Liberty ID-SAFE MRD - Marketing Requirements Document for Strong Authentication

Version: 1.0

Editors:

Shelagh Callahan, Intel
Eric Malville, France Télécom
Rob Marano, Falcon Systems LLC/Individual Sponsor
Dave Nadig, Wave Systems Corp.
Sreeram Thirukkonda, Fidelity Investments

Contributors:

Shin Adachi, NTT
Alvaro Armentaros, Telefónica Móviles
Siddharth Bajajm, Verisign
Ingo Friese, Deutsche Telecon - T-Systems
Dietmar Krueger, Deutsche Telecon - T-Systems
Rob Lockhart, IEEE-ISTO
Dave Nadig, Wave Systems Corp.
Kapil Sachdeva, Gemalto
Uppili Srinivasan, Oracle
Kenji Takahashi, NTT

Abstract:

This MRD describes real-life usage of strong authentication that the Liberty Alliance Project should address. This shall be done broadly for all applicable environments requiring strong authentication.

Filename: liberty-id-safe-mrd-v1.0.pdf

This Market Requirements Document (MRD) has been developed by the Business and Marketing Expert Group of Liberty Alliance to capture the business requirements for an identity governance framework. Liberty Alliance is making this MRD publicly available to the industry at large for review and consideration. In addition, this MRD is being provided to the appropriate technical standards development group within Liberty Alliance for consideration of new technical work to address the requirements identified herein. This publication does not constitute a commitment by Liberty Alliance, explicit or implied, to develop technical specifications in full compliance with the requirements herein, now or in the future.

Table of Contents

1 Introduction

The goal of this MRD is to describe real-life usage of strong authentication that the Liberty Specification should address. This shall be done broadly for all applicable environments requiring strong authentication.

In order to tackle this goal, a two-phased approach has been used.

In the first phase, which is the objective of this MRD, the real-life scenarios relating to use of Strong Authentication are described. This phase does not include any use cases or requirements for interoperability related to lifecycle-supporting activities for Strong Authentication. It is strongly recommended that reviewers of this MRD first familiarize themselves with Liberty's Strong Authentication Expert Group's (SAEG) taxonomy as given in the [Taxonomy] section at the end of this MRD.

2 Foundational Scenarios

2.1 Background

It is assumed that the reader is familiar with the [Taxonomy] used here.

Building on that basic assumption, here is an attempt to differentiate between the functional roles of Authentication Delegate (AD) and Validation Delegate (VD). It must be stressed that the terms signify functional roles. No assumptions are made as to the physical implementation or location of these entities. They may be implemented in one logical entity or may be located in separate physical systems provided by separate business entities. Thus, the definition of their requirements and interactions will always assume the general case of complete separation.

Topic	**Authentication Delegate**	**Validation Delegate**
Identity Mapping	Maps authentication mechanisms and assurance levels to user identities. Owns direct Identity relationship with the Principal.	Maps one or more Credential IDs per mechanism to pseudonyms provided by requesting AD. Is not able to directly relate the user identity to the credential.
Authentication Mechanism Discovery	Can discover additional VD or AD which can validate the Principal. Publishes discovery meta data for use by SP.	Does not perform additional discovery functions. However, publishes discovery meta data for use by AD.
Authentication coordination	Coordinates the authentication on behalf of the Principal (i.e., maps required assurance level to assigned assurance level and helps the user select the appropriate authentication mechanism).	Does not perform coordination function.
Credential Lifecycle management	Helps coordinate the lifecycle management of the credentials (i.e., provision, activate, deactivate and de-provision).	Responds to lifecycle management requests from the AD. However, it relies on the AD for coordination.

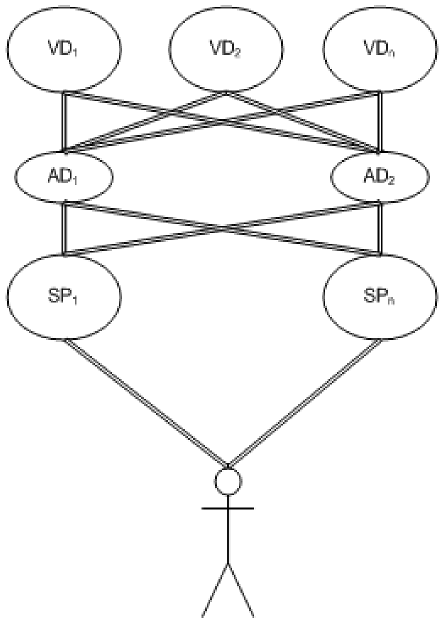
Credential Collection	Yes.	Yes, in selected mechanisms.
Credential Validation	No.	Yes.

2.2 Foundational Scenario 1

Foundational scenario 1 is intended to describe the base level of interactions between service providers (SP), Authentication Delegate (AD) and Validation Delegate (VD). The basic characteristics of foundational scenario 1 are:

1. Each service provider has only one authentication delegate per user identity per authentication mechanism;
2. Multiple service providers can share one AD. However, there is no need to orchestrate distributed sessions unless the service providers share a single sign-on relationship via the AD;
3. An AD always has a maximum of one VD per authentication mechanism, using which, the corresponding authentication factors can be validated.

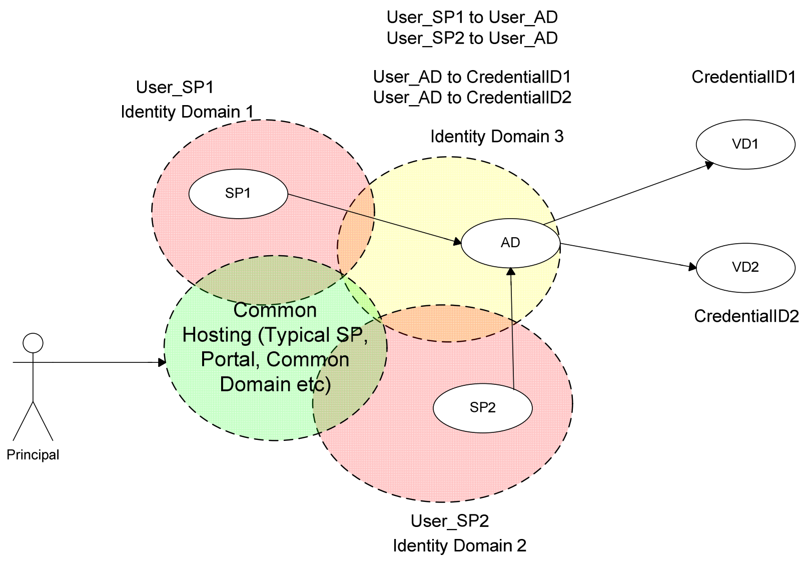
The most general representation of these constraints is shown below:



In this context, it is important to recognize two high level use cases that arise from interoperable strong authentication: Single Sign-On and Credential Portability.

2.2.1 Single Sign-On

The single sign-on use cases discuss the use of authentication factors in the context of single sign-on relationships between service providers, authentication delegates and validation delegates. The SSO model can be depicted as follows:

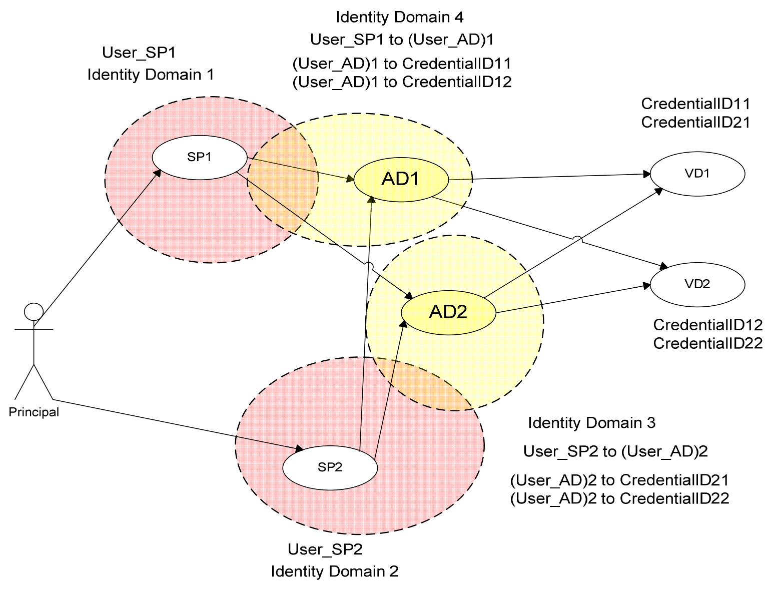


The single sign-on use cases also illustrate potential scenarios for re-use of authentication resultants. In a typical SSO use case in foundational scenario 1, different service providers share the same authentication delegate. This authentication delegate can validate different types of credentials by establishing relationships with multiple validation delegates for different types of credentials. By virtue of business agreements with the authentication delegate, the service provider now has access to established assurance levels which can be set for different assets under its control.

In addition, one service provider in the network may act as the portal and host access to assets that belong to different service providers. Hence, the base level of authentication can be shared and used as a basis for stronger authentication.

2.2.2 Credential Portability

The credential portability use cases discuss the reuse of credentials such as tokens, certificates, and other factors without sharing the logical process (e.g., authentication) itself. This does not appear to carry direct implications for orchestrating authentication, step-down, or logout decisions. The credential portability scenario can be depicted as follows:



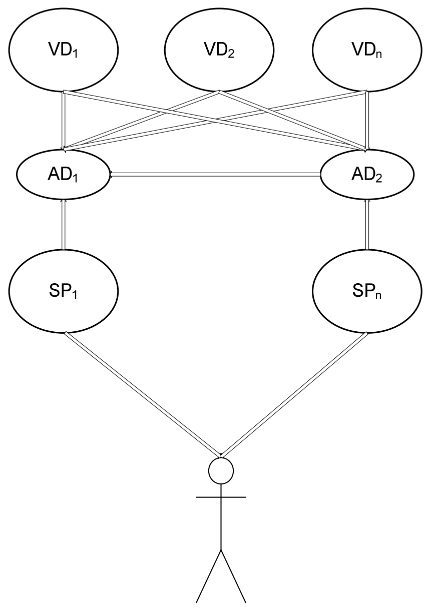
In the flow above, it is assumed that the user identities at each service provider are not related to each other. Hence, an access to a protected resource at each service provider would be managed by separate challenges even if the same entities (AD, VD1, and VD2) are involved in performing this action.

