Identity Chaining Across Trust Domains

IETF 118 OAuth Working Group Meeting

Arndt Schwenkschuster (Microsoft)
Pieter Kasselman (Microsoft)
Kelley Burgin, (MITRE)
Mike Jenkins (NSA-CSS)
Brian Campbell (Ping)


LÁVKA

Agenda

- The challenge of Identity Chaining
- A (proposed) approach
- What’s in the draft
- Next Steps

Why Identity Chaining Across Trust Domains

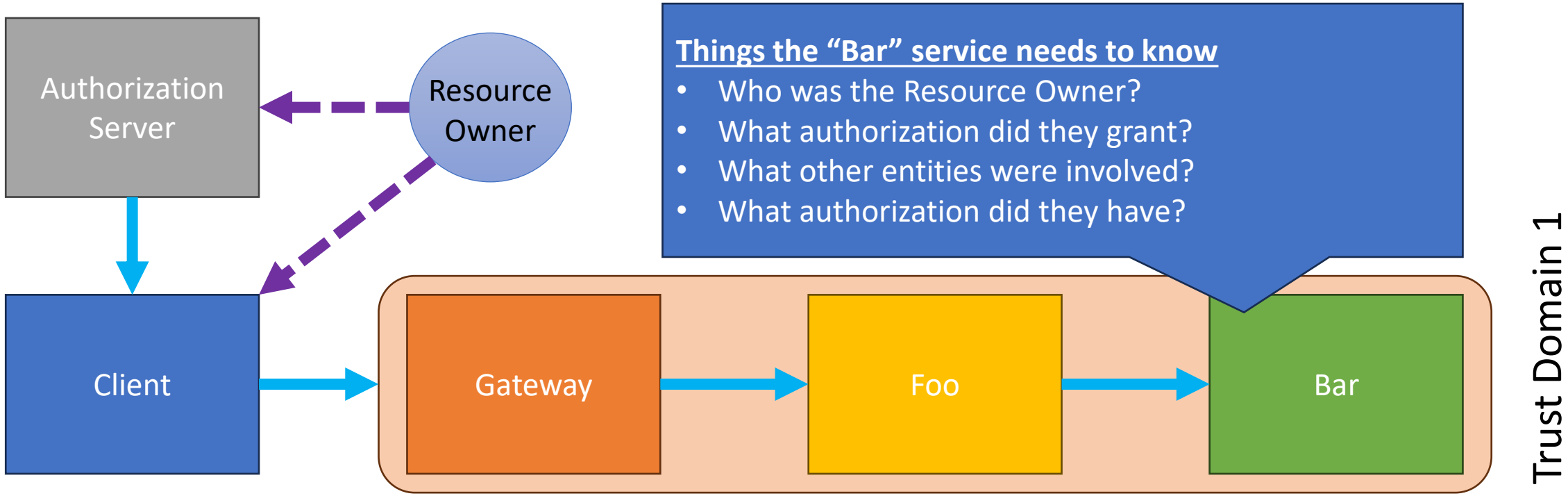


Image Courtesy Justin Richer (justin@bspk.io)

