Message Security Mechanisms Specification

Version 1.0.7 23 September 2013

Message Security Mechanisms Specification Version 1.0.7

Notice:

As of the date of publication, this document is a release candidate specification subject to DECE Member review and final adoption by vote of the Management Committee of DECE in accordance with the DECE LLC Operating Agreement. Unless there is notice to the contrary, this specification will become an adopted “Ecosystem Specification” on 5 November 2013.

Contact Information:

Licensing inquiries and requests should be addressed to us at: http://www.uvvu.com/uv-for-business.php

The URL for the DECE web site is http://www.uvvu.com

Message Security Mechanisms Specification draft 1.0.7

Contents

Table of Figures

Table of Tables

1 Introduction

1.1 Scope

This Specification details the security requirements for the communication between Nodes and the Coordinator, between Media Clients and the Device Portal, and between user agents and the Web Portal within the DECE Ecosystem. It additionally specifies Security Token Profile s that shall be used in conjunction with Coordinator API invocations, and User Credential requirements.

1.2 Document Notation and Conventions

1.2.1 Notations

The following terms are used to specify conformance elements of this specification. These are adopted from the ISO/IEC Directives, Part 2, Annex H [ISO-P2H].

SHALL and SHALL NOT indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

SHOULD and SHOULD NOT indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required, or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

MAY and NEED NOT indicate a course of action permissible within the limits of the document.

Terms defined to have a specific meaning within this specification will be capitalized, e.g. “Track”, and should be interpreted with their general meaning if not capitalized. Normative key words are written in all caps, e.g. “SHALL”.

1.2.2 Glossary of Terms

The following terms have specific meanings in the context of this specification. Additional terms employed in other specifications, agreements or guidelines are defined there. Many terms have been consolidated within [DSystem].

Delegation: the act of transferring rights and privileges to another party

Delegation Token: a Security Token used to demonstrate Delegation.

DECE Data: Data or information that Coordinator provides to Licensee via technical interfaces, including Account.

Federation Token Profile: A Security Token profile that defines the protocols and representation of a Security Token that enables the authentication of a user from one Node to another Node.

Delegation Token Profile: A Security Token profile that defines the protocols and representations of a Security Token that enables the proper identification of a User to the Coordinator as part of the Coordinator’s authorization decision processes.

Asserting Node: A Node that can locally authenticate a User, and asserts the identity of the User to the another Node. Typically, a Retailer or LASP may assert a User identity to the Coordinator's 's' host.

1.2.3 DECE References

[DCoord]	Coordinator API Specification
[DDevice]	Device Specification
[DGeo]	Geography Policies Specification

Table 1: DECE Technical Specifications

1.2.4 External References

The following external references are made:

[SAMLTC]	The OASIS Security Services Technical Committee. See
[SAMLCORE]	S. Cantor et al. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-core-2.0-os. See http://www.oasis-open.org/committees/security/.
[SAMLPROF]	S. Cantor et al. Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-profiles-2.0-os. See http://www.oasis-open.org/committees/security/.
[SAMLBIND]	S. Cantor et al. Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-bindings-2.0-os. See http://www.oasis-open.org/committees/security/.
[SAML-XSD]	S. Cantor et al., SAML assertions schema. OASIS SSTC, March 2005. Document ID saml-schema-assertion-2.0. See http://www.oasis-open.org/committees/security/
[SAMLP-XSD]	S. Cantor et al. SAML protocols schema. OASIS SSTC, March 2005. Document ID saml-schema-protocol-2.0. See http://www.oasis-open.org/committees/security/.
[SAMLMETA]	S. Cantor et al. Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-metadata-2.0-os. See http://www.oasis-open.org/committees/security/.

[SAMLTechOvw]	J. Hughes et al. SAML Technical Overview. OASIS, February 2005. Document ID sstc-saml-tech-overview-2.0-draft-03. See http://www.oasisopen.org/committees/security
[SAMLGloss]	J. Hodges et al. Glossary for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-glossary-2.0-os. See http://www.oasis-open.org/committees/security/.
[SAML2SECC]	F. Hirsch et al. Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0 OASIS SSTC, March 2005. See http://docs.oasis-open.org/security/saml/v2.0/saml-sec-consider-2.0-os.pdf
[SAMLCTX]	J. Kemp et al. Authentication Context for the OASIS Security Assertion Markup Language (SAML) V2.0, March 2005. See http://docs.oasis-open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf
[SSL3]	A. Frier et al. The SSL 3.0 Protocol. Netscape Communications Corp, November 1996.
[RFC1951]	P. Deutsch. DEFLATE Compressed Data Format Specification version 1.3 IETF RFC 1951, May 1996. See https://www3.ietf.org/rfc/rfc1951.txt
[RFC2045]	N. Freed et al. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies IETF RFC 2045, November 1996. See https://www3.ietf.org/rfc/rfc2045.txt
[HTTP11]	R. Fielding et al. Hypertext Transfer Protocol -- HTTP/1.1 IETF RFC 2616, June 1999
[RFC2246]	T. Dierks. The TLS Protocol Version 1.0. IETF RFC 2246, January 1999. See http://www.ietf.org/rfc/rfc2246.txt.
[ISO-P2H]	ISO/IEC Directives, Part 2, Annex H http://www.iec.ch/tiss/iec/Directives-part2-Ed5.pdf
[RFC4346]	T. Dierks et al. The Transport Layer Security (TLS) Protocol Version 1.1 RFC 4346, April 2006
[RFC 5280]	D. Cooper et al. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile IETF RFC 5280, May 2008
[SANSPP]	SANS Password Policy - http://www.sans.org/resources/policies/Password_Policy.pdf
[CAList]	CA Forum Cert Authority List URI

Table 2: External References

2 Introduction

This document specifies security mechanisms for use within the DECE Ecosystem. This includes mechanisms for authentication, integrity, and confidentiality protection, and the means for sharing information necessary for performing authorization decisions. The mechanisms build on accepted technologies including SSL [SSLv3], TLS [RFC4346], HTTP Authentication mechanisms, and SAML assertions. HTTP request headers [HTTP11] are used for message-level security, to communicate relevant security information, for example using SAML [SAMLCORE] assertions, along with the protected message.

Many of the DECE protocol messages to the Coordinator require that Users consent to explicit Delegations to Nodes, in order for the Node to communicate to the Coordinator on the Users behalf. These Delegations are recorded with the Coordinator, and require interactions with the User for their establishment. The result of a successful Delegation is a Security Token, introduced in Section 4, and an associated policy as defined in [DCoord] Section 5.

Delegations may be established for prescribed periods of time, ranging from short-lived Delegations to persistent, long-lived Delegations.

The general security requirements are specified in Sections 3 and 4. Specific security profiles are specified in Sections 5 and 6, allowing the future addition of security profiles using other methods.

3 DECE Security Requirements

This chapter establishes the transport and storage security requirements for communications between Nodes and the Coordinator, between Devices and the Device Portal, and between user agents and the Web Portal.

3.1 Common Requirements (informative)

The following apply to all mechanisms in this specification, unless specifically noted by the individual mechanism.

Messages may need to be kept confidential and inhibit unauthorized disclosure, either when in transit or when stored persistently. Confidentiality may apply to the entire message, payload, or XML portions depending on application requirements.

Messages may need to arrive at the intended recipient with data integrity. HTTP intermediaries may be authorized to make changes, but no unauthorized changes should be possible without detection. Integrity requirements should apply to the entire message, payload, or XML portions depending on application requirements.

The authentication of a message sender and/or initial sender may be required by a receiver to process the message. Likewise, a sender may require authentication of the response.

Protection against replay or substitution attacks on requests and/or responses may be needed.

The privacy requirements of the participants with respect to how their information is shared or correlated must be met, and will appear in this specification, [Dsystem], or [DGeo].

3.2 Confidentiality and Privacy Mechanisms

Some service interactions described in this specification include the conveyance of information that is only known by a trusted authority and the eventual recipient of a resource access request. This section specifies the measures to be employed to attain the necessary confidentiality and privacy controls.

3.2.1 Transport Layer Channel Protection

When communicating peers interact directly (i.e., no active intermediaries in the message path) then transport layer protection mechanisms may suffice to ensure the integrity and confidentiality of the message exchange.

Messages between sender and recipient SHALL have their integrity protected and confidentiality SHALL be ensured. This requirement SHALL be met with suitable SSL/TLS cipher suites. The security of the SSL or TLS session depends on the chosen cipher suite. An entity that terminates an SSL or TLS connection needs to offer (or accept) suitable cipher suites during the handshake. The following list of TLS 1.0 cipher suites (or their SSL 3.0 equivalent) is recommended:

- TLS_RSA_WITH_RC4_128_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA

The above list is not exhaustive. The recommended cipher suites are among the most commonly used. New cipher suites using the Advanced Encryption Standard have been standardized by the IETF [RFC3268] and are just beginning to appear in TLS implementations. It is anticipated that these AES-based cipher suites will be widely adopted and deployed.

- TLS_RSA_WITH_AES_CBC_SHA
- TLS_DHE_DSS_WITH_AES_CBC_SHA

For signing and verification of protocol messages, communicating entities SHOULD use certificates and private keys that are distinct from the certificates and private keys applied for SSL or TLS channel protection.

Other security protocols (e.g., Kerberos, IPSEC) MAY be used as long as they implement equivalent security measures.

3.2.2 Confidentiality and Privacy Protection

As much of the data in the DECE Ecosystem is sensitive and private in nature, all communications between entities in the architecture must ensure data privacy, integrity, and end-point authenticity. There are two major origins of communication specified here. The first are the communications amongst Nodes (e.g. Retailers, LASPs, DSPs) and between Nodes and the Coordinator. The second are the communications between a User (via a user agent), DECE Device, or other devices, including LASP Clients. Nodes and Devices are collectively called API Clients. API Clients SHALL ensure that the exchange of Security Tokens occurs in accordance with Section 3.2.1

Communication between a User's user-agent or Device and any Node and communication between Nodes SHOULD employ transport layer channel protection in a manner consistent with Section 3.2.1 above, when such communications involves DECE Data.

3.3 Data Custodial Guidelines (Informative)

The following guidelines serve as recommendations to API Clients for the proper protection of DECE Data:

- Controls are deployed to protect against unauthorized connections to services (e.g. firewalls, proxies, access control lists, etc.)
- Controls are deployed to protect against malicious code execution (e.g. antivirus, anti-spyware, etc.)
- Controls deployed to protect against malicious code execution are kept up to date (e.g. software version, signatures, etc.)
- Host-based intrusion detection and/or prevention software is deployed and monitored
- Local system accounts that are not being used are disabled or removed
- Default or vendor supplied credentials (e.g. username and password) are changed prior to implementation
- Services that are not being used are disabled or removed
- Applications that are not being used are removed
- Auto-run for removable electronic storage media (e.g. CDs, DVDs, USB drives, etc.) and network drives is disabled
- Active sessions are locked after a period of inactivity
- Native security mechanisms are enabled to protect against buffer overflows and other memory based attacks (e.g. address space layout randomization, executable space protection, etc.)
- Procedures for monitoring for new security vulnerabilities are documented and followed
- Operating system and software security patches are deployed in a timely manner
- Mitigating controls are deployed for known security vulnerabilities in situations where a vendor security patch is not available
- System is periodically tested for security vulnerabilities (e.g. vulnerability scanning, penetration testing, etc.)
- Successful attempts to access Information Systems are logged
- Failed attempts to access Information Systems are logged
- Attempts to execute an administrative command are logged
- Changes in access to an Information System are logged
- Changes to critical system files (e.g. configuration files, executables, etc.) are logged
- Process accounting is enabled, where available
- System logs are reviewed on a periodic basis for security events
- System logs are protected against tampering

3.4 Authentication

Accurate and secure identification and authentication of API Clients and DECE Users is required to ensure controlled access to all DECE resources and data.

3.4.1 User Authentication

Users are authenticated via their Coordinator managed User Credential or a defined Security Token. Users shall be authenticated directly using one of the prescribed User Credential Profiles or indirectly using a defined Authentication Security Token Profiles.

The Coordinator SHALL provide at least one authentication mechanism used to uniquely identify Users to the Coordinator, Nodes and Devices. In addition, the Coordinator SHALL provide at least one authentication mechanism for API Clients to acquire a Security Token representing the User.

All Security Token and User Credential exchanges SHALL occur over TLS/SSL [TLS].

The following User Resource Statuses SHALL NOT be successfully authenticated by the Coordinator: `urn:dece:type:status:deleted` and `urn:dece:type:status:forceddeleted`. Other statuses may prevent or minimize User activities, but shall be allowed to successfully authenticate.

The minimum size of any graphical control dialogue employed on a general purpose computer SHALL be 350 pixels wide by 600 pixels high. Devices and other clients do not have any specific dimension requirements, as their capabilities vary significantly, however, it shall be suitable to display the necessary form controls, and other contextual information which may include brand and assistive language.

Node must have a “reasonable policy” for maximum time between ID verifications and a SecurityMechanisms-C1.0.5maximum session idle time, as it relates to locally employed user authentication.

Nodes are required to be capable of authenticating a User.

Nodes MAY use a username and password to authenticate Users (independent of the credentials stored with the Coordinator), in which case, these security tokens must be at least 4 chars in length. Other mechanisms to authenticate a user may be used, but must meet or exceed this minimum security requirement.

Coordinator or Web Portal may invalidate any outstanding Security Token(s) for a User if it detects a security threat or potential fraudulent use, using the revocation methods defined for the applicable Security Token.

3.4.2 Node Authentication

Nodes SHALL be authenticated to the Coordinator via a TLS server certificate issued by the Coordinator provided Certificate Authority. This certificate SHALL conform to [RFC 5280].

Nodes SHALL verify the Coordinator's certificate validity and status as specified by the certificate.

The Coordinator SHALL be authenticated to Nodes and Devices via a TLS server certificate issued by the Coordinator provided Certificate Authority.

The Coordinator SHALL verify a Node's certificate validity and status as specified by the certificate, and maintain the association of that certificate with the applicable Node.

The NodeID of the Node SHALL be included in the certificates Subject Distinguished Name (DN) and at a minimum SHALL contain the following DN attributes:

- Common Name (CN): the NodeID of the Node
- Organization (OU): the Registered Business name of the organization
- Country (C): the Country of organization
- Additional identifying Subject DN attributes, such as the Organizational Unit (OU), State (ST), and Locality (L) MAY be included.

Nodes that interact with Users SHALL obtain Extended Validation Certificates (EV Certs) as defined in [EVCert]. The Certificate Authorities employed for such certificates SHOULD be one of those commonly distributed with user agent clients. A list of these CA's can be found in [CAList].

Certificates employed for Coordinator API calls SHALL be obtained from the Coordinator Certificate Authority. The CN relative distinguished name of the subject of the certificate shall be used by the Coordinator to identify the Node as a valid bearer of Security Tokens presented to the Coordinator APIs.

3.4.3 Requirements for the Prevention of Automated Attacks

The Coordinator and Nodes may be required to implement defenses against replay attacks, denial of service attacks and attempts by attackers to test for potential valid values stored at the Coordinator. To achieve this, a combination of Reverse Turing Test and other strategies will be required to frustrate such attacks. Such tests must meet the following minimum requirements:

- The test SHALL make reasonable attempts at distinguishing automated use from human use of the service

- The tests SHOULD provide mechanisms to impede the progress of the use of the service by the introduction of delays in the service between a failed attempt and the next attempt,

- The test SHALL be provided in a way that meets appropriate regional requirements for accessibility, and may be clarified in [DGeo]

APIs defined in [DCoord] may indicate a Reverse Turing Test be required, in which case, the test SHALL be required after DSECMECH_FAILED_AUTHN_ATTEMPTS failures within DSECMECH_AUTHN_ATTEMPT_PERIOD.

Examples of strategies include: CAPTCHA challenges, picture-based CAPTCHA challenges, prose-based challenges ("What is the third word in the sentence above"), knowledge-based challenges, etc.

3.5 Handling of Security Tokens

Security Tokens that are employed as bearer tokens SHALL be stored in a secure fashion, such that its confidentiality and integrity can be reasonably achieved. This may include local encryption, secure file systems, or other mechanisms. This is especially true of Device storage of Security Tokens (including the SAML Tokens defined in section 5 and the Username/Password tokens defined in section 6.

Entities, including Nodes and Devices, that maintain local persistent storage of Security Tokens SHALL ensure such tokens are removed from all persistent caches and other storage medium when instructed to do so by the Coordinator (e.g. Security Token Revocation in section 5.10), or as a consequence of a Device Leave operation as defined in [DDevice] section 4.2.

3.6 User API Authorization

Security Tokens may be employed, as allowed for in the individual profile, by devices. In particular Licensed Clients are able to present Security Tokens to certain Coordinator APIs as defined in [DCoord].

3.7 Coordinator Security-related endpoints

Security endpoints are provided as part of the onboarding process with the Coordinator. The details of these URLs will be shared with Nodes during initial setup and onboarding.

The structure of the endpoints is defined as part of specific Delegation or Federation Security Token Profiles (including those specified in Security Token Profiles specified in Section 8).