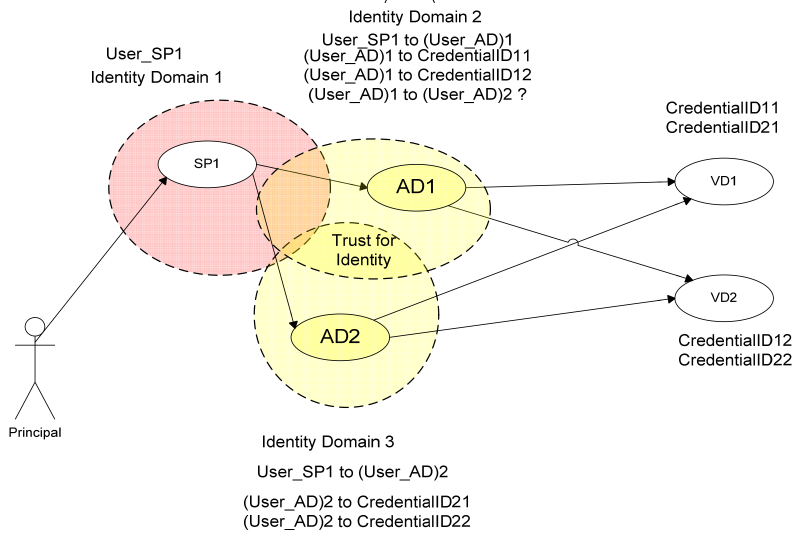
Special Cases include:

1. CredentialID11 = CredentialID12 and CredentialID21=CredentialID22
2. In addition to 1, above, (User_AD)1 = (User_AD)2

2.3 Foundational Scenario 2

Foundational Scenario 2 is a superset of functions to those in Foundational Scenario 1. In addition to FS1, this scenario captures Circle of Trust relationships between authentication delegates. This can be represented in a general case as follows:



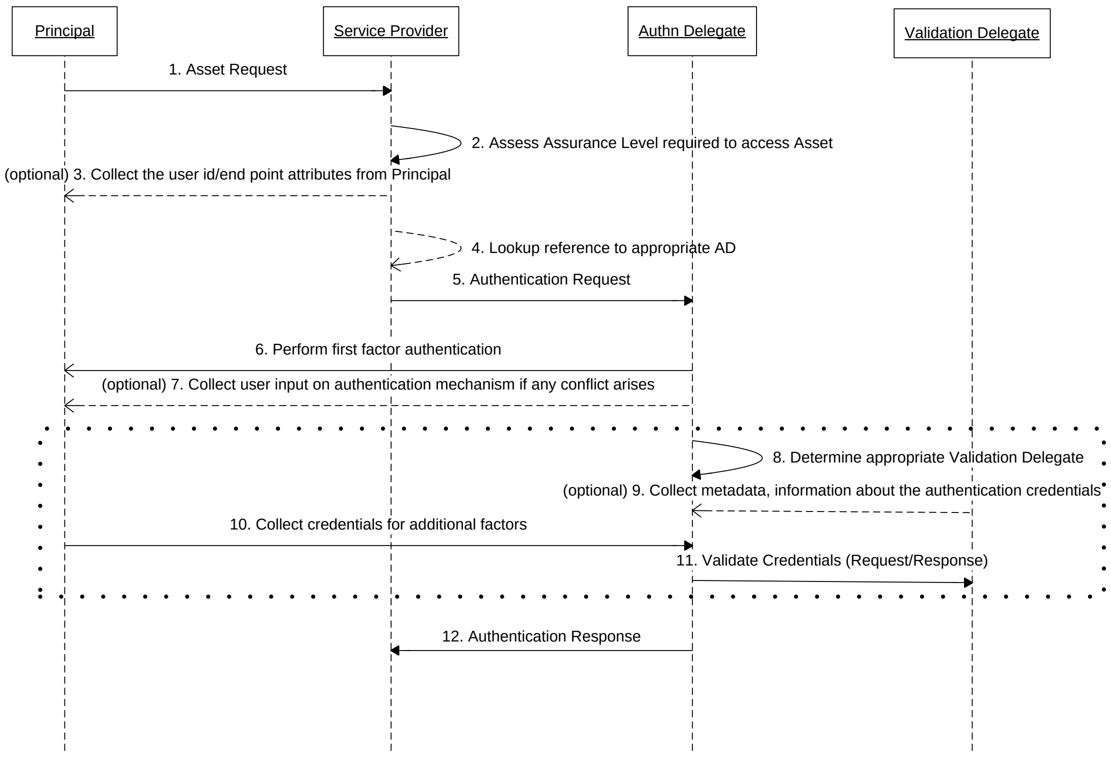


The above figure indicates how Authentication Delegates might share Circle of Trust relationships which enable one authentication delegate to perform higher factor upgrades based on previously performed authentications from a different delegate.

3 Use Cases: Foundational Scenario 1

3.1 Unauthenticated Principal, Credential Collection by AD

3.1.1 Graphical Description


Principal
Service Provider
Authn Delegate
Validation Delegate
1. Asset Request
2. Assess Assurance Level required to access Asset
(optional) 3. Collect the user id/end point attributes from Principal
4. Lookup reference to appropriate AD
5. Authentication Request
6. Perform first factor authentication
(optional) 7. Collect user input on authentication mechanism if any conflict arises
8. Determine appropriate Validation Delegate
(optional) 9. Collect metadata, information about the authentication credentials
10. Collect credentials for additional factors
11. Validate Credentials (Request/Response)
12. Authentication Response

Denotes Operations that can be repeated over multiple VDs

3.1.2 Description

Title/ID	Unauthenticated Principal, Service Provider Interaction
Pre conditions	1. Principal does not possess any session state (or) authentication resultants corresponding to a currently active session[1]. 2. All data elements required to execute the use case have been successfully provisioned at the different entities/constituents.
Constituents	Principal, Service Provider, Authentication Delegate
Use Case	1. Principal requests access to desired asset from Service Provider

[1] Principal may possess expired resultants which could be used to obtain user identity.

	2. Service Provider compares the current assurance level of the request (non-existent since there is not an existing Principal authentication for that session) with the required assurance level for the requested asset. 3. (Optional) The Service Provider optionally collects endpoint (e.g., device) information, user identity information prior to creating Authentication Request. a. The Service Provider uses the reference to the Authentication Delegate that can match the required assurance level for the asset in addition to any available optional endpoint attributes, OR b. Based on preliminary examination of endpoint attributes, Service Provider may reject the request (e.g., an internet café does not provide the required assurance level for a certain asset). 4. Service Provider looks up reference: a. If the Service Provider can look up a valid reference to a valid Authentication Delegate, it creates a request with the required attributes, OR b. Service Provider cannot find an appropriate Authentication Delegate that matches required the assurance level (or) attributes and returns an error. 5. Service Provider forwards the Principal over to the Authentication Delegate along with the Authentication Request. 6. Authentication Delegate performs base level authentication. In case the user does not possess first factor authentication, the flow moves to Step 7 where the user is prompted for alternate mechanisms or combination of mechanisms. (Refer to the flexible authentication use case at Section 3.1.5 (Mapping to other business cases).) 7. (Optional) If multiple authentication mechanisms exist for the user, input from the user resolves conflicts between multiple mechanisms. 8. Authentication Delegate looks up reference: a. If the AD can look up a valid reference to an appropriate VD, It creates an request with the required attributes, OR b. AD cannot find an appropriate VD that matches the required assurance level (or) attributes and returns an error. 9. (Optional) Once the additional authentication mechanisms are known, the Authentication Delegate can optionally collect metadata describing the mechanisms by interacting with the corresponding validation delegates.

	10. The AD collects all credentials required for the secondary mechanisms and cycles through the appropriate VD for the mechanisms. 11. The AD maps the credential ID for the user and generates a validation request for the Principal to be sent to the VD. 12. Upon successful completion of validation requests, the AD creates the authentication context and returns the resultant to the SP. The resultant contains the context (either by reference or by value) for all authentications performed for this Principal. The SP in turn may issue local cookies/references to sessions for subsequent intra-session access.
Post conditions	Principal is successfully authenticated for all authentication factors. Steps 8-11 above can be repeated for multiple authentication factors
Out of band / error conditions	1. Principal requests invalid asset and the request is rejected. 2. Service Provider cannot contact appropriate Authentication Delegate and returns error. 3. AD cannot refer to any VD and returns error. 4. Authentication fails for one or more factors and failure is returned to the Principal.

226

227 ### 3.1.3 Data Elements

Constituent	Data Element	Function	Description
Principal	Endpoint Attributes	Manage	The end user/terminal may possess attributes which are required by the Service Provider. These may be collected automatically by the Service Provider.
Service Provider	Asset	Manage	Assets are owned and managed entirely by the Service Provider.
	Assurance Level for given asset	Manage	Assigns and maintains (upgrades, downgrades) assurance levels for assets.
	Reference to the AD(s)	Use	This is a static reference to the Authentication Delegate Location/Address.
	AD attributes per reference	Use	Service Provider needs to be aware of user endpoint attributes and assurance levels that are supported by each AD in order to map a request to the appropriate AD.

	Authentication Request	Create	Service Provider uses identity attributes, endpoint attributes, assurance levels to create a Request.
Authentica-tion Delegate	Mapping between user identity & appropriate mechanisms	Manage	These links are used as part of the Validation Request created by the AD. AD maps the user identity to one or more CredentialID.
	Endpoint attributes & supported assurance levels	Manage	Manages all supported endpoint attributes and assurance levels.
	CredentialIDs & identifiers for the Principal	Manage	These links are used as part of the Validation Request created by the AD. AD maps the user identity to one or more CredentialIDs.
	Validation Request	Create	AD creates the validation request and passes it to the appropriate VD.
	Authentication Response	Create	Creates response which contains the resultant. In cases where the credential collection is performed by the VD, this may also include the credential ID for use by the AD in credential mapping.
	Reference to the VD(s)	Manage	This is a static reference to the Validation Delegate Location/Address.
Validation Delegate	Validation Response	Create	The VD validates the credentials for appropriate factors and returns the response back to the AD.

3.1.4 Mapping to the Business Cases

3.1.4.1 Strong and Flexible AuthN Use Case

3.1.4.1.1 Background

Authentication is a very important process in order to make business, enable personalization, and privacy. Authentication can be of different qualities. Strong Authentication now has the notion of very high assurance level, but most services do not really need a very high assurance level. The basic mechanism of Strong AuthN, as described in the MRD, is the combination of two or more AuthN methods.

This use case aims on using the basic mechanisms of Strong AuthN, also, for lower and medium assurance level. This makes AuthN more flexible in terms of usable AuthN methods and achievable assurance level (Flexible and Strong AuthN).

3.1.4.1.2 Summary

Alice usually accesses SP by using AuthN method A1 (e.g., SIM card in her IMS handheld device). A1 is supposed as a rather strong AuthN.

Now Alice wants to access SP in a situation where A1 is not available (e.g., other device, other location). Alice is able to use AuthN method A2 or A3. Both of them are rather weak AuthN methods (e.g., IP-address authentication (in fixed network) or login/Password).

Alone, none of them is sufficient for SP since they are not as strong as A1.

Alice gets access by combining them, first A2 and the step-up with A3.

3.1.4.1.3 Pre-condition

Alice has several credentials or is able to provide different AuthN methods.

A1 is for higher assurance levels.

A2 and A3 is for rather lower assurance levels.

3.1.4.1.4 Description

1. Alice requests access to SP.
2. SP requests IDP to authenticate Alice with certain assurance level.
3. IdP asks Alice to go for AuthN A1 to fulfill assurance level.
4. Alice indicates that she cannot provide AuthN A1 but A2 and A3 instead (A2 and A3 alone not matching the requested assurance level, but the combination does).
5. IdP authenticates Alice using A2 and A3.
6. Alice accesses SP.

For a visual example of these, please see the associated slide set "Use Case for Flexible Strong Authentication" [MRD.Strong.Auth.Slides].

3.1.5 Out of Scope

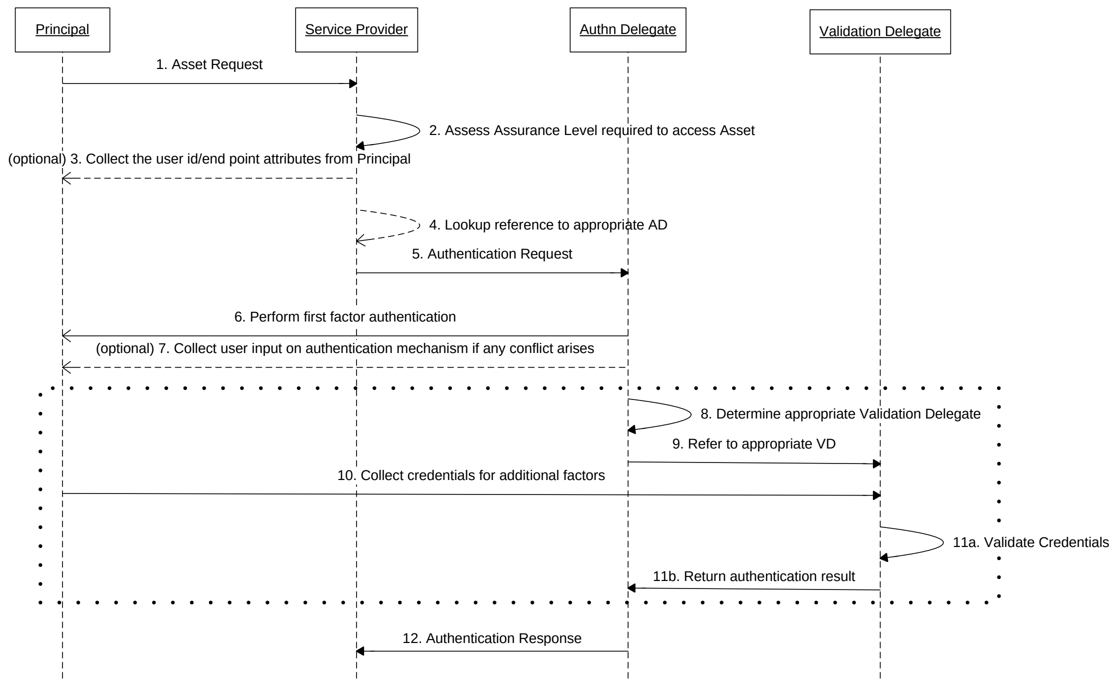
Error conditions presented to the user.

3.1.6 Dependencies with Other Use Cases

None

3.2 Unauthenticated Principal, Credential Collection by VD

3.2.1 Graphical Description


Principal
Service Provider
Authn Delegate
Validation Delegate
1. Asset Request
2. Assess Assurance Level required to access Asset
(optional) 3. Collect the user id/end point attributes from Principal
4. Lookup reference to appropriate AD
5. Authentication Request
6. Perform first factor authentication
(optional) 7. Collect user input on authentication mechanism if any conflict arises
8. Determine appropriate Validation Delegate
9. Refer to appropriate VD
10. Collect credentials for additional factors
11a. Validate Credentials
11b. Return authentication result
12. Authentication Response

Denotes Operations that can be repeated over multiple VDs

3.2.2 Description

Title/ID	Unauthenticated Principal, Credential collection by the VD
Pre conditions	Same as Use Case 3.1.
Constituents	Same as Use Case 3.1.
Use Case	Steps 1-8 are the same as in Use Case 3.1. 9. The AD performs the CredentialID mapping for the user and generates a validation request for the Principal to be sent to the VD. ***Please note:*** In this context, it is assumed the AD still retains the role to coordinate credential provisioning and management for the Principal with the VD. Hence, this role is still aware of all the other credentials (via credential ID only) available for a Principal. This is also used to redirect the Principal to the VD. 10. The VD collects all credentials required for its corresponding mechanism directly by interacting with the Principal. 11. The following two steps are implicit in validation:

	a. Credential Validation, and b. Upon successful completion of validation requests, the VD sends the results back to the AD. 12. The AD repeats steps 9-11 for all appropriate VDs and composes a result which is sent back to the Principal as proof of success.
Post conditions	Same as Use Case 3.1.
Out of band / error conditions	Same as Use Case 3.1.

3.2.3 Data Elements

Same as Use Case 3.1.

3.2.4 Mapping to BMEG Use Cases

Same as Use Case 3.1.

3.2.5 Out of scope

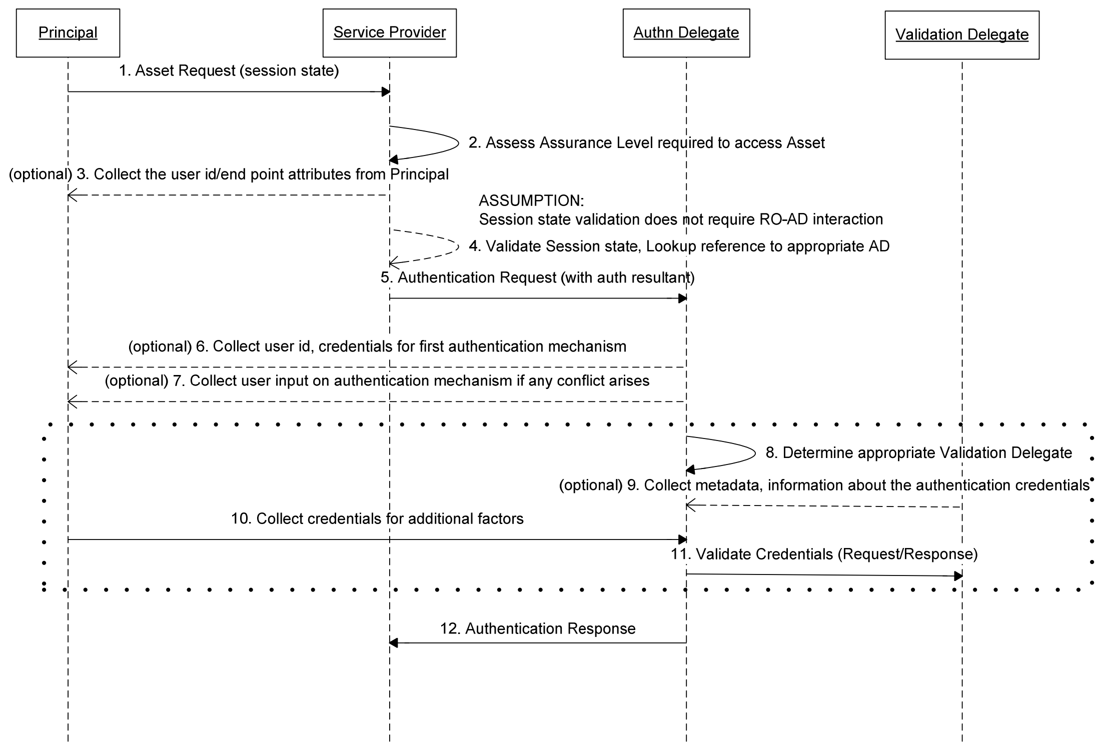
Same as Use Case 3.1.

3.2.6 Dependencies with Other Use Cases

Same as Use Case 3.1.

3.3 Authenticated Principal: System-Enforced Step-up Access

3.3.1 Graphical Description


Principal
Service Provider
Authn Delegate
Validation Delegate
1. Asset Request (session state)
2. Assess Assurance Level required to access Asset
(optional) 3. Collect the user id/end point attributes from Principal
ASSUMPTION:
Session state validation does not require RO-AD interaction
4. Validate Session state, Lookup reference to appropriate AD
5. Authentication Request (with auth resultant)
(optional) 6. Collect user id, credentials for first authentication mechanism
(optional) 7. Collect user input on authentication mechanism if any conflict arises
8. Determine appropriate Validation Delegate
(optional) 9. Collect metadata, information about the authentication credentials
10. Collect credentials for additional factors
11. Validate Credentials (Request/Response)
12. Authentication Response

Denotes Operations that can be repeated over multiple VDs

3.3.2 Assumptions

1. Active session in a steady-state where a resultant exists from prior authentication and is stored in a cache at the Service Provider.
2. Principal returns to access the same resource at the same SP.
3. Once a high assurance level is attained, the latest resultant replaces any previous resultants in the same context. However, no restrictions are imposed on the Principal possessing resultants of multiple assurance levels in different contexts.

300 3.3.3 Description

Title/ID	Authenticated Principal, Service Provider Interaction
Pre conditions	1. Principal possesses a reference to an active session state that could optionally contain resultants corresponding to that session[2]. 2. All data elements required to execute the use case have been successfully provisioned at the different entities/constituents.
Constituents	Same as Use Case 3.1.
Use Case	1. Principal requests Service Provider for access to the desired asset(s) using a previously authenticated context. 2. Service Provider compares the current assurance level of the request (as manifested in the current active session) with that required for the requested asset. 3. The Service Provider optionally collects endpoint information, user identity information prior to creating the Authentication Request. a. The Service Provider uses the reference to the Authentication Delegate that can match the required assurance level for the asset in addition to any available optional endpoint attributes. b. Based on preliminary examination of the endpoint attributes, Service Provider rejects request (e.g., an internet café does not provide the required assurance level for a certain asset). 4. Subsequently, the service provider validates the session assurance state and looks up a reference to the AD: a. If the Service Provider can look up a valid reference to a valid Authentication Delegate, It creates an authentication request which may optionally contain the previous authentication resultant, OR b. Service Provider cannot find an appropriate Authentication Delegate that matches the required assurance level (or) attributes and returns an error. 5. Service Provider forwards the Principal over to the Authentication Delegate along with the Request which now contains the Resultant.. 6. (Optional) As the resultant is already passed as part of the request, step 6 of Use Case 3.1 is optional in this use case and required only if mandated by policy. Steps 7-12 are captured in Use Case 3.1 and Use Case 3.2.

[2] Principal may possess expired resultants which could be used to obtain user identity.

Post conditions	Service Provider creates the request and forwards the Principal over to the Authentication Delegate.
Out of band / error conditions	1. Principal requests an invalid asset and the request is rejected. 2. Service Provider cannot contact an appropriate Authentication Delegate and returns an error. 3. The resultant is invalid and rejected by AD.

3.3.4 Data Elements

In addition to data elements captured in Use Case 3.1:

Constituent	Data Element	Function	Description
Principal	Session State	Use	Principal is able to provide a reference to an active session state.
Service Provider	Session State	Manage	Service Provider is responsible for creating a reference to a session state and expiring the reference per its policies.
	Session State x Resultant Mapping	Manage	The Service Provider is responsible for creating the session state and managing the mapping of session state to previous resultants. In a trivial case, the session state may itself contain the resultant.
Authentica-tion Delegate	User tracking at different service providers (i.e., SP and assurance level required for the session at the SP)	Manage	Authentication Delegate needs to track user authentication and the assurance level at which user is authenticated at different service providers from the perspective of sign-out.

3.3.5 Mapping to BMEG Use Cases

All relevant mappings captured in Use Case 3.1.

3.3.6 Out of Scope

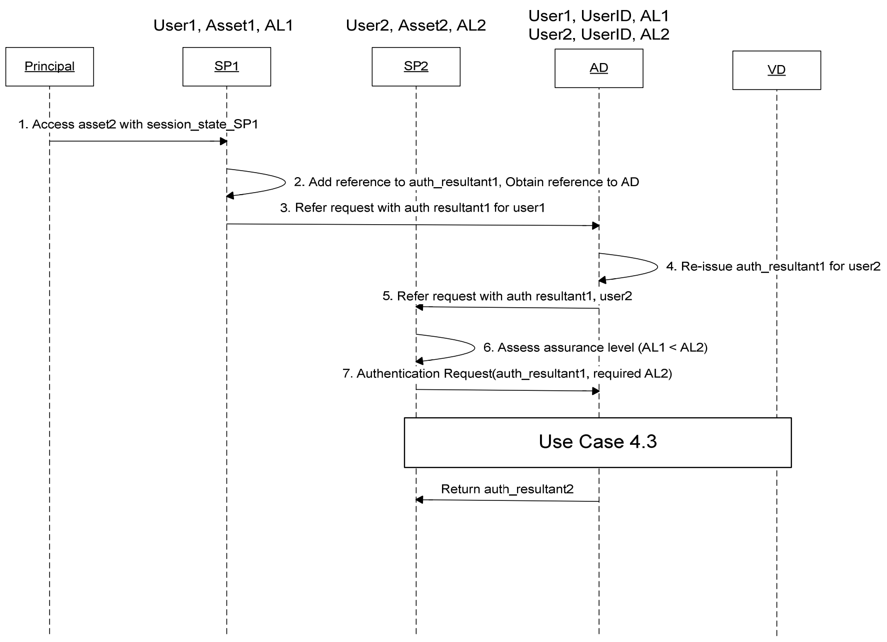
Error conditions presented to the user.

3.3.7 Dependencies with Other Use Cases

Same as Use Case 3.1.

3.4 Single Sign-On with Step-up Access

3.4.1 Graphical Description


User1, Asset1, AL1
User2, Asset2, AL2
User1, UserID, AL1
User2, UserID, AL2
Principal
SP1
SP2
AD
VD
1. Access asset2 with session_state_SP1
2. Add reference to auth_resultant1, Obtain reference to AD
3. Refer request with auth resultant1 for user1
4. Re-issue auth_resultant1 for user2
5. Refer request with auth resultant1, user2
6. Assess assurance level (AL1 < AL2)
7. Authentication Request(auth_resultant1, required AL2)
Use Case 4.3
Return auth_resultant2

3.4.2 Description

Constituents	Principal, SP1, SP2, AD, VD
Pre conditions	1. Principal has been previously allowed to access asset1 at SP1 at assurance level1 (AL1) and has a currently active session at SP1. 2. Asset2 is managed by SP2, but asset2 is also hosted at SP1 as a single sign-on resource. As part of this single sign-on, SP2 also knows or can derive the identity domain for SP1. 3. SP1 and SP2 have agreed to use AD in common. The AD can also perform conversion between user identities at the different SP using indicative data. 4. It is also assumed that in the most general case, AL1 < AL2, since AL1 ≥ AL2 would not trigger flows 7 and beyond in this use case. Also, if the assurance levels are shared between SP1 and SP2, AL1 would effectively be 0.

	5. This use case depends on Use Case 3.3 for incremental step-up access.
Use Case	1. Principal creates a request for asset2 which is hosted at SP1 or in a portal scenario at the AD or a common domain SP. 2. SP1 validates the session state and injects the auth_resultant1 (previously issued) or a suitable reference. 3. The Principal is referred to the authentication delegate along with the auth resultant and asset which is requested (i.e., asset2 and the identity domain in which asset2 is managed). 4. The AD re-issues the auth resultant while performing the conversion between user 1 and user 2. 5. The AD refers the request to SP2 along with requested asset2 and auth_resultant1 which is AL1 for user2. 6. SP2 assesses AL1 and determines it to be less than AL2. If AL1 is not known to SP2, it assumes the current assurance level to be 0. 7. SP2 then creates an authentication request with the current auth resultant, user identity, and required assurance level AL2. 8. This triggers execution of Use Case 3.3. 9. At the end of this use case, SP2 has an auth_resultant2 which has user2 for the user issues and is issued at AL2.
Post conditions	User now has access to asset2 at SP2.
Out of band / error conditions	Same as Use Case 3.1.

319

320 ### 3.4.3 Data Elements

Constituent	Data Element	Function	Description
Service Provider	Identity Domains for all hosted resources	Use	SP1 knows or is able to obtain the identity domain for SP2.

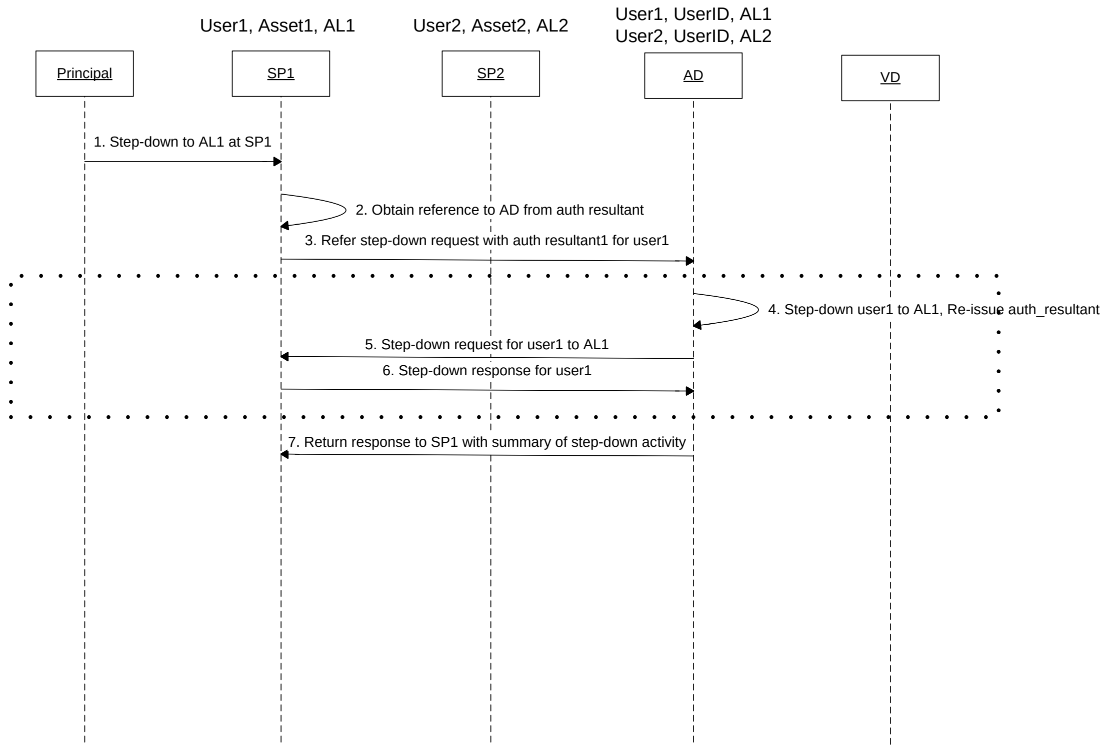
321

322 ### 3.4.4 Dependencies with Other Use Cases

323 Same as Use Case 3.3.

3.5 Single Sign-On and Step-Down Access

3.5.1 Graphical Description


User1, Asset1, AL1
User2, Asset2, AL2
User1, UserID, AL1
User2, UserID, AL2
Principal
SP1
SP2
AD
VD
1. Step-down to AL1 at SP1
2. Obtain reference to AD from auth resultant
3. Refer step-down request with auth resultant1 for user1
4. Step-down user1 to AL1, Re-issue auth_resultant
5. Step-down request for user1 to AL1
6. Step-down response for user1
7. Return response to SP1 with summary of step-down activity

⁚⁚⁚ Denotes Operations that can be repeated over multiple SPs

3.5.2 Description

Title/ID	Single Sign-On and Step Down Access
Pre conditions	1. User has received varying assurance levels and is single signed on to different service providers. 2. There is only one authentication delegate in this use case.
Constituents	Same as Use Case 3.4.
Use Case	1. User initiates a step-down request at SP1 to the new assurance level AL1. Alternately, this can also be initiated at the AD in which case the flow continues from Step 4 below. 2. SP1 obtains reference to Authentication Resultant 1. 3. Creates and refers step-down request to the appropriate AD. In case of system-directed step-down, the flow starts with this step.

	4. AD steps down the user to AL1 at SP1. 5. AD issues step-down request to SP1. 6. SP1 returns step-down response: Steps 4-6 are repeated for all SP for which current assurance level is greater than target assurance level AL2. 7. A summary of the step-down process is returned to the initial requestor and displayed to the user.
Post conditions	User is stepped down to AL1 at all service providers.
Out of band / error conditions	1. Same as Use Case 3.4. 2. One or more step down requests could have failed.

3.5.3 Data Elements

All relevant data elements captured in Use Case 3.1.

3.5.4 Mapping to BMEG Use Cases

All relevant mappings captured in Use Case 3.1.

3.5.5 Out of Scope

Error conditions presented to the user.

3.5.6 Dependencies with Other Use Cases

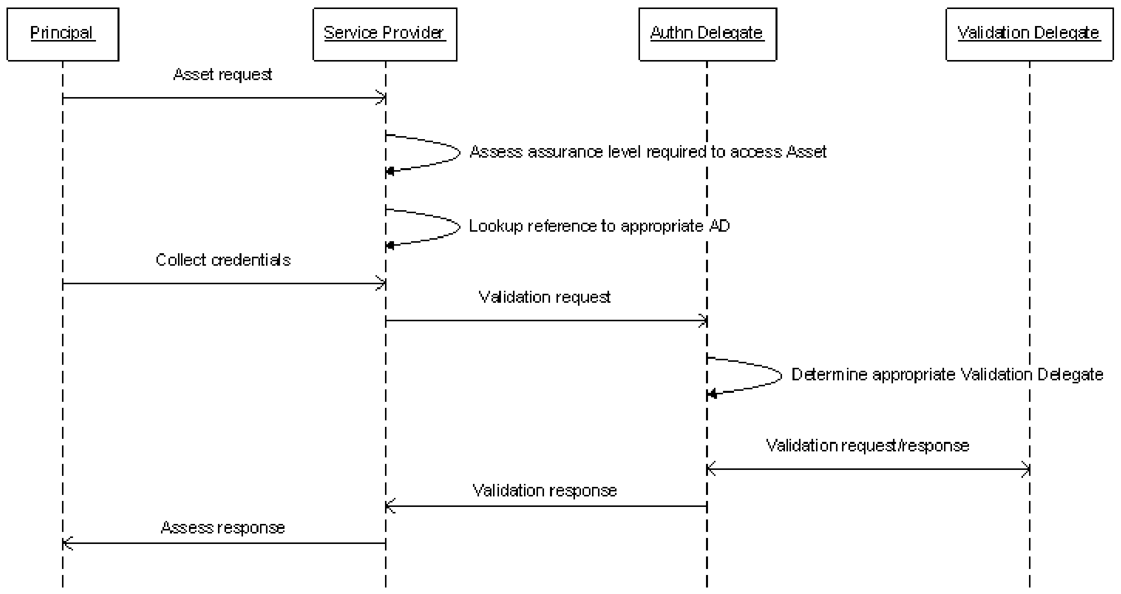
None.

3.6 Credential Portability

The following use cases describe the various constituents' experiences in order to realize the benefits of credential portability.

3.6.1 High Level Use Case for Credential Portability

The following use case highlights the difference between credential portability and single sign-on.


Principal
Service Provider
Authn Delegate
Validation Delegate
Asset request
Assess assurance level required to access Asset
Lookup reference to appropriate AD
Collect credentials
Validation request
Determine appropriate Validation Delegate
Validation request/response
Validation response
Assess response

Title/ID	Credential Portability
Pre conditions	1. Principal has already federated/bound his credential at the AD to his identity at the SP. 2. The Principal's credentials at the VD have to be bound, using an opaque handle aka Credential ID, to the Principal's identity at the AD prior to the actual authentication.
Constituents	Principal, Service Provider, Authentication Delegate, Validation Delegate
Use Case	1. Principal requests Service Provider for access to an asset. 2. Service Provider compares the current assurance level of the request (non-existent since there is not an appropriate existing session state) with that required for the requested asset. 3. The Service Provider determines which Authentication Delegate can match the required assurance level for the asset. 4. If the Service Provider can look up a valid reference to a valid Authentication Delegate, it collects credentials. Otherwise, it

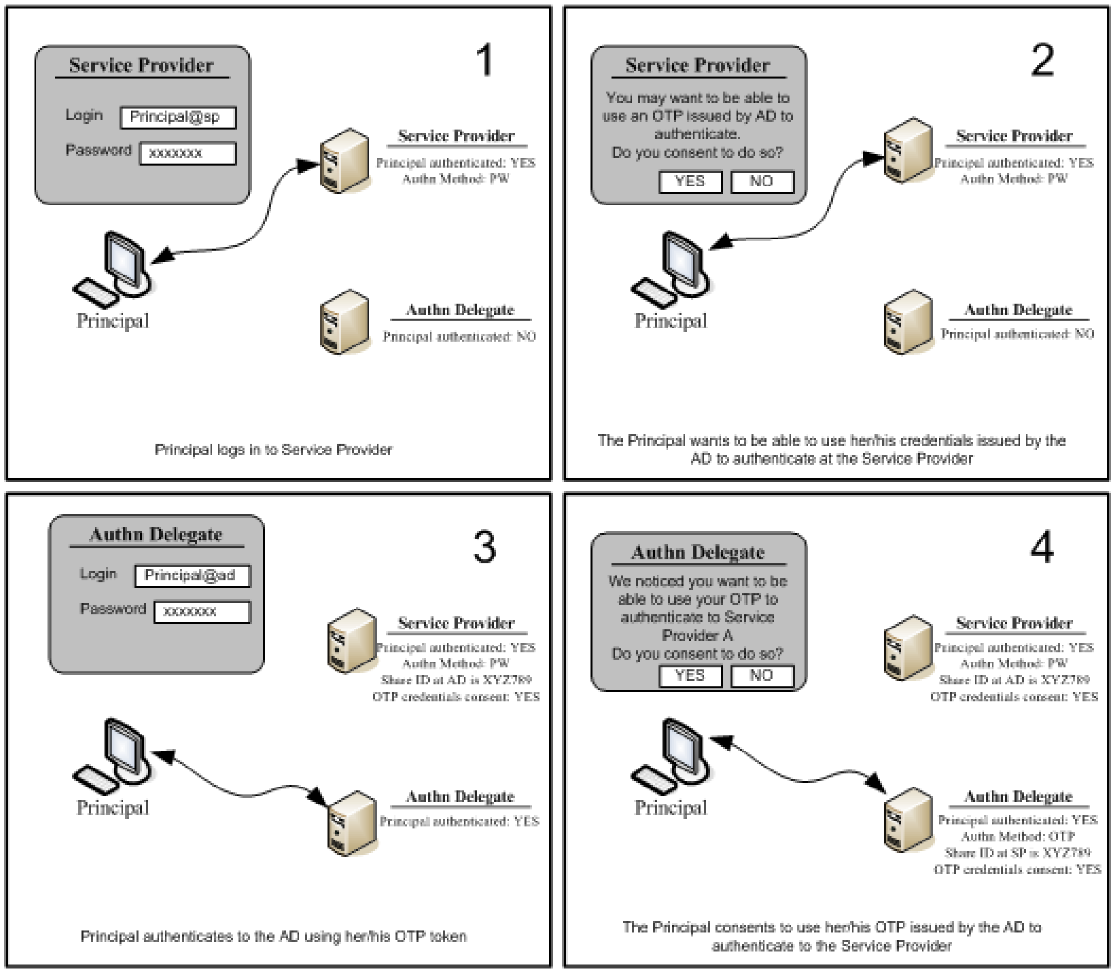
	returns an error. 5. The Service provider creates a request with the required attributes and sends it to the Authentication Delegate. 6. The Authentication Delegate determines which Validation Delegate can validate the provided credentials. If AD cannot find an appropriate VD, it returns an error. 7. The AD creates a validation request with the required attributes and passes it to the Validation Delegate. 8. The VD validates the credentials and generates a validation response specifying the result of the actual credential validation. 9. Upon successful completion of the validation requests, the AD creates the authentication context and returns the resultant to the SP. The resultant contains, either by reference or by value, the context for all authentications performed for this Principal.

3.6.2 Service Provider-Local Credentials

The Principal has credentials local to the SP and uses AD-issued credentials for a stronger authentication.

3.6.2.1 Principal Links Credentials at AD to Identity at SP

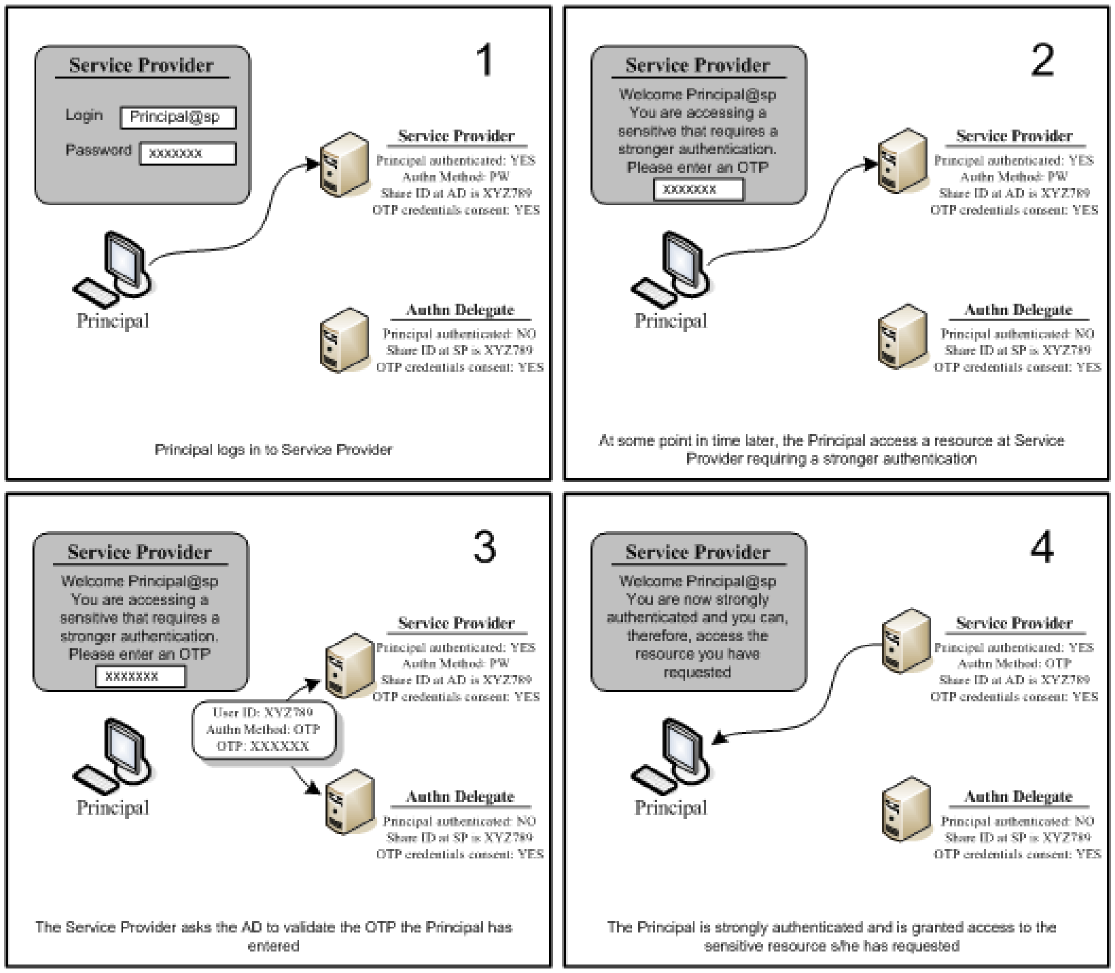
Linking the Principal's AD-issued credentials to the Principal's SP identity is a prerequisite for being able to use these credentials to authenticate at the SP. This can be done on a different channel than the one on which the credentials are actually evaluated.


1
Service Provider
Login Principal@sp
Password xxxxxxx
Principal
Service Provider
Principal authenticated: YES
Authn Method: PW
Authn Delegate
Principal authenticated: NO
Principal logs in to Service Provider
2
Service Provider
You may want to be able to use an OTP issued by AD to authenticate.
Do you consent to do so?
YES NO
Principal
Service Provider
Principal authenticated: YES
Authn Method: PW
Authn Delegate
Principal authenticated: NO
The Principal wants to be able to use her/his credentials issued by the AD to authenticate at the Service Provider
3
Authn Delegate
Login Principal@ad
Password xxxxxxx
Service Provider
Principal authenticated: YES
Authn Method: PW
Share ID at AD is XYZ789
OTP credentials consent: YES
Principal
Authn Delegate
Principal authenticated: YES
Principal authenticates to the AD using her/his OTP token
4
Authn Delegate
We noticed you want to be able to use your OTP to authenticate to Service Provider A
Do you consent to do so?
YES NO
Service Provider
Principal authenticated: YES
Authn Method: PW
Share ID at AD is XYZ789
OTP credentials consent: YES
Principal
Authn Delegate
Principal authenticated: YES
Authn Method: OTP
Share ID at SP is XYZ789
OTP credentials consent: YES
The Principal consents to use her/his OTP issued by the AD to authenticate to the Service Provider

Title/ID	Principal Links Credentials at AD to Identity at SP
Pre conditions	1. Principal has an identity and strong credentials (typically an OTP token) at an Authentication Delegate. 2. Principal has an identity and credentials (typically a login/password) at the Service Provider. 3. The Authentication Delegate has entered into an agreement with the Service Provider by which the Service Provider accepts authentication of the Principal with credentials issued by the Authentication Delegate, including what authentication mechanism(s) the SP will accept. 4. The Authentication Delegate has followed Liberty guidelines for provisioning identities and managing the linking of a Principal's identity at the Authentication Delegate to the Principal's identity at the Service Provider. 5. The Authentication Delegate tells the Principal how their identity at the Authentication Delegate can be linked to their identity at this Service Provider, as well as any other Service Providers with whom the Authentication Delegate has a relationship, in accordance with Liberty guidelines. 6. The Principal gives consent to link their identity at the Authentication Delegate with their identity at the Service Providers with whom the Authentication Delegate has a relationship. 7. The Authentication Delegate binds the Principal’s consent to the notice in accordance with Liberty guidelines. 8. The Authentication Delegate stores the bound notice and choice such that the Principal can gain later access to this information in accordance with Liberty guidelines. 9. The Service Provider has followed Liberty guidelines for provisioning identities and managing the linking of a Principal’s identity at the Authentication Delegate to the Principal’s identity at the Service Provider.
Constituents	Principal, Authentication Delegate, Service Provider
Use Case	1. The Principal navigates to the Service Provider, verifies the identity of the Service Provider, and authenticates to the Service Provider using her/his local credentials (e.g., her/his login/password). The Service Provider recognizes that the Principal is currently authenticated with the assurance level consistent with password verification. 2. The Service Provider provides the Principal with notice that she/he has the opportunity to use the credentials issued by the AD, in addition to their local credentials, to strongly

	authenticate at the SP. Principal chooses to link her/his identities and gives the Service Provider her/his consent. 3. The Principal is redirected to the AD and authenticates using credentials issued by the AD. 4. The Authentication Delegate tells the Principal that she/he has asked to use credentials issued by the AD to strongly authenticate at the SP. The Principal chooses to link her/his identities and gives the Authentication Delegate her/his consent.

3.6.2.2 Principal Authenticates to SP by Using Credentials at SP and Credentials at AD


1
Service Provider
Login Principal@sp
Password xxxxxxx
Service Provider
Principal authenticated: YES
Authn Method: PW
Share ID at AD is XYZ789
OTP credentials consent: YES
Principal
Authn Delegate
Principal authenticated: NO
Share ID at SP is XYZ789
OTP credentials consent: YES
Principal logs in to Service Provider
2
Service Provider
Welcome Principal@sp
You are accessing a sensitive that requires a stronger authentication. Please enter an OTP
xxxxxxx
At some point in time later, the Principal access a resource at Service Provider requiring a stronger authentication
3
User ID: XYZ789
Authn Method: OTP
OTP: XXXXXX
The Service Provider asks the AD to validate the OTP the Principal has entered
4
Welcome Principal@sp
You are now strongly authenticated and you can, therefore, access the resource you have requested
Authn Method: OTP
The Principal is strongly authenticated and is granted access to the sensitive resource s/he has requested

Title/ID	Principal Authenticates to SP by Using both Credentials at SP and Credentials at AD
Pre conditions	1. The Principal has an identity and credentials (typically an OTP token) at an Authentication Delegate. 2. The Principal has an identity and credentials (typically a login/password) at the Service Provider. 3. The Authentication Delegate has entered an agreement with the Service Provider by which the Service Provider accepts authentication of the Principal with credentials issued by Authentication Delegate and has agreed what Authentication mechanism(s) they will accept. 4. The Authentication Delegate has followed Liberty guidelines for provisioning identities and has given notice and obtained consent from the Principal to accept credential validation requests from the Service Provider. 5. The Service Provider has followed Liberty guidelines for provisioning identities and has given notice and obtained consent from the Principal to use credentials issued by the Authentication Delegate to authenticate her/him. 6. The Principal's identity at the Service Provider is linked with her/his credentials issued by the Authentication Delegate.
Constituents	Principal, Authentication Delegate, Service Provider
Use Case	1. The Principal navigates to the Service Provider, verifies the identity of Service Provider and authenticates to the Service Provider using her/his local credentials (e.g., her/his login/password). The Service Provider recognizes that the Principal is currently assigned an assurance level consistent with password validation. 2. The Principal requests a sensitive resource or service at the Service Provider. The Service Provider recognizes that the Principal can be authenticated with a credential managed by the Authentication Delegate (e.g., an OTP) and asks the Principal to use the credentials issued by the Authentication Delegate to initiate stronger authentication mechanism. The Principal provides the requested input. 3. The Service Provider asks the Authentication Delegate to validate the credential the Principal has provided. The Authentication Delegate obtains validation of the credential and securely asserts the Principal's authentication and the type of authentication to the Service Provider. 4. The Service Provider recognizes that the Principal is currently authenticated with a stronger authentication method (e.g., OTP)

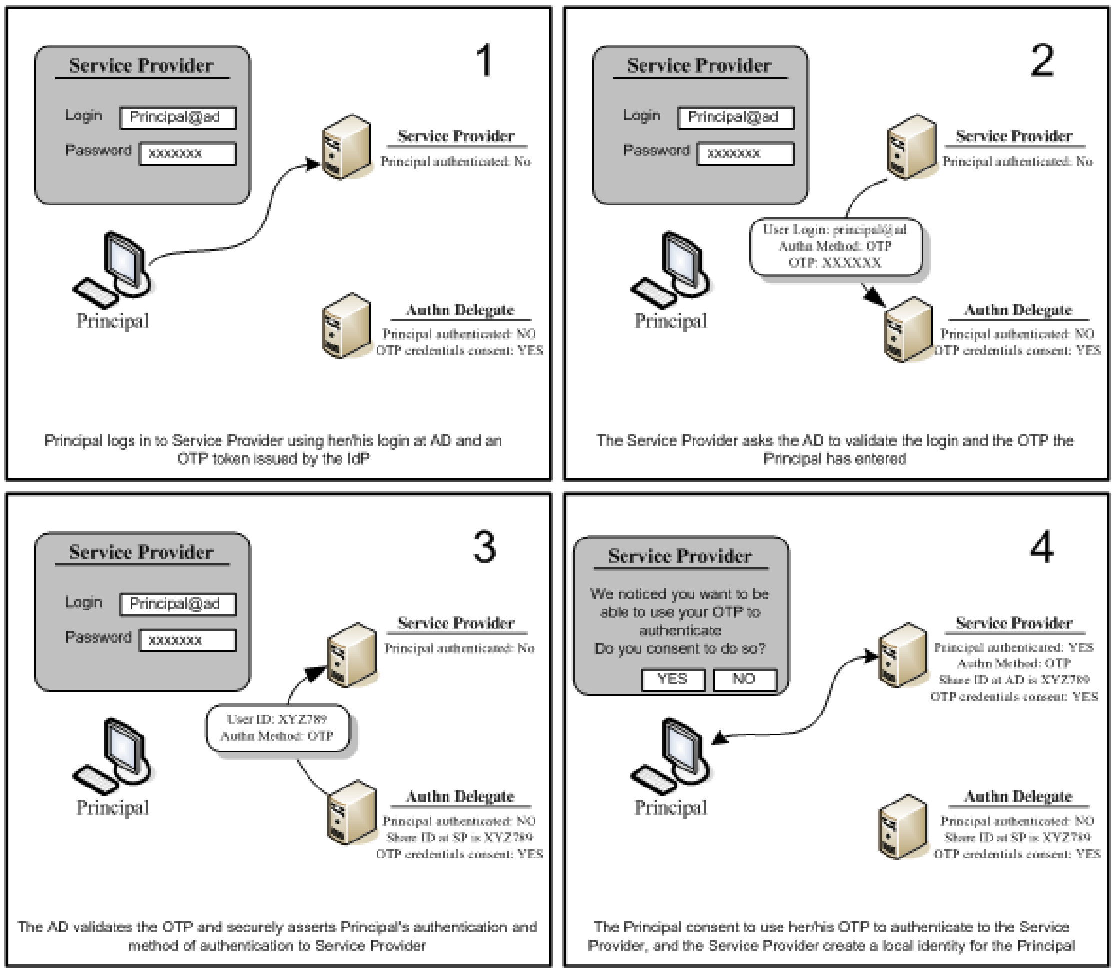
	and recognizes that thus type of authentication is acceptable to access the requested service or asset. The Service Provider delivers service to the Principal.

3.6.3 No SP-Local Credentials

Principal has no credentials local to the SP and instead uses only AD-issued credentials to authenticate to the SP.

3.6.3.1 Principal Links Credentials at AD to a Newly Created Identity at SP

Linking the Principal's AD-issued credentials to a newly created identity at the SP is a prerequisite for being able to use these credentials to authenticate at the SP.



Principal logs in to Service Provider using her/his login at AD and an OTP token issued by the IdP

The Service Provider asks the AD to validate the login and the OTP the Principal has entered

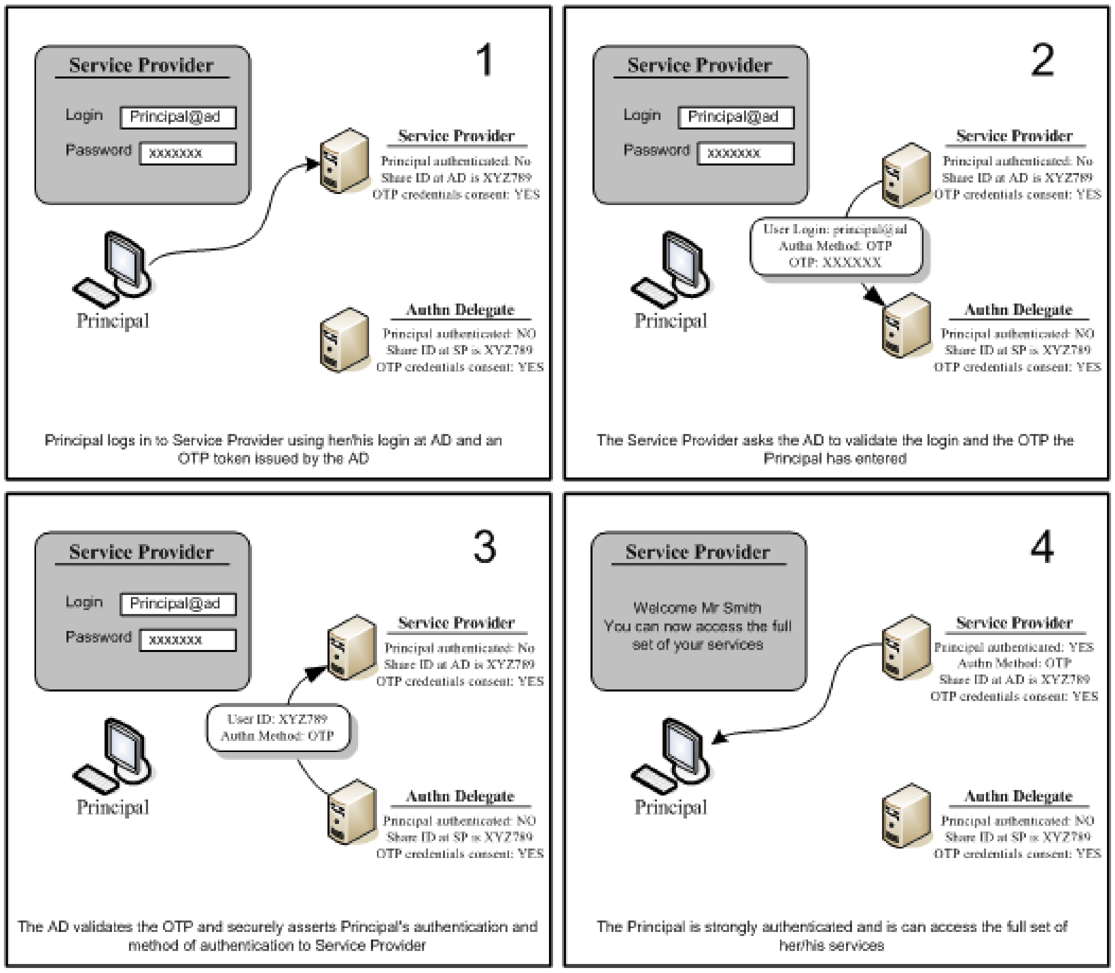
The AD validates the OTP and securely asserts Principal's authentication and method of authentication to Service Provider

The Principal consent to use her/his OTP to authenticate to the Service Provider, and the Service Provider create a local identity for the Principal

Title/ID	Principal Links Credentials at AD to a Newly Created Identity at SP
Pre conditions	1. The Principal has an identity and stronger credentials (e.g., an OTP token) at an Authentication Delegate. 2. The Principal has no identity at the Service Provider yet. 3. The Authentication Delegate has entered an agreement with the Service Provider by which the Service Provider accepts authentication of the Principal using AD-issued credentials and has agreed what authentication mechanism(s) will be accepted. 4. The Authentication Delegate has followed Liberty guidelines for provisioning identities and managing the linking of a Principal's identity at the Authentication Delegate to identities at Service Provider. 5. The Authentication Delegate tells the Principal how their identity at the Authentication Delegate can be linked to their identity at the Service Provider, and any other Service Providers with whom that Authentication Delegate has a relationship, in accordance with Liberty guidelines. 6. The Principal gives consent to link their identity at the Authentication Delegate with a newly created identity at the Service Providers with whom that Authentication Delegate has a relationship. 7. The Authentication Delegate binds the Principal’s consent to the notice in accordance with Liberty guidelines. 8. The Authentication Delegate stores the bound notice and choice such that the Principal can gain later access this information in accordance with Liberty guidelines. 9. The Service Provider has followed Liberty guidelines for provisioning identities and managing the linking of a Principal’s identity at the Authentication Delegate to the Principal’s identity at Service Provider.
Constituents	Principal, Authentication Delegate, Service Provider
Use Case	1. The Principal navigates to the Service Provider, verifies the identity of the Service Provider, and authenticates to the Service Provider using this AD-issued credentials. 2. The Service Provider determines which Authentication Delegate can handle the credentials provided by the Principal and asks the Authentication Delegate to obtain validation for them. 3. The Authentication Delegate obtains validation of the provided credentials and returns a resultant (e.g., an authentication assertion) back to the Service Provider. 4. The Service Provider determines that the Principal has no identity yet and tells the Principal that she/he must use her/his AD-issued

	credentials authenticate at the SP. The Principal chooses to create a new identity local to the Service Provider and to link her/his identities, then gives Service Provider her/his consent.

3.6.3.2 Principal Authenticates to SP by Only Osing Credentials at AD



Principal logs in to Service Provider using her/his login at AD and an OTP token issued by the AD

The Service Provider asks the AD to validate the login and the OTP the Principal has entered

The AD validates the OTP and securely asserts Principal's authentication and method of authentication to Service Provider

The Principal is strongly authenticated and is can access the full set of her/his services

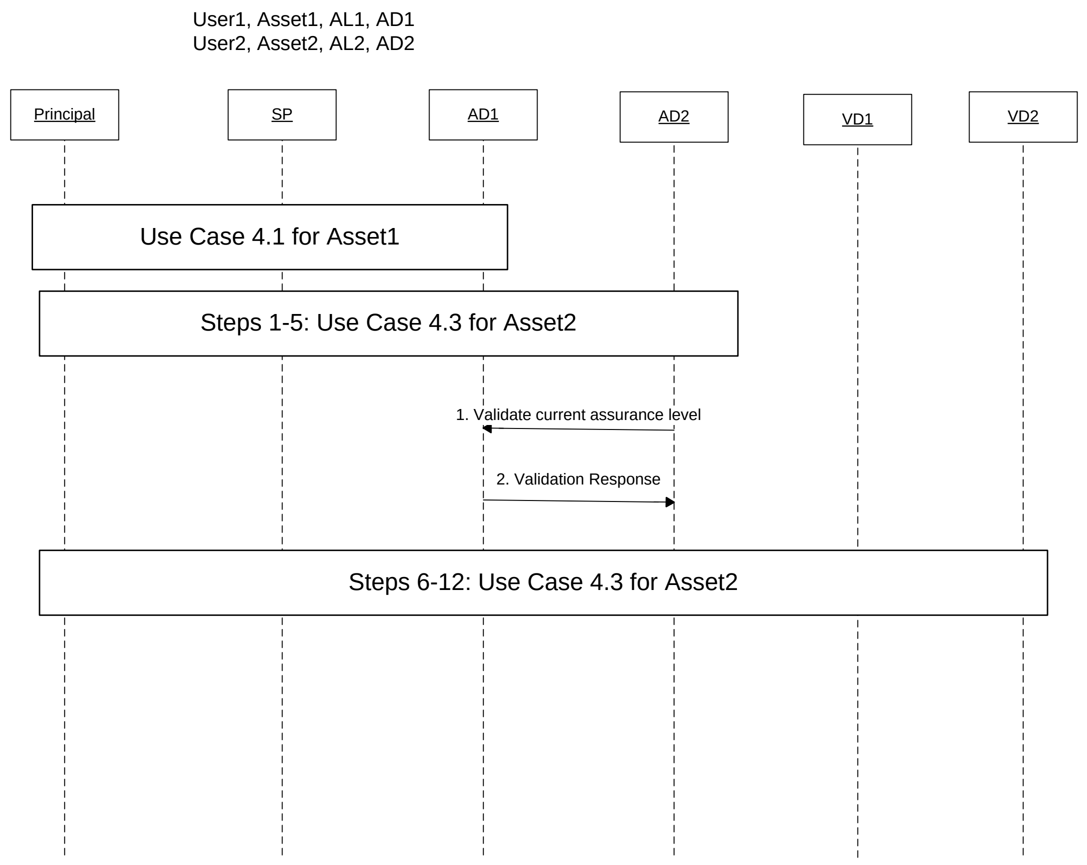
Title/ID	Principal Authenticates to SP by Only Using Credentials at AD
Pre conditions	1. The Principal has an identity and credentials (e.g., an OTP token) at an Authentication Delegate. 2. The Principal has an identity but no credentials at the Service Provider to make assertions about that identity. 3. The Authentication Delegate has entered an agreement with the Service Provider by which the Service Provider accepts authentication of the Principal with AD-issued credentials and has agreed what authentication mechanism(s) they will accept. 4. The Authentication Delegate has followed Liberty guidelines for provisioning identities and has given notice and obtained consent from the Principal to accept credential validation requests from the Service Provider. 5. The Service Provider has followed Liberty guidelines for provisioning identities and has given notice and obtained consent from the Principal to use credentials issued by the Authentication Delegate to authenticate her/him. 6. The Principal's identity at the Service Provider is linked with her/his credentials issued by the Authentication Delegate.
Constituents	Principal, Authentication Delegate, Service Provider
Use Case	1. The Principal navigates to the Service Provider, verifies the identity of the Service Provider, and authenticates to the Service Provider using her/his AD-issued credentials (e.g., her/his login and an OTP). 2. The Service Provider asks the Authentication Delegate to obtain validation of the login and the OTP that the Principal has just provided. 3. The Authentication Delegate obtains validation of the OTP and securely asserts Principal's authentication and type of authentication to the Service Provider. 4. The Service Provider recognizes that the Principal is currently authenticated with a stronger authentication method (e.g., OTP).

NB: This use case raises privacy issue since a same login may be provided to several SPs, enabling these SPs to directly establish a link between the Principal's identities. The SP should, therefore, commit to not storing this information and, instead, use the name ID provided by the AD.

4 Use Cases: Foundational Scenario 2

4.1 Single Sign-On: Reuse of Authentication Resultants

4.1.1 Graphical Description



4.1.2 Description

Title/ID	Single Sign-On: Reuse of Authentication Resultants
Pre conditions	1. There exists a trusted relationship between AD1 and AD2 using which authentication requests made by AD1 can be verified by AD2 and used to establish a basic level of assurance on which further authentication can be performed to achieve higher assurance levels. Elements of the trusted relationship between AD include: a. Assurance Levels, b. CredentialID mapping,

	c. Shared identities, and d. Hierarchy of authentication mechanisms: for example, OTP based on passwords, certificates based on smart cards. 2. AD2 is capable of using the current authentication resultants issued to the Principal in order to bootstrap or upgrade to the next level of assurance. 3. (Open note) Does AD1 need to perform CredentialID mapping? Would the SP be responsible for this mapping? 4. (Open note) Is there a need for the notion of a primary authentication delegate? For points 3and 4, per TEG sync up, need to look into SAML Proxying model for further interaction details. 5. (Open note) Need to specify auditing requirements and limitations of dynamic provisioning and tracking of sessions across AD. 6. In a browser-based profile, AD1 may redirect the Principal to AD2. In a smart client-based profile, the client would make a call with the current authentication resultant prior to higher level authentication.
Constituents	SP, AD1, AD2, VD1, VD2
Use Case	1. Principal initiates request for asset1 which is protected by AD1. This leads to execution of Use Case 3.1. 2. Principal now requests asset2 for which higher assurance level AL2 is required. 3. The SP creates a request to send to AD2 and embeds the resultant or a reference to the resultant in the request along with the verifier URL at AD1 (Steps 1-5 of Use Case 3.3). 4. AD2 on receiving the request makes a direct call to AD1 to verify the assertion. 5. On receiving confirmation, AD2 performs the higher level authentication and issues a higher level authentication resultant (Steps 6-12 of Use Case 3.3).
Post conditions	Principal is able to access higher assurance level asset2 at SP with having to authenticate only the higher level factor.
Out of band / error conditions	1. Authentication Resultant from AD1 is expired or invalid in which case AD2 could perform a full authentication for the user. 2. Other out of band conditions from Use Cases 3.1 and 3.3.

4.1.3 Data Elements

In addition to data elements in Use Cases 3.1 and 3.3, the following are net new elements:

Constituent	**Data Element**	**Function**	**Description**
AD2	Validation Request	Create, Use	Out of band trusted connections may enable the Validation request. However, the request contains the requested assurance level in addition to the base resultant.
AD1	Validation Response	Create, Use	Returns successful confirmation of user identity along with optional CredentialID mapping functions.

4.1.4 Out of Scope

4.1.5 Dependencies with Other Use Cases

1. Use Case 3.1.
2. Use Case 3.3.

5 Supporting Service

A Supporting Service fulfills a required role in multiple use cases. Its required set of features, including any variances between data sets, are the union of the requirements across all of the foundational use cases and any other enhanced use cases, such as those described in the [Identity Theft Whitepaper]. The description of a Supporting Service does not in itself constitute a unique use case.

5.1 Discovery of Delegated Service

A range of Authentication Delegates and Validation Delegates may be available in any given authentication cycle, and, in fact, using more than one (1) authentication mechanism for a given user identity or more than one (1) validator for a credential may increase the surety of the authentication. The following discussion applies to both Authentication Delegates and Validation Delegates, where the "interactor" refers to the service making the discovery request and "evaluator" refers to the service being sought. In other words, an Authentication Delegate is an Evaluator when it is being discovered and then becomes an Interactor when it searches for a Validation Delegate.

It is assumed that the Discovery Service used to locate Evaluators is built from an existing Discovery Service. Thus, no new requirements are introduced that read on the composition of the basic discovery protocol (e.g., whether the discovery is a network-based broadcast or a directory look-up).

No assumption is made of a pre-existing relationship between the Interactor and the Evaluator, although a pre-existing relationship is allowed, and, should either party desire, signatures may be used to verify that relationship.

The attributes are basically the same for both the Interactor and the Evaluator. Any non-null attributes provided by the Interactor are to be used as search criteria for an Evaluator. Each Evaluator discovered will be described with all available attributes and the Interactor may or may not choose to actually engage with any of the discovered Evaluators.

Attribute	Description
Location	The source (Interactor) of the discovery request, or if this is the result of a discovery request, then this is the location of the Evaluator service.
Service Name	Identifying tag for the service. If a prior relationship exists, this may be signed. Any signature or other trust assurance data may be included in sub-attributes to this attribute.
Assurance Level	The degree of assurance that will be provided by the Evaluator. In general, if there is no prior relationship, this value has little to no meaning.

Source	Source of the Evaluator's knowledge. In the case of an Authentication Delegate, this may identify the Validation Delegate. In the case of a Validation Delegate, this may indicate a repository, as an example.
Type of Source	Direct or indirect.
Mechanism	Type of evaluator mechanism.
Credential	Type of credential.
Private Data	Additional attributes that may be provided by an Interactor with a prior relationship with the Evaluator.

The attributes used in the discovery process are used by the Interactor to choose an evaluator. They do not actually perform the evaluation activity. The Interactor may choose to save the discovery attributes provided by an Evaluator to qualify the Evaluator's results. It is important, however, that the Interactor NOT provide user identity data (e.g., in the Private Data attributes) during the discovery process.

6 Taxonomy

6.1 SAEG-Specific Terms and Definitions

1. **Asset**
   Something to which the Principal seeks access. Interchangeable with the term "resource."
   a. Either data related to some identity or identities or a service acting on behalf of some identity or group of identities. An example of a resource is a calendar containing appointments for a particular identity.
2. **Principal** (as adopted from Main [LibertyGlossary].)
3. **Service Provider (SP)**
   A system entity that provides or denies access to an asset based upon policy. The SP has an identity store, pointer to authentication mechanisms, and a policy store. The SP knows the real identity of the Principal.
4. **Authentication Delegate (AD)**
   In a Liberty context, the Authentication Delegate can be thought of as an **enhanced IdP**. In addition to conventional IdP functions, the AD is required to perform the following tasks:
   a. Identity functions related to mapping authentication mechanisms and assurance levels to user identities.
      i. Mapping/un-mapping.
         1. User to authentication mechanism(s).
         2. Authentication mechanism to assurance level.
      ii. Discovery.
         SP and AD must exist in a circle of trust in order to have a trusted exchange of sensitive strong-authentication information:
         1. Discovery of services
            a. AD discovery by SP.
            b. VD discovery by AD.
         2. Discovery of strong authentication-related attributes per service and per user/group, e.g., respond to requests for discovery of available authentication mechanisms per user.
      iii. Coordination of authentication of the Principal.
      iv. Lifecycle (provision, activation, and management) for strong authentication credentials for set of Principals under management.
   b. Credential collection from SP or Principal and respective transmission to another AD and/or VD.
   c. Correlation of the authentication mechanisms per Principal against the authentication strength as required by the SP.
5. **Validation Delegate**
   The Validation Delegate is aware of the mapping between a Credential ID and the

Credential for one or more authentication mechanisms without the ability to correlate the mechanisms for a given Principal, that is the actual user's identity is masked with a handle or equivalent, for purposes of privacy protection. One or more Credential IDs can be associated to one authentication mechanism.

The VD performs the following services:

a. Optionally sources/collects the Credentials,
b. Validates given authentication credentials for a claimed Principal,
c. Maps one or more Credential IDs for a Credential per authentication mechanism, and
d. Enumerates the attributes of itself including, but not limited to, its supported authentication mechanisms and associated metadata.

6. **Credential Alias**
An opaque reference that is bound to an identity at the AD and is also bound to one credential at the VD to be validated. A VD may establish a unique credential alias per AD for enhanced privacy.
7. **Credential Identifier (ID)**
A unique identifier of the credential at the VD.
8. **Authentication Resultant**
Stores/represents/contains result and its respective authentication context of a previous authentication process in the form of a token. One implementation of the resultant could be the SAML assertion.
9. **Authentication Context**
Container of descriptive attributes required as an input to select an authentication mechanism (or) derived as an output from execution of the selected authentication mechanism. This does not assume successful authentication. Additional data communicated related the specific result of a transaction.
    a. Empty.
        i. Nothing.
    b. Rich (extensible).
        i. Audit trail.
        ii. States.
        iii. Strikes.
        iv. System information.
        v. Previous authentications.
10. **Assurance Level**:
An assurance level can be defined as the required set of criteria that must be satisfied in order for a Principal to obtain access to an asset. In the current context, these criteria are expected to be restricted to those that are conveyed within an authentication resultant. In this context:
    a. Each asset can be thought off as possessing a "Required" Assurance Level. This is the basic set of criteria which must be met in order to obtain access to the asset.

   b. Each Principal can be thought off as possessing a "Resultant" Assurance Level. This is the basic set of criteria which the principal has already fulfilled by virtue of a previous authentication event. In the most trivial case, lack of authentication implies only unrestricted criteria which are possessed by all principals.

11. **Authentication Discovery**
    The request-response interaction by which a Resource Owner (RO)/Asset Owner (AO) can interrogate and identify one or more AuthD entities that can to the fullest extent possible the passed authentication context attributes.
12. **Authentication Upgrade**
    The process by which an RO/AO can request issuance of security resultants which can be used by the requester to access resources/perform operations which demand levels of assurance higher than that conveyed by the requestors most recent security resultants.
13. **Factor**
    A unique type of information that can be tested to validate a claimed attribute. Commonly considered factors at this point:
    a. Possession.
    b. Knowledge.
    c. Essence.
14. **Transaction**
    A unique request and response exchange between any two system entities.
15. **Policy store**
    Holds a set of policies used to make authorization decisions which are based upon authentication results and additional data.
16. **Identity store**
    Local repository of known Principals.
17. **Provisioning**
    The aggregation of fulfillment and activation.
18. **Credential store**
    Local repository of factor information associated to an identity through a unique identifier.
19. **Fulfillment**
    The delivery of a credential to a Principal.
20. **Activation**
    The association of a credential with a credential store so that authentication event can take place.

6.2 Liberty-Specific Terms and Definitions Taken from the Master Glossary

From the master [LibertyGlossary]:

1. **authenticated identity**
   An identity, representing a system entity, which often is a Principal, that is asserted to have been the subject of a successful authentication.
2. **authenticating entity**
   A system entity that engages in the process of authenticating itself to another system entity, the latter typically being an Identity Provider (see also authentication). More formally, an authenticating system entity.
3. **authentication** (authn)
   Authentication is the process of confirming a system entity's asserted identity with a specified, or understood, level of confidence [TrustInCyberspace].
4. **authentication mechanism**
   An authentication mechanism is a particular, identifiable, process or technique that results in a confirmation of a system entity's asserted identity with a specified, or understood, level of confidence.
5. **authentication session**
   The period of time starting after A has authenticated B and until A stops trusting B's identity assertion and requires reauthentication. Also known simply as a session, it is the state between a successful login and a successful logout by a Principal.
6. **authorization** (authz)
   The process of determining, by evaluating applicable access control information, whether a subject is allowed to have the specified types of access to a particular resource. Usually, authorization is in the context of authentication. Once a subject is authenticated, it may be authorized to perform different types of access [SAMLGloss].
7. **client**
   A role assumed by a system entity who makes a request of another system entity, often termed a server [RFC2828]. A client is at varying times a sender or a receiver.
8. **credentials**
   Data that is transferred or presented to establish either a claimed identity or the authorizations of a system entity.
9. **federation**
   a. The act of establishing a relationship between two entities.
   b. An association comprising any number of service providers and identity providers.

10. **identity**
    The essence of an entity. One's identity is often described by one's characteristics, among which may be any number of identifiers. A Principal may wield one or more identities. See also Principal identity.
11. **Identity Provider** (IdP)
    A Liberty-enabled system entity that manages identity information on behalf of Principals and provides assertions of Principal authentication to other providers.
12. **login**
    The act of a Principal proving their identity to a system entity, which typically establishes a session.
13. **logout**
    The termination of a session.
14. **metadata**
    Definitional data that provides information about other data or system entities managed within an application or environment. In Liberty, metadata is Provider information that is necessary for interacting with Providers [LibertyMetadata].
15. **opaque handle**
    An identifier that has meaning only in the context between a specific identity provider and specific service provider.
16. **permission**
    Privileges granted to a system entity with respect to operations that may be performed on some resource.
17. **policy**
    A logically defined, enforceable, and testable set of rules.
18. **Policy Decision Point**
    A system entity that evaluates decision requests in light of applicable policy and information describing the requesting entity or entities and renders an authorization decision.
19. **Policy Enforcement Point**
    A system entity that performs access control by making decision requests and enforcing authorization decisions. If the authorization decision is pushed to the PEP, there will be no need for it to create a request.
20. **Principal**
    Succinctly, a principal is a system entity whose identity can be authenticated. In Liberty usage, the term Principal is often synonymous with "natural person" or "user." A Principal's identity may be federated. Examples of Principals include individual users, groups of individuals, organizational entities (e.g., corporations), or a component of the Liberty architecture.
21. **Principal identity**
    An identity being wielded by a Principal or that is mapped to a Principal in some fashion.

22. **privacy**
    Proper handling of personal information throughout its life cycle, consistent with the preferences of the subject.
23. **processing context**
    A processing context is the collection of specific circumstances under which a particular processing step or set of steps take place.
24. **processing context facet**
    A processing context facet is an identified aspect, inherent or additive, of a processing context.
25. **profile**
    Data comprising the broad set of attributes that may be maintained on behalf of an system entity (usually a Principal), over and beyond its various identifiers. At least some of this information (for example, addresses, preferences, card numbers) is typically provided by the Principal.
26. **provider**
    A provider is a Liberty-enabled entity that performs one or more of the provider roles in the Liberty architecture, for example Service Provider or Identity Provider. Providers are identified in Liberty protocol interactions by their Provider IDs or optionally their Affiliation ID if they are a member of an affiliation(s) and are acting in that capacity.
27. **proxy**
    a. An entity authorized to act for another [Merriam-Webster].
    b. A system entity whose authenticated identity, according to the recipient, differs from that of the system entity making the invocation under consideration.
28. **pseudonym**
    An arbitrary identifier assigned by the identity or service provider to identify a Principal to a given relying party so that the name has meaning only in the context of the relationship between the parties.
29. **recipient**
    An entity that receives a message and acts as the message's ultimate processor.
30. **receiver**
    A role taken by a system entity when it receives a message sent by another system entity.
31. **relying party**
    The recipient of a message that relies on a request message and associated assertions to determine whether to provide a requested service.
32. **requester**
    A system entity which sends a service request to a provider.
33. **resource**
    Either data related to some identity or identities, or a service acting on behalf of some identity or group of identities. An example of a resource is a calendar containing appointments for a particular identity.

34. **role**
    A function or part performed, especially in a particular operation or process [Merriam-Webster].
35. **security token**
    In Liberty, a security token is a collection of security-related information that is used to represent and substantiate a claim [LibertyIDWSFSecurityPrivacyGuidelines] [LibertySecMech]. Outside of Liberty, the term "security token" often refers to hardware-based devices, e.g., so-called "token cards." One should not confuse the latter and the former definitions. However, it is possible for some given authentication mechanism to employ token cards in the process of authentication.
36. **sender**
    a. A role donned by a system entity when it constructs and sends a message to another system entity. See also SOAP sender in [SOAPv1.2].
        i. An initial SOAP sender. A sender is a proxy when its identity differs from the invocation identity.
37. **server**
    A role donned by a system entity that provides a service in response to requests from other system entities called clients [RFC2828]. Note that in order to provide a service to clients, a server will often be both a sender and a receiver.
38. **session**
    [Merriam-Webster] defines session (in its sixth sense [*sic*]) as: "a meeting or period devoted to a particular activity." Thus, a given interaction between some set of system entities may involve a notion of session, especially if one or more of the system entities maintain session state.
39. **session state**
    If an interaction between system entities involves one or more of the system entities maintaining information pertaining to the interaction itself—such as who the other involved system entity(ies) are, when the interaction began, etc.—then there likely is an explicit notion of session and thus this information is termed session state information.
40. **single sign-on (SSO)**
    From a Principal’s perspective, single sign-on encompasses the capability to authenticate with some system entity—in the Liberty context, an Identity Provider—and have that authentication honored by other system entities, termed Service Providers in the Liberty context. Note that upon authenticating with an Identity Provider, the Identity Provider typically establishes and maintains some notion of local session state between itself and the Principal’s user agent. Service Providers may also maintain their own distinct local session state with a Principal’s user agent.
41. **system entity**
    An active element of a computer/network system. For example, an automated

process or set of processes, a subsystem, a person or group of persons that incorporates a distinct set of functionality [SAMLGloss].

42. **token**
    See security token.
43. **Trusted Third Party** (TTP)
    In general, a security authority or its agent, trusted by other entities with respect to security-related activities. In the context of Liberty, these other entities are, for example, Principals and Service Providers, and the trusted third party is typically the Identity Provider(s) involved in the particular interaction of interest.

7 References

[Identity Theft Whitepaper] Duserick, William, eds, "Whitepaper on Liberty Protocol and Identity Theft," Version 1.0, Liberty Alliance Project (February 20, 2004). *http://www.projectliberty.org/liberty/content/download/389/2726/file/Liberty_Identity_Theft_Whitepaper.pdf*

[LibertyIDWSFSecurityPrivacyGuidelines] Landau, Susan, eds. "Liberty ID-WSF Security and Privacy Overview," Version 1.0, Liberty Alliance Project (8 October 2003). *http://www.projectliberty.org/specs*

[LibertyGlossary] Hodges, Jeff, eds. "Liberty Technical Glossary," Version v2.0, Liberty Alliance Project (30 July, 2006). *http://www.projectliberty.org/specs*

[LibertyMetadata] Davis, Peter, eds. "Liberty Metadata Description and Discovery Specification," Version 2.0-02, Liberty Alliance Project (25 November 2004). *http://www.projectliberty.org/specs*

[LibertySecMech] Hirsch, Frederick, eds. "Liberty ID-WSF Security Mechanisms Core," Version 2.0-errata-v1.0, Liberty Alliance Project (21 April, 2007). *http://www.projectliberty.org/specs*

[Merriam-Webster] "Merriam-Webster Dictionary," *http://www.merriam-webster.com/*

[MRD.Strong.Auth.Slides] Friese, Ingo, eds. "Use Case for Flexible Strong Authentication," Liberty Alliance Project (http://www.projectliberty.org/liberty/content/download/4298/28776/file/MRD.Strong.Auth.slides.pdf)

[RFC2828] Shirey, R., eds. (May 2000). "Internet Security Glossary," RFC 2828., Internet Engineering Task Force. *http://www.ietf.org/rfc/rfc2828.txt*

[SAMLGloss] Hodges, Jeff, Maler, Eve, eds. (05 November 2002). "Glossary for the OASIS Security Assertion Markup Language (SAML)," SAML V1.0, OASIS Standard, Organization for the Advancement of Structured Information Standards *http://www.oasis-open.org/specs/index.php#samlv1.0*

[SOAPv1.2] "SOAP Version 1.2 Part 1: Messaging Framework," Gudgin, Martin, Hadley, Marc, Mendelsohn, Noah, Moreau, Jean-Jacques, Nielsen, Henrik Frystyk, eds. World Wide Web Consortium W3C Recommendation (07 May 2003). *http://www.w3.org/TR/2003/PR-soap12-part1-20030507/*

[TrustInCyberspace] Schneider, Fred B., eds. "Trust in Cyberspace," National Research Council (1999) *http://www.nap.edu/readingroom/books/trust/*