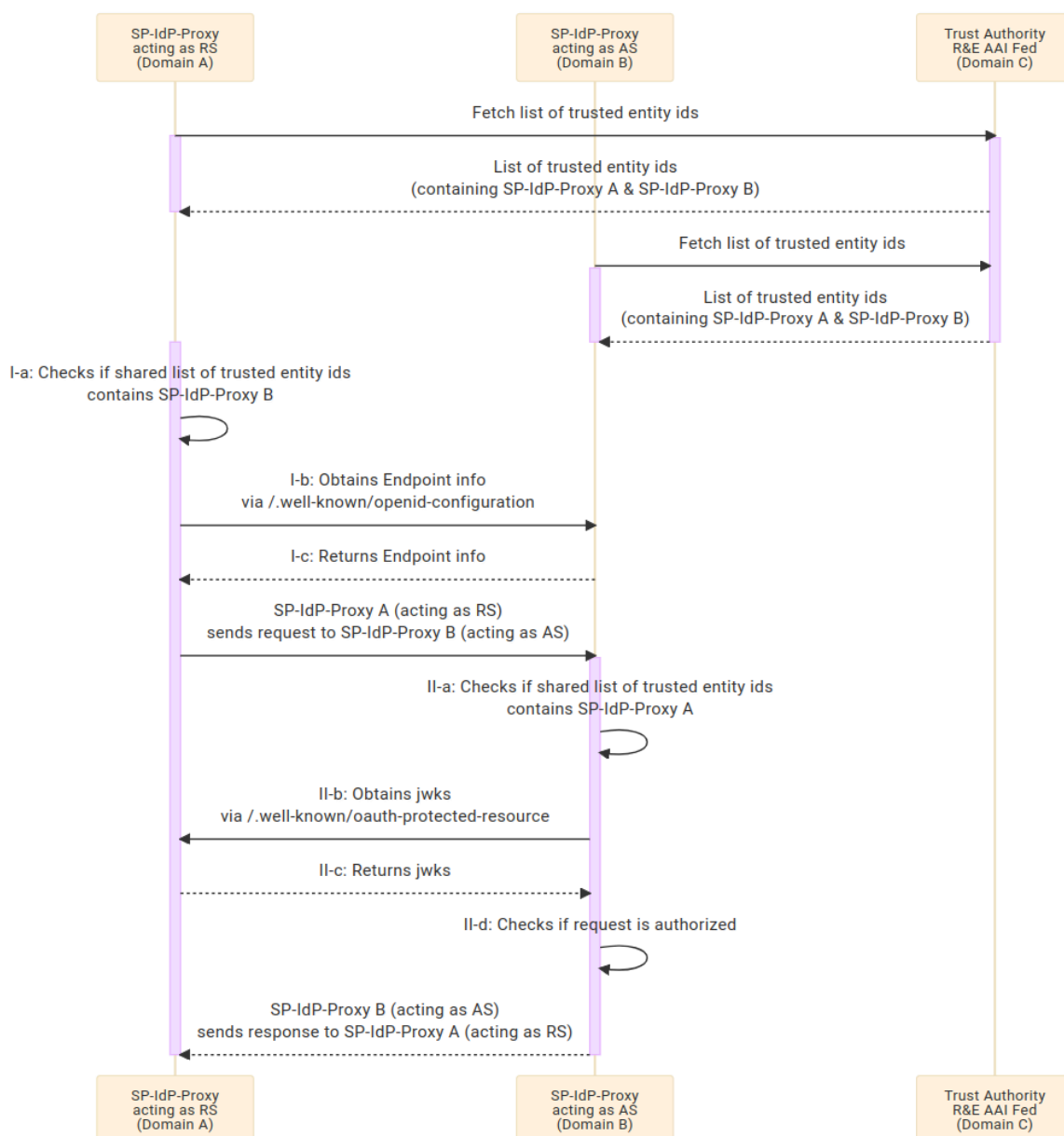
The Trust Authority is required to:

- maintain a list of trusted entity identifiers (or multiple, one for each policy).
- provide a mechanism for the SP-IdP-Proxies to retrieve the list of trusted entity identifiers.

The SP-IdP-Proxies are required to:

- implement automatic client registration as defined in the OpenID Federation specification.

The diagram below illustrates the flow for this approach, for how an SP-IdP-Proxy in Domain A can establish trust, acting as a resource server, with another SP-IdP-Proxy in Domain B acting as an OAuth 2.0 Authorization Server, both relying on the shared Trust Authority in Domain C.



4.2 Establishing trust with shared aggregated entity statements

In this approach, a Trust Authority's role is to distribute entity statements for all federation members, where the entity statements encompass the entire entity configuration as defined in the OpenID Federation specification, including metadata and Trust Marks. The SP-IdP-Proxies periodically retrieve this information from their Trust Authority(ies), via the "resolve" endpoint. This endpoint is defined in Section 4.2.1, and it returns the aggregated entity statements for all federation members. However, it is not part of the OpenID Federation specification, and there is no standard protocol for publishing or updating the entity statements either.

To establish trust between two SP-IdP-Proxies that have no direct cross-domain trust relationship, each proxy must check if the shared list of entity statements contains a statement for the other proxy, and then check the required Trust Marks. The approach then relies on the Automatic Client registration mechanism defined in the OpenID Federation specification.

Support for multiple federation policies can be implemented using Trust Marks. As per the definition in the OID-Fed specification, Trust Marks are statements of conformance to sets of criteria determined by an accreditation authority. Each federation policy can be expressed as a Trust Mark. In this approach, the Trust Authority can also play the role of the Trust Mark issuer and would be responsible for issuing and checking the status of a Trust Mark. Since the mechanism for publishing entity statements is done out-of-band by the Trust Authority, it would make sense for the Trust Authority to also be responsible for adding the Trust Marks to the entity statements.

This approach is easier to implement for the proxies, as they only need to retrieve the entity statements from the Trust Authority(ies) and cache them. However, it requires the Trust Authority to maintain the entity statements for all federation members, which can be a significant overhead.

The Trust Authority is required to:

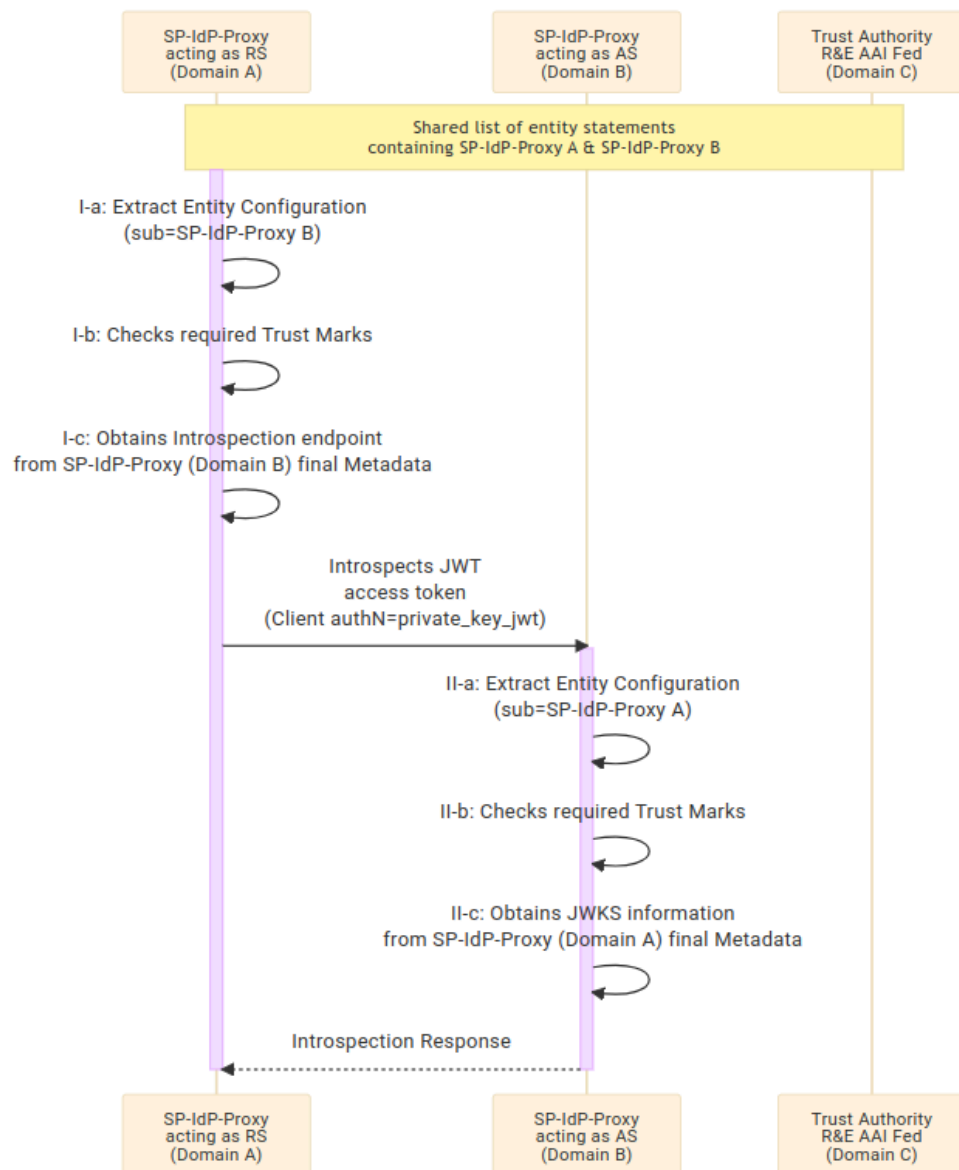
- provide the non-standard "resolve" endpoint for the SP-IdP-Proxies to retrieve the aggregated entity statements.
- provide a mechanism for the SP-IdP-Proxies to publish and update their entity statements.
- issue Trust Marks and add them to the entity statements
- provide a Trust Mark status endpoint

The SP-IdP-Proxies are required to:

- implement automatic client registration as defined in the OpenID Federation specification.
- be able to validate Trust Marks

The diagram below illustrates the flow for this approach, for how an SP-IdP-Proxy in Domain A can establish trust, acting as a resource server, with another SP-IdP-Proxy in Domain B

acting as an OAuth 2.0 Authorization Server, both relying on the shared Trust Authority in Domain C.



4.2.1 API for retrieving entity statements

4.2.1.1 Resolve Entities Request

The request MUST be an HTTP request using the GET method and the https scheme to a resolve endpoint with the following query string parameters, encoded in application/x-www-form-urlencoded format.

type

OPTIONAL. A specific Entity Type to resolve. Its value is an Entity Type Identifier, as specified in Section 5.1 of the OpenID Federation specification. If this parameter is not present, then all Entity Types are returned.

The following is a non-normative example of a resolve request:

```
GET /resolve?type=openid_provider HTTP/1.1
Host: ta.example.org
```

4.2.1.2 Resolve Entities Response

A successful response MUST use the HTTP status code 200 with the content type application/json, containing a JSON array with the resolved metadata and verified Trust Marks of all entities. These are the claims in each array entry of the resolve response:

- **iss** REQUIRED. String. Entity Identifier of the issuer of the resolve response.
- **sub** REQUIRED. String. Entity Identifier of the subject of the resolve response.
- **iat** REQUIRED. Number. Time when this resolution was issued. This is expressed as Seconds Since the Epoch, as defined in [RFC7519].
- **exp** REQUIRED. Number. Time when this resolution is no longer valid. This is expressed as Seconds Since the Epoch, as defined in [RFC7519]. It MUST correspond to the exp value of the Trust Chain from which the resolve response was derived.
- **metadata** REQUIRED. JSON object containing the resolved subject metadata, according to the requested type and expressed in the metadata format defined in Section 5 of the OpenID Federation specification.
- **trust_marks** REQUIRED. As defined in Section 3 of the OpenID Federation specification, array of objects, each representing a Trust Mark (see Section 7 of the OpenID Federation specification).
- **trust_chain** OPTIONAL. Array containing the sequence of Entity Statements that compose the Trust Chain, starting with the subject and ending with the selected Trust Anchor, sorted as shown in Section 4 of the OpenID Federation specification.

4.3 Establishing trust with shared aggregated entity statements: via OID-Fed's extended listing

In this approach, a Trust Authority's role is to distribute subordinate entity statements for all federation members, as well as metadata and Trust Marks, as defined in the OpenID Federation specification. The SP-IdP-Proxies periodically retrieve this information from their Trust Authority(ies), via the "extended_list" endpoint, as defined in the OpenID Federation Extended Subordinate Listing specification [\[OID-Fed-ExtList\]](#). The proxies must request additional claims to be included in the extended subordinate listing response, including at the very least "metadata" and "trust_marks". The proxies cache this information and periodically refresh it through calls to the Trust Authority's extended_list endpoint. To this end, the proxies may use features in the OpenID Federation Extended Subordinate Listing

specification, such as pagination and the “updated_after” parameter. There is no standard protocol for entities to publish or update their metadata stored at the Trust Authority.

To establish trust between two SP-IdP-Proxies that have no direct cross-domain trust relationship, each proxy must check if the shared list of entity statements contains a statement for the other proxy, and then check the required Trust Marks. The approach then relies on the Automatic Client registration mechanism defined in the OpenId Federation specification.

Similar to Approach 4.2, Trust Marks can be used to provide support for multiple federation policies, with the Trust Authority being responsible for issuing Trust Marks and adding them to the entity statements.

This approach is easier to implement for the proxies, as they only need to retrieve the entity statements from the Trust Authority(ies) and cache them. However, it requires the Trust Authority to maintain the entity statements for all federation members, which can be a significant overhead. In addition, this approach will not scale to hierarchical federations, since the extended_list endpoint only returns information about the Trust Authority's direct subordinates.

The Trust Authority is required to:

- implement the OpenID Federation Extended Subordinate Listing specification
- provide a mechanism for the SP-IdP-Proxies to publish and update their entity statements
- issue Trust Marks and add them to the entity statements
- provide a Trust Mark status endpoint.

The SP-IdP-Proxies are required to:

- implement automatic client registration as defined in the OpenID Federation specification.
- be able to validate Trust Marks

The process is the same as the one depicted in the diagram in [Section 4.2](#). The only difference is the way in which the shared list of entity statements is retrieved from the Trust Authority.

4.4 Establishing trust with targeted entity statement resolution: Resolve by subject ID

In this approach, a Trust Authority's role is to distribute entity statements for the entities that the SP-IdP-Proxies need to interact with. The SP-IdP-Proxies retrieve this information from their Trust Authority(ies) via the standard “resolve” endpoint, as defined in the OpenID Federation specification. This endpoint returns the entity statement for the requested entity. The SP-IdP-Proxies can cache the resolved entity statements to avoid repeated requests.

To establish trust between two SP-IdP-Proxies that have no direct cross-domain trust relationship, each proxy must retrieve the entity statement for the other proxy from the Trust

Authority, and then check the required Trust Marks. The approach then relies on the Automatic Client registration mechanism defined in the OpenID Federation specification.

Similar to Approach 4.2, Trust Marks can be used to provide support for multiple federation policies, with the Trust Authority being responsible for issuing Trust Marks and adding them to the entity statements.

This approach is more efficient than the previous one, as the SP-IdP-Proxies only need to retrieve the entity statements for the entities they need to interact with. In addition, it relies on the standard "resolve" endpoint defined in the OpenID Federation specification. However, there is still no standard protocol for publishing or updating the entity statements.

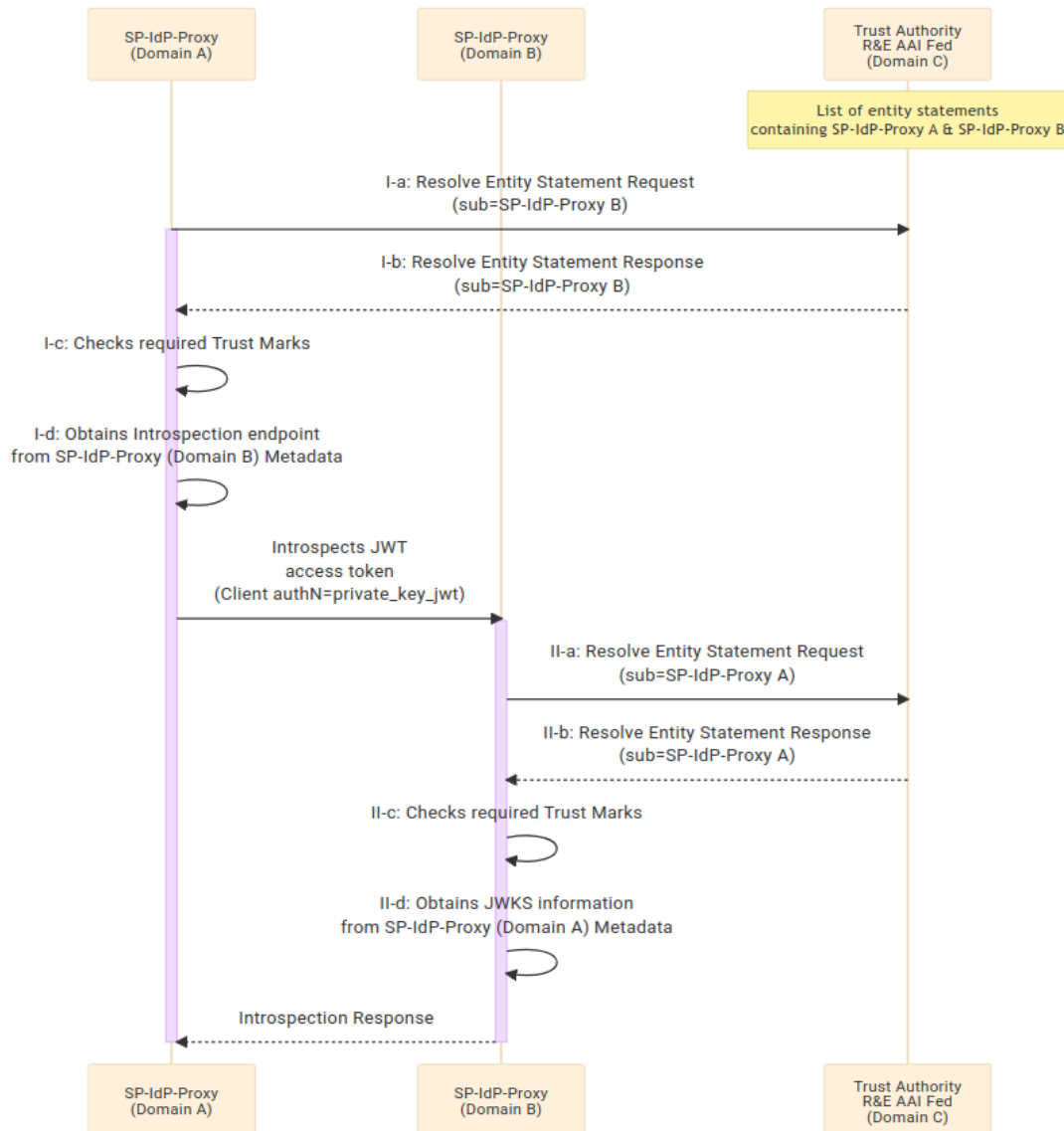
The Trust Authority is required to:

- implement the "resolve" endpoint as defined in the OpenID Federation Specification
- provide a mechanism for the SP-IdP-Proxies to publish and update their entity statements
- issue Trust Marks and add them to the entity statements
- provide a Trust Mark status endpoint.

The SP-IdP-Proxies are required to:

- implement automatic client registration as defined in the OpenID Federation specification.
- be able to validate Trust Marks

The diagram below illustrates the flow for this approach, for how an SP-IdP-Proxy in Domain A can establish trust, acting as a resource server, with another SP-IdP-Proxy in Domain B acting as an OAuth 2.0 Authorization Server, both relying on the shared Trust Authority in Domain C.



4.5 Establishing trust with shared list of trusted entity identifiers with OpenID Federation Metadata

In this approach, a Trust Authority's role is to distribute a list of trusted entity identifiers. The SP-IdP-Proxies retrieve this list from their Trust Authority(ies), either proactively, or on-demand, when interaction is required. In addition, each proxy must implement publishing and obtaining entity configurations as defined by OID-Fed.

To establish trust between two SP-IdP-Proxies that have no direct cross-domain trust relationship, each proxy must check if the shared list of trusted identifiers contains the other proxy. Then, each proxy must retrieve the entity configuration for the other proxy from its well-known endpoint, and check the required Trust Marks. The approach then relies on the Automatic Client registration mechanism defined in the OpenId Federation specification.

In order to support multiple federation policies, there are two options:

- 1) the Trust Authority must maintain a list of trusted entity identifiers for each policy, or
- 2) the federation members must publish Trust Marks in their entity configuration.

When relying on Trust Marks, the Trust Authority is assumed to play the role of Trust Mark issuer. However, since the proxies publish their own entity configurations as defined by OID-Fed, they are also responsible for requesting and adding Trust Marks to their entity configurations.

There is no standard available for the Trust Authorities to distribute these lists to the proxies, but the approach relies on OpenID Federation specification for exposing and consuming metadata.

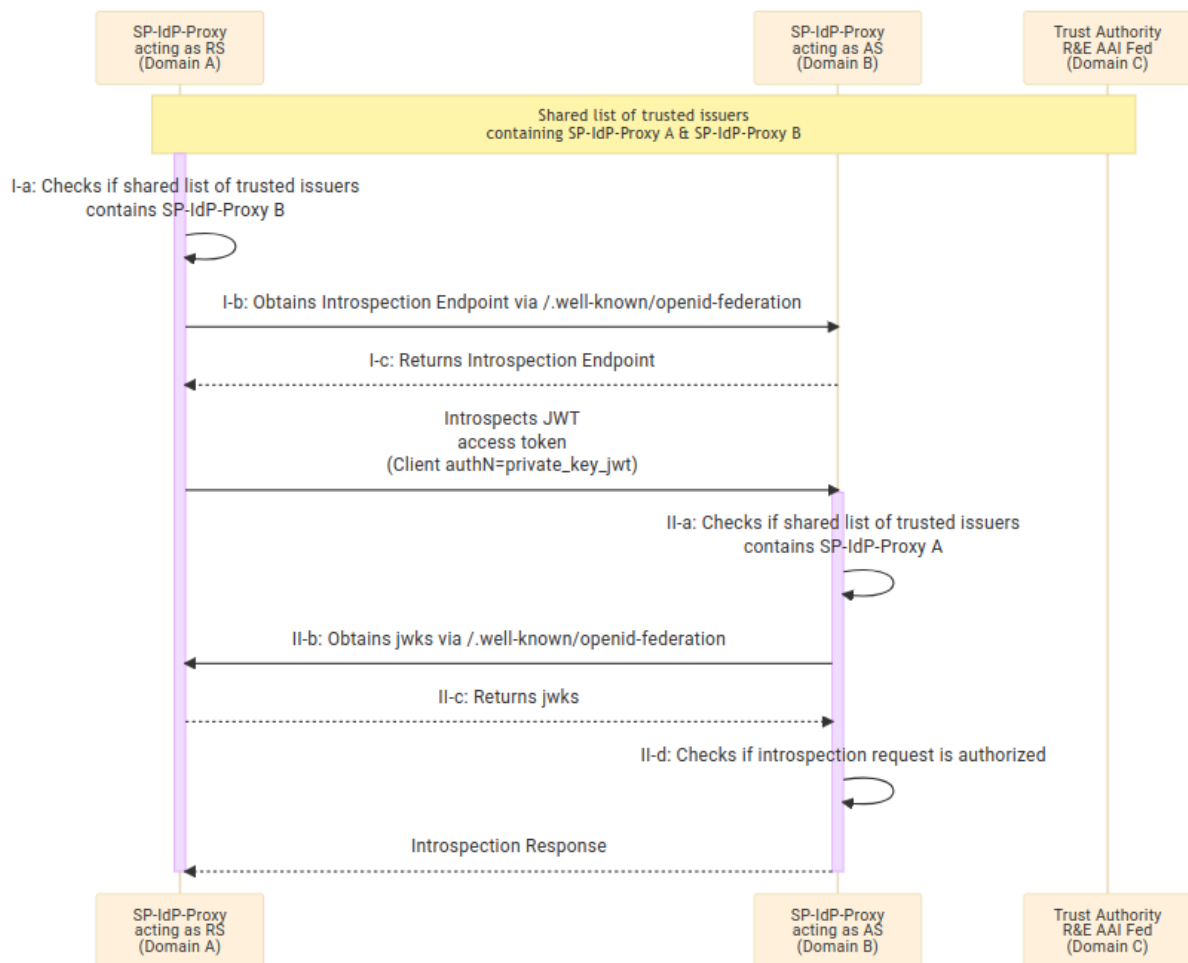
The Trust Authority is required to:

- maintain a list of trusted entity identifiers (or multiple, one for each policy).
- provide a mechanism for the SP-IdP-Proxies to retrieve the list of trusted entity identifiers.
- provide Trust Mark endpoint and Trust Mark status endpoint, as defined by OID-Fed spec

The SP-IdP-Proxies are required to:

- expose entity configurations via the well-known endpoint, as defined by OID-Fed
- implement mechanisms for regularly requesting Trust Marks from the Trust Mark issuer and adding them to their entity configuration
- implement automatic client registration as defined in the OpenID Federation specification.
- be able to validate Trust Marks

The diagram below illustrates the flow for this approach, for how an SP-IdP-Proxy in Domain A can establish trust, acting as a resource server, with another SP-IdP-Proxy in Domain B acting as an OAuth 2.0 Authorization Server, both relying on the shared Trust Authority in Domain C.



4.6 Federation entity discovery

This approach is based on the OpenID Federation specification, which defines a process for discovering entity statements for all federation members. The SP-IdP-Proxies retrieve the entity statements for the entities they need to interact with, and construct a trust chain from the proxy they intend to interact with to the Trust Anchors.

This approach is more complex than the previous ones, as the SP-IdP-Proxies need to retrieve sufficient entity statements to construct a trust chain. However, it is more flexible, as it allows for more complex, hierarchical federations to be modelled. Multiple federation policies can be supported using Trust Marks.

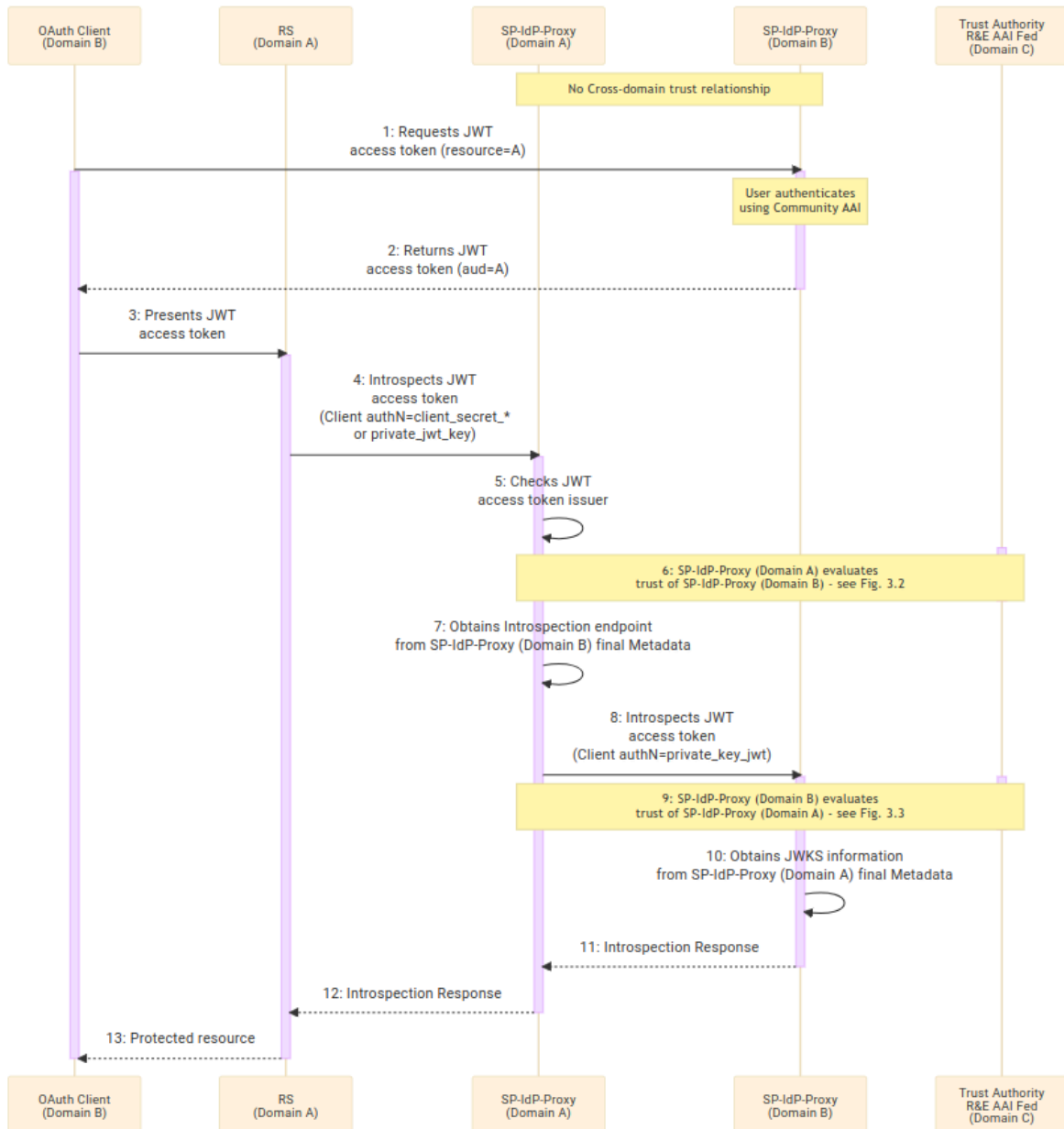
The Trust Authority is required to:

- provide the "fetch" endpoint for subordinate statements as defined by the OpenID Federation specification
- act as a Trust Marks issuer for the supported federation policies by implementing the Trust Mark endpoint and Trust Mark status endpoint, or list third party federation entities that take on the role of Trust Mark issuers.

The SP-IdP-Proxies are required to:

- expose entity configurations via the well-known endpoint, as defined by OID-Fed
- implement mechanisms for regularly requesting Trust Marks from the Trust Mark issuer and adding them to their entity configuration
- implement trust chain construction as defined in the OpenID Federation specification.
- implement automatic client registration as defined in the OpenID Federation specification.
- be able to validate Trust Marks

The trust evaluation is based on OpenID Federation using Automatic Client Registration. In turn, the Proxy in Domain B needs to evaluate the trust of Proxy in Domain A in order to authorise the introspection request. The diagram illustrates the message flow and trust relationships among these participants in this cross-domain token validation scenario.



The following sequence diagram represents the interactions between the SP-IdP-Proxy (Domain A), the SP-IdP-Proxy (Domain B), and the Trust Authority (Trust Anchor in OID-Fed terminology) during a trust evaluation made by the SP-IdP-Proxy (Domain A) for the SP-IdP-Proxy (Domain B):

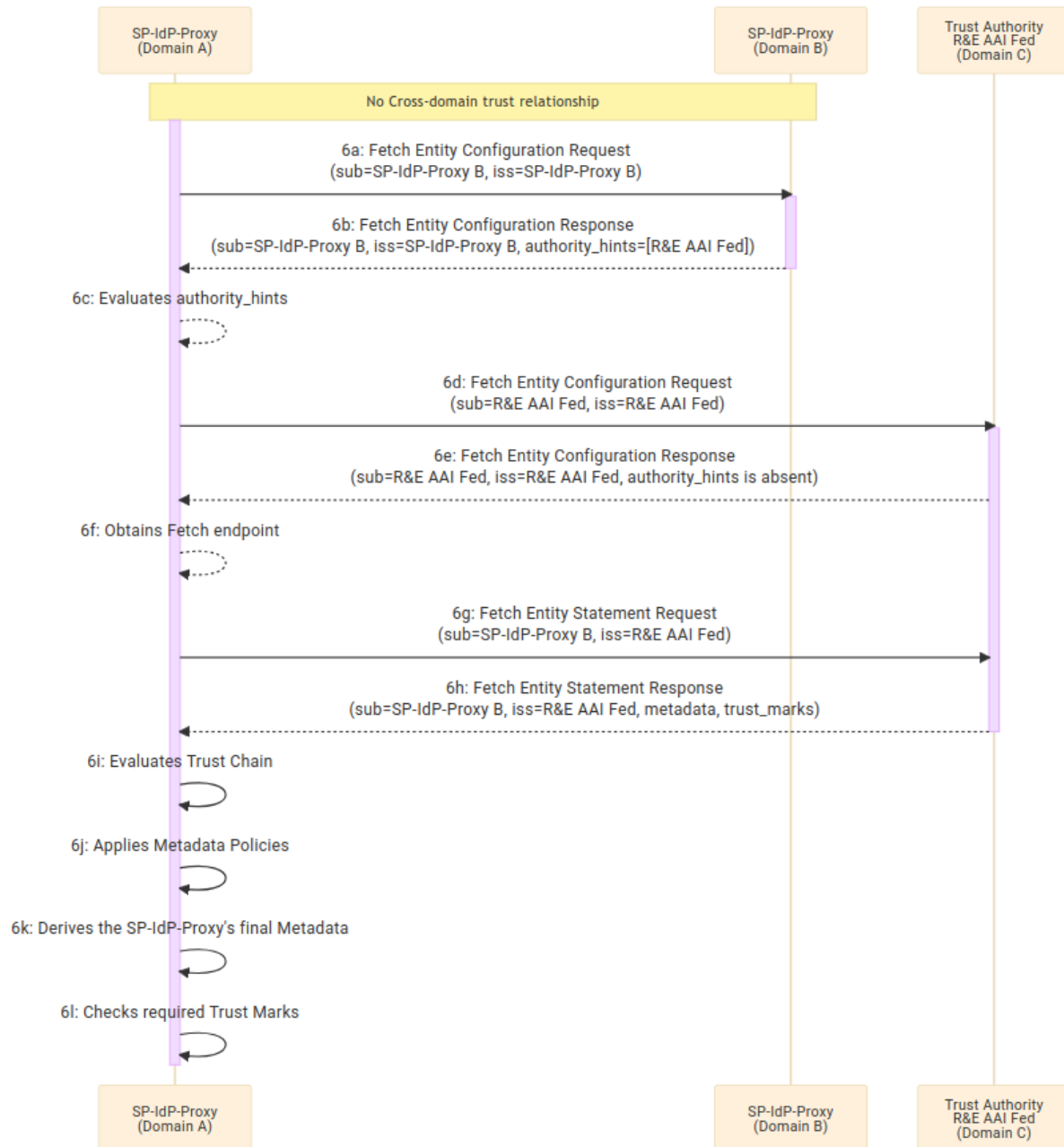


Fig. 3.2 SP-IdP-Proxy (Domain A) evaluates trust of SP-IdP-Proxy (Domain B)

Please note that in version OID-Fed v39 the iss parameter was removed from fetch endpoint.

The following sequence diagram represents the interactions between the SP-IdP-Proxy (Domain A), the SP-IdP-Proxy (Domain B), and the Trust Anchor during a trust evaluation made by the SP-IdP-Proxy (Domain B) for the SP-IdP-Proxy (Domain A):

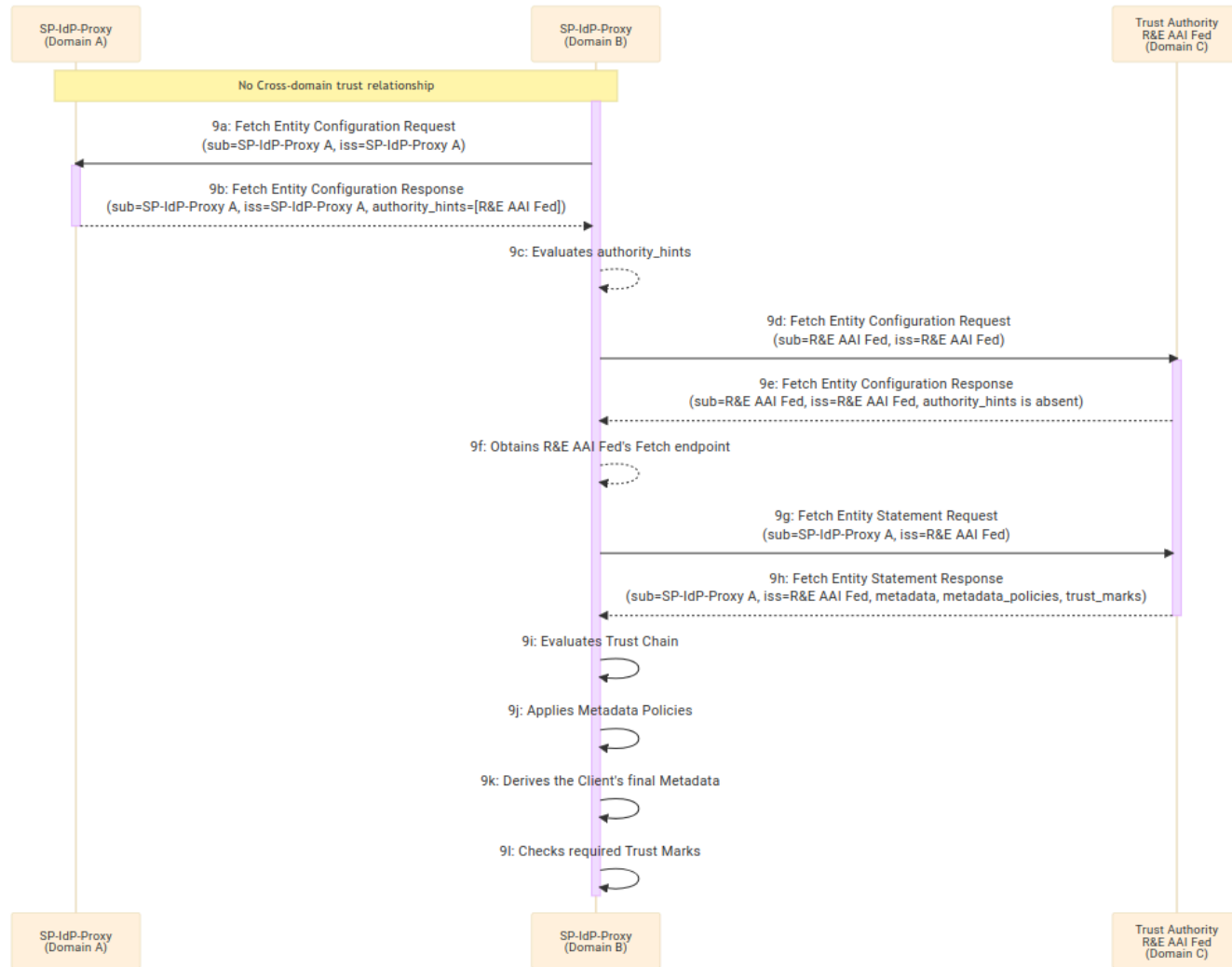


Fig. 3.3 SP-IdP-Proxy (Domain B) evaluates trust of SP-IdP-Proxy (Domain A)

References

- [AARC-G045]** AARC Community members;AppInt members;Nicolas Liampotis (ed.), "AARC Blueprint Architecture 2019", DOI: 10.5281/zenodo.3672784, November 2019,
<<https://aarc-community.org/guidelines/aarc-g045/>>.
- [OAuth2-PRM]** Jones, M. B., Hunt, P., and Parecki, A., "OAuth 2.0 Protected Resource Metadata", May 2024,
<<https://www.ietf.org/archive/id/draft-ietf-oauth-resource-metadata-05.html>>.
- [OIDC-Discovery]** Sakimura, N., Bradley, J., Jones, M., and Jay, E., "OpenID Connect Discovery 1.0 incorporating errata set 2", December 2023,
<https://openid.net/specs/openid-connect-discovery-1_0.html>.
- [OID-Fed]** Hedberg, R., Ed., Jones, M.B., Solberg, A.Å., Bradley, J., De Marco, G., and Dzhuvinov, V., "OpenID Federation 1.0", Draft 37, August 2024,
<https://openid.net/specs/openid-federation-1_0.html>.
- [OID-Fed-ExtList]** De Marco, G., Fraser, M., Jaromin, L., and Jones, M.B., "OpenID Federation Extended Subordinate Listing 1.0 draft 00", August 2024,
<<https://openid.github.io/federation-extended-listing/main.html>>.
- [RFC2119]** Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997,
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC6749]** Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012,
<<http://www.rfc-editor.org/info/rfc6749>>.
- [RFC7159]** Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <<https://www.rfc-editor.org/info/rfc7159>>.
- [RFC7515]** Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015,
<<https://www.rfc-editor.org/info/rfc7515>>.
- [RFC7519]** Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015,
<<https://www.rfc-editor.org/rfc/rfc7519>>.
- [RFC7662]** Richer, J., "OAuth 2.0 Token Introspection", RFC 7662, DOI 10.17487/RFC7662, October 2015,
<<https://www.rfc-editor.org/rfc/rfc7662>>.
- [RFC8414]** Jones, M., Sakimura, N., and J. Bradley, "OAuth 2.0 Authorization Server Metadata", RFC 8414, DOI 10.17487/RFC8414, June 2018,
<<https://www.rfc-editor.org/info/rfc8414>>.
- [RFC8693]** Jones, M., Nadalin, A., Campbell, B., Ed., Bradley, J., and C. Mortimore, "OAuth 2.0 Token Exchange", RFC 8693, DOI

- 10.17487/RFC8693, January 2020,
<<https://www.rfc-editor.org/info/rfc8693>>.
- [RFC8707]** Campbell, B., Bradley, J., and H. Tschofenig, "Resource Indicators for OAuth 2.0", RFC 8707, DOI 10.17487/RFC8707, February 2020,
<<https://www.rfc-editor.org/info/rfc8707>>.
- [RFC8725]** Sheffer, Y., Hardt, D., and M. Jones, "JSON Web Token Best Current Practices", BCP 225, RFC 8725, DOI 10.17487/RFC8725, February 2020, <<https://www.rfc-editor.org/info/rfc8725>>.
- [RFC9068]** Bertocci, V., "JSON Web Token (JWT) Profile for OAuth 2.0 Access Tokens", RFC 9068, DOI 10.17487/RFC9068, October 2021,
<<https://www.rfc-editor.org/info/rfc9068>>.