CableLabs® Specifications
Online Content Access

Authentication and Authorization Interface 1.1 Specification

CL-SP-AUTH1.1-I01-160616

ISSUED

DISCLAIMER

This document is furnished on an "AS IS" basis and neither CableLabs nor its members provides any representation or warranty, express or implied, regarding the accuracy, completeness, noninfringement, or fitness for a particular purpose of this document, or any document referenced herein. Any use or reliance on the information or opinion in this document is at the risk of the user, and CableLabs and its members shall not be liable for any damage or injury incurred by any person arising out of the completeness, accuracy, or utility of any information or opinion contained in the document.

CableLabs reserves the right to revise this document for any reason including, but not limited to, changes in laws, regulations, or standards promulgated by various entities, technology advances, or changes in equipment design, manufacturing techniques, or operating procedures described, or referred to, herein.

This document is not to be construed to suggest that any company modify or change any of its products or procedures, nor does this document represent a commitment by CableLabs or any of its members to purchase any product whether or not it meets the characteristics described in the document. Unless granted in a separate written agreement from CableLabs, nothing contained herein shall be construed to confer any license or right to any intellectual property. This document is not to be construed as an endorsement of any product or company or as the adoption or promulgation of any guidelines, standards, or recommendations.

Document Status Sheet

Document Control Number:	CL-SP-AUTH1.1-I01-160616			
Document Title:	Authentication and Authorization Interface 1.1 Specification			
Revision History:	I01 - Released 06/16/16			
Date:	June 16, 2016			
Status:	~~Work in Progress~~	~~Draft~~	**Issued**	~~Closed~~
Distribution Restrictions:	~~Author Only~~	~~CL/Member~~	~~CL/ Member/ Vendor~~	**Public**

Key to Document Status Codes

Work in Progress	An incomplete document, designed to guide discussion and generate feedback that may include several alternative requirements for consideration.
Draft	A document in specification format considered largely complete, but lacking review by Members and vendors. Drafts are susceptible to substantial change during the review process.
Issued	A generally public document that has undergone Member and Technology Supplier review, cross-vendor interoperability, and is for Certification testing if applicable. Issued Specifications are subject to the Engineering Change Process.
Closed	A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

Trademarks

Contents

Figures

Tables

This page left blank intentionally.

1 INTRODUCTION

1.1 Overview

Initiated by CableLabs member companies, the Online Content Access (OLCA) interoperable protocol specification provides technical requirements and architecture for the delivery of video to a Multichannel Video Programming Distributor (MVPD) Customer from different online sources.

A number of MVPDs have already deployed either field trials or commercial services of online content access, in collaboration with a number of Programmers. There is an interest among MVPDs and Programmers in developing common architectures, interfaces, and operations for online content access services. It is their belief that a common approach offers greater choice to consumers, expanding the service to include more service providers as well as enable competition among technology providers to support the market.

As shown in Figure 1, this specification addresses the common standards and specifications to support authentication and authorization.