Why Identity Chaining Across Trust Domains

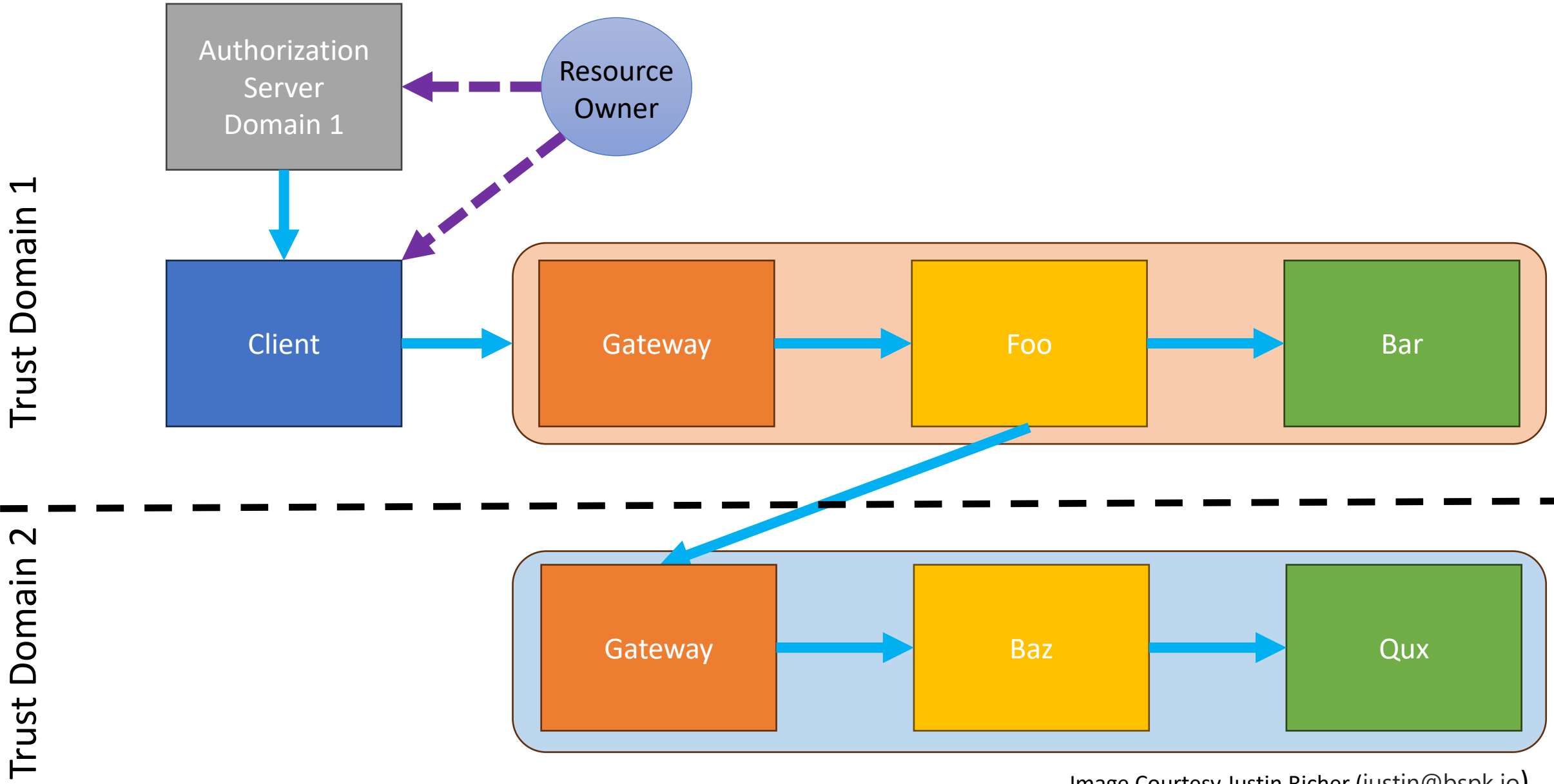


Image Courtesy Justin Richer (justin@bspk.io)

Why Identity Chaining Across Trust Domains

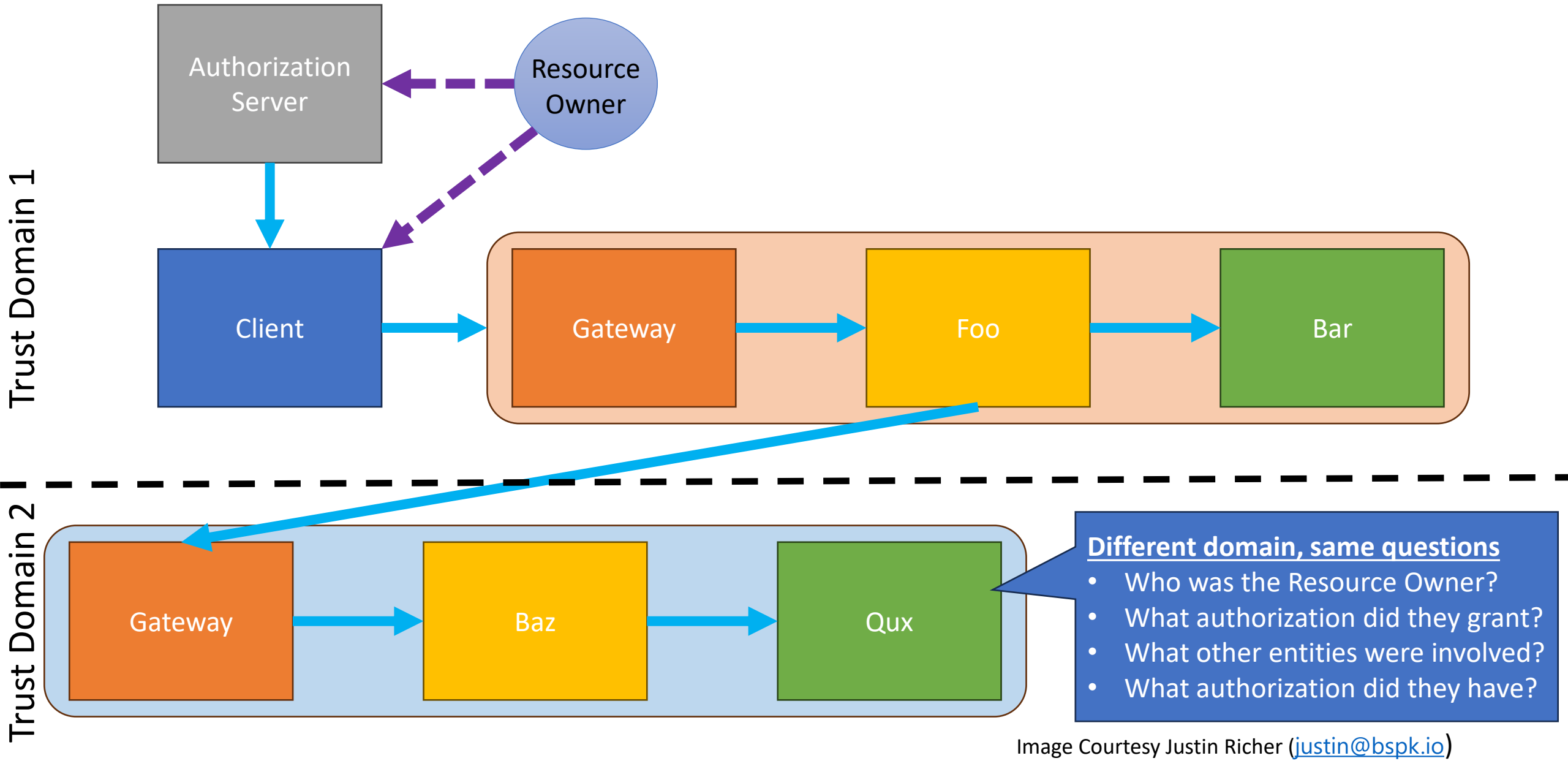


Image Courtesy Justin Richer (justin@bspk.io)

Why Identity Chaining Across Trust Domains

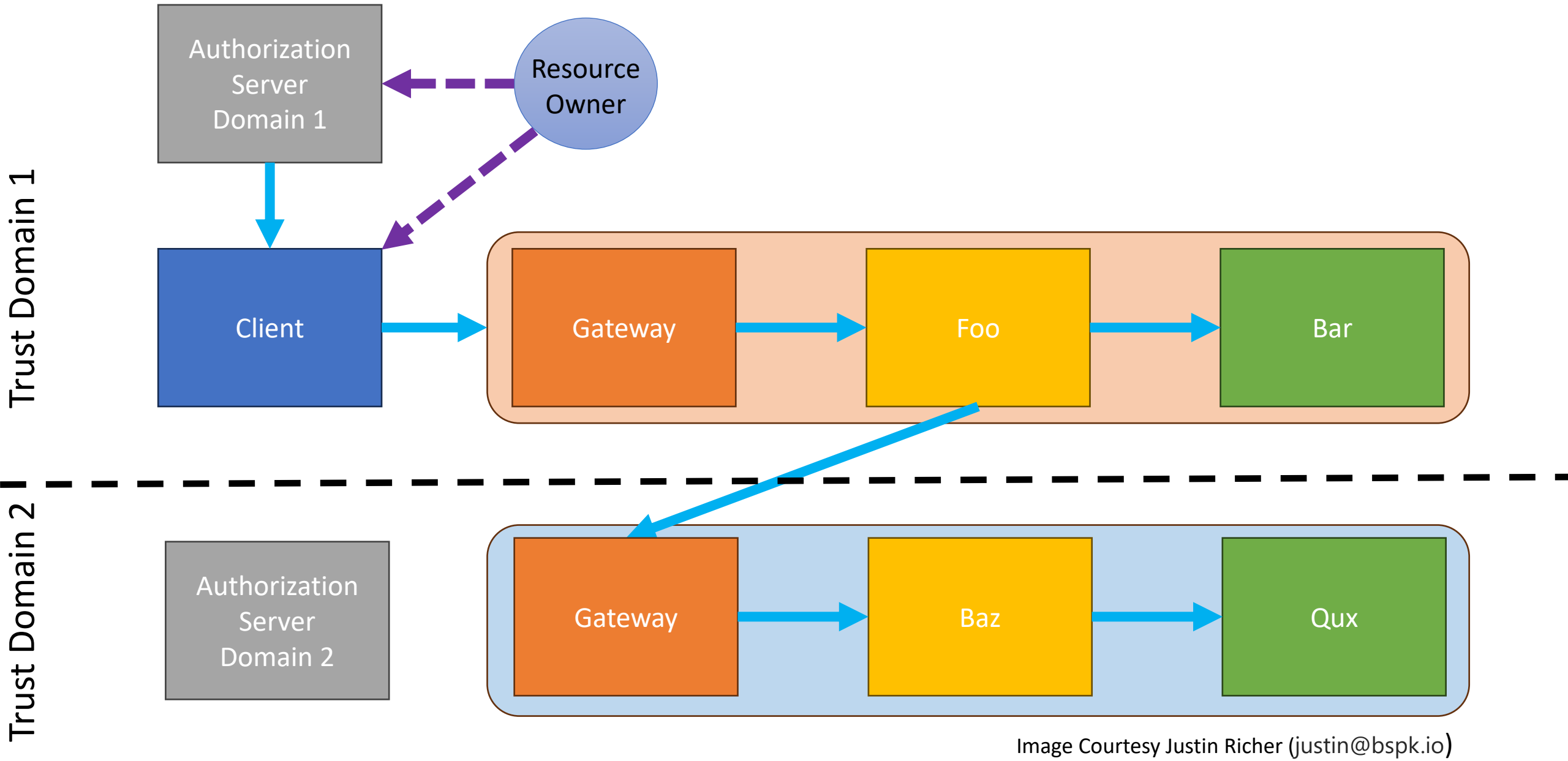
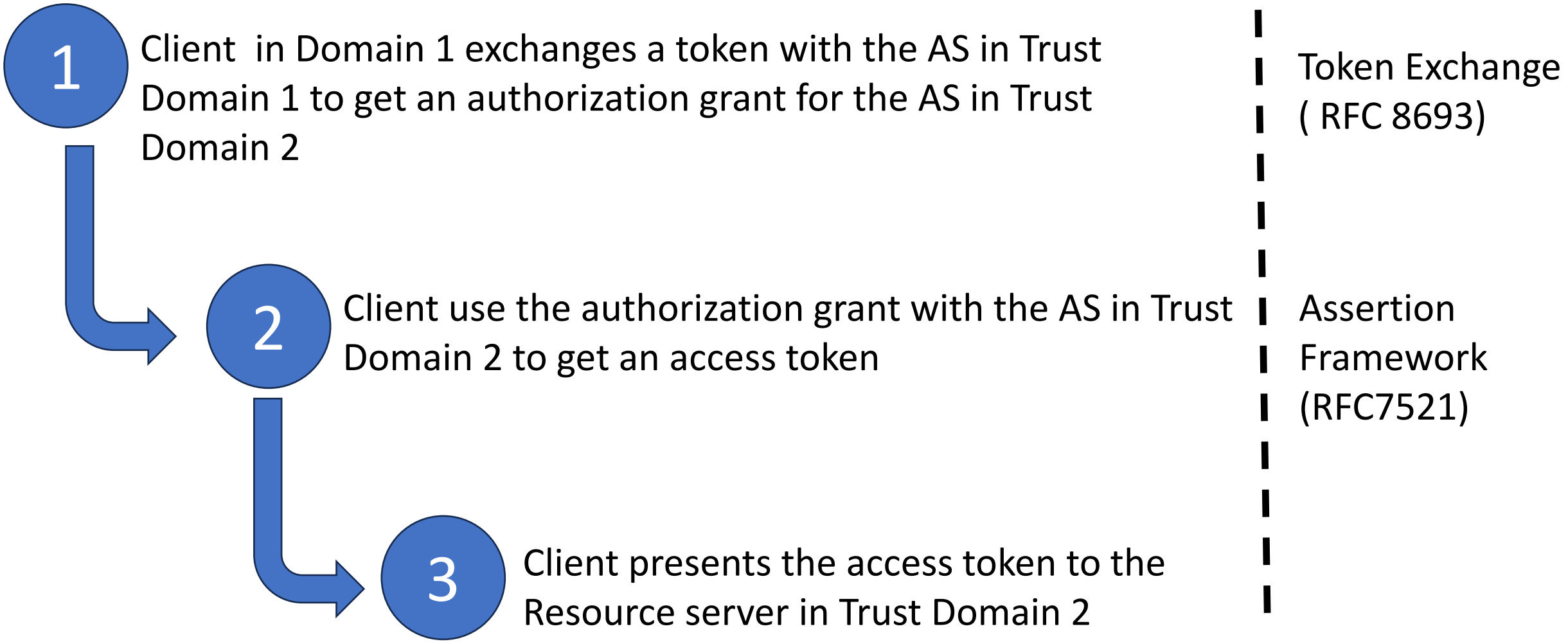


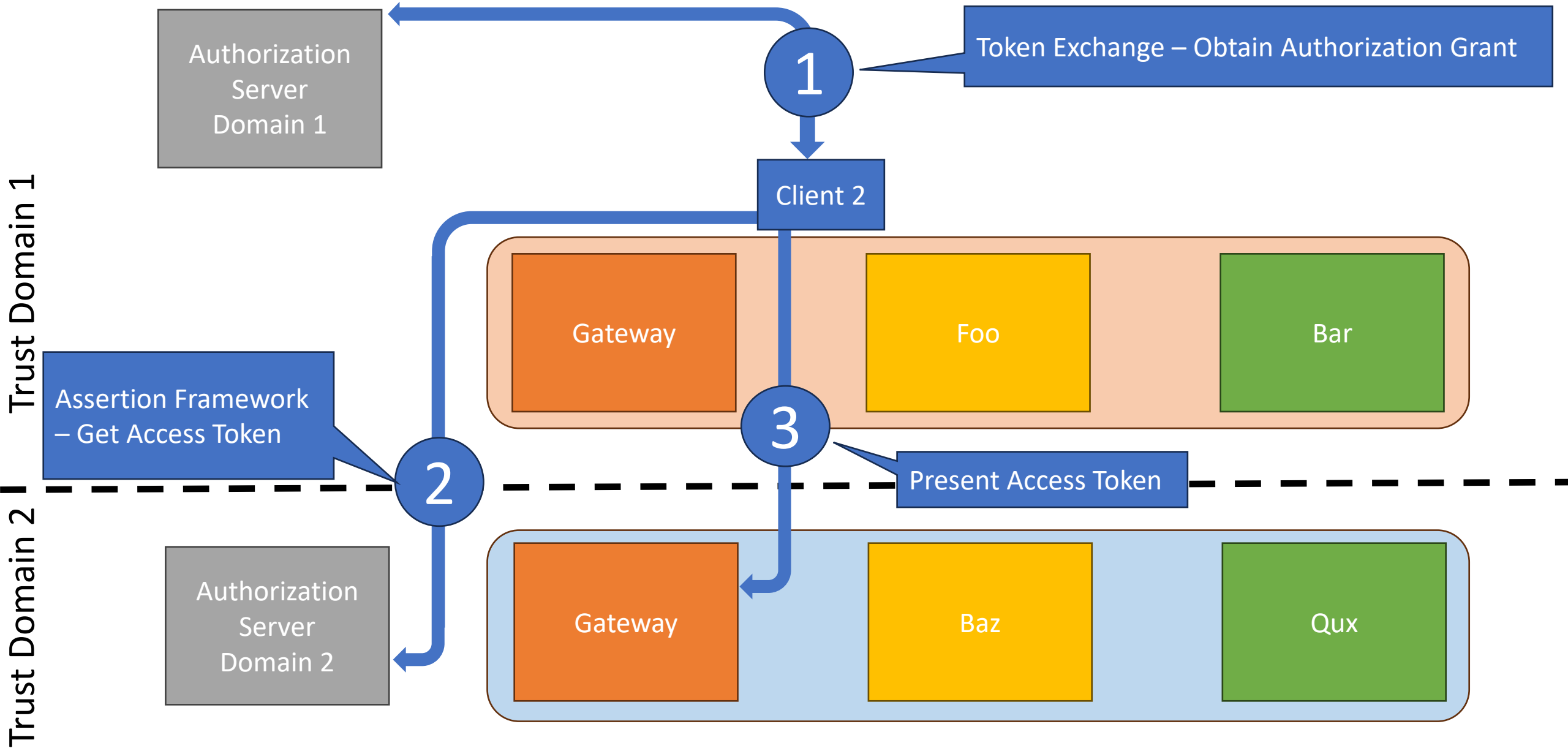
Image Courtesy Justin Richer (justin@bspk.io)

Proposal Concepts

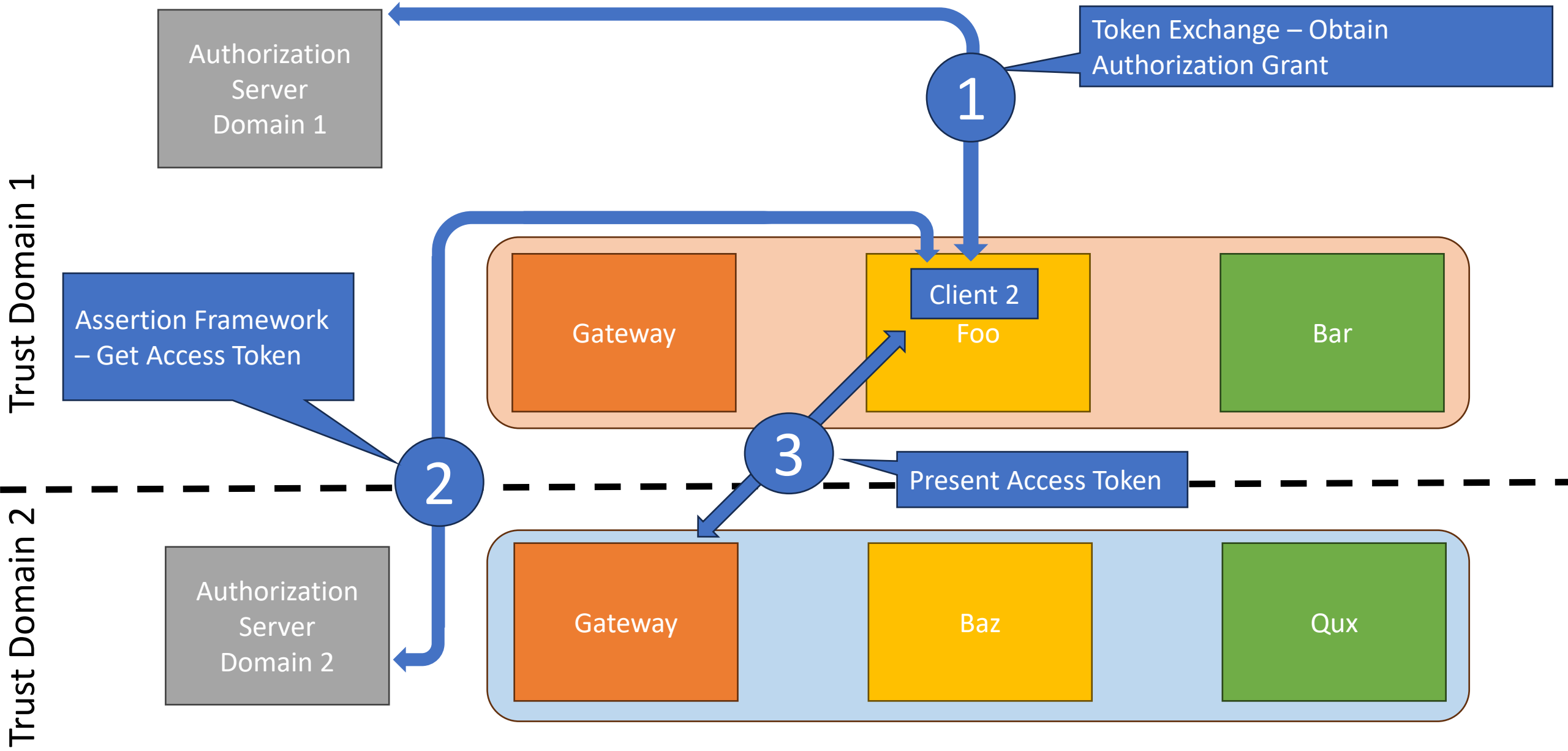
Getting an Authorization Grant for another Trust Domain



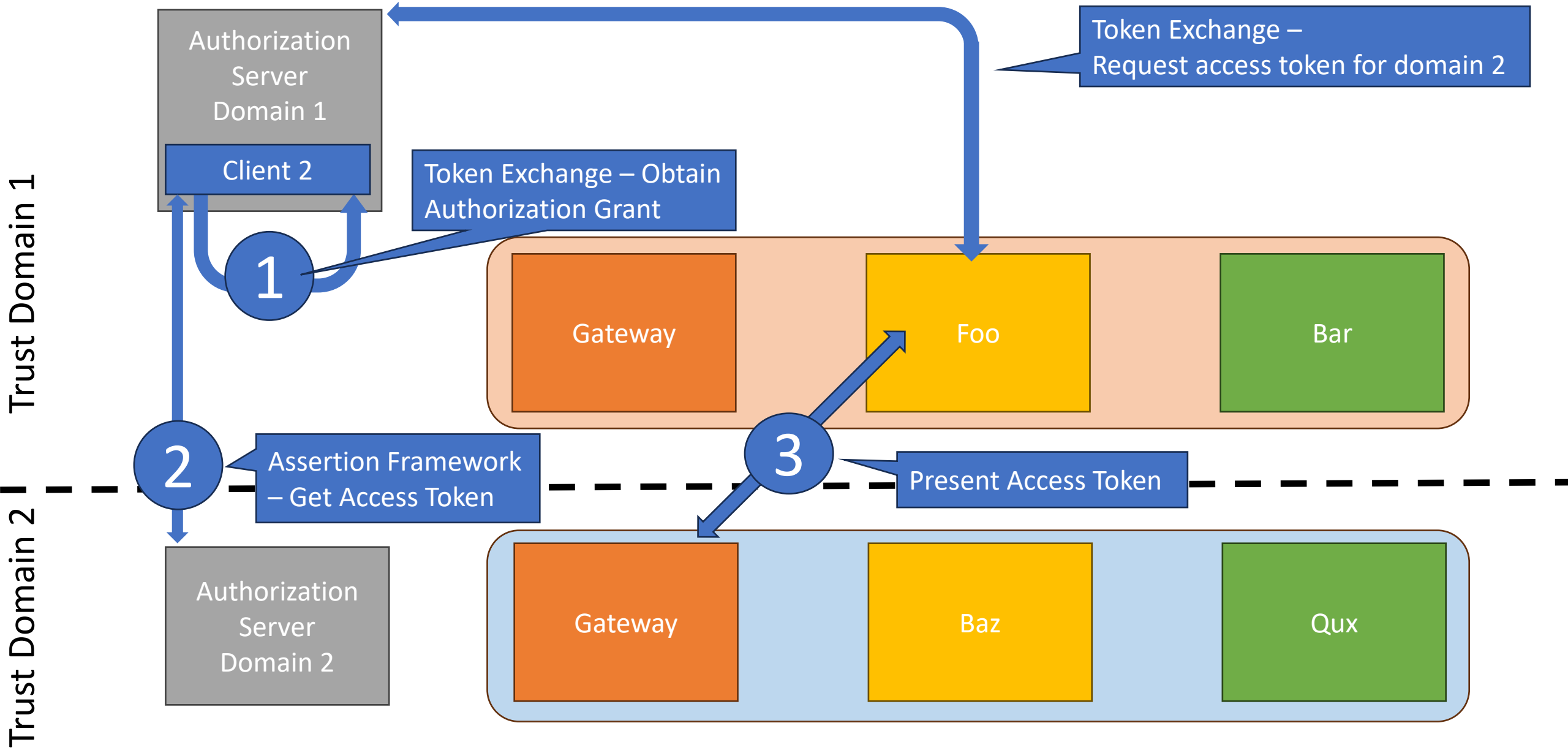
Generic Cross-Domain Identity Chaining



Resource Server as Client



Authorization Server as Client



What’s in the draft

https://datatracker.ietf.org/doc/draft-identity-chaining/

Section 2.2
Generic Cross-Domain Identity Chaining

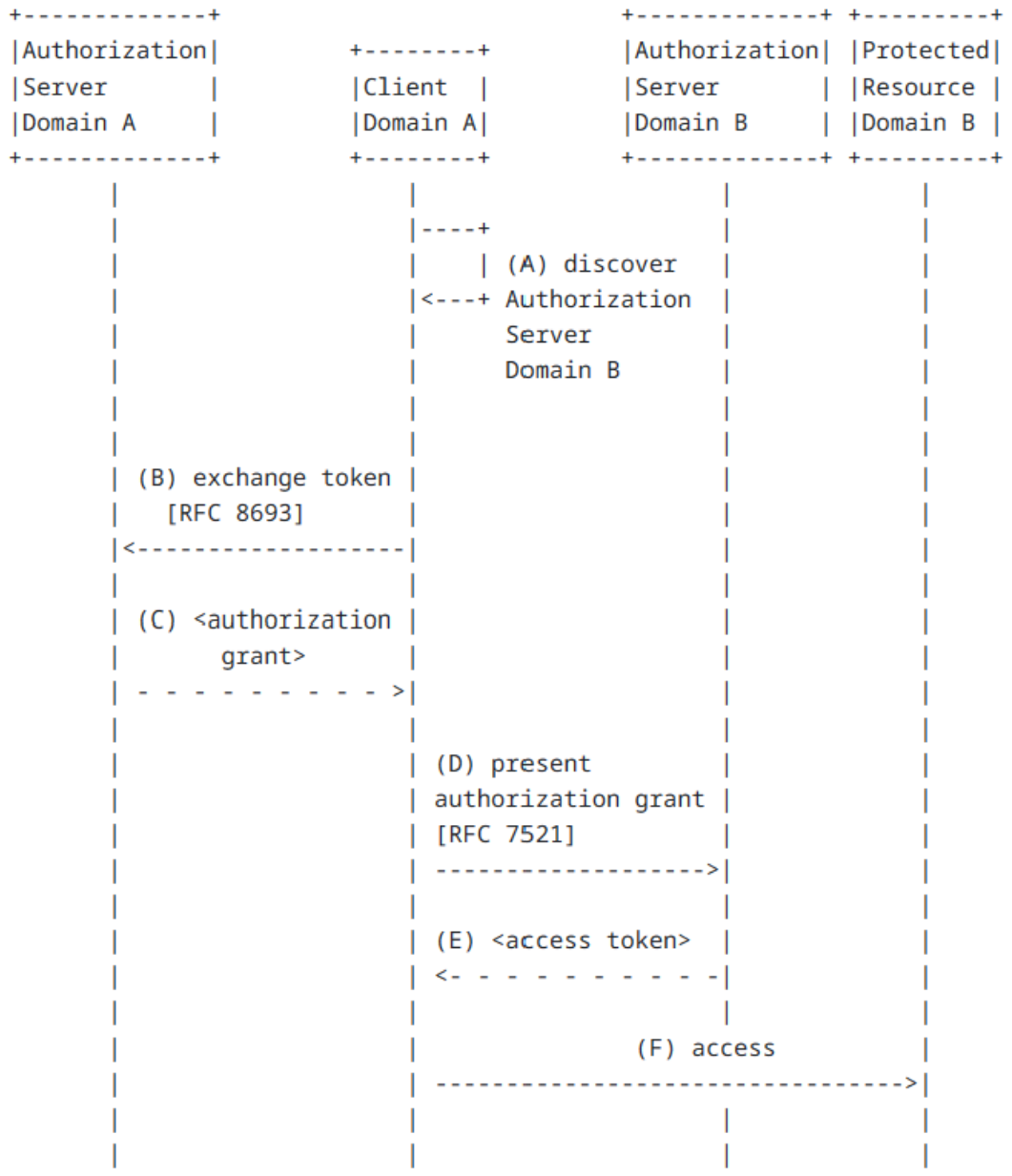


Figure 1: Identity Chaining Flow

Appendix A.1 Resource Server acting as Client

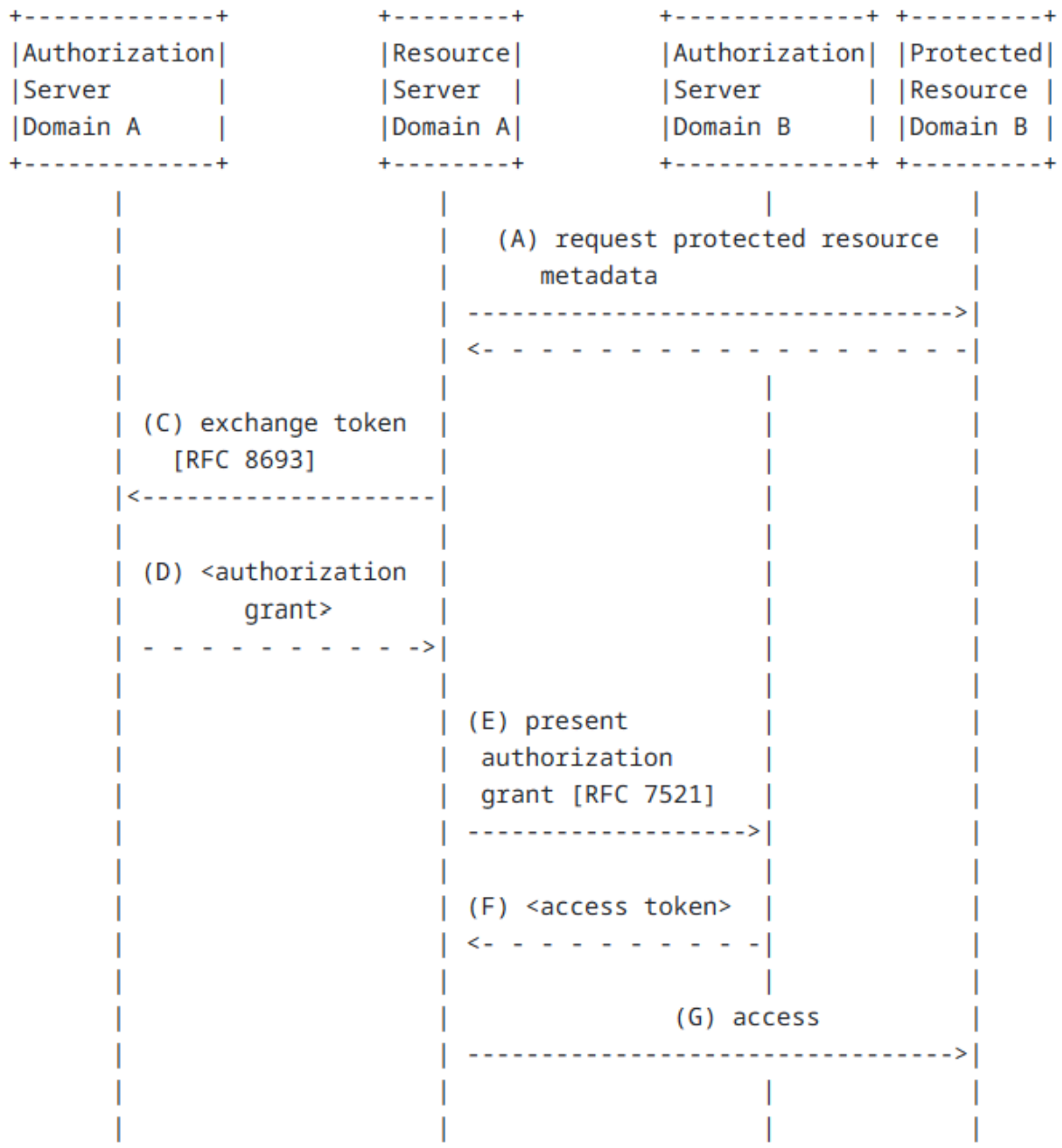


Figure 6: Resource server acting as client

Appendix A.2 Authorization Server acting as Client

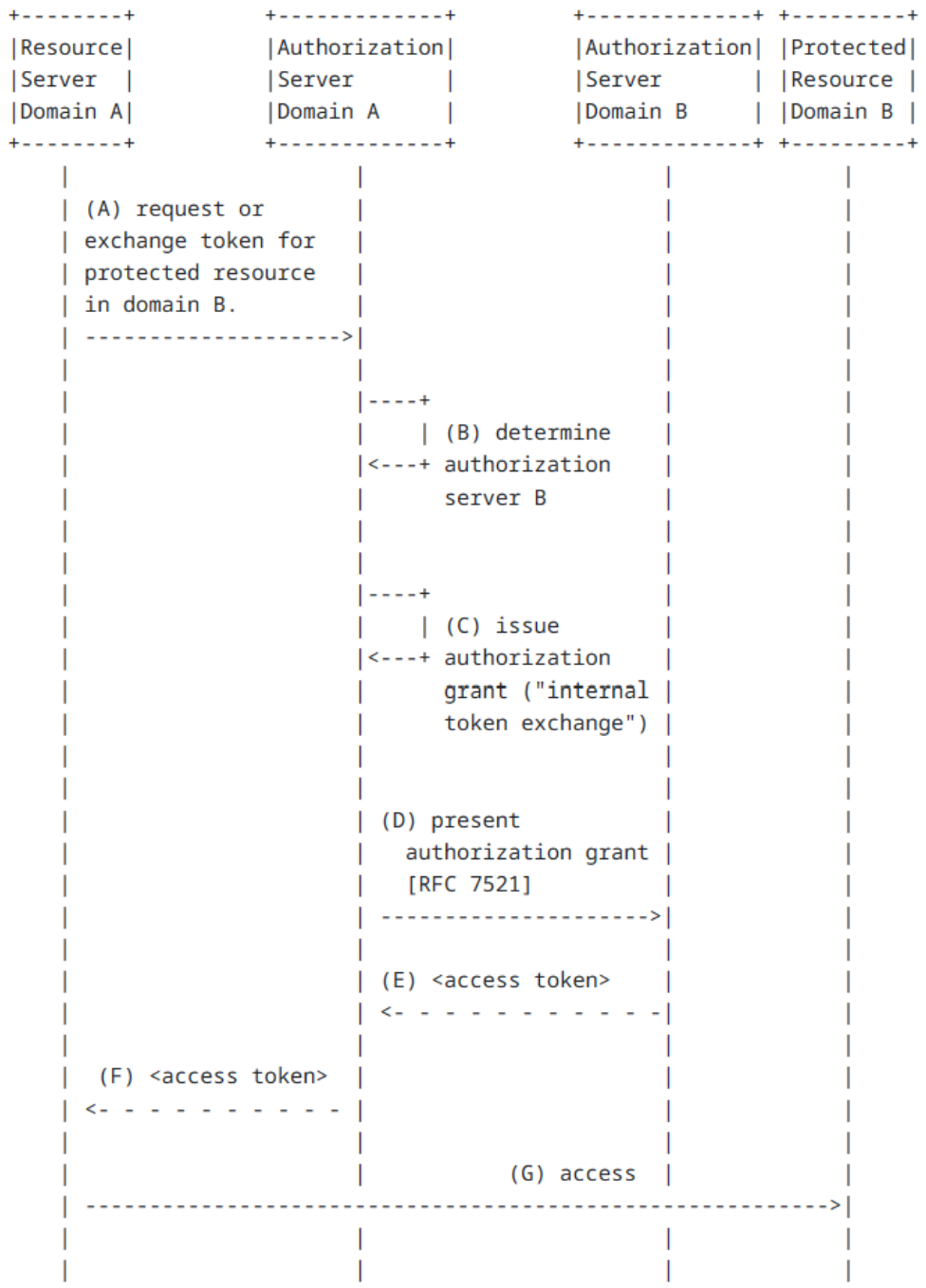


Figure 7: Authorization server acting as client

Token Exchange Profile

2.4. Token Exchange

The client performs token exchange as defined in [RFC8693] with the authorization server for its own domain (e.g., Domain A) in order to obtain an authorization grant that can be used with the authorization server of a different domain (e.g., Domain B) as specified in section 1.3 of [RFC6749].

2.4.1. Request

The parameters described in section 2.1 of [RFC8693] apply here with the following restrictions:

requested_token_type
OPTIONAL according to [RFC8693]. In the context of this specification this parameter SHOULD NOT be used. See Authorization grant type (Section 2.4.3).

Token type agnostic

Open Question:

Should this be restricted to JWT?

Assertion Flow Profile

2.5. Authorization Grant

The client presents the authorization grant it received from the authorization server in its own domain and presents it to the authorization server in the domain of the resources server it wants to access as defined in the "Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants" [RFC7521].

2.5.1. Request

If the authorization grant is in the form of a JWT bearer token, the client SHOULD use the "JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants" as defined in [RFC7523]. Otherwise, the client SHOULD request an access token using the "Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants" as defined in [RFC7521] (Section 4.1). For the purpose of this specification the following descriptions apply:

grant_type
: REQUIRED. In context of this specification clients SHOULD use the type identifier returned by the token exchange (issued_token_type response). See authorization grant type (Section 2.4.3) for more details.

assertion
: REQUIRED. Authorization grant returned by the token exchange (access_token response).

scope
: OPTIONAL.

The client MAY indicate the audience it is trying to access through the scope parameter or the resource parameter defined in [RFC8707].

Claims Transcription

2.6. Claims transcription

Authorization servers MAY transcribe claims when either producing authorization grants in the token exchange flow or access tokens in the assertion flow.

* *Transcribing the subject identifier*: Subject identifier can differ between the parties involved. For instance: A user is known at domain A by "johndoe@a.org" but in domain B by "doe.john@b.org". The mapping from one identifier to the other MAY either happen in the token exchange step and the updated identifer is reflected in returned authorization grant or in the assertion step where the updated identifier would be reflected in the access token. To support this both authorization servers MAY add, change or remove claims as described above.

* *Selective disclosure*: Authorization servers MAY remove or hide certain claims due to privacy requirements or reduced trust towards the targeting trust domain. To hide and enclose claims [I-D.ietf-oauth-selective-disclosure-jwt] MAY be used.

* *Controlling scope*: Clients MAY use the scope parameter to control transcribed claims (e.g. downscoping). Authorization Servers SHOULD verify that requested scopes are not higher priveleged than the scopes of presented subject_token.

* *Including authorization grant claims*: The authorization server performing the assertion flow MAY leverage claims from the presented authorization grant and include them in the returned access token. The populated claims SHOULD be namespaced or validated to prevent the injection of invalid claims.

Controlled by Authorization Servers

1. Subject identifier change
2. Selective disclosure
3. Controlling scope/down-scoping
4. Preserving claims

Open Question:

Should we define how the claims are transcribed?

Changes since IETF 117

- Update docname to draft-schwenkschuster-oauth-identity-chaining-00
- Editorial: Remove repetitive text
- Replace cURL commands with "on-the-wire" examples
- Add correct reference for RFC 7523
- Clarify requirements for "aud" claim
- Update Acknowledgements
- Correct/Update Authorization Server Discovery

Next steps

Open Issues

Scope

- Consider limiting token formats to JWT
- How to transcribe claims

https://github.com/arndt-s/ietf-identity-chaining/issues

Next Steps

- Interest in the WG to pursue this work?

Questions?