Except when specifically defined by a particular Security Token Profile, the following structure will be used to define security-related endpoints:

```
[securityBaseHost] = DGEO_API_DNSNAME

[type] = “/security”

[purpose] = [“delegation” | “federation”]

[proto] = [“saml”]

[sHost] = s.[securityBaseHost]

[baseURL] = https://[sHost][type]/[purpose]/[proto]
```

For example, the SAML Delegation Security Token Profile might utilize a base URL of

```
https://s.uvvu.com/security/delegation/saml
```

whereas the SAML Federation Security Token Profile would utilize a base URL of

```
https://s.uvvu.com/security/federation/saml
```


4 Security Token Profiles

Security Tokens are employed in DECE protocol messages to demonstrate Delegation by the User to a Node or Device, to act on their behalf, or to enable the unique identification of a User (as is the case with User Credentials).

The following sections discuss the common requirements for all Security Tokens, a framework for defining new profiles, and an initial set of profiles. Additional profiles may be added and specified here or in another DECE publication.

4.1 Security Token Profile Common Requirements

Nodes and other clients that are authorized or required to query and provision data within the Coordinator SHALL utilize a valid Security Token to identify the invoking User. These tokens represent a Delegation by the User to the Node, authorizing the Node to query and provision with the Coordinator on the User's behalf.

To successfully process Security Token requests by API Clients, the Coordinator SHALL authenticate the User in a manner specified in the Security Token Profile.

Whenever the Coordinator receives a Security Token request message, the Coordinator SHALL collect or confirm the User's acknowledgement of the Delegation to the requesting Node and this acknowledgement is conveyed in the response message in the manner specified in the profile. While each Security Token Profile differs in how this consent is conveyed, each Profile will define how it is encoded in the token.

4.1.1 Roles Requiring Security Tokens

The following Roles SHALL utilize Security Tokens, to be authorized for use of Coordinator APIs:

Role Identifiers
urn:dece:role:dece:customersupport
urn:dece:role:retailer
urn:dece:role:retailer:customersupport
urn:dece:role:lasp
urn:dece:role:lasp:linked
urn:dece:role:lasp:linked:customersupport
urn:dece:role:lasp:dynamic
urn:dece:role:lasp:dynamic:customersupport
urn:dece:role:dsp

Role Identifiers
urn:dece:role:dsp:customersupport
urn:dece:role:device
urn:dece:role:device:customersupport
urn:dece:role:portal
urn:dece:role:portal:customersupport
urn:dece:role:dece
urn:dece:role:dece:customersupport
urn:dece:role:coordinator:customersupport
urn:dece:role:accessportal
urn:dece:role:accessportal:customersupport

Table 3: Roles requiring Security Tokens

Section 5 of this specification defines one Security Token Profile.

Section 6 defines one User Credential profile.

It is RECOMMENDED that the `urn:dece:role:device` role limit it's use of the User Credential Token Profile, and instead utilize the Security Token Exchange mechanism defined in section 8.1 to exchange the User Credential Token for another token type.

The following policies apply for all Security Token Profiles:

- Unless otherwise defined, the maximum Security Token validity period SHALL be 1 year.
- Consent collection performed by the Coordinator SHOULD clearly identify the longevity of the Security Token and MAY provide options for more than one time duration.
- Security Tokens that are established for a user in a *pending* status SHALL NOT exceed DCOORD_MAX_PENDING_USER_TOKEN_DURATION
- Security Tokens that are established for a user who does not elect to a permanent link (via the establishment of the `urn:dece:type:policy:UserLinkConsent` policy to the Node) SHALL NOT exceed DSECMECH_MIN_TOKEN_DURATION_DEFAULT
- If a User elects to remove the `urn:dece:type:policy:UserLinkConsent` policy for the Node, the corresponding Security Token SHALL be revoked.
- Security Tokens issued exclusively to `urn:dece:role:dsp` role SHALL NOT exceed DSECMECH_MIN_TOKEN_DURATION_DEFAULT

4.1.2 Combining Roles for a Delegation Token

Due to the special restrictions on Security Tokens provided to the LASP roles which are not required for other roles (most notably the LINK_LASP_ACCOUNT_LIMIT limits the number of Security Tokens outstanding for and Account to a Linked LASP), LASP roles SHOULD NOT be combined with other roles, when the Security Token Profile provides a mechanism to share the

Security Token across multiple Nodes within an Organization (e.g., the SAML AudienceRestriction).

If the intention of a Node is to include a Linked LASP, it SHALL include the LASP NodeID in the token request, and the Coordinator or the requesting Node SHALL indicate to the User that the request will consume one of the allowed Linked LASP quota as specified by the LINK_LASP_ACCOUNT_LIMIT defined in [DSystem] appendix A. Note that the Token Duration a Linked LASP will generally be longer than other Nodes (e.g., 10 years), however, it is still the responsibility of the Node to evaluate the validity period of the Delegation Security Token.

4.2 Consent Collection

In order to establish a Security Token, in addition to authenticating a User, the Coordinator SHALL obtain the proper consent from the User, indicating the Users agreement to the Delegation represented by the Security Token. The Coordinator SHOULD indicate to the User the nature of the token request, its purpose, and its lifespan. The acceptance by the User SHALL be conveyed to the Node in manner that must be specified by the token profile being employed.

A record of the agreement by the User is retained by the Coordinator as a Policy, as defined in Section 5 of [DCoord].

The following processing rules apply to all Security Token Profiles consent collection mechanism(s):

The Security Token profile SHALL NOT require the replacement of a delegation token when consent policies are changed.

The Security Token profile SHALL require that the PolicyList resource be used to convey requested policies and established policies.

The Security Token profile SHALL allow all Policy resource elements in its request that are identical to the capabilities and restrictions defined for the PolicyCreate PolicyUpdate and PolicyDelete Coordinator APIs in [DCoord] section 5, however, only the UserLinkConsent and DataSharingConsent policies are evaluated. Other policy requests are accepted but ignored.

4.3 Delegation

Security Token Profiles may specify usage as a Delegation Token, which will be employed by Nodes to covey User identity information during interactions with the Coordinator. Such profiles SHALL specify the processing rules, consent, and durability of such Delegations.

4.3.1 Delegation Scope

Delegation Security Token Profiles may be defined to include mechanisms or procedures for the distribution of a Security Token across multiple Nodes. Implementations SHOULD take reasonable measures to share such tokens in a secure and reliable means.

Because of the need to enforce and convey to users the applicable parties for the establishment of consent policy classes as defined in [DCoord] Section 5.5, the scope of the delegation SHALL NOT cross organization boundaries. That is, within a given organization (in which multiple Nodes may be defined), the set of Nodes identified with a given policy SHALL all be part of the same organization. This does not preclude the provision of services by third parties, rather, such services must operate under the span of control of the Organization.

4.3.2 Delegation Revocation

All Security Token Profiles SHALL specify how a Delegation Token is revoked. A request for revocation may be made by any API Client authorized to wield the Security Token. Methods for revocation may vary greatly from profile to profile, and therefore only the profile-defined mechanisms may be used for a given Security Token. For example, the revocation method defined in the SAML Token Profile section 5.10, employs the SAML SingleLogout profile, which is used to initiate the Delegation Revocation process.

When a Delegation Security Token is revoked, any UserLinkConsent policies in place for the Node initiating the revocation message are preserved. UserLinkConsent policies for other Nodes that may have been authorized to wield the Delegation Security Token are not removed.

If a UserLinkConsent policy is deleted, related Delegation Security Tokens are revoked. Determination of impacted Delegation Security Tokens is done by identifying all named Nodes and Orgs within the UserLinkConsent policy, and selecting all Delegation Security Tokens naming those Nodes as an authorized bearer, including Nodes belonging to a named OrgID in the policy.

4.3.2.1 Delegation Durations

Delegation Security Token durations are issued with the following durations, depending on

a) requesting Node and the audience specified in the request,

b) the existence of the UserLinkConsent policy for the requesting Node:

- When the UserLinkConsent policy is not present:
  DSECMECH_MIN_TOKEN_DURATION_DEFAULT

- When the UserLinkConsent policy is present:

- If the Delegation Security Token requestor is a LLASP: DSECMECH_LLASP_TOKEN_DURATION_DEFAULT
- If a DSP is the only bearer: DSECMECH_MIN_TOKEN_DURATION_DEFAULT
- All other cases: DSECMECH_MAX_TOKEN_DURATION_DEFAULT

A Node request for a longer duration (as documented in section 8.1.4) will be ignored. A Node request for a shorter duration will reduce the duration of the issued Security Token.

Issuance of new Delegation Security Tokens is subject to constraints due to a User's ResourceStatus as follows:

- Users in active status within an account in active status MAY obtain Delegation Security Tokens valid for durations specified above.
- Users in pending status MAY obtain Delegation Security Tokens valid for no longer than DSECMECH_MIN_TOKEN_DURATION_DEFAULT
- Users in blocked:tou status MAY obtain Delegation Security Tokens valid for no longer than DSECMECH_MIN_TOKEN_DURATION_DEFAULT
- Users in any other status SHALL NOT have Delegation Security Tokens issued.

Issued Delegation Security Tokens are affected by change in User status as follows:

- Users moved to the deleted status invalidates outstanding Delegation Security Tokens (see [DCoord] 14.1.4.3)
- Any other transition of User status has no effect on outstanding Delegation Security Tokens

Account deletion invalidates outstanding Delegation Security Tokens for all Users in the Account. Any other change in Account status does not affect outstanding Delegation Security Tokens.

4.3.3 Delegation Token Issuance Impacts to Consent Policies

When Delegation Security Tokens are requested (new or replacements), their issuance may result in the creation or removal of certain consent Policies, involving one or more Nodes in the following manner:

- All successful Delegation Security Token Requests will result in having the LockerViewAllConsent, EnableManageUserConsent, and EnableUserDataUsageConsent

Policies created by the Coordinator for each Node named in the request. If such Policies already are in place, they are not altered.

- If the request for a new Delegation Security Token includes a UserLinkConsent policy request, all Nodes identified in the request will have this Policy created by the Coordinator. If such Policies already are in place, they are not altered.

- If the request for a new Delegation Security Token includes a UserLinkConsent policy request, all Nodes identified in the request will have the LockerViewAllConsent, EnableManageUserConsent, and EnableUserDataUsageConsent Policies created by the Coordinator. If such Policies already are in place, they are not altered.

- If the request would result in altering a previous set of authorized Nodes, replacing the previous Delegation Security Token and includes a UserLinkConsent policy request, Policies for the Nodes that were excluded are not altered.

- If the request would result in altering a previous set of authorized Nodes, replacing the previous Delegation Security Token and does not include a UserLinkConsent policy request, UserLinkConsent Policies for the authorized Nodes that were in the request are deleted. Policies for the Nodes that were excluded are not altered.

4.4 Subject Scope of Security Tokens

The scope of a Security Token SHALL be at the level of an individual User. However, some Roles, due to operational characteristics or constraints of the Role, require the subject scope of Security Tokens be evaluated at the Account level by the Coordinator. The Coordinator SHALL evaluate Security Tokens at the Account level for the following Roles:

- All Customer Support roles
- Linked LASPs
- Devices

All other Roles will have the presented Security Token evaluated in the context of the User represented in the token.

4.5 Guidelines for Specifying Security Token Profiles

This section provides a checklist of issues that SHALL be addressed by each profile.

Specify a URI that uniquely identifies the profile and provide reference to previously defined profiles that the new profile updates or obsoletes.

Specify if the profile is for Delegation, Authentication or both.

Describe the set of interactions between parties involved in the profile. Any restrictions on applications used by each party and the protocols involved in each interaction must be explicitly called out.

Specify applicable HTTP WWW-Authenticate challenge response values as required by [DCoord] section 2.3.2

Identify the parties involved in each interaction, including how many parties are involved and whether intermediaries may be involved.

Specify the method of authentication of parties involved in each interaction, including whether authentication is required and acceptable authentication types.

Identify the level of support for message integrity, including the mechanisms used to ensure message integrity.

Identify the level of support for confidentiality and whether the profile requires confidentiality, and the mechanisms recommended for achieving confidentiality.

Identify the error states, including the error states at each participant.

Identify security considerations, including analysis of threats and description of countermeasures.

Identify any required confirmation methods specific to the profile.

Identify relevant metadata required by a Node that shall be required by the profile.

Extend, as required, any necessary extensions to the Security Token Service specified in section 8.

5 Security Assertion Markup Language (SAML) Token Profile

This profile specifies the application of Security Assertion Markup Language (SAML) [SAMLTC] Assertions for use as Delegation Security Tokens for API Clients in order to communicate User identity and Account identifiers to the Coordinator in Coordinator API endpoints.

Section 5.3 defines the request protocol. Section 5.6 defines the response protocol.

These tokens are then composed with Coordinator protocol messages using the HTTP Authorization Binding specified in Section 5.11 to demonstrate the Delegation between the API Client and the Coordinator by the User.

An assertion is a package of information that supplies zero or more statements made by a SAML authority; SAML authorities are sometimes referred to as asserting parties in discussions of assertion generation and exchange, and system entities that use received assertions are known as relying parties. (Note that these terms are different from requester and responder, which are reserved for discussions of SAML protocol message exchange.)

SAML assertions are usually made about a subject, represented by the <Subject> element. Typically there are a number of service providers that can make use of assertions about a subject in order to control access and provide customized service, and accordingly they become the relying parties of an asserting party called an identity provider.

The SAML technical overview [SAMLTechOvw] and glossary [SAMLGloss] provide more detailed explanation of SAML terms and concepts.

5.1 SAML Assertion as Delegation Token

This profile of SAML describes the use of a SAML Assertion (“Security Token”) in DECE protocol messages between API Clients and the Coordinator. Schema for the Security Token is defined by [SAML-XSD] and [SAMLP-XSD]. The Security Token is provided by the Coordinator within the SAML response message. The Security Token performs 2 functions:

Acts as a Delegation bearer token for use by authorized entities as an indication of consent

Conveyance of subject data (specifically, the UserID and the AccountID) that are used to compose protocol messages to the Coordinator.

This Security Token may be wielded by more than one Node (described by the audience restriction), and may also be borne by Devices, in order to authenticate such Devices to the Coordinator.

Devices SHOULD provide a secure storage facility for such Security Token, inaccessible to other applications, other than the applications necessary for Node interactions.

5.2 Profile Required Information

Identification: urn:dece:type:tokenprofile:saml2

Updates: None

Purpose: This profile may be used for Delegation and Authentication

Description: See Section 5.3

Authorized Roles: any role identified in section 4.1.1

WWW-Authenticate challenge: SAML2

Security Message Protocol Endpoints: the Coordinator SAML metadata defines the specific endpoints for Delegation Security Token requests for each of the supported bindings.

5.3 Overview of SAML Request / Response Messages (Non-normative)

The following diagram depicts the protocol exchange between the Node, the user agent client and the Coordinator, and covers positive outcome flows only:

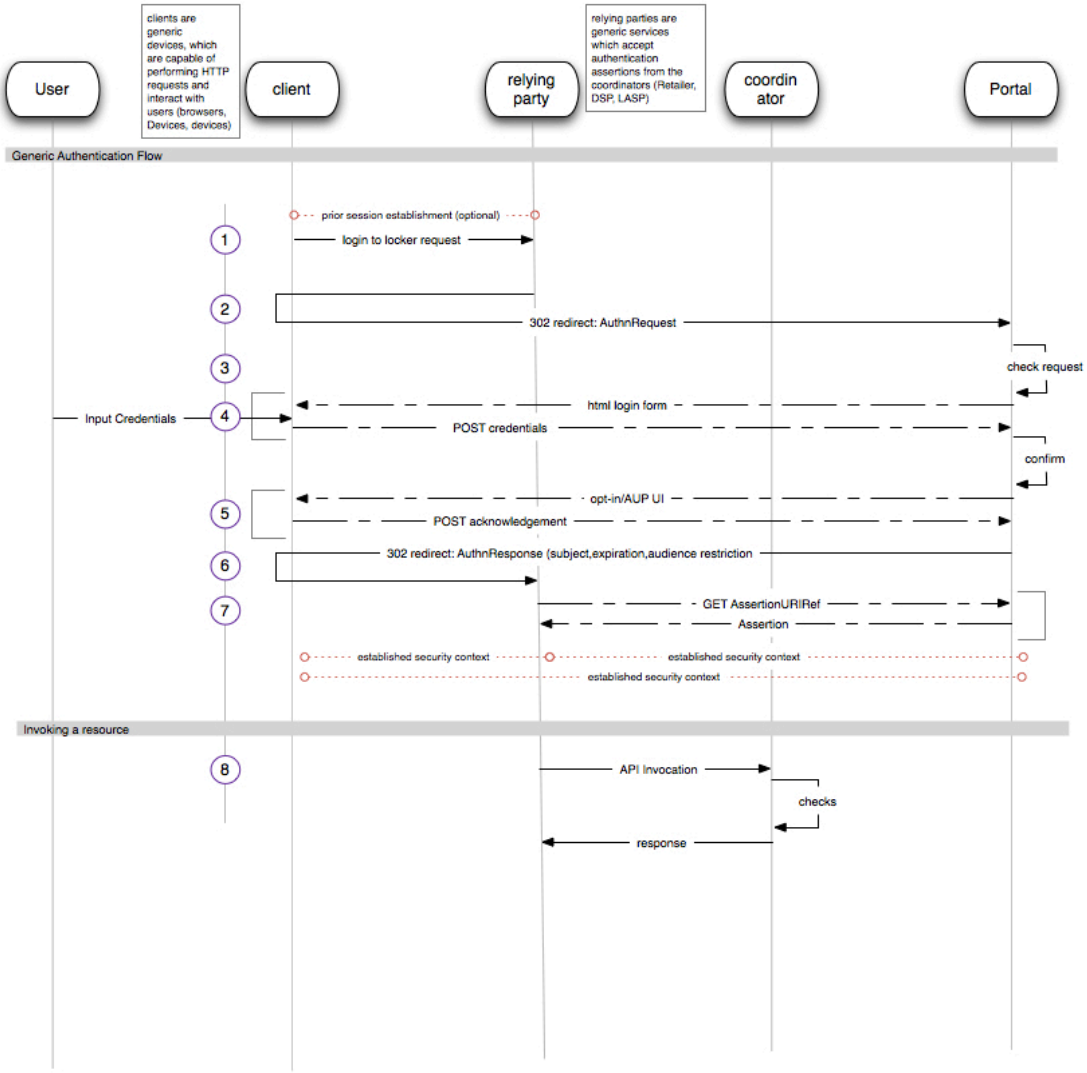

Figure 1: SAML Request and Response sequence

The details of the steps identified in the figure are as follows:

1. The User, via the user agent client, indicates to the SAML relying party (Node) that a persistent or temporary Delegation is desired

2. The relying party (SAML Requestor) forms a signed SAML Request using one of the message bindings specified in Section 5.5 targeted to the ‘s’ host

3. The ‘s’ host verifies the request including the authentication of the SAML Requestor

4. The ‘s’ host conditionally presents to the user agent client an authentication challenge for the collection of User Credentials, which:

a. Has a representation suitable for display to the user agent client, which may include HTTP Basic or forms-based authentication

b. The 's' host may incorporate through the initial representation, any necessary consent agreements required to fulfill the SAML Request, either as indicated in the SAML request, or as may be required by the applicable Geography Profile defined in [DGeo].

5. Any consent agreements collected in step 4 are submitted to the Portal

6. The 's' host conditionally presents to the user agent client in a representation suitable for display to the user agent client a resource to collect any necessary agreements relating to the SAML Request, or usage of UltraViolet

7. The 's' host verifies the User Credential, the necessary consents and agreements, and forms a SAML Response targeted at the SAML Requestor using one of the message bindings specified in Section 5.5

8. If the SAML Response utilizes the SAML URI Reference Binding, the SAML Requestor dereferences the resource, and obtains the SAML Assertion from the 's' host

9. For subsequent interactions with the Coordinator, the API Client incorporates the SAML Assertion in the request to the Coordinator using the HTTP Authorization Binding specified in Section 5.12.

5.4 General Constraints on SAML Tokens

The use of SAML as a Security Token requires that the token validity period be established in a manner that does not introduce unnecessary risks to the system. The limits defined in Section 4.1 shall apply to this profile.

All SAML messages SHALL be signed by requestors and responders to ensure message integrity and authenticity of the sender and the recipient. These signing keys are exchanged during initial Node provisioning into the Coordinator, and are expressed in SAML Metadata, detailed in Section 5.11

5.5 SAML Assertion Request

The process of obtaining assertions from the Coordinator shall use the SAML2 Web Browser SSO Profile [SAMLPROF], which uses browser URL encoding or HTML Form encoding of assertion requests and responses to convey SAML Assertions.

Using an existing HTTP interaction between a User and the Node ('Service Provider'), the Service Provider constructs the SAML Assertion Request following the requirements of Section 4.1 Web Browser SSO Profile of the SAML Profiles specification [SAMLPROF].

The binding employed by requestors (Nodes) SHALL be either the POST or Redirect Binding (depicted in Figure 1) as defined by [SAMLBIND]. When determining the desired SAML Binding to employ, implementers should be aware that the Redirect Binding may require very long request URLs. Some HTTP User Agents may have limitations on URL lengths. For that reason, Nodes SHOULD use the POST Binding, when possible.

Nodes SHALL specify, during certification and enrollment with the Coordinator, which response bindings are supported, and their associated protocol endpoints. Node SAML Metadata [SAMLMETA] is detailed in see Section 5.11. This metadata is managed and maintained by the Coordinator (and provisioned at the time the Node is certified for Coordinator interactions).

The Coordinator SHALL support the following response bindings:

- the HTTP POST Binding specified in [SAMLBIND] Section 3.5
- the HTTP Redirect Binding specified in [SAMLBIND] Section 3.4
- the SAML URI Binding specified in [SAMLBIND] Section 3.7

SAML requests SHALL be signed with the keys provided to the Coordinator by the Node, as defined in SAML Metadata [SAMLMETA].

SAML Requestors and SAML Responders NEED NOT include x509Data in their use of SAML protocols. Nodes SHALL verify protocol and Assertion signatures using x509Data information in the SAML metadata document provided to the Node by the Coordinator.

Requestors and responders SHALL include a Cache-Control header field set to "no-cache, no-store".

Requestors and responders SHALL include a Pragma HTTP header field set to "no-cache".

The Destination XML attribute in the root SAML element of the protocol message SHALL contain the URL to which the sender has instructed the User agent to deliver the message. The recipient SHALL then verify that the value matches the location at which the message has been received.

All Node SAML Endpoints SHALL use SSL 3.0 [SSL3], TLS 1.0 [RFC2246] or TLS 1.1 [RFC4346] to maintain confidentiality of the messages. Certificates SHALL conform to the requirements of Section 3.4.2.

Requestors SHALL include the ID attribute in a request, and the responder SHALL indicate that ID in the responses inResponseTo attribute.

In addition to the above browser-based bindings, [DCoord] section 14 allows Delegation Security Token requests to submitted directly to the Coordinator, initiating an email-based interaction with a user. Nodes SHALL use the SAML POST binding to properly encode the request body for the UserValidationTokenCreate API, including the use of the "application/x-www-form-urlencoded" Content-type header. This is normally performed through a browser, so Nodes will need to properly encode the SAML request as would a browser, including name-value pairs in the API request body. The SAMLRequest parameter is required. The RelayState parameter is optional. An example can be found in [SAMLBIND] section 3.5.8.

5.5.1 SAML Assertion Request Message Elements

The assertion request messages contain elements from both the [SAML-XSD] and [SAMLP-XSD] schema. The semantics and processing rules found in [SAMLCORE] SHALL be used. This profile further refines the processing requirements of the request as follows:

samlp:AuthnRequest@Version : SHALL have the value "2.0"
samlp:AuthnRequest@IssueInstant : SHALL be the time instant the request was formed, conform to processing rules specified in [SAMLCORE] Section 1.3.3, except for relaxing time granularity, such that requestors and responders SHOULD NOT rely on time resolution finer than seconds.
samlp:AuthnRequest@ForceAuthN : Requestors MAY request the Coordinator to re-authenticate a User at the Coordinator (thus producing a fresh Assertion).
samlp:AuthnRequest@IsPassive : Requestors MAY request that the Coordinator not interact with a User in a noticeable fashion by providing this attribute. However, if the present security context between the User and the Coordinator has expired, the Coordinator SHALL respond with a second-level SAML error response code:
`urn:oasis:names:tc:SAML:2.0:status:NoPassive`
samlp:AuthnRequest@AssertionConsumerServiceIndex : Specifies which requestor endpoint described in [SAMLMETA] shall be used for the response. This endpoint SHALL have been already identified by the requestor in their metadata. Omission of this attribute will result in the response being returned to the endpoint indicated as the default endpoint in metadata for the requestor
samla:Issuer : SHALL be the entity identifier for the Node (NodeID)
samla:Conditions/samla:AudienceRestriction/samla:Audience : if the requestor requires that the SAML assertion be shared amongst a set of affiliated Nodes, these Nodes SHALL be identified in SAML metadata via the AffiliationDescriptor (and defined in Section 5.11 below) and SHALL utilize the Coordinator supplied identifiers for these entities

samlp:RequestedAuthnContext/samla:AuthnContextClassRef : this version of the SAML Token Profile specifies support for the authentication class:
`urn:oasis:names:tc:SAML:2.0:ac:classes:Password`
samlp:RequestedAuthnContext@Comparison : indicates the relative comparison of the requested authentication context with those authentication mechanisms the Coordinator is capable of supporting. Future versions of this specification may provide for additional contexts, and in so doing shall specify the relative ranking of each context employed by an entity.

Requestors SHALL adhere to the precise encoding strategies defined for the Redirect binding ([SAMLBIND] Section 3.4.4) and POST Binding ([SAMLBIND] Section 3.5.4) for SAML messages.

5.5.2 Processing Requirements for SAML Requests

Upon receipt of a SAML Request from a Node, the Coordinator SHALL:

- Verify the signature of the request, and verify the Node is authorized to send such a request
- Map the identity of the requestor to a valid Node and Organization
- Verify the mapping between the Node's SAML EntityID, the subject of the Node's TLS certificate which is used for API invocations at the Coordinator, and the DECE Node identifier and Organizational Identifier (the syntax for which is defined in [DSystem] Section 5.
- Authenticate the User, if required and permitted by IsPassive directive of the request
- Obtain consent from the User, if required, in order to establish a permanent link (allowing the Node to persistently store the SAML Token)
- Ensure the User has acknowledged the most recent end-User license agreement(s) (See [DCoord] section 5.5.2)
- Verify that the requested audience corresponds with an established affiliation, as provided for in the SAML metadata of the Node. If a request includes Audience member Nodes which are not eligible to be included in a SAML Audience restriction, the Coordinator will remove the Node from the Audience list, and continue to process the request. If no Nodes remain in the Audience restriction, and the requestor is also not eligible, the Coordinator will respond with the appropriate SAML protocol error message to the requestor.

5.6 Creation of the SAML Token Response

During the assertion request message handling, the Coordinator SHALL:

- Establish the identity of the Subject (User) involved in the authentication request (by directly authenticating the User, if required by policy, explicitly in the requester's message, or by User preferences and Coordinator policy). This is accomplished using the User Credential Token Profile defined in Section 6 or through HTTP Basic or Forms-based authentication. The Coordinator shall select from these methods based on the capabilities of the User's user-agent.
- Ensure the Subject has agreed to a token exchange with the Node, and record urn:dece:type:policy:UserLinkConsent policy, if requested by the Node and not declined by the User, as a Policy for the Policy Class as defined in [DCoord] Section 5.1.2
- Authenticate the Requestor (Node) by evaluating the signature on the request, which SHALL match the corresponding signing key identified in the Node's SAML metadata

The Coordinator shall then produce an appropriate assertion targeted at the requestor's requested audience. The Subject of this assertion SHALL BE the authenticated User, and will be delivered to the requestor using the response transport binding specified in their metadata to the requested AssertionConsumerServiceIndex or the default AssertionConsumerService endpoint if the endpoint index is omitted from the request. The details of the token are specified below in section 5.7.

Any outstanding Delegation Security Token issued to the requesting Node SHALL be deleted by the Coordinator if the SAML response contains a new Assertion.

5.7 SAML Response Elements

In response to assertion requests, the Coordinator SHALL verify the identity of the requestor, and SHALL verify the intended audience is identical or narrower than the requesters affiliation definition in SAML metadata, and SHALL verify a security context with the User bearing the request.

Responses to valid, verified requests are detailed in the following sections.

5.7.1 Assertions

Issuer: The `<Issuer>` element conveys the entity who produced the assertion (in this case, always the Coordinator), and shall be of type `urn:oasis:names:tc:SAML:2.0:nameid-format:entity`

For example:

```
<saml2:Issuer
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:entity">http://c.d
ecellc.com/</saml2:Issuer>
```

Advice/AssertionURIRef: used to convey the URI reference to the assertion. Only authenticated Nodes cited in the audience restriction may obtain the assertion located at this reference endpoint. Employed when the intended recipient specifies support for the SAML URI Binding in metadata, and is always employed when the Security Token Exchange is used.

Subject: Conveys the details of the described entity of the assertion (the User).

NameID: The <NameID> element shall be used to convey the subject of the assertion. It SHALL be of type `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`. This identifier, SHALL be unique to the audience the token was issued to. The `NameID` identifies the User to the Node and the Coordinator, and is unique in the Coordinator-Node namespace. It will be provided in a form suitable for direct insertion into API invocation requests.

For example:

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">abcxyz93nd90wjdos</saml2:NameID>
```

SubjectConfirmation: The subject confirmation conveys the mechanism by which the recipient can confirm the subject of the message with the entity which the recipient is communicating with. The Coordinator SHALL support the methods:

- `urn:oasis:names:tc:SAML:2.0:cm:sendervouches`, and
- `urn:oasis:names:tc:SAML:2.0:cm:bearer`.

SubjectConfirmationData: Requestors SHALL verify the validi ty of the `InResponseTo`, `NoOnOrAfter` and `Recipient.` If

For Example:

```
<saml2:SubjectConfirmation
Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
<saml2:SubjectConfirmationData
InResponseTo="_someuniqueidhere"
NotOnOrAfter="2010-02-21T23:17:15.203Z"
Recipient="http://www.example.com" />
</saml2:SubjectConfirmation>
```

5.7.2 Conditions

Conditions convey the validity period of the assertion and authorized relying parties to the assertion. The Coordinator shall perform verification that the wielder of the Security Token is authorized.

NotBefore: The dateTime value after which the assertion may be used and considered valid

NotOnOrAfter: The dateTime value after which the Security Token SHALL be discarded and considered invalid, and a new token should be obtained

AudienceRestriction: An enumeration of <Audience> entities who are authorized by the Coordinator to wield the Security Token and employ it in protocol messages to the Coordinator

For example:

```
<saml2:Conditions NotBefore="2011-06-22T19:27:09.179Z"
NotOnOrAfter="2012-06-22T19:27:19.179Z">
<saml2:AudienceRestriction>
     <saml2:Audience>urn:dece:org:org:dece:retailer:acmesto
re</saml2:Audience>
     <saml2:Audience>urn:dece:org:org:dece:lasp:acmestore</
saml2:Audience>
</saml2:AudienceRestriction>
</saml2:Conditions>
```

5.7.3 Advice

Assertion Advice element contains any additional information that the SAML authority wishes to provide. This information MAY be ignored by applications without affecting either the semantics or the validity of the assertion.

Advice/AssertionURIRef: The URI from which the token may be re-obtained. Only entities cited in the Assertion/AudienceRestriction may obtain the token from the Coordinator.

AuthNStatement: Conveys details of the authentication mechanism used to identify the subject.

AuthnInstant: the dateTime when the User was authenticated by the Coordinator.

AuthNContext: the mechanism used to authenticate the User. Defined values are:

- urn:oasis:names:tc:SAML:2.0:ac:classes:Password
- urn:oasis:names:tc:SAML:2.0:ac:classes:Session
- urn:oasis:names:tc:SAML:2.0:ac:classes:x509

5.7.4 AttributeStatement

The attribute statement SHALL convey the Coordinator managed AccountID for the associated User, which is suitable for use in the construction of certain Coordinator API endpoints. This attribute will be named "accountid", indicated in the <Attribute> element, it's NameFormat will be indicated as urn:dece:type:accountid, and its value shall be of type xs:string This AccountID, as with the Coordinator userID expressed in the <Subject>, SHALL be unique in the Coordinator-Node (or affiliation) namespace.

Example:

```
<saml2:AttributeStatement>
<saml2:Attribute Name="accountID"
NameFormat="urn:dece:type:accountID">
     <saml2:AttributeValue
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">urn:dece:accountid:org:dece:A5F2CD62D2
```

```
6CDB9BE0405B0A0B3464B0</saml2:AttributeValue></saml2:Attrib
ute></saml2:AttributeStatement>
```

5.7.5 Protocols

Status/StatusCode: provides an indication of SAML Protocol errors, which are defined in [SAMLCORE] (see also section 5.7.7)

Status/StatusMessage: a textual message, which may be returned to a requestor

5.7.6 Response

The Response portion indicates information pertaining to the responder, and includes:

Destination: identifies the intended recipient identifier

ID: a unique identifier for the response body, suitable for incorporation in as a signature reference

InResponseTo: indicates the Request Message ID to which this response is associated with

IssueInstant: the time instant the response was formed (this is not the issueInstant of the Assertion itself)

Version: the SAML protocol version

Example:

```
<saml2p:Response
xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
Destination="http://www.example.com"
ID="acmeidp1266793933406"
InResponseTo="someuniqueidhere"
IssueInstant="2010-02-21T23:12:15.203Z"
Version="2.0">
```

5.7.7 Handling Authentication Failure

If the Coordinator fails to be able to authenticate the User, it shall respond with an appropriate SAML second-level protocol error as discussed on [SAMLCORE] section 3.4.1.4:

- `urn:oasis:names:tc:SAML:2.0:status:AuthnFailed` – indicates that the Coordinator could not authenticate the User. Typically, this is due to the User selecting a cancelling action at the Coordinator.
- `urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal` – indicates that the User selected a credential recovery option, preventing authentication from immediately occurring.

An example of such a message sequence is provided in Figure 2 below, which shows the SAML POST binding, in conjunction with a User indicating they require credentials recovery.

Providing this return path allows the Node to take whatever corrective action is required. If the `urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal` is provided in the response, Nodes SHALL NOT attempt to initiate credential recovery immediately, since this protocol error message specifically indicates that the Coordinator has already done so.

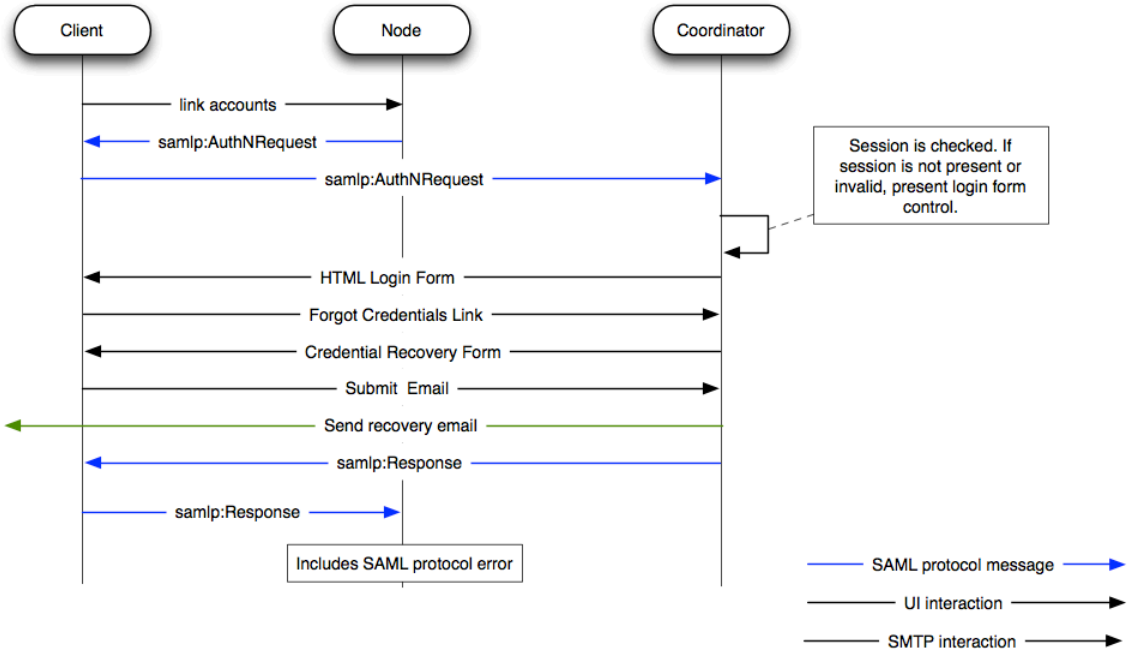

Figure 2: Handling Authentication Failures

5.8 XML Signature Processing

A SAML assertion obtained by a SAML relying party from an entity other than the SAML asserting party SHALL be signed by the SAML asserting party. A SAML protocol message arriving at a destination from an entity other than the originating sender SHALL be signed by the sender.

5.9 Consent Identifiers

It is required that the Coordinator collect consent from a User when a request for a Delegation Token has been made. Consent is collected during the handling of the SAML Request message.

One of the following consent identifiers SHALL be used in any protocol message:

`urn:oasis:names:tc:SAML:2.0:consent:unspecified` - No claim as to principal consent is being made.

`urn:oasis:names:tc:SAML:2.0:consent:obtained` - Indicates that a principal's consent has been obtained by the issuer of the message.

`urn:oasis:names:tc:SAML:2.0:consent:prior` - Indicates that a principal's consent has been obtained by the issuer of the message at some point prior to the action that initiated the message.

`urn:oasis:names:tc:SAML:2.0:consent:current-implicit` - Indicates that a principal's consent has been implicitly obtained by the issuer of the message during the action that initiated the message, as part of a broader indication of consent. Implicit consent is typically more proximal to the action in time and presentation than prior consent, such as part of a session of activities.

`urn:oasis:names:tc:SAML:2.0:consent:current-explicit` - Indicates that a principal's consent has been explicitly obtained by the issuer of the message during the action that initiated the message.

`urn:oasis:names:tc:SAML:2.0:consent:unavailable` - Indicates that the issuer of the message did not obtain consent.

When these consent identifiers are employed in a successful SAML Response that incorporates a SAML Assertion, their meaning shall convey the consent of the User to link their Account with the Node to which the Assertion is issued.

The Coordinator, during the processing of the SAML Request message, SHALL ensure consent is obtained via one of the specified mechanisms above, or SHALL return a SAML Response indicating `urn:oasis:names:tc:SAML:2.0:consent:unavailable` and the appropriate SAML Error.

5.9.1 SAML-based Consent Collection at the Coordinator

[DCoord] section 5.5.3.1 requires that Security Token Profiles specify a mechanism to enable User consent collection, via an HTTP User-agent. This section defines a mechanism using established protocol binding defined in [DSecMech] section 5.5 .

5.9.1.1 General Requirements

When handling the Authentication or Delegation request, the Coordinator shall allow any valid policy or policies which would be allowed in the respective Policy APIs defined in [DCoord] section 5.6, and as allowed for in section 4.2

Any protocol binding defined in section 5.5 may be used to create the request and the response:

- the HTTP POST Binding specified in [SAMLBIND] Section 3.5
- the HTTP Redirect Binding specified in [SAMLBIND] Section 3.4

SAML requests and responses SHALL be signed with the keys provided to the Coordinator by the Node, as defined in SAML Metadata [SAMLMETA].

The requestor identified in the Issuer element SHOULD be named in all requested Policies. All named Nodes in the request SHALL be of the same Organization as discussed in [DCoord] section 2.3.

If Policies in a request result in Policies that need to remain in a pending status (for example, approval by another user is required), Policies are still returned and SHALL include the ResourceStatus/Current indicating the pending status.

The Coordinator shall provide 2 variants of display renderings for handling requests:

- Display of the consent and authentication form controls suitable for full browser display, and
- Display of the consent and authentication form controls intended for use within an embedded display (e.g. an I-Frame)

Requestors may choose which display is desired by selecting the appropriate IDPSSODescriptor indicated in the Coordinators SAML Metadata by the included dece:EmbeddedInteraction attribute. If this attribute is false or omitted, the identified endpoint supports full-browser interactions only. If the attribute is true, the identified endpoint supports the embedded display form.

If one or more Policies are requested via the protocol extensions defined below, the Coordinator SHALL respond with either fully populated Policies or Policy references (that is, <Policy> elements expressing only their PolicyID attribute). Policy references may then be used as defined in [DCoord] section 5.

5.9.2 Protocol Extensions

Protocol extensions, if needed SHALL be placed in the //AuthNRequest/Extensions element. Nodes may include zero or more of these extensions. Some extensions may prohibit certain extension combinations.

5.9.2.1 Policy Request Extension

This extension may be included in an AuthNRequest message sent to the Coordinator.

To include one or more User consent requests in a SAML Delegation Security Token request, the Node MAY include a PolicyList resource in the AuthNRequest/Extension element defined in [SAMLCORE]. For such requests, only the PolicyClass is provided within the Policy element. The Coordinator will populate the remainder of the Policy (that is, the Resource and RequestingEntity values). The following Policy Classes are supported using this method:

- urn:dece:type:policy:UserLinkConsent

This extension allows the inclusion of a PolicyList resource. For such requests, Nodes SHALL only include the `//Policy/PolicyClass` element.

To determine the Token's validity period, the Coordinator provides the User the option of accepting (which is the default state) or refusing the UserLinkConsent policy.

An existing UserLinkConsent policy on the User from the Node has no influence on the handling of this request.

If the UserLinkConsent policy is included in the request, the Coordinator SHALL present a checkbox form control pre-checked along with potentially other form controls, such as credential collection form controls. Upon submission:

- If the UserLinkConsent form control remains checked, the Coordinator SHALL create a UserLinkConsent policy for the requesting Node, and any other Node named in the Audience Restrictions of the SAML message, if the policy is not already present

- If the UserLinkConsent form control is not checked, any UserLinkConsent Policies already established for the requesting Node, and any other Node named in the Audience Restrictions of the SAML message, SHALL be deleted by the Coordinator.

If the UserLinkConsent Policy is not included in the SAML request, no checkbox form control will be presented to the User. Any UserLinkConsent Policies already established for the requesting Node, and any other Node named in the Audience Restrictions of the SAML message, SHALL be deleted by the Coordinator.

This allows a simple Delegation Security Token-only request where UserLinkConsent can be separately collected from the User by the Node.

The resulting Assertion's duration is influenced by the presence or absence of the UserLinkConsent Policy as described in 4.3.2.1.

If the User accepts the policy request, the Coordinator creates the UserLinkConsent policy on the User for the Node, and includes a Delegation Security Token in its response, with a validity period as described in section 4.3.2.1. The Response extension point will also include a PolicyList including a UserLinkConsent policy reference (see section 5.9.1).

If the User declines the policy request, the Coordinator deletes any existing UserLinkConsent policy on the User for the Node, and includes a Delegation Security Token in its response, with a validity period of DSECMECH_MIN_TOKEN_DURATION_DEFAULT.When the Coordinator completes the collection of consent policies requested by the Node, it shall include the list of Policies the User agreed to in the samlp:Response/Extension element in its response to the Node for either the AuthNRequest protocol or the SubjectQuery protocol requests.

The SAML response wll only include Policies related to the SAML request, however, other policies may have been altered during the Coordinator's interactions with the User. These policies will be indicated by reference only (e.g., only indicating the policyID). To discern the outcome of the policy request, Nodes will be required to request the full policy from the Coordinator using the policyID as a request parameter to the PolicyGet() API defined in [DCoord] section 5.

5.9.2.2 Presentation Language Selection Extension

This extension allows the inclusion of a dece:Language element in a SAML request extension.

This extension allows the Node to indicate to the Coordinator the presentation language to be used when rendering user interface elements to a user agent. Its' value SHALL be one of the language identifiers defined in a corresponding DGeo version (e.g. languages identified as supported by the identically versioned DGeo specification).

If included, the `isPrimary` attribute is ignored.

If this request extension is not included, the Coordinator will determine the presentation language based on the provided HTTP `Accept-Language` header as defined in [RFC2616]. If this header is not provided, the Coordinator will default to the language `en-US`.

5.10 Security Token Revocation

The Coordinator shall implement and support the SingleLogout Profile for SAML as defined in [SAMLPROF] Section 4.4. SAML Logout is the means by which Security Tokens are revoked. The message bindings supported for this profile are:

- HTTP Redirect Binding
- HTTP POST Binding

As discussed above, and specified in [SAMLBIND].

As with earlier uses of these bindings, these messages SHALL occur over SSL/TLS.

The single logout protocol provides a message exchange protocol by which all sessions provided by a particular session authority are near-simultaneously terminated. The single logout protocol is used either when a principal logs out at a session participant or when the principal logs out directly at the session authority. This protocol may also be used to log out a principal due to a timeout. The reason for the logout event can be indicated through the Reason attribute.

> **LogoutRequest**: SHALL be signed, and indicates the sender wishes to initiate the termination of session with the recipient, and the recipient SHALL do so, and, in addition, SHALL dispose of the Security Token. Should the recipient require a new Security Token, it SHALL initiate a new login request with the Coordinator.
>
> **LogoutResponse**: The recipient of a `<LogoutRequest>` message SHALL respond with a `<LogoutResponse>` message, of type StatusResponseType, with no additional content specified. The `<LogoutResponse>` message SHALL be signed or otherwise authenticated and integrity protected by the protocol binding used to deliver the message.

If the logout profile is initiated by the Coordinator, or upon receiving a valid `<LogoutRequest>` message from a Node, the Coordinator processes the request as defined in [SAMLCore].

When handling Delegation Token Revocation messages (during SAML Logout), Nodes SHOULD not provide any user interaction.

Nodes SHALL accept the Logout request, validate the request, and respond with either a success or failure response message to the Coordinator.

LogoutRequest messages SHALL include the //NameID element, which identifies the User whose Delegation Revocation is being sought.

The Coordinator SHALL issue `<LogoutRequest>` messages to each Node in the audience scope of the associated, previously issued SAML Assertion, as determined by the Node presenting the `<LogoutRequest>`. Nodes receiving Logout request for which they did not initiate SHOULD handle the logout message according to SAML Logout profile guidelines, and return the User to the SAML Authority (Coordinator).

Upon receiving a valid, signed `<LogoutRequest>`, Nodes SHALL dispose of any associated Security Token for the subject User. This does not require that any sessions established solely between the Node and the User needs to be terminated, however.

Under circumstances where the User (SAML Subject) is not present, the Coordinator SHALL accept the logout request, however other audience members identified in the Assertion cannot be notified by the Coordinator. Nodes MAY use other means to notify audience members that the Assertion is no longer valid.

The Coordinator SHALL NOT accept API invocations that include a SAML Assertion that has been deleted.

5.11 Required SAML Metadata

The following minimal required information is necessary for the Coordinator to receive, confirm, and provision for the purposes of servicing Node assertion requests and for the proper authorization of Node invocations of the Coordinator API. Each Node which requires a Security Token SHALL provide this metadata to the Coordinator.

samlmd:EntityDescriptor@entityID : the Coordinator issued organization identifier for the Node (identical to NodeID)

samlmd:SPSSODescriptor@protocolSupportEnumeration : its value SHALL be urn:oasis:names:tc:SAML:2.0:protocol

samlmd:SPSSODescriptor@AuthnRequestsSigned : its value SHALL be true

samlmd:SPSSODescriptor@WantAssertionsSigned : its value SHALL be true

samlmd:SPSSODescriptor@validUntil : the longevity of the provisioned data. Its value SHALL be no greater than 2 months prior to the earliest certificate expiration dateTime value for certificates cited in the metadata document.

samlmd:SPSSODescriptor/samlmd:KeyDescriptor@use : `signing` keys SHALL be specified

samlmd:SPSSODescriptor/samlmd:SingleLogoutService@Binding : identifies the binding supported at the referenced endpoint for servicing Single Logout Requests to be used for Security Token Revocation messages by the Coordinator. Nodes SHALL support at least one of

`urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST`

`urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect`

samlmd:SPSSODescriptor/samlmd:SingleLogoutService@Location : specifies the endpoint for the identified binding supporting the SingleLogout request profile for Nodes

samlmd:SPSSODescriptor/samlmd:AssertionConsumerService@index : used by requestors to indicate in their request (via AssertionConsumerServiceIndex) which endpoint assertions from the Coordinator will be directed.

samlmd:SPSSODescriptor/samlmd:AssertionConsumerService@isDefault : indicates which endpoint, in the absence of specifying a preferred endpoint in their request, Coordinator responses should be directed to

samlmd:SPSSODescriptor/samlmd:AssertionConsumerService@Binding : the protocol binding support by the indicated endpoint

samlmd:SPSSODescriptor/samlmd:AssertionConsumerService@Location : the endpoint URL for the AssertionConsumerService

Affiliation Descriptors:

In SAML, affiliations describe the set of entities (Nodes) that shall be allowed to posses the *same* token for use in API calls. Typical deployments will include, for example, the primary nodeID of a retailer role, and the corresponding customer support Node. The Coordinator uses this affiliation description as a complete set of possible audience members (saml:AudienceRestriction) that can be requested in an assertion request.

`samlmd:EntityDescriptor/samlmd:AffiliationDescriptor` : Describes the set of Nodes who shall be authorized to include the Security Token in an API invocation (see [DCoord] section 12 on Node Delegation).

`samlmd:AffiliationDescriptor@affiliationOwnerID`: the nodeID of the entity who is operating as the primary Node in an affiliation

`samlmd:AffiliationDescriptor/samlmd:AffiliateMember`: one or more nodeIDs who shall be authorized to use a SAML assertion issued as a delegation token.

When Nodes are provisioned with the Coordinator for access, they will be provided with the necessary Coordinator metadata.

5.12 HTTP Authorization Binding for SAML Tokens

5.12.1 Including the SAML Assertion in HTTP Requests

Binding of SAML Assertions (Security Tokens) to REST API requests to the Coordinator is achieved by encoding the assertion using the DEFLATE mechanism described in [SAMLBIND] Section 3.4.4.1, further base64 encoding the DEFLATEd assertion, and placing the encoded assertion in the Authorization header of the request.

The complete algorithm is as follows:

1. Extract the saml2:Assertion from the samlp:Response (including the ds:Signature within the saml2:Assertion)

2. The DEFLATE compression mechanism, as specified in [RFC1951] is then applied to the extracted saml2:Assertion.

3. The compressed data is subsequently base64-encoded according to the rules specified in RFC 2045 [RFC2045]. Linefeeds or other whitespace SHALL be removed from the result of the base64 encoding process.

4. The base-64 encoded data is then placed in the HTTP Authorization header field, indicating that the token type is a SAML2 token:

```
Authorization: SAML2 assertion="<encoded SAML Assertion>"
```

   Where the assertion parameter (<encoded SAML Assertion>) conveys the DEFLATEd and base64 encoded SAML Assertion surrounded by double quotes.

5. The requestor SHALL prevent intermediary caching by specifying the HTTP headers:

```
Cache-Control: no-cache, no-store
Pragma: no-cache
```

RelayState SHALL NOT be conveyed in the use of this binding and in this binding, any <ds:signature> element signing the Assertion element and its contents SHALL NOT be removed.

5.12.2 HTTP Authorization Security Token Processing

The Coordinator SHALL validate the Security Token (SAML assertion) by:

1. Verify the Node TLS Certificate subject matches with the audience restriction in the Security Token and corresponding metadata

2. Verify the Security Token is well-formed and valid

3. Verify that the Security Token has not been revoked or otherwise deleted procedurally by the Coordinator

4. Verify the subject (UserID) and Account (from the Attribute Statement) are consistent with the API URI of the request

Upon successful validation of the assertion, the Coordinator will have established a Security Token subject scope that is documented for each API of [DCoord], and will enable the Coordinator to identify the User and Account associated with the request, independent of the invocation URI.

5.13 Confirmation Methods

This profile allows for the following SAML Confirmation methods:

- `urn:oasis:names:tc:SAML:2.0:cm:bearer`: The subject of the assertion is the bearer of the assertion. This confirmation method is only used for SAML Assertions issued to Devices. Tokens of this form SHOULD include constraint attributes within `SubjectConfirmationData` which establish a binding between the Licensed Application and the Device. Since the Coordinator exclusively produces and relies upon bearer tokens, they are opaque to the Device.

- `urn:oasis:names:tc:SAML:2.0:cm:sender-vouches`: No other information is available about the context of use of the assertion. This method is only employed when the presented token is conveyed over mutually authenticated communications channels. The

Coordinator SHALL verify that the sender (e.g. the Node) is identified in the assertions `AudienceRestriction` based on the Nodes presented certificate.

In the future, reliance upon the LicAppHandle may be incorporated into this profile, which would then provide a `urn:oasis:names:tc:SAML:2.0:cm:holder-of-key` confirmation method for Devices.

5.14 Token Integrity

Nodes and the Coordinator SHALL sign and verify the signature of all Assertions and SAML protocol messages.

5.15 Security Token Exchange requirements

The Security Token Service specified in section 8 defines 2 methods for the creation of, and the exchange of SAML assertions.

5.16 Security Considerations

All protocol messages occur over integrity-protected channels provided by TLS. Security considerations detailed in [SAML2SECC], however, still should be consulted. In particular:

- Section 6.1, which discusses SOAP Binding considerations but is applicable to the HTTP Authorization Bind defined in this specification.
- Sections 6.3 and 6.4 – Redirect and POST Binding considerations
- Section 6.6 – URI Bindings
- Section 7.1.1 and 7.1.4 – SSO Profile and Single Logout Profiles employed in this specification

6 User Credential Token Profile

During User creation, the User establishes a User Credential that is a pair of shared secrets held by the Coordinator. These secrets are:

- Username, which SHALL have a minimum length of 6 alphanumeric characters and a maximum length of 64 alphanumeric characters and MAY contain the non-alphanumeric characters:
  '@', '.', '-', '_' (UTF-8 Hex: 0x40, 0x2E, 0x2D, 0x5F)

- a Password, with a minimum length of 6 characters and a maximum length of 256 characters, constructed in a manner consistent with [SANSPP]. It may consist of any characters in the UTF-8 character set (represented here in hexadecimal form with Unicode code points) within the following ranges, inclusive:

  - 0x21 through 0x7E (Unicode code points U+0021 – U+007E)
  - 0xC2A1 through 0xC2AC (Unicode code points U+00A1 – U+00AC)
  - 0xC2AE through 0xC3BF (Unicode code points U+00AE – U+00FF)

The password SHALL NOT be based on personal information or information associated with the User (e.g. GivenName, SurName, UserName, etc.). Such similarities shall be determined over a minimum of 5 characters.

These secrets, when incorporated into protocol messages or submitted via graphical User interfaces, SHALL be conveyed over a properly secured transport mechanism, such as TLS.

The username SHOULD NOT be an email address. A User's username SHALL be unique in the Coordinator namespace. The Coordinator SHALL NOT require User passwords to be changed.

6.1 Profile Required Information

Identification: urn:dece:type:tokentype:usernamepassword

Updates: None

Purpose: This profile may be used for Authentication

Description: This profile is employed when authenticating a device or browser to the 's' host and by the Security Token Service defined in section 8.

Authorized Roles: any role identified in section 4.1.1

WWW-Authenticate challenge: Basic

6.2 User Credential Verification

User Credentials may only be collected and verified by the Coordinator, with the sole exception of when a Node is collecting User Credentials for the purposes of populating a UserCreate or UserUpdate API invocation.

There are three transport bindings supported in this profile:

- HTTP Basic authentication, as defined in [RFC2617]
- HTML Forms-based authentication
- a Coordinator Security Token Service API as defined in Section 8

The HTTP Basic authentication mechanism shall be used for Coordinator clients not capable of rendering HTML3.0 or greater representations.

The HTML Forms-based authentication utilizes HTML form controls to request and handle the submission of User Credentials to the Coordinator.

The Security Token Service API makes allowances for some deployment scenarios where Nodes preclude direct interaction between the 's' host and the User. The Security Token Service API also provides mechanisms for the exchange of one Security Token for another (including the exchange of User Credentials for a SAML Assertion)

Nodes other than the Coordinator Role SHALL NOT store User Credentials .

6.3 Security Considerations

Repeated failed attempts to authenticate a User to the Coordinator with an HTML Forms interface and using the User Credential profile SHALL, after DSECMECH_FAILED_AUTHN_ATTEMPTS failed attempts within DSECMECH_AUTHN_ATTEMPT_PERIOD, require the client to also pass a reverse Turing test (e.g. a CAPTCHA or similar technique) in addition to the verification of User Credentials when possible.

After repeated failed attempts to authenticate a User (either through a Device, or through interactions via the 's' host, the Coordinator may either:

- Present a reverse Turing test, if possible, or
- After DSECMECH_FAILED_AUTHN_ATTEMPTS failed attempts within DSECMECH_AUTHN_ATTEMPT_PERIOD, the Coordinator SHALL prohibit requests to APIs and the 's' host based on the origin IP address of the request for a period not to exceed DSECMECH_AUTHN_LOCK_PERIOD. This restriction does not apply to the Security Token Service. Note that restrictions may apply to the Security Token Service in the future.

The Coordinator MAY notify the effected User, using their primary email address, about the temporary login lock on their User account.

User-Agents which fail DSECMECH_FAILED_AUTHN_ATTEMPTS login attempts using the HTTP Basic Authentication transport binding MAY be denied access by the Coordinator until a successful HTML Forms authentication has been completed.

6.4 Proper Selection of Binding

The ‘s’ host shall allow for either HTTP Basic authentication or Forms-based authentication of the User using this User Credential profile. The Coordinator shall determine the proper binding to use based on the HTTP Accept header provided by the UserAgent, which indicates Mime-Types as an ordered set of supported types [RFC2045].

If the UserAgent indicates a preference for mime-types text/html or text/xhtml, the Coordinator shall respond with the Forms Binding.

If the UserAgent indicates a preference for text/xml or application/xml, the Coordinator shall respond with an HTTP Basic Challenge (WWW-Authenticate) Binding.

7 Federated Authentication Token Profiles

The federated authentication profile provides a mechanism by which a user can log in at a Node (e.g. a Retailer site) and subsequently move on to the Web Portal for Account management or other functions without requiring a separate authentication at the Coordinator, or conversely, enable a User to authenticate to a Node using their User Credentials managed by the Coordinator.

The primary purpose of this profile is for the transfer of an authenticated session with a User (or more precisely, their User Agent) from one party to the other.

In the case where the Coordinator issues the assertion in response to a request by a Node, the resultant response will be a Federation Security Token, and the specific profile may allow additional information in the response, including a Delegation Security Token. See section 5.12.

The section below defines general requirements for this federated authentication profile. The following sections define various profiles that map those requirements to a particular authentication technology.

7.1 Requirements

To preserve the trust between all the actors of the ecosystem, federated authentication must be provided within a carefully crafted framework. The following subsections describe the requirements defined to support this framework.Authorized Roles

The following Roles shall be permitted to assert a customer's identity to the Coordinator:

```
urn:dece:role:accessportal[:customersupport]
urn:dece:role:lasp[:customersupport]
```

`urn:dece:role:retailer[:customersupport]`The following Roles shall be permitted to rely upon assertions made by the Coordinator to the Node (federating the User's identity from the Coordinator):

```
urn:dece:role:accessportal[:customersupport]
urn:dece:role:lasp[:customersupport]
urn:dece:role:portal[:customersupport]
urn:dece:role:retailer[:customersupport]
```

Session duration requirements vary based on User requests and which role is providing the assertion:

- Idle sessions, described by DSECMECH_MAX_IDLE_REMOTE_SESSION_DURATION, trigger a re-authentication requirement to establish a new session. This re-authentication may be achieved via a Federation Security Token or via local credentials.
- A maximal constraint on session duration at the asserting party is described by DSECMECH_MAX_REMOTE_SESSION_DURATION.
- Nodes may not make an assertion that exceeds DSECMECH_MAX_REMOTE_ASSERTION_DURATION
- When a User expresses the desire for a limited session duration, the assertion is constrained by DSECMECH_SHORT_REMOTE_SESSION_DURATION (for example, not selecting a 'keep me signed in' form control
- The Coordinator may not produce an assertion of a session for less than DSECMECH_MIN_FEDERATION_DURATION

7.1.1 Requirements on Any Asserting Party

If a party is asserting the identity of a User to the Coordinator, it SHALL support the following requirements:

- Asserting parties SHALL re-authenticate idle users before issuing an authentication assertion. This re-authentication SHALL occur no later than DSECMECH_MAX_IDLE_REMOTE_SESSION_DURATION after the last user's activity/interaction.
- If a user session at the asserting party exceeds DSECMECH_MAX_REMOTE_SESSION_DURATION, it SHALL re-authenticate the user before issuing an authentication assertion.
- Asserting parities SHALL establish Federation Security Tokens for a session no less than DSECMECH_MIN_FEDERATION_DURATION. If capable, the asserting party SHOULD attempt to extend its own session with the User such that an assertion greater than or equal to DSECMECH_MIN_FEDERATION_DURATION can be made. If the asserting party cannot lengthen its session, or the presence of the `IsPassive` attribute in the request prevents it from lengthening the session, the request recipient SHALL respond with the SAML top-level code of `urn:oasis:names:tc:SAML:2.0:status:Requester` and second level error code `urn:oasis:names:tc:SAML:2.0:status:NoPassive` or `urn:oasis:names:tc:SAML:2.0:status:AuthnFailed` as appropriate.

- A Node SHALL NOT issue an assertion with an expiration that would exceed DSECMECH_MAX_REMOTE_SESSION_DURATION, or the expiration of the user session at the asserting party. If the Coordinator already has a session established with the User, the Coordinator will compare the identity named in the assertion and the identity associated with the Coordinator session.
    - If the identities differ, the Coordinator SHALL destroy the previous session, and establish a new session with the user.
    - If the identities match, then the Coordinator SHALL ensure the user session duration is the greater of the Node-asserted session and the Coordinator's session.
- If the user session at the asserting party is terminated (she logs out), the asserting party SHALL send a logout or revocation notification to assertion relying parties as defined in the applicable Token Profile.
- An asserting party SHALL provide a mechanism (e.g., form control with "Keep me signed in" box that can be unchecked), to allow the indication of the users preference for a short session. If a short session is requested by the User:
    - Such a session SHALL NOT exceed DSECMECH_SHORT_REMOTE_SESSION_DURATION
    - Any user agent session management techniques (for example, a cookie) SHALL be temporary. For HTTP cookies, this means setting a session cookie (one without an Expires parameter) instead of a permanent cookie.

When the Coordinator is asserting the identity of a User, it SHALL support the following requirements:

- All Federation Security Token protocol messages SHALL originate and terminate on the 'S' Host (see section 3.7)
- The 'S' Host is solely responsible for the administration of the master session for a User, using browser-stored session cookies when possible. This enables a more seamless user experience, minimizing the number of authentication requests the user is presented with.
- Any browser-stored information (such as a cookie) SHALL NOT contain any sensitive information such as usernames or other specific, static user identifiers.
- The 'S' Host oversees session validity for a User using (at least) the following criteria:

- the request includes a session cookie that internally maps to a set of criteria (which are not incorporated into the cookie itself)
- the interaction with the user originates from a previously used client IP address
- certain client-set static HTTP headers (for example, user-agent) map to the same session cookie
- the frequency of requests is within a reasonable tolerance (e.g. not more than 1 per second). Note that there may be circumstances where user agents will perform several messages in tight succession. The Coordinator MAY adjust such a tolerance as circumstances require.

- If an authorized message recipient (the request sender or audience member) includes a Dynamic LASP operating in Single-session User Mode, the Federation Security Token profile must meet the requirements defined in [DSystem] section 4.4.2. This is achieved in a profile-specific manner.

7.1.2 Requirements on Relying Parties

If the Coordinator is the relying party, it SHALL support the following requirements upon receiving an identity assertion from a Node:

- All Federation Security Token protocol messages SHALL terminate on the 'S' Host (see section 3.7)
- The Coordinator SHALL be able to verify the asserted identity.
- The Coordinator SHALL verify the asserting Node credentials (Role etc.)
- The Coordinator SHALL be able to reject the asserting Node authentication assertion for security purposes (e.g. if fraud is perceived or the authentication mechanism used is deemed inappropriate).
- The Coordinator SHALL establish a session with the browser that will persist until the lesser of DSECMECH_MAX_REMOTE_SESSION_DURATION and the assertion's expiration.

7.1.3 Targeting Web Portal resources

Federation Profiles must specify a mechanism for indicating a target URL (a URL to direct a user-agent to after validating an asserted identity to a location at the Portal). Values for specific Portal pages are defined in Appendix C.

7.2 SAML v2.0 Federation Profile

7.2.1 Overview

As described in section 5, SAML is already used by the Coordinator to communicate a user's identity to Nodes as Delegation Security Tokens. However those Delegation Security Tokens are issued by the Coordinator and presented by requesting Nodes to invoke the Coordinator's API. In other words, the Coordinator acts as the Identity Provider for the purposes of delegation. In the federated authentication scenario, a SAML assertion may be passed between a Node and the Coordinator in either direction, for the purpose of establishing an authenticated session between the User and the relying party via the asserting party.

In addition, this profile adds unsolicited response support to the use of SAML. Unsolicited responses, in SAML, are defined as `<samlp:Response>` messages, where there is no corresponding `<samlp:AuthnRequest>` message issued by a relying party.

Although less common, emitting such unsolicited authentication response is supported in the SAML specification [SAMLPROF]. It is thus possible to leverage the same SAML Web SSO profile and the POST and Redirect transport bindings defined in the SAML Token Profile section (see section 5).

The following diagram depicts the unsolicited response protocol exchange between an asserting Node, the user agent client and the Coordinator, and covers positive outcome flows only:

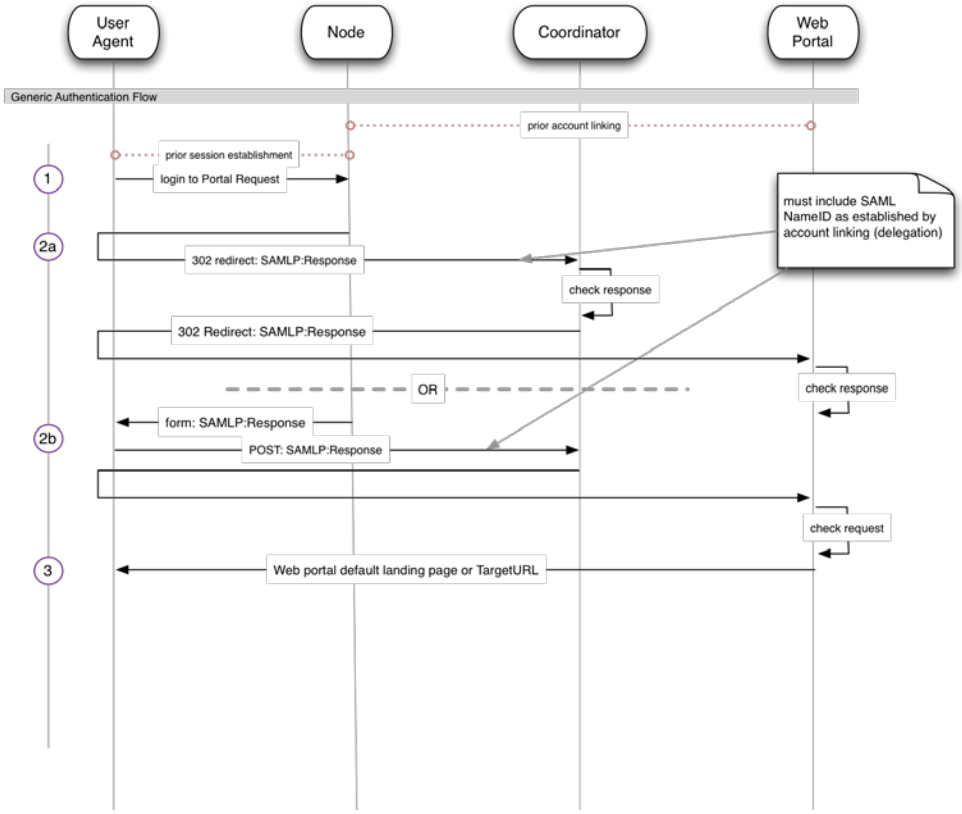

Figure 3: SAML message flow for Unsolicited Responses

The details of the steps identified in Figure 3 are as follows:

1. The user visits the Node and logs in using his local credentials
2. Following verification of the user's credentials, the Node performs one of the following:
    a. In the case of the Redirect binding, the user agent is redirected to the Coordinator (its assertion consumer service URL endpoint, the 'S' host) with a SAML <Response> message
    b. In the case of the POST binding, the Node sends an XHTML document that contains a Form and the SAML <Response> message. The user agent then delivers the <Response> message to the Coordinator (the 'S' host).
3. Upon successful verification of the <Response> message, the Coordinator directs the user agent to the Web Portal's default landing page, or another endpoint which may be indicated by the TargetURL parameter (see section 7.2.5).

See Figure 1: SAML Request and Response sequence for an example message exchange where the Coordinator is the asserting party.

7.2.2 Supported SAML Protocols

The following SAML Protocols SHALL be supported by the Coordinator and Asserting Node:

- Authentication Request Protocol which is defined in [SAMLCORE, section 3.4]
- Single Logout Protocol which is defined in [SAMLCORE, section 3.7]

These protocols define the message exchanges for SAML Requests and SAML Responses.

The Coordinator's SAML metadata defines the specific endpoints for Delegation Security Token requests for each of the supported bindings, and are derived from the Coordinator Security-related endpoints defined in 3.7.

7.2.3 Supported SAML Bindings and Profiles

The following SAML Bindings SHALL be supported by the Coordinator and Asserting Node:

- HTTP Redirect Binding which is defined in [SAMLBIND, section 3.4], and identified by the binding identifier `urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect`
- HTTP POST Binding which is defined in [SAMLBIND,section3.5], and identified by the binding identifier `urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST`

These bindings define the message exchanges mechanisms for the conveyance of SAML Requests and SAML Responses. Both bindings require the presence of the user and an HTTP User Agent.

The following SAML Profiles SHALL be supported by the Coordinator and Asserting Node:

- Web Browser SSO Profile which is defined in [SAMLPROF, section 4.1], and identified by the profile identifier `urn:oasis:names:tc:SAML:2.0:profiles:SSO:browser`.
- Single Logout Profile which is defined in [SAMLPROF, section 4.4], and identified by the profile identifier `urn:oasis:names:tc:SAML:2.0:profiles:SSO:logout`.

SAML Requestors and SAML Responders NEED NOT include x509Data in their use of SAML protocols. Requestors and Responders SHALL verify protocol and Assertion signatures using x509Data information in the SAML metadata document provided to the Node by the Coordinator.

7.2.4 Protocol Extensions

Protocol extensions, if needed SHALL be placed in the //AuthNRequest/Extensions element or the //Response/Extensions element as appropriate. Nodes may include zero or more of these extensions. Some extensions may prohibit certain extension combinations.

The dualSession and ProvideDateOfBirth extensions defined below are included in the RequestedResponseType element, defined in [DCSchema], which is of type dece:EntityID-type.

The UserLinkConsent and DataSharingConsent policies (defined in [DCoord] section 5) are included within the PolicyList element defined in [DCoord] section 5. Nodes SHALL only include the //Policy/PolicyClass element.

7.2.4.1 dualSession Extension

This extension may only be included in an AuthNRequest message sent to the Coordinator.

This extension is placed within the RequestedResponseType element with a value of urn:dece:type:policy:dualSession. This extension indicates to the Coordinator that the requestor wishes to maintain two concurrent sessions in a browser. For example, when a Node is preparing to perform an AccountMerge defined in [DCoord] section 13.2.

The Coordinator will be required to maintain 2 user sessions concurrently within a single UserAgent.

The requestor SHALL include in the AuthNRequest request the ForceAuthn attribute with a value of true which will force the Coordinator to authenticate the User regardless of any present session state. If not included, the SAML protocol error urn:oasis:names:tc:SAML:2.0:status:RequestDenied will be sent as a response to the requestor.

No Delegation Security Token will be included in the Response.

This extension may only be combined with the temporaryDelegation extension.

7.2.4.2 temporaryDelegation Extension

This extension may only be included in an AuthNRequest message sent to the Coordinator.

This extension is placed within the RequestedResponseType element with a value of urn:dece:type:policy:temporaryDelegation when a Node requires a Delegation Security

Token with a validity period of DSECMECH_MIN_TOKEN_DURATION_DEFAULT, in addition to the Federation Security Token in the response.

A Delegation Security Token will be included in the `Response`.

When combined with the dualSession extension, the AuthNRequest SHALL include the attribute ForceAuthn attribute with a value of true which will force the Coordinator to authenticate the User regardless of any present session state. If not included, the SAML protocol error urn:oasis:names:tc:SAML:2.0:status:RequestDenied will be sent as a response to the request sender.

This extension may be combined with the dualSession, provideAgeInfo and dataSharing extensions.

7.2.4.3 ProvideAgeInfo Extension

This extension may only be included in an AuthNRequest message sent to the Coordinator.

This extension is placed within the `RequestedResponseType` element with a value of `urn:dece:type:policy:ProvideAgeInfo`. This extension indicates to the Coordinator that the Node requests an Assertion of the age of the User, in addition to a Federation Security Token. See 7.2.6.3 for Assertion details.

This extension may be combined with the UserLinkConsent and DataSharing extensions.

7.2.4.4 UserLinkConsent Policy Extension

This extension may be included in an AuthNRequest message sent to the Coordinator and a Response message from the Coordinator.

This extension allows the inclusion of a PolicyList resource. For such requests, Nodes SHALL only include UserLinkConsent in the `//Policy/PolicyClass` element.

Use of this extension indicates that a Node is requesting a Delegation Security Token in addition to a Federation Security Token.

To determine the Token's validity period, the Coordinator provides the User the option of accepting (which is the default state) or refusing the UserLinkConsent policy.

An existing UserLinkConsent policy on the User from the Node have no influence on the handling of this request.

If the User accepts the policy request, the Coordinator creates the UserLinkConsent policy on the User for the Node, and includes a Delegation Security Token in its response, with a validity period as described in section 4.3.2.1. The Response extension point will also include a PolicyList including a UserLinkConsent policy reference (see section 5.9.1).

If the User declines the policy request, the Coordinator deletes any existing UserLinkConsent policy on the User for the Node, and includes a Delegation Security Token in its response, with a validity period of DSECMECH_MIN_TOKEN_DURATION_DEFAULT.

This extension may be combined with the ProvideAgeInfo and DataSharing extensions.

7.2.4.5 DataSharingConsent Policy Extension

This extension may only be included in an AuthNRequest message sent to the Coordinator.

This extension allows the inclusion of a PolicyList resource. For such requests, Nodes SHALL only include DataSharingConsent in the `//Policy/PolicyClass` element.

If the User accepts the policy request, the Coordinator creates the DataSharingConsent policy on the User for the Node or Organization. This policy makes the UserGet DataSharing API endpoint available to the Node or Organization as described in [DCoord] 14.1.3. The Response extension point will also include a PolicyList including a DataSharingConsent policy reference (see section 5.9.1).

If an existing and applicable DataSharingConsent policy is in place and the User accepts the policy request a second or subsequent time, the existing DataSharingConsent policy is updated to reflect the most recent acceptance by the User.

If the User declines the policy request, the Coordinator will adhere to any existing applicable policies, otherwise the Coordinator will deny access to the UserGet DataSharing API endpoint to the Node. No change to any existing and applicable DataSharingConsent policies are made (previous policies and UserGet access window duration remain in place).

This extension may be combined with the temporaryDelegation, ProvideAgeInfo and UserLinkConsent extensions.

7.2.4.6 ReturnToURI Extension

This extension is only used in unsolicited response messages, and SHALL be accompanied with the TargetURL extension, defined below.

The <Response> MAY contain, in its <Extensions> element, the element ReturnToURI of type xs:anyURI. If the Asserting Node provides a URL, the response recipient MAY use it to return the User to the response sender.

The Coordinator, upon receipt of such an extension, will forward this URL to the TargetURL, to enable the application to provide the ability to return the User to their original page at the asserting Node.

This extension will always be accompanied by the TargetURL extension.

7.2.4.7 TargetURL Extension

The <Response> MAY contain, in its <Extensions> element, the element TargetURL of type xs:anyURI. This URI allows the asserting Node to define the desired URL to direct a user-agent to, once the recipient has successfully verified the Response.

If the intention is to target a specific page on the Portal, the values defined in Appendix C SHALL be used.

This extension may appear by itself, or may be accompanied by the ReturnToURI extension.

7.2.4.8 Presentation Language Selection Extension

This extension allows the inclusion of a dece:Language element in a SAML request or reponse extension.

This extension allows the Node to indicate to the Coordinator the presentation language to be used when rendering user interface elements to a user agent. Its' value SHALL be one of the language identifiers defined in a corresponding DGeo version (e.g. languages identified as supported by the identically versioned DGeo specification).

If included, the isPrimary attribute is ignored.

If this request extension is not included, the Coordinator will determine the presentation language based on provided HTTP Accept-Language header as defined in [RFC2616]. If this header is not provided, the Coordinator will default to the language en-US.

7.2.5 SAML Request Messages

AuthNRequest messages SHALL conform to the requirements as stated in [SAMLCORE], [SAMLPROF] and [SAMLBIND].

AuthNRequest messages MAY include a //Subject/NameID, in which case, the recipient of the request SHALL verify that the authenticated user matches the requested NameID. If these do not match, the recipient SHALL respond with the SAML protocol error InvalidNameIDPolicy.

AuthNRequest messages MAY include protocol extensions as defined in section 7.2.4.

Requests may use either of the supported bindings, but it is STRONLY RECOMMENDED that the HTTP Post binding be employed.

If an authorized message recipient (the request sender or audience member) includes a Dynamic LASP operating in Single-session User Mode, the request SHALL include a date/time value in the //AuthNRequest/Conditions/@SessionNotOnOrAfter to meet the session requirements defined in [DSystem] section 4.4.2.

Requestors MAY request that the Coordinator not interact with a User in a noticeable fashion by setting the `isPassive` attribute to TRUE. The Coordinator SHALL respond with one of the following SAML status values:

- `urn:oasis:names:tc:SAML:2.0:status:Success` indicating that the existing Federation Security Token issued to the requestor is still valid and has remaining duration greater than `DSECMECH_MIN_FEDERATION_DURATION`.
- `urn:oasis:names:tc:SAML:2.0:status:NoPassive` indicating that the Coordinator could not service the request without requiring noticable interactions with the UserAgent for any of a number of possible reasons.
- `urn:oasis:names:tc:SAML:2.0:status:AuthnFailed` indicating that there was a problem with the request.

Requests that include `isPassive` set to TRUE SHALL NOT also include the request extensions `temporaryDelegation` or `UserLinkConsent`. If included, the Coordinator SHALL respond with `urn:oasis:names:tc:SAML:2.0:status:AuthnFailed` status.

7.2.6 SAML Response Message

The asserting party SHALL ensure the SAML subject has agreed to the creation of a remote session with the relying party.

Response messages MAY include protocol extensions as defined in section 7.2.4.

7.2.6.1 Common Requirements for SAML Responses

Unless expressed otherwise, the constraints and rules defined in section [5.4] apply for this profile and SHALL be followed. In addition, the following constraints are defined:

- The <AuthnStatement> SHALL include a SessionIndex attribute to enable per-session logout from the Coordinator.
- The <AuthnStatement> SHALL include a SessionNotOnOrAfter attribute that defines the maximum session duration at the relying party. Presence of this attribute is the sole differentiator between a Delegation Security Token and a Federation Security Token. If the message recipient receives more than one AuthNStatement in a response, it SHALL use this attribute to identify the Federation Security Token.

7.2.6.2 Node Originated Responses

Since Nodes will never be the recipient of a SAML Request, only unsolicited responses will be generated by Nodes.

7.2.6.2.1 Node Originated Unsolicited Responses

The semantics and processing rules found in [SAMLPROF] and in section [5] SHALL be applied. As described in [SAMLPROF] unsolicited responses carry additional rules that SHALL be enforced by the sender and recipient:

- The <Response> SHALL NOT contain an InResponseTo attribute.
- Any bearer <SubjectConfirmationData> elements of the <Response> SHALL NOT contain an InResponseTo attribute.
- The <Response> SHOULD be delivered to the <md:AssertionConsumerService> endpoint of the service provider designated as the default.

This profile further refines the processing requirements of the response as follows:

- **samlp:Response/Issuer** : SHALL be the same NodeID as one of the Nodes contained in the AudienceRestriction of the embedded (or referenced) Delegation Security Token.
- **samlp:Response@IssueInstant** : the date SHALL NOT be earlier than the IssueInstant contained in the embedded (or referenced) Delegation Security

Token. Similarly, this date SHALL fall in between the (optional) window defined by the `NotBefore` and `NotAfter` dates in the `Conditions` elements of the embedded (or referenced) Delegation Security Token.

- When asserting an identity to the Coordinator, the Node SHALL use the `NameId` provided by the Coordinator in the Delegation Security Token issued to the asserting Node about the subject User (typically as a result of a previously established Delegation Security Token, or in cases where the asserting Node was responsible for the creation of the User).
- When asserting an identity to the Coordinator, the Nodes <`Assertion`> SHALL either:
    - Embed an <`Assertion`> which SHALL be a valid Delegation Security Token previously issued to the asserting Node by the Coordinator. This <`Assertion`> SHALL be put in the `saml:Advice/saml:Assertion` element.
    - Contain a reference to a valid Delegation Security Token previously issued to the asserting Node by the Coordinator. This reference SHALL be put in the `saml:Advice/saml:AssertionURIRef` element.
    - **samla:IssueInstant** : the date SHALL NOT be earlier than the `IssueInstant` contained in the embedded (or referenced) Delegation Security Token.
    - **samla:Conditions@NotBefore** : if present, the date SHALL NOT precede the corresponding `NotBefore` date contained in the embedded (or referenced) Delegation Security Token.
    - **samla:Conditions@NotOnOrAfter** : if present, the date SHALL NOT exceed the corresponding `NotOnOrAfter` date contained in the embedded (or referenced) Delegation Security Token. Additional restrictions on @NotOnOrAfter defined in section 7.1.2 SHALL be followed.
    - **samla:AuthnStatement@SessionNotOnOrAfter** : if present, the date SHALL NOT exceed the `NotOnOrAfter` date contained in the embedded (or referenced) Delegation Security Token. Additional restrictions on @SessionNotOnOrAfter defined in section 7.1.2 SHALL be followed.
    - **samla:Conditions/AudienceRestriction/Audience** : if present, there SHALL NOT be more than one NodeID. That NodeID SHALL be the Coordinator NodeID.

If for some reason the `Response` message is found to be unacceptable to the recipient, the Response recipient SHALL respond to the sender a SAML Response to the Node's `AssertionConsumerServiceURL` specified in SAML Metadata with a SAML protocol major

error and an appropriate minor error identifier. This error response SHALL include the original `AuthNRequest/@ID` value in its `Response/@InResponseTo` attribute.

7.2.6.3 Coordinator Originated Responses

If the UserLinkConsent or temporaryDelegation extension is present in the request, the Coordinator SHALL include both a Federation Token (session assertion) and a SAML Delegation Security Token in its response, as defined in section 5. In addition, the Coordinator SHALL include within the response extension point a PolicyList including a UserLinkConsent policy reference (see section 5.9.1).

Any requested audience members SHALL BE included in both the Federation and Delegation tokens.

If the dualSession with delegation extension is present in the request, the Coordinator SHALL include both a Federation Token (session assertion) and a SAML Delegation Security Token in its response, as defined in section 5.

If the ProvideAgeInfo extension is present in the request, the Coordinator MAY respond within a separate assertion of

a) the User's //User/DateOfBirth value, or

b) a calculated age based on the //User/DateOfBirth value, or

c) a Boolean value indicating if the User's age is at or above (true) or below (false) the applicable geography's DGEO_AGEOFMAJORITY.

[DGeo] section 2.6.2 indicates whether or not this assertion is provided, and how it is expressed.

This Assertion contains an attribute statement for age or date of birth, where the `NameFormat` options the Coordinator uses are:

- `urn:dece:type:DateOfBirth` with a data type of the `saml2:AttributeValue` as `dece: DayOptionalDate-type`, as defined in [DCoord] section 14.
- `urn:dece:type:UserAge` with a data type of the `saml2:AttributeValue` of `xs: nonNegativeInteger`.
- `urn:dece:type:MeetsAgeOfMajority` with a data type of `xs:Boolean`.

For example:

```
<saml2:AttributeStatement>
  <saml2:Attribute
  Name="DateOfBirth"
  NameFormat="urn:dece:type:DateOfBirth">
      <saml2:AttributeValue
      xmlns:dece="http://www.decellc.org/schema/2011/08/coor
dinator"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="dece:DayOptionalDate-
type">10/22/1970</saml2:AttributeValue>
  </saml2:Attribute>
</saml2:AttributeStatement>
```

If the DataSharingConsent extension is present in the request, and the User agrees to the data sharing policy request, the Coordinator SHALL include within the response extension point a PolicyList including a DataSharingConsent policy reference (see section 5.9.1).

If the ReturnToURI extension is present in the request or response, the Coordinator SHALL forward this extension to the Portal's assertion consumer service endpoint within the unsolicited response to the Portal.

If the TargetURL extension is included in the request or response, the Coordinator SHALL create an unsolicited response including the requested Portal page identified in the TargetURL in the response.

As a consequence of processing request extensions, Nodes may receive up to three statements from the Coordinator:

- an assertion to be used as a Federation Security Token (declaring a browser session),
- a Delegation Security Token for use in Coordinator APIs
- an attribute statement containing a `dece:DateOfBirth-type` value.

7.2.6.3.1 Coordinator Originated Unsolicited Responses

The semantics and processing rules found in [SAMLPROF] and in section [5] SHALL be applied.

As described in [SAMLPROF] unsolicited responses carry additional rules that SHALL be enforced by the sender and recipient:

- The `<Response>` SHALL NOT contain an `InResponseTo` attribute.
- Any bearer `<SubjectConfirmationData>` elements of the `<Response>` SHALL NOT contain an `InResponseTo` attribute.

- The <Response> SHOULD be delivered to the <md:AssertionConsumerService> endpoint of the service provider designated as the default.

If for some reason the <Response> message is found to be unacceptable, the recipient of the initial <Response> SHALL respond to the response sender a SAML <Response> to the Coordinator's AssertionConsumerServiceURL specified in SAML Metadata with a SAML protocol major error and an appropriate minor error identifier. This error response SHALL include the original AuthNRequest/@ID value in its Response/@InResponseTo attribute.

7.3 Security Considerations

7.3.1 Compromised Credential

If the user's credential information at the Node is compromised, the Node SHALL take the following measures:

- If any, immediately terminate the user's session and send a Single Logout request to the Coordinator when possible.
- Notify DECE of the security breach.

7.3.2 Authentication Levels

In order to reduce the risk of rejection of the authentication assertion by the Coordinator, it is expected that the credential strength required at the Node is very comparable to those defined in section 6 or better (i.e. authentication mechanisms that may be used). Additional authorized Security Mechanisms will be evaluated from time to time, but the Coordinator SHALL accept:

```
urn:oasis:names:tc:SAML:2.0:ac:classes:Password
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession
```

as defined in [SAMLCTX].

Nodes SHALL support, at a minimum the following Security Mechanisms:

```
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport

urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession
```

8 Security Token Service

The Coordinator provides a token exchange service that enables API Clients to exchange one Security Token for another, or to extend the validity period and other properties of a Token. New Security Tokens incorporated into this specification should incorporate applicable token exchange requirements to this section, when published.

8.1 SecurityTokenExchange()

8.1.1 API Description

This service allows for the exchange of a security token in place of another security token. The 2 tokens may differ in type (e.g. a username/password token exchanged for a SAML assertion, or a SAML assertion in exchange of another SAML assertion) or have different characteristics (that is, lifetime, time constraints, or targeted audience).

There are two types of invocation for this API:

- The API Client has no existing Security Token for a User with the Coordinator. In this case, the token to be replaced must be provided. Transformations of this type may be used by an API Client for the Username/Password Token and Device Authentication Token.
- The token to be replaced was previously issued by the Coordinator to an API Client identified in the present token. The URI that corresponds to the previous token SHALL be used.

The Coordinator supports a limited set of Security Token formats. Currently supported conversions include

- Username/Password (User Credential) Token and Device Authentication Token, which is converted to a SAML assertion (as a Delegation Security Token).
- A SAML assertion (Delegation Security Token), which may only be exchanged for another SAML assertion (as a Delegation Security Token).

For all SecurityTokenExchange APIs, the [baseURL] is as defined in [DCoord] section 3.12. Since this API is not considered a query, the [baseURL] uses the [pHost] form defined there.

Federation Security Tokens may not be exchanged for a Delegation Security Token.

8.1.2 API Details

Path:

When the token to be replaced was not issued by the Coordinator:

```
[BaseURL]/SecurityToken/SecurityTokenExchange?tokentype={type}
```

When the token to be replaced was issued by the Coordinator:

```
{SecurityTokenURIRef}/SecurityTokenExchange?tokentype={type}
```

Method: POST

Authorized Roles:

For the `usernamepassword` token type:

`urn:dece:role:accessportal`[1]

`urn:dece:role:device`

`urn:dece:role:lasp`[1]

`urn:dece:role:portal`[1]

`urn:dece:role:retailer`[1]

For the SAML token type:

`urn:dece:role:accessportal[:customersupport]`

`urn:dece:role:device`

`urn:dece:role:lasp[:customersupport]`

`urn:dece:role:portal[:customersupport]`

`urn:dece:role:retailer[:customersupport]`

(1) The Access Portal, LASP and Retailer roles may only use this tokentype during initial Account and User creation to simplify the user creation process via a Node.

For the saml2 token type: `urn:dece:role:node:any`

Security Token Subject Scope: None

Opt-in Policy Requirements:

when the token type is not the User Credential type and the requestor is a Node: `urn:dece:type:policy:TermsOfUse`

when the requestor is a device: `urn:dece:type:policy:TermsOfUse`

when the token type is the User Credential type and the requestor is a Node: none

Request Parameters:

`{type}` is the type of Security Token the Node would like in response, and SHALL be one of the following values:

Token Type	Description
`urn:dece:type:tokentype:saml2`	SAML v2.0 assertion as defined in section 5

Table 4: Security Token Exchange Token types

{SecurityTokenURIRef} is the absolute URI of the token to be replaced. This is provided within the Security Token itself. When presenting the token to the Coordinator, the Client SHALL replace the [iHost] BNF production with the [pHost] BNF production (see [DCoord] section 3.11.

If the supplied {type} parameter is unknown or unsupported, the response type SHALL BE a SAML2 response. This is done to support backwards compatibility with earlier implementations, where the Coordinator expected the {type} parameter to be the input type, and not the output type. This treatment of {type} will be deprecated in a future version of the specifications, however, so implementations should adhere to the intended use of {type} as soon as possible.

Request Body:

The Token to be exchanged for a Security Token of type {type}.

If the requestor is a Node, and is not presently in possession of a Coordinator-issued Security Token, it shall provide Credentials element:

Element	Attribute	Definition	Value	Card.
Credentials		The Credentials Security Token to be exchanged.	`dece:UserCredentials-type`	
Username		The Username element, as specified in [DCoord].	`xs:string`	1
Password		The Password element, as specified in [DCoord]	`xs:string`	1

Table 5: Username/Password Token type

If the requestor is a Device, it shall provide the `DeviceAuthToken` element:

Element	Attribute	Definition	Value	Card.
DeviceAuthToken			`dece:DeviceAuthToken-type`	

Table 6: Device Authentication Token

Element	Attribute	Definition	Value	Card.
Dece:DeviceAuthToken-type				

Element		Attribute	Definition	Value	Card.
Choice	DeviceJoinCode		The Device authentication code input into the Device, which must match the corresponding value generated by the Coordinator. See [DCoord] section 9.1.6 and [DDevice] section 4.1.1.2.	`xs:string`	
	DeviceString		The Retailer POS-issued join string (DeviceUniqueString). See [DDevice] section 4.1.1.4	`xs:string`	

Table 7: DeviceAuthToken-type

Response Body: None

8.1.3 Requestor Behavior

If the API Client is not in possession of any token types above, they shall employ the first form of this API, which uses the Credentials element to convey this information to the Coordinator. The Requestor receives the User Credentials, and submits them to the Coordinator to exchange for the requested token type. The Node SHALL obtain the Credentials from the User employing a confidentiality-protected channel, such as is described in Section 3.2.1 in [DSecMech]. The Node SHALL dispose of these credentials immediately after their use in this API exchange.

If the API Client is in possession of the `urn:dece:type:tokentype:saml2` token type, the API Client SHALL extract the `saml2:AssertionURIRef` from the current SAML token, and use that ID as the {SecurityTokenURIRef} in the API endpoint.

8.1.4 Responder Behavior

For the Username/Password Token and Device Authentication Token forms:

- The Coordinator SHALL verify the Credentials supplied by the requestor. If the token fails to validate, the Coordinator responds with a 403 Forbidden response.
- When a Node is making the urn:dece:type:tokentype:usernamepassword token service request, the Coordinator SHALL only authorize the request if the Node that created the User is making the request and the User was created within DSECMECH_STS_USERCREDENTIAL_TIMELIMIT. In such circumstances, if the newly created User had its password calculated by the Coordinator, the User Credential token SHALL NOT include the Password element (as it is not known to the User or the Node). Devices are not subject to this restriction.

For the SAML Token form:

- The Coordinator SHALL verify that the token supplied, including ensuring that the Node is identified in the presented token's `saml:Conditions/saml:AudienceRestrictions/saml:Audience`.
- The token SHALL be valid at the time of presentation. The Coordinator SHALL perform any integrity and validity checks as defined in of [DSecMech] section 5.
- Nodes will be allowed to exchange Security Tokens that were invalidated as a result of the use of the AccountMerge and AccountMergeUndo APIs defined in [DCoord].

Tokens created as a result of a Device Authentication Token exchange SHALL require the presentation of the original DeviceAuthToken during Security Token retrieval. This requires Devices to retain the DeviceAuthToken or DeviceString (referred to as DeviceUniqueString in [DDevice] Section 4.1.1.4) until the Security Token is successfully obtained from the Coordinator. Device Authentication Tokens are not returned to the Device in the response by the Coordinator.

Similarly, when Devices use the UsernamePassword token profile to exchange for another token type, they SHALL present the users Credentials via HTTP Basic authorization when retrieving the created Security Token.

If no error conditions occur, the Coordinator SHALL respond with an HTTP 201 status code (*Created*) and a Location header containing the URL of the created resource. The 201 response is used in order to remain consistent with other Coordinator messages, and to enable retrieval by other Nodes named in an AudienceRestriction (in lieu of passing an assertion, the assertion reference may be passed). The resulting resource will be of the type requested by the `tokentype` query parameter. The requester may then retrieve the token at the indicated URL. The Coordinator MUST authenticate Nodes at this URL as defined in [DSecMech] section 3, and verify that the Node identity matches an entry in the `saml:Conditions/saml:AudienceRestrictions/saml:Audience`.

In cases where the requested `tokenType` supports multiple named Nodes in the Security Token, the following query parameters MAY be appended to the request URL:

audience=nodeid1;nodeid2;…

In cases where the requested `tokenType` supports a defined validity period, the following query parameters MAY be appended to the request URL:

duration=number (measured in days)

The following processing rules SHALL be enforced by the Coordinator:

- The duration query parameter SHALL not exceed DSECMECH_MAX_TOKEN_DURATION_DEFAULT. The duration parameter value must be a

positive integer. The Coordinator SHALL discard any supplied decimal value, preserving only the integer portion of the parameter value.

- Security Tokens exchanged via this API are subject to the constraints defined in section 4 and any additional constraints defined in the corresponding Security Token profile.

- The Coordinator will respond with the error urn:dece:errorid:org:dece:invalidDurationvalue if the duration parameter is negative. When a requestor includes a duration that exceeds defined limits, the Coordinator SHALL adjust the duration to the maximum allowed duration.

- The Nodes listed in the audience query parameter SHALL each have SAML metadata document registered at the Coordinator.

- Requests that include Nodes in the audience request parameter who do not have SAML metadata registered at the Coordinator will result in an HTTP 403 Forbidden response.

- If a request includes Audience member Nodes which are not eligible to be included in a SAML Audience restriction (for example, prohibited Role combinations or different Organizations) but does have SAML metadata registered at the Coordinator, the Coordinator will remove such Nodes from the Audience list, and continue to process the request. If no Nodes remain in the Audience restriction, and the requestor is also not eligible, the Coordinator will respond with the appropriate SAML protocol error message to the requestor..

Example:

{SecurityTokenURIRef}/SecurityTokenExchange?tokentype=urn:dece:type:tokentype:saml2&audience=urn:dece:retailer:mycompany;urn:dece:lasp:mycompany&duration=24

The above example request the exchange of a SAML token for another one in which the audience will contain 2 Node IDs (urn:dece:retailer:mycompany and urn:dece:lasp:mycompany) and the lifetime is expected to be of 24 days.

Although, when supported, these extensions will allow for more felicity, additional security constraints will be necessary to maintain an adequate control over the issuance of SAML assertions.

The audience in the query has to be within the boundaries of the affiliation descriptor in the SAML metadata.

8.1.4.1 Special Considerations for the urn:dece:role:dece:customersupport Role

The urn:dece:role:dece:customersupport Role exclusively can obtain a Security Token based only on Username. The following conditions apply:

- The urn:dece:role:dece:customersupport Role SHALL supply only the Username and its parent Credentials element when using the UserCredential token request.

- The Coordinator SHALL NOT issue a Security Token with a validity period greater than 24 hours.

- The Coordinator SHALL issue the Security Token with identifiers targeted specifically to the urn:dece:role:dece:customersupport Node making the request.

8.1.5 Errors

- Unsupported token type
- Input token is malformed
- Invalid token
- LinkLASPLimitExceeded

8.2 Device Authentication Token Exchange Retrieval

In order to authorize a Device to retrieve a Security Token created via the Security Token Exchange Service, Devices SHALL present the Device Authentication Token, the Device Unique Token string, or User Credentials to the Security Token Resource created after a successful SecurityTokenExchange() invocation by including the applicable Authorization header in the request. For the User Credential (UserNamePassword) form, HTTP Basic Authorization SHALL be used.

The Device Authentication Token is incorporated into the HTTPS GET request of the resource created by including its value in the HTTP Authorization header as follows:

```
Authorization: DeviceCode value="[devicecode]"
```

where `[devicecode]` is either the Device Authentication Token or the Device Unique Token string.

The Coordinator SHALL verify the association between the generated Token at the resource location with the provided `DeviceCode`.

The following diagram depicts this exchange:

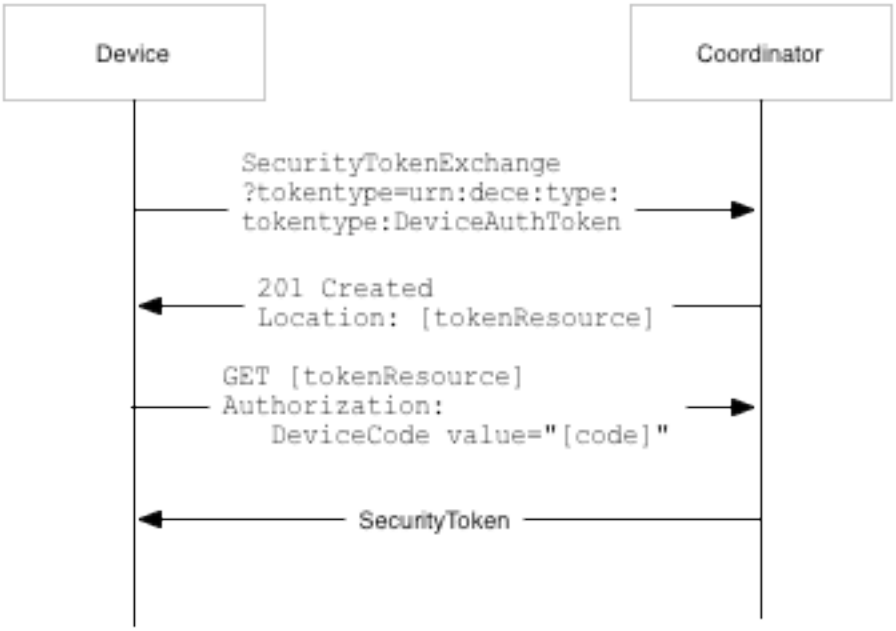

Figure 4: Device Authentication Token Exchange

Appendix A. Subject Query Profile of SAML

Note: This feature has been removed. It may, however, be re-introduced in the future.

This profile enables a Requestor to construct a structured subject query to a SAML responder. To implement this profile requires supporting the HTTP Redirect, HTTP Post and HTTP Artifact bindings.

It is assumed that the user is using a standard commercial browser and can authenticate to the identity provider by some means outside the scope of SAML.

A.1 Required Information

Identification: urn:dece:type:tokenprofile:saml2:subjectquery

SAML Confirmation Method Identifiers: The SAML V2.0 "bearer" confirmation method identifier, urn:oasis:names:tc:SAML:2.0:cm:bearer, is used by this profile.

Description: Given below.

Updates: None.

A.2 Profile Overview

The Subject Query profile provides a generalized message exchange profile, which is derived from the Web Browser SSO Profile defined in [SAMLPROF]. It is expected the implementations of this profile will further define message processing instructions, and make use of one or more of the provided message extension points (for example, the samlp:Request/Extension extension point). Figure 5 illustrates the basic template for performing a subject query.

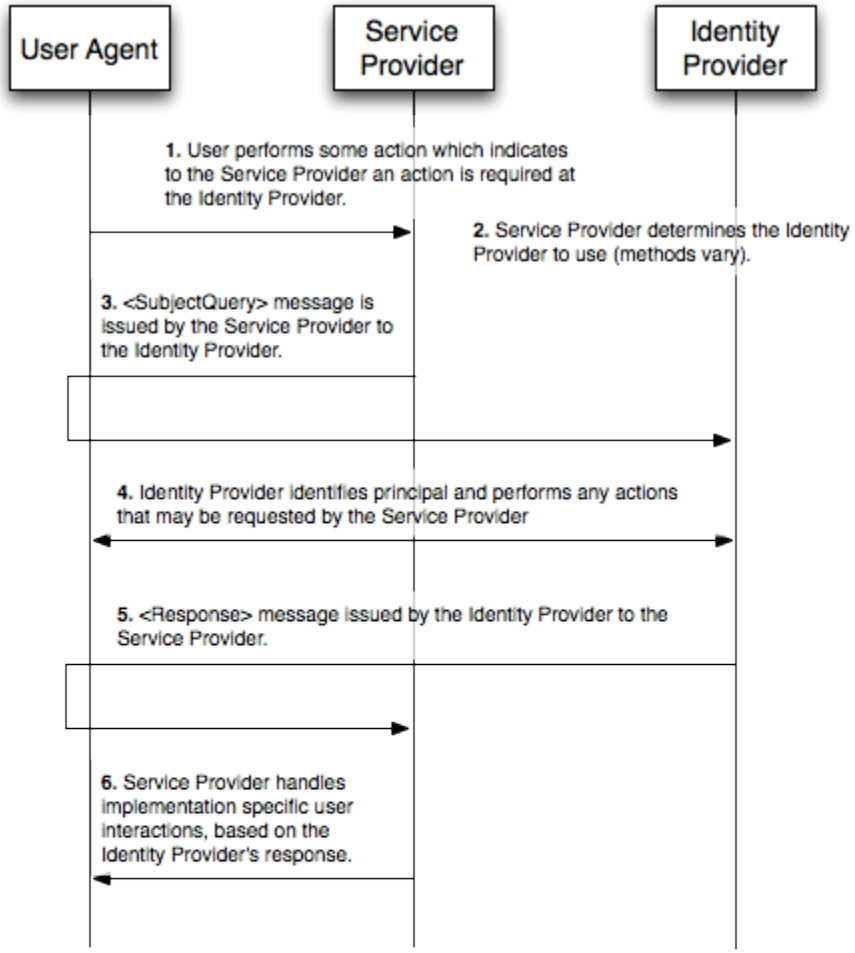

Figure 5: Subject Query Message exchange

1. HTTP Request to Service Provider

In step 1, the principal, via an HTTP User Agent, makes an HTTP with an established security context.

2. Service Provider Determines Identity Provider

In step 2, the service provider obtains the location of an endpoint at an identity provider for the subject query protocol that supports its preferred binding. The means by which this is accomplished is implementation-dependent. The service provider MAY use the SAML identity provider discovery profile described in [SAMLProf] section 4.3.

3. <SubjectQuery> issued by Service Provider to Identity Provider

In step 3, the service provider issues an <SubjectQuery> message to be delivered by the user agent to the identity provider. Either the HTTP Redirect, HTTP POST, or HTTP Artifact binding can be used to transfer the message to the identity provider through the user agent.

4. Identity Provider identifies Principal

In step 4, the principal is identified by the identity provider by some means outside the scope of this profile. This may require a new act of authentication, or it may reuse an existing authenticated session. The identity provider performs implementation-specific operations with the principal as may be indicated in the <SubjectQuery>.

5. Identity Provider issues <Response> to Service Provider

In step 5, the identity provider issues a <Response> message to be delivered by the user to the service provider. Either the HTTP POST, or HTTP Artifact binding can be used to transfer the message to the service provider through the user agent. The message may indicate an error, or will include (at least) appropriate implementation-specific responses (for example, information placed in the <samlp:Extension> point.

6. Service Provider grants or denies access to Principal

In step 6, having received the response from the identity provider, the service provider can respond to the principal's user agent with its own error, or can otherwise interact with the principal in accordance with implementation-specific requirements.

A.3 Profile Description

This profile allows SAML implementations to leverage established SAML protocol bindings in a generalized fashion, and employ the extension point in the <SubjectQuery> and <Response> to convey application-specific requirements.

Requestors and Responders MUST conform to all processing instructions given in [SAMLProf] section 4.1 Web Browser SSO Profile.

A.3.1 HTTP Request to Service Provider

As specified in [SAMLProf] section 4.1.3.1

A.3.2 Service Provider Determines Identity Provider

As specified in [SAMLProf] section 4.1.3.2

A.3.3 <SubjectQuery> is Issued by Service Provider to Identity Provider

This profile requires the request to be issued as <samlp:SubjectQuery> instead of the <AuthnRequest> indicated in [SAMLProf] section 4.1.3.3.

A.3.4 Identity Provider Identifies Principal

This profile does not include the <RequestedAuthnContext> message element, and therefore, the identity provider may choose any authentication mechanism available to it.

A.3.5 Identity Provider Issues <Response> to Service Provider

As specified in [SAMLProf] section 4.1.3.5

A.3.6 Service Provider Processes Response

No security context can be inferred from a response to a <SubjectQuery>. Any response should be considered informative only. The service provider SHOULD confirm the response directly from the identity provider.

A.4 Use of Subject Query

Applications which make use of this profile MUST specify any applicable processing instructions for the identity provider and service provider. Specifically, information which may be conveyed in the request extension point.

If the identity provider wishes to return a SAML protocol error, is SHOULD NOT return any information in the response extension point.

If a Subject is present in the request, the identity provider MUST positively identify the principal indicated in the request.

All response level processing instructions in [SAMLProf] section 4.1.4.3 MUST be adhered to. This includes verification that the IssueInstant, InResponseTo and Destination attributes conform to the requirements set forth in [SAMLProf] section 4.1.4.3

If the HTTP POST binding is used to deliver the <Response>, the response MUST be signed.

The service provider MUST ensure that responses are not replayed, by maintaining the set of used ID values for the length of time for which the assertion would be considered valid based on the NotOnOrAfter attribute.

A.5 Unsolicited Responses

The identity provider MAY initiate this profile as specified in [SAMLProf] section 4.1.5.

A.6 Use of Metadata

Any [SAMLMD] defined Endpoint-type may indicate support for this profile as urn:dece:type:tokenprofile:saml2:subjectquery

Appendix B. Security Mechanism Parameters

This section describes the parameters used elsewhere in this document. Additional usage model variables are defined in Appendix A of [DSystem] and Appendix E of [DCoord].

Parameter	Value	Description
DSECMECH_MAX_IDLE_REMOTE_SESSION_DURATION	7 days	The maximum amount of time between interactions with the User, after which the Node is required to re-authenticate the User before an assertion of identity may be made by the Node to the Coordinator.
DSECMECH_MAX_REMOTE_SESSION_DURATION	6 months	The maximum amount of time that may elapse before a Node must re-authenticate a User before an assertion of identity may be made by the Node to the Coordinator.
DSECMECH_MAX_REMOTE_ASSERTION_DURATION	7 days	The maximum amount of time an assertion of identity shall be valid when issued by a Node to the Coordinator
DSECMECH_MAX_TOKEN_DURATION_DEFAULT	1 year	The default maximum amount of time a Security Token shall be allowed to be considered to be, or allowed to be valid.
DSECMECH_MIN_TOKEN_DURATION_DEFAULT	6 hours	The minimum validity period for Delegation and Federation Security Tokens issued by the Coordinator. The duration of Delegation Security Tokens without UserLinkConsent.
DSECMECH_LLASP_TOKEN_DURATION_DEFAULT	10 years	The default validity period for Delegation Security Tokens Issued to the LLASP role.
DSECMECH_STS_USERCREDENTIAL_TIMELIMIT	15 minutes	The allowed delay between a UserCreate API call, and a corresponding Security Token Service invocation employing the User Credential token profile.
DSECMECH_AUTHN_LOCK_PERIOD	30 minutes	The period of time a User may not authenticate to a Coordinator service when reverse Turing tests cannot me employed (for example, when a Device attempts the HTTP Basic authentication)
DSECMECH_FAILED_AUTHN_ATTEMPTS	3	The number of failed authentication attempts before either a reverse Turing test is required, or the DSECMECH_AUTHN_LOCK_PERIOD is enforced.

Parameter	Value	Description
DSECMECH_AUTHN_ATTEMPT_PERIOD	30 minutes	The period of time in which DSECMECH_FAILED_AUTHN_ATTEMPTS are measured.
DSECMECH_SHORT_REMOTE_SESSION_DURATION	24 hours	The length of time a session with a browser is valid when a User indicates a preference for a short session
DSECMECH_MIN_FEDERATION_DURATION	14 days	The minimum validity period for Federation Security Tokens (minimum session length).

Appendix C. Web Portal TargetURL Values

The following table defines the set of URLs which may be used within Federation Token Profiles to direct a user-agent to a specific page within the Web Portal. These values will remained fixed, with mappings to the relevant pages managed by the Web portal.

TargetURL Parameter	Description
`urn:dece:portal:home`	The main, authenticated page of the Web Portal. May be the same as urn:dece:portal:library
`urn:dece:portal:account`	The account management entry point for the Web Portal
`urn:dece:portal:library`	The library (locker) main page for the Web Portal
`urn:dece:portal:library:{rightsTokenID}`	The details page of the provided `rightsTokenID`
`urn:dece:portal:device`	The main device entry point for the Web Portal
`urn:dece:portal:tou`	The Web Portal-operated terms of use collection URL.
`urn:dece:portal:user`	The user's profile page of the Web Portal (as determined by the User identified in the Federation Token.
`urn:dece:portal:user:{UserID}`	The user profile page of the Web Portal for the provided `UserID`

Appendix D. SAML Request Message Example (Informative)

```
<?xml version="1.0" encoding="UTF-8"?>
<samlp:AuthnRequest
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
ID="urn:dece:org:org:dece:iot:retailer:acmestore_1314391818.1105"
Version="2.0" IssueInstant="2011-08-26T20:50:18+00:00"
AssertionConsumerServiceIndex="1"
Destination="https://iot.p.uvvu.com:7001/rest/1/0/loginservice/login
/">
<saml:Issuer>urn:dece:org:org:dece:iot:retailer:acmestore</saml:Issu
er>
    <samlp:NameIDPolicy Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent"/>
    <saml:Conditions>
        <saml:AudienceRestriction>

<saml:Audience>urn:dece:org:org:dece:iot:retailer:acmestore</saml:Au
dience>
        </saml:AudienceRestriction>
    </saml:Conditions>
    <samlp:RequestedAuthnContext>

<saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:Pa
ssword</saml:AuthnContextClassRef>
    </samlp:RequestedAuthnContext>
</samlp:AuthnRequest>
<?xml version="1.0" encoding="UTF-8"?>
<samlp:AuthnRequest
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
ID="urn:dece:org:org:dece:iot:retailer:acmestore_1314391818.1105"
Version="2.0" IssueInstant="2011-08-26T20:50:18+00:00"
AssertionConsumerServiceIndex="1"
Destination="https://iot.p.uvvu.com:7001/rest/1/0/loginservice/login
/">
<saml:Issuer>urn:dece:org:org:dece:iot:retailer:acmestore</saml:Issu
er>
    <samlp:NameIDPolicy Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent"/>
    <saml:Conditions>
        <saml:AudienceRestriction>

<saml:Audience>urn:dece:org:org:dece:iot:retailer:acmestore</saml:Au
dience>
<saml:Audience>urn:dece:org:org:dece:iot:lasp:acmestore</saml:Audien
ce>
        </saml:AudienceRestriction>
    </saml:Conditions>
    <samlp:RequestedAuthnContext>
```



```
<saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:Pa
ssword</saml:AuthnContextClassRef>
    </samlp:RequestedAuthnContext>
</samlp:AuthnRequest>
```

Appendix D. SAML Response Message Example (Informative)

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2p:Response
Consent="urn:oasis:names:tc:SAML:2.0:consent:unspecified"
Destination="https://example.com/service/login/POST"
ID="urn:dece:coordinator"
InResponseTo="5FFFC00BD297649B037A66D75FA3B620" IssueInstant="2010-
11-08T17:36:34.133Z" Version="2.0"
xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol">
<saml2:Issuer
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">http://c.decellc
.com/</saml2:Issuer>
<saml2p:Status>
<saml2p:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</saml2p:Status>
<saml2:Assertion ID="72541381-a0f6-4d79-aecf-380eed5cade8"
IssueInstant="2010-11-08T17:36:34.133Z" Version="2.0"
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
<saml2:Issuer>http://c.decellc.com/</saml2:Issuer><ds:Signature
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-
exc-c14n#"/>
<ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
<ds:Reference URI="#72541381-a0f6-4d79-aecf-380eed5cade8">
<ds:Transforms>
<ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
<ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"><ec:InclusiveNamespaces PrefixList="ds saml2 xs"
xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"/></ds:Transform>
</ds:Transforms>
<ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
<ds:DigestValue>2s13ZHI0pjQY0f2xgy0BtDZiLtc=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
[signaturedata]
</ds:SignatureValue>
<ds:KeyInfo><ds:X509Data>
<ds:X509Certificate>[Certificate data]</ds:X509Certificate>
</ds:X509Data></ds:KeyInfo></ds:Signature>
<saml2:Subject><saml2:NameID
Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">urn:dece:userid:org:dece:9457119E91628C73E0405B0A
0B344B4C</saml2:NameID>
<saml2:SubjectConfirmation
Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
<saml2:SubjectConfirmationData
InResponseTo="5FFFC00BD297649B037A66D75FA3B620" NotOnOrAfter="2010-
11-09T17:36:34.133Z"
```

```
Recipient="https://example.com/service/login/POST"/>
</saml2:SubjectConfirmation></saml2:Subject>
<saml2:Conditions NotBefore="2010-11-08T17:36:24.133Z"
NotOnOrAfter="2011-11-08T17:36:34.133Z">
<saml2:AudienceRestriction>
<saml2:Audience>urn:dece:org:org:dece:200</saml2:Audience>
<saml2:Audience>urn:dece:org:org:dece:200:002</saml2:Audience>
<saml2:Audience>urn:dece:org:org:dece:200:003</saml2:Audience>
</saml2:AudienceRestriction>
</saml2:Conditions>
<saml2:Advice>
<saml2:AssertionURIRef>https://iot.q.uvvu.com:7001/dece/SecurityToke
n/Assertion/72541381-a0f6-4d79-aecf-
380eed5cade8</saml2:AssertionURIRef>
</saml2:Advice>
<saml2:AuthnStatement AuthnInstant="2010-11-08T17:36:34.133Z">
<saml2:AuthnContext>
<saml2:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:P
assword</saml2:AuthnContextClassRef>
<saml2:AuthenticatingAuthority>urn:dece:coordinator</saml2:Authentic
atingAuthority>
</saml2:AuthnContext></saml2:AuthnStatement>
<saml2:AttributeStatement>
<saml2:Attribute Name="accountID"
NameFormat="urn:dece:type:accountID">
<saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">urn:dece:userid:org:dece:948F0849800D7F59E0405B
0A0B346405</saml2:AttributeValue>
</saml2:Attribute>
</saml2:AttributeStatement>
</saml2:Assertion>
</saml2p:Response>
```

Appendix E. SAML Metadata Example (Informative)

```
<?xml version="1.0" encoding="UTF-8"?>
<md:EntitiesDescriptor
xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <md:EntityDescriptor entityID="urn:dece:org:example">
        <md:SPSSODescriptor AuthnRequestsSigned="true"
WantAssertionsSigned="true"
protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"
validUntil="2012-01-01T00:00:00Z">
            <md:KeyDescriptor use="signing">
                <ds:KeyInfo>
                    <ds:X509Data>
                        <ds:X509Certificate>
                            [PEMEncoded x509 certificate]
                        </ds:X509Certificate>
                    </ds:X509Data>
                </ds:KeyInfo>
            </md:KeyDescriptor>
            <md:ContactPerson contactType="technical">
                <!-- optional identification of the person/persons
responsible for the SAML aspects of the Node -->
                <md:Company>Example Org</md:Company>
                <md:GivenName>Joe</md:GivenName>
                <md:SurName>Plumber</md:SurName>

<md:EmailAddress>joe.plumber@example.org</md:EmailAddress>
                <md:TelephoneNumber>+1 (212) 555
1212</md:TelephoneNumber>
            </md:ContactPerson>
            <md:SingleLogoutService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
                Location="https://saml.example.org/logout/POST"/>
            <md:SingleLogoutService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
                Location="https://saml.example.org/logout/GET"/>
            <md:AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
                Location="https://saml.example.org/login/POST"
index="1" isDefault="true"/>
            <md:AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
                Location="https://other.saml.example.org/login/POST"
index="2"/>
            <md:AssertionConsumerService
                Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
Redirect"
                Location="https://saml.example.org/login/GET"
index="3"/>
        </md:SPSSODescriptor>
    </md:EntityDescriptor>
<!—the affiliation entityID must be different than the entityID of
the sopnsoring organization -->
```

```
    <md:EntityDescriptor
entityID="urn:dece:org:example:affiliation">
        <md:AffiliationDescriptor
affiliationOwnerID="urn:dece:org:example"
            validUntil="2012-02-21T23:12:15.203Z">

<md:AffiliateMember>urn:dece:org:example:node001</md:AffiliateMember
>

<md:AffiliateMember>urn:dece:org:example:node002</md:AffiliateMember
>

<md:AffiliateMember>urn:dece:org:example:node003</md:AffiliateMember
>
        </md:AffiliationDescriptor>
    </md:EntityDescriptor>

</md:EntitiesDescriptor>
```

END