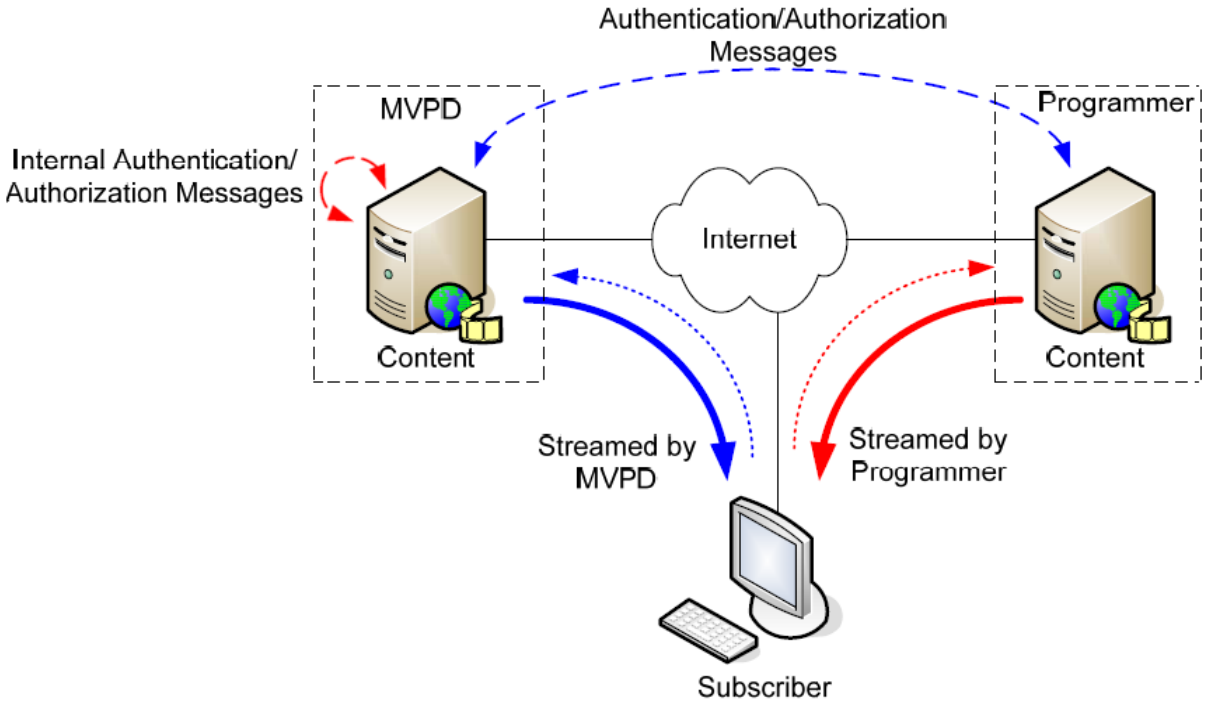

Figure 1 - High-level MVPD, Programmer Ecosystem

OLCA 1.1 contains the following changes to OLCA 1.0 that provide the ability to:

- Identify users as belonging to a household or subscriber grouping - needed for home based authentication, concurrency monitoring/fraud management
- Identify parents and children for the purposes of parental control and/or account management
- Enable PIN entry functions to temporarily elevate a user's privilege on Content Provider's sites and apps
- Enable Content Providers to request attributes necessary to enabling specific scenarios (e.g., In Home/Out of Home, Device Type, future requirements)
- Pre-screen authentication to determine a user's authorization status (passive AuthZ)
- Enable MVPDs to handle an authorization at the same time as authentication - improving user experience

In addition, emphasis and clarification has been added regarding how to use Passive/Forced Authentication to facilitate frictionless authentication and improve user assurance.

1.2 Principal Objectives

The principal service objectives of the cable operators engaged in providing online access to subscription cable TV services are listed below.

- Enable consumer choice and convenience:
  - Enable consumer choice of devices: TV, PC, and mobile devices
  - Enable easy access by consumers, avoiding multiple logins and complex navigation
- Protect content owner's rights by addressing security threats:
  - Theft of service
  - Appropriate measures for user authentication and fraud prevention
- Provide the appropriate level of security to safeguard the consumer's privacy and protect the consumer's identity
- Enable both ad-supported and subscription channel statistics:
  - Provide appropriate viewership statistics to support ratings measurement while protecting consumer's privacy and identity
- Leverage common open standards to the extent possible
- Enable interoperability among the major system components allowing for multiple technology providers to provide products and services to the ecosystem

1.3 Scope

Online Content Access service can take on many forms, use different technologies and business models, and be developed by various entities in the online video arena. For example, an MVPD may enable its Customers access to entitled video content online at a Programmer's website.

Figure 2 shows these components, their interfaces, and some of their sub-functions.

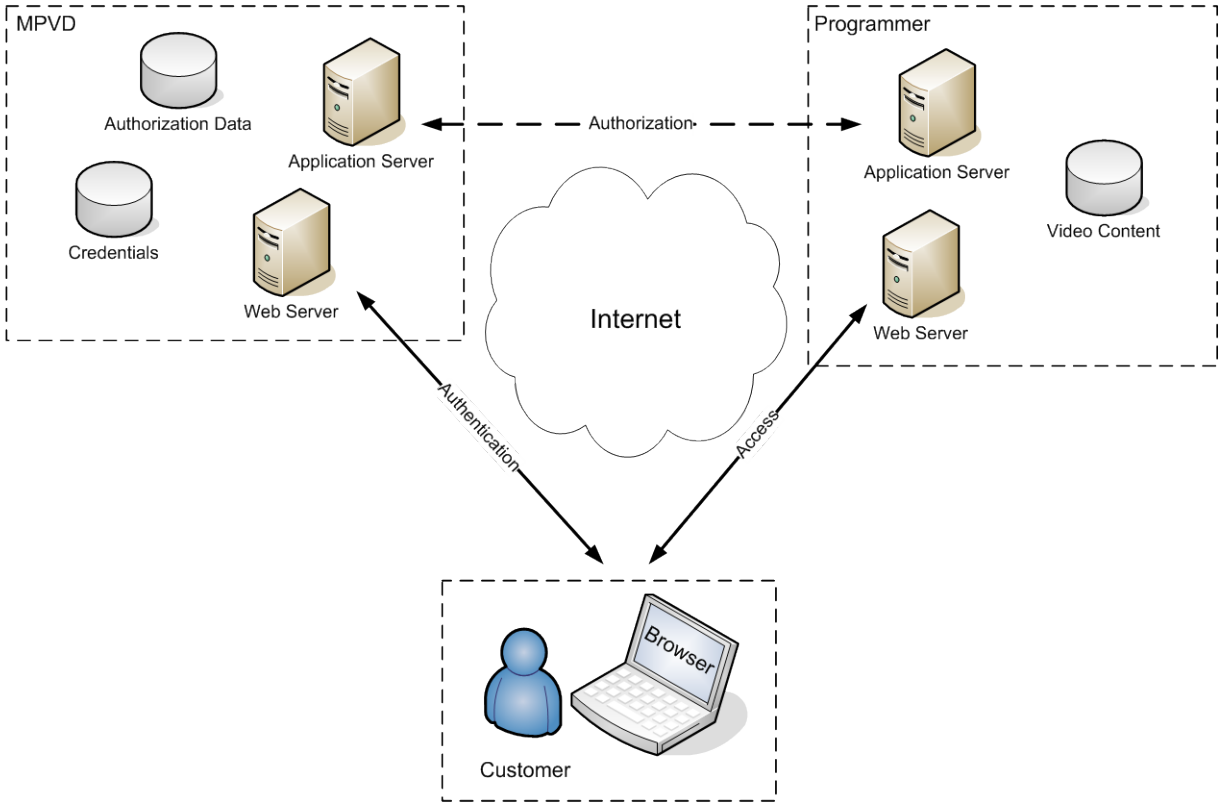

Figure 2 - Sample OLCA Architecture

This specification focuses on the interface between two main players in this space, the MVPD and the Programmer, and defines the authentication and authorization interfaces between these two entities to enable an online content access service.

1.3.1 Actors

The three main actors in the online content access service are the MVPD, the Customer, and the Programmer.

1.3.1.1 MVPD

Multichannel Video Programming Distributor (MVPD) is a term defined by the Federal Communications Commission (FCC) to mean an entity such as, but not limited to, a cable operator, a Multiple System Operator (MSO), a multiple channel distribution service, or a television receive-only satellite program distributor who makes available for purchase by Subscribers or customers, multiple channels of video programming. MVPD encompasses all providers of multichannel TV, including MSOs, Private Cable Operators (PCOs), and Competitive Local Exchange Carriers (CLECs).

1.3.1.2 Customer

The Customer is any person within a household, or business that legally receives video service. A household or business entity account may support more than one Customer with its own authentication credentials and authorization status.

1.3.1.3 Programmer

The Programmer is any entity that, by means other than broadcasting, provides or distributes programming for retransmission by MVPD systems.

1.3.2 Roles

This specification also uses roles such as Service Provider, Authentication Provider, Authorization Provider, and Content Provider to describe the roles that an MVPD or Programmer can take on in the Online Content Access service.

1.3.2.1 Service Provider

The Service Provider (SP) role provides the Subscriber interface and access control to online video content. An MVPD or Programmer can be a Service Provider. The Service Provider relies on the Authentication Provider for Customer authentication, the Authorization Provider for authorization, and the Content Provider for the content delivery to the Customer.

1.3.2.2 Authentication Provider

The Authentication Provider (AnP) role creates, maintains, and manages the Customer identity information. The AnP provides Customer authentication for the Service Provider. Within the scope of this document, only an MVPD can play the role of an Authentication Provider.

1.3.2.3 Authorization Provider

The Authorization Provider (AzP) role creates, maintains, and manages the Customer authorization information. The AzP provides Customer authorization for the Service Provider. Within the scope of this document, only an MVPD can play the role of an Authorization Provider.

1.3.2.4 Content Provider

The Content Provider role delivers content to the Customer. The MVPD or Programmer can be a Content Provider. The Content Provider is responsible for the delivery and protection of the content.

1.3.3 Subscriber

The Subscriber serves as the primary role for the Customer. The Subscriber possesses credentials used to authenticate with the AnP.

1.3.4 Actor/Role Mappings

1.3.4.1 MVPD

As shown in Figure 3, and within the scope of this document, an MVPD can play the role of a Service Provider, Content Provider, Authentication Provider, Authorization Provider, or a combination thereof.

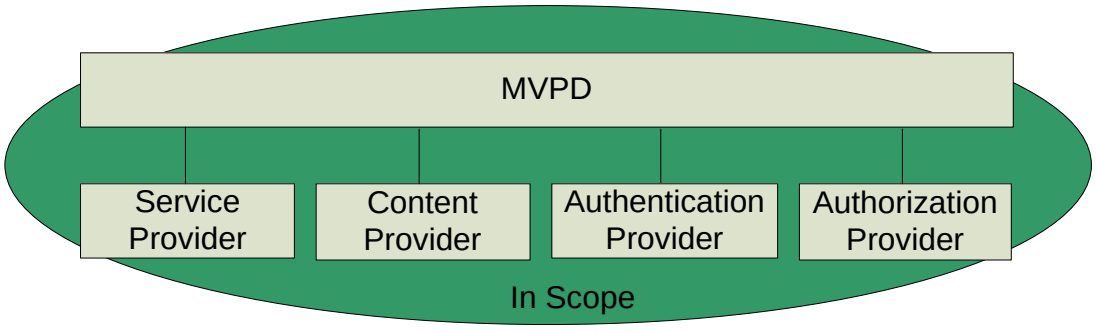

Figure 3 - MVPD Roles

1.3.4.2 Programmer

As shown in Figure 4, and within the scope of this document, a Programmer can play the role of a Service Provider, a Content Provider, or a combination thereof, but the role of a Programmer as an Authentication or Authorization Provider is out of scope for this specification.

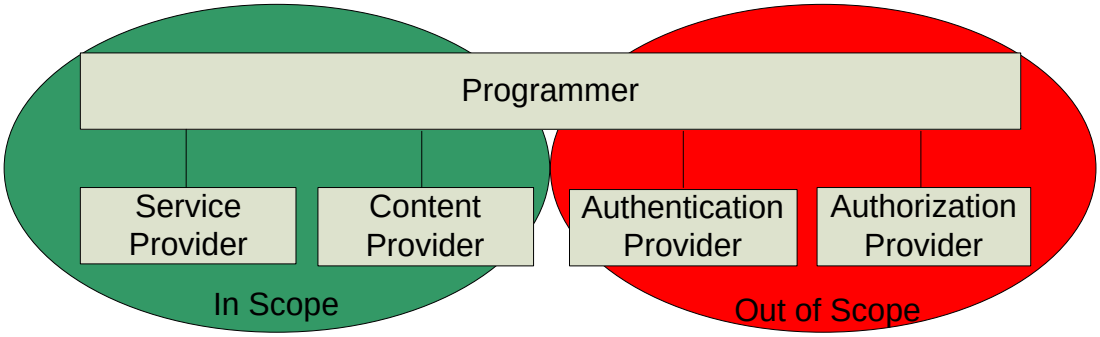

Figure 4 - Programmer Roles

1.3.4.3 Customer

As shown in Figure 5, and within the scope of this document, a Customer can take on the sole role of the Subscriber.

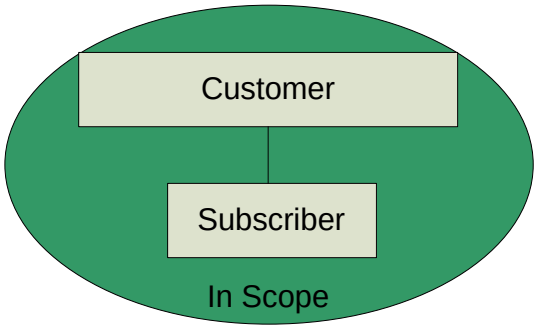

Figure 5 - Customer Roles

1.3.5 User Experience Task Flow

Figure 6 outlines the generalized task flow for the use cases covered herein and can be broken down into the following tasks:

- Portal Entry – Subscriber navigates to the SP web portal
- Portal Exit – Subscriber exits the SP web portal
- Authentication – Subscriber logs in using AnP provided credentials
- Authorization – Subscriber selects content for playback and receives authorization from the AzP
- Browse – Subscriber browses for content of interest

All use cases defined below refer directly to this task flow.

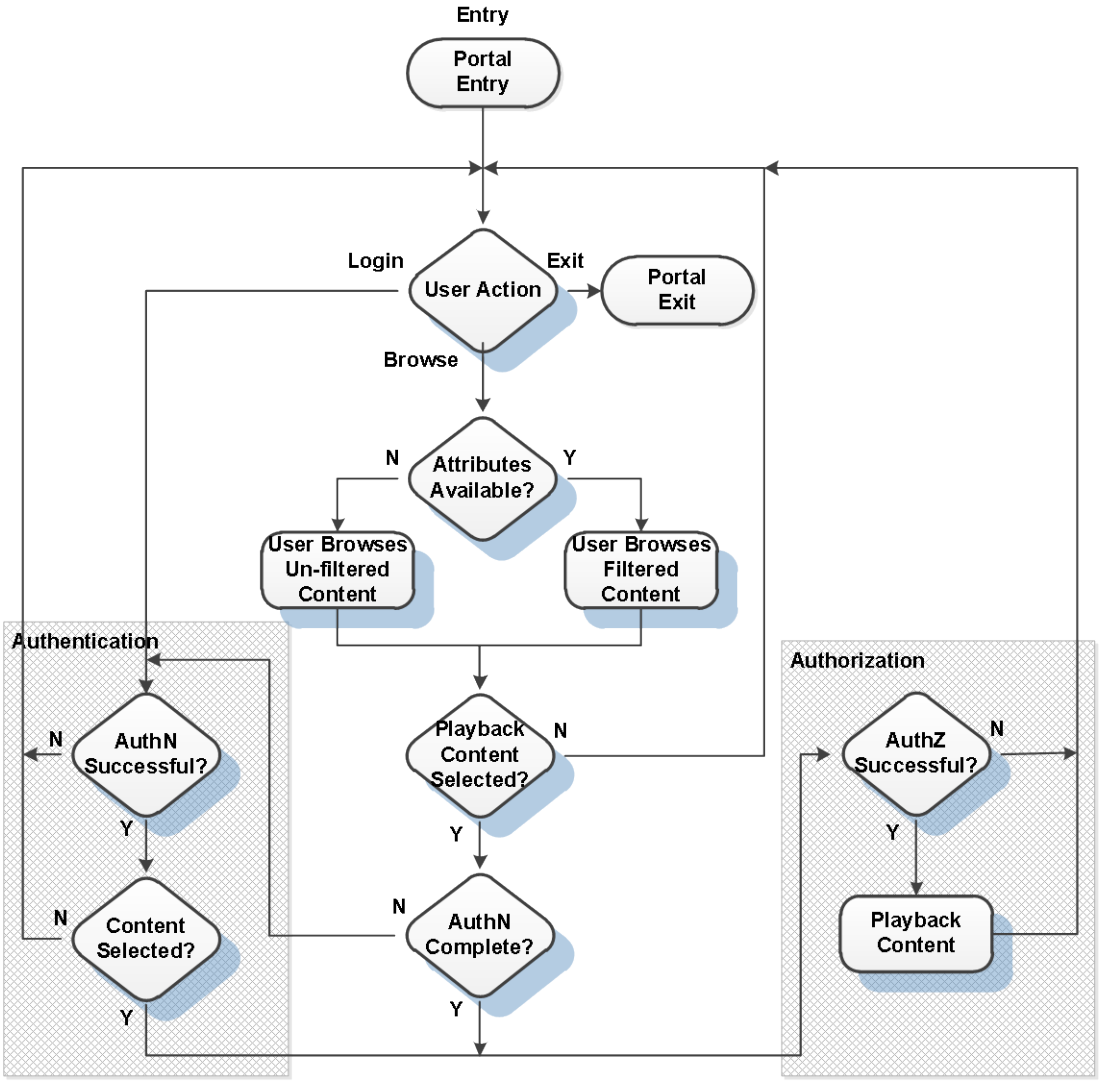

Figure 6 - User Experience Task Flow

1.4 Requirements (Conformance Notation)

Throughout this document, the words that are used to define the significance of particular requirements are capitalized. These words are:

"MUST"	This word means that the item is an absolute requirement of this specification.
"MUST NOT"	This phrase means that the item is an absolute prohibition of this specification.
"SHOULD"	This word means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
"SHOULD NOT"	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
"MAY"	This word means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

2 REFERENCES

2.1 Normative References

In order to claim compliance with this specification, it is necessary to conform to the following standards and other works as indicated, in addition to the other requirements of this specification. Notwithstanding, intellectual property rights may be required to use or implement such normative references.

[ISO 3166]	ISO 3166, Country Codes, http://www.iso.org/iso/country_codes/country_codes
[RFC 7519]	IETF RFC 7519, JSON Web Token (JWT), May, 2015.
[RFC 2459]	IETF RFC 2459, Internet X.509 Public Key Infrastructure Certificate and CRL Profile, January 1999.
[RFC 5246]	IETF RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2, August 2008.
[SAML 2.0 BINDINGS]	http://docs.oasis-open.org/security/saml/v2.0/saml-bindings-2.0-os.pdf
[SAML 2.0 CONFORMANCE]	http://docs.oasis-open.org/security/saml/v2.0/saml-conformance-2.0-os.pdf
[SAML 2.0 CORE]	http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf
[SAML 2.0 Metadata]	http://docs.oasis-open.org/security/saml/v2.0/saml-metadata-2.0-os.pdf
[SAML 2.0 Profile XACML 2.0]	http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-profile-spec-os.pdf
[SAML 2.0 PROFILES]	http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf
[SAML 2.0 Security]	http://docs.oasis-open.org/security/saml/v2.0/saml-sec-consider-2.0-os.pdf
[XACML 2.0 Spec Core]	http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

2.2 Informative References

This specification uses the following informative references.

[AUTH1.0]	Authentication and Authorization Interface 1.1 Specification, CL-SP-AUTH1.0-C01-160616, June 16, 2016, Cable Television Laboratories, Inc.
[ID-IdP]	RFC IEFT Draft: PingPong IdP Discovery Protocol, draft-efazendin-pingpong-idp-discovery-00, December 2010.
[XPath]	www.w3.org/TR/xpath/

2.3 Reference Acquisition

- Internet Engineering Task Force (IETF) Secretariat, 48377 Fremont Blvd., Suite 117, Fremont, California 94538, USA, Phone: +1-510-492-4080, Fax: +1-510-492-4001, http://www.ietf.org
- OASIS – SAML documents, http://wiki.oasis-open.org/security.
- World Wide Web Consortium (W3C), http://www.w3.org.

3 TERMS AND DEFINITIONS

This specification uses the following terms:

Assertion	A piece of data produced by a SAML authority regarding either an act of *authentication* performed on a *subject*, *attribute* information about the subject, or *authorization* data applying to the subject with respect to a specified *resource*.
Attribute	A distinct characteristic of a *subject*. A subject's attributes are said to describe it, such as size, type of encoding, network address, etc. Attributes are often represented as pairs of "attribute name" and "attribute value(s)", e.g., "count" has the value 1. Often, this is referred to as "attribute value pairs."
Authentication	To confirm a *system entity's* asserted *principal identity* with a specified, or understood, level of confidence.
Authentication Provider	A kind of *service provider* that creates, maintains, and manages identity information for *principals* and provides principal authentication to other service providers within a *federation*, such as with web browser *profiles*.
Authorization	The act of evaluating *access control information*, as to whether a *subject* is allowed the specified types of access to a particular *resource*.
Authorization Provider	A kind of *service provider* that creates, maintains, and manages authorization information for *principals* and provides principal authorization to other service providers within a *federation*, such as with web browser *profiles*.
Back Channel	A direct connection between two entities (not going through end user's device).
Content	A unit of video.
Content ID	A unique identifier for a unit of video programming OR channel. Please refer to the Content ID team's documentation for further details.
Credentials	Credentials in cryptography establish the identity of a party to communication. Usually they take the form of machine-readable cryptographic keys and/or passwords. Cryptographic credentials may be self-issued, or issued by a trusted third party; in many cases, the only criterion for issuance is unambiguous association of the credentials with a specific, real individual or other entity. Cryptographic credentials are often designed to expire after a certain period, although this is not mandatory. An x.509 certificate is an example of a cryptographic credential.
Customer	See Subscriber.
Digital Rights Management (DRM)	A technology that manages access to digital content or services to enable access and use as designated by the provider, and to prevent unauthorized access and use. DRM may prevent the sharing or copying of digital content or tie the use or viewing of content to specific individuals, operating systems, or hardware.
Multichannel Video Programming Distributor (MVPD)	A service provider delivering video programming services, usually for a subscription fee. These providers include cable operators, direct-broadcast satellite (DBS) providers, and wireline video providers.
Party	Informally, one or more *principals* participating in some process or communication, such as receiving an *assertion* or accessing a *resource*.
Principal	A *system entity* whose identity can be authenticated.
Programmer	A Programmer is any person, firm or corporation that, by means other than broadcasting, provides or distributes programming for retransmission by MVPD systems.

Resource	Data contained in an information system (for example, in the form of files, information in memory, etc.), as well as: a) A service provided by a system b) An item of system equipment (in other words, a system component such as hardware, firmware, software, or documentation) SAML refers to resources by means of URI references.
Role	*System entity* function or position. System entities can take on various types of roles serially and/or simultaneously, for example, active roles and passive roles. The notion of an Administrator is often an example of a role.
Security Assertion Markup Language (SAML)	The set of specifications describing security assertions that are encoded in XML, profiles for attaching the assertions to various protocols and frameworks, the request/response protocol used to obtain the assertions, and bindings of this protocol to various transfer protocols (for example, SOAP and HTTP).
Service Provider	A role donned by a system entity where the system entity provides services to principals or other system entities.
Subject	A *principal* in the context of a security domain. SAML assertions make declarations about subjects.
Subscriber	A person, household, or business that legally receives and pays for video service for its own use.
User	A person who makes use of a system and its resources for any purpose.

4 ABBREVIATIONS, ACRONYMS, SYMBOLS

This specification uses the following abbreviations:

AA Attribute Authority

AnP Authentication Provider

AzP Authorization Provider

CLEC Competitive Local Exchange Carriers

CRL Certificate Revocation List

CSRF Cross Site Request Forgeries

CP Content Provider

DBS Direct Broadcast Satellite

DoS Denial of Service Attack

DNS Domain Name Service

DRM Digital Rights Management

FCC Federal Communications Commission

FQDN Fully Qualified Domain Name

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

IP Internet Protocol

ISM Inter-System Messaging

JSON JavaScript Object Notation

JWT JSON Web Token

MPAA Motion Picture Association of America

MSO Multiple System Operator

MVPD Multichannel Video Programming Distributor

OASIS Organization for the Advancement of Structured Information Standards

OLCA Online Content Access

PAP Policy Administration Point

PCO Private Cable Operators

PDP Policy Decision Point

PEP Policy Enforcement Point

PIP Policy Information Point

PKI Public Key Infrastructure

SAML Security Assertion Markup Language

SOAP Simple Object Access Protocol

SP	Service Provider
SSL	Secure Socket Layer
SSO	Single Sign On
TCP	Transmission Control Protocol
TLS	Transport Layer Security
URL	Uniform Resource Locator
UX	User Experience
XACML	Extensible Access Control Markup Language
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
ZSO	Zero Sign On

5 ONLINE CONTENT ACCESS SCENARIOS

A Customer of a multi-channel video programming distributor (MVPD), as part of his subscription, has access to content from multiple Programmers. Today, the Customer enjoys these programs on a TV in the living room. For convenience and other reasons, the Customer is also interested in watching these programs, inside and outside the house, for example, on a PC with a browser and internet connectivity. The Customer wants to visit a Programmer's or MVPD's website from anywhere in the world, navigate among the various content offerings, select from the content offerings and watch content for which he/she is already subscribed to via the MVPD. Additionally, the Customer would like to use MVPD credentials (e.g., username and password) to access content online, and should not be required to create new credentials at the Service Provider's site.

This section provides a list of user-centric use cases and service requirements for the Online Content Access. It also includes assumed user-centric use cases and service requirements for OLCA that may drive Authentication and Authorization requirements.

Different scenarios are considered, based primarily on the roles played by the MVPD and Programmer actors.

Implementation requirements defined in this specification support scenario 1 of Section 5.1 only. Requirements may be updated in the future to support the other scenarios.

5.1 Scenario 1

Scenario 1 consists of the Customer visiting a Programmer's web portal with the interest of streaming content. Customer authentication and authorization functions are performed by the Customer's MVPD to verify subscription to the desired content. Upon authentication and authorization, the Programmer allows access to the requested video content.

5.1.1 Actors and Roles

The Customer plays the role of the Subscriber, the Programmer plays the role of a Service Provider and Content Provider, and the MVPD plays the role of the Authentication and Authorization Providers.

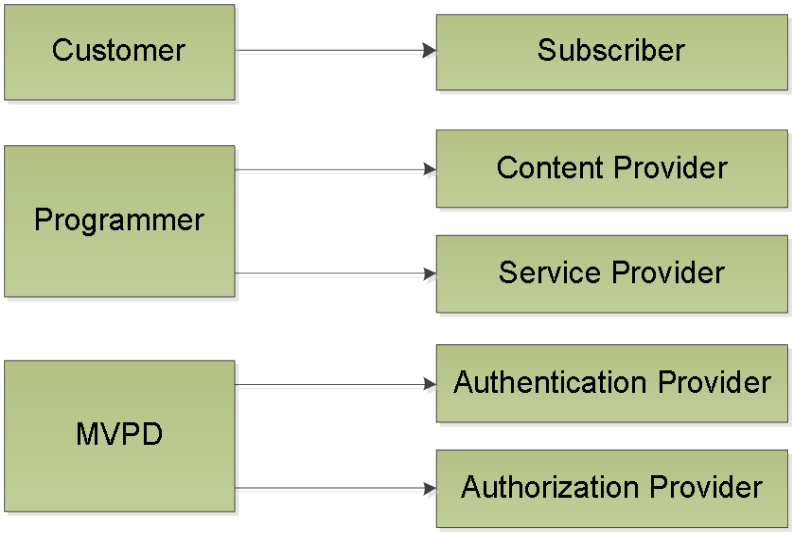

Figure 7 - Actors and Roles for Scenario 1

5.1.2 Architecture

In scenario 1, the MVPD and Programmer actors are separate business entities. As indicated in the previous section, each of them plays different roles to enable the MVPD Customers access to their subscribed video content at the

Programmer's web portal. Figure 8 provides an architectural diagram that shows where these functional roles are located with respect to the actors in scenario 1. It also indicates the main messaging interfaces.

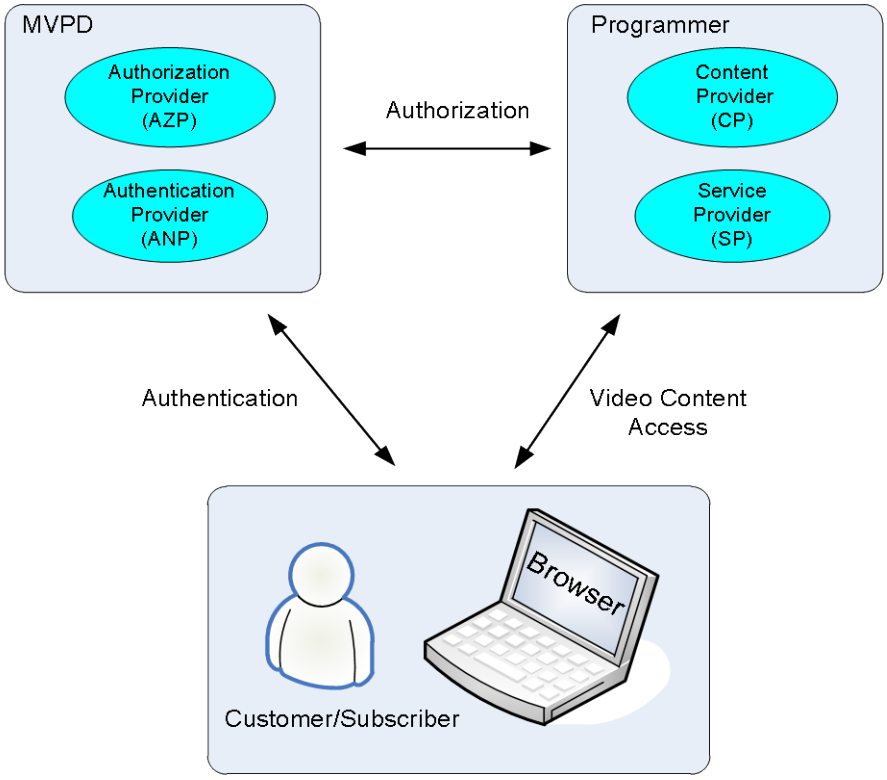

Figure 8 - Scenario 1: System Architecture

Using the web browser, the Customer visits the Programmer web portal to access video content. Normally, the portal web page will have a login button. The SP role controls access to video content and therefore, when the Customer clicks the login button it will want to authenticate the Customer. At this point, the SP will prompt the Customer for the MVPD name or the user name and password for the Programmer account. If the Customer selects an MVPD name, the SP redirects his browser to the MVPD for authentication. The AnP role at the MVPD site authenticates the Customer using his Subscriber credentials (e.g., username password). Once the Customer has successfully authenticated, the AnP redirects the browser back to the Programmer's site. The SP then verifies that the MVPD properly authenticated the Customer.

Once the SP verifies Customer authentication it displays available video content for selection. When the Customer selects video content, the SP sends an authorization status request for that content to the MVPD. The AzP at the MVPD determines the authorization status based on the Customer's subscription level and responds back to the Programmer. If the Customer has been authorized to view the content, the SP allows access to the Content Provider, which then streams the selected video content to the Customer.

5.1.3 Pre-conditions

The following pre-conditions must be met for all use cases described below.

1. Customer will have an active Video subscription with the MVPD.
2. Customer will have access to a PC with web browser, video player, and a broadband internet connection.
3. There is an existing business relationship between the Programmer and the MVPD.

4. Customer knows the name of his MVPD of interest.
5. MVPD and Programmer have a business relationship.

5.1.4 Use Cases

5.1.4.1 Use Case 1

Name	Customer Accesses Programmer Website With Initial Login Authorization Attributes Available
Summary	Customer requests access to Online Content Access service at a Programmer website. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume. Authorization attributes per Section 7 are used to filter content.
Actor(s)	Customer plays role of Subscriber Programmer plays role of Service Provider and Content Provider MVPD plays role of Authentication and Authorization Provider
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to Programmer web portal
Terminates	Customer closes session at Programmer web portal
Main Steps	1. Customer navigates to Programmer's web portal, which is acting as the Service Provider. 2. Customer selects "Sign In" and is presented with a list of supported MVPDs. 3. Customer selects the MVPD with which they have a video subscription. 4. The Programmer, as the SP, redirects the Customer's browser to the MVPD (acting as the Authentication Provider) for authentication. 5. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 6. The AnP sends the results of the login to the SP along with authorization attributes and redirects the Customer's browser back to the SP. 7. SP verifies Customer authentication and displays filtered video content based on the Customer's authorization attributes for selection. 8. Customer selects video content. 9. SP sends authorization status request to MVPD, now acting as the Authorization Provider, for selected content. 10. AzP determines Customer authorization status for requested content and responds to SP. 11. SP verifies that Customer is authorized to access content. 12. SP allows Customer to access/consume selected video content from Content Provider.
Post-Conditions	Subscriber has access to desired video content online.

5.1.4.2 Use Case 2

Name	Customer Accesses Programmer Website Without Initial Login
Summary	Customer selects content from a Programmer website prior to Online Content Access. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume.
Actor(s)	Customer plays role of Subscriber Programmer plays role of Service Provider and Content Provider MVPD plays role of Authentication and Authorization Provider
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to Programmer web portal
Terminates	Customer closes session at Programmer web portal

Name	Customer Accesses Programmer Website Without Initial Login
Main Steps	1. Customer navigates to Programmer's web portal, which is acting as the Service Provider. 2. Customer selects video content. 3. Customer is presented with a list of supported MVPDs. 4. Customer selects the MVPD with which they have a video subscription. 5. The Programmer, as the SP, redirects the Customer's browser to the MVPD (acting as the AnP) for authentication. 6. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 7. The AnP sends the results of the login to the SP and redirects the Customer's browser back to the SP. 8. SP verifies Customer authentication and sends authorization status request to MVPD, now acting as the Authorization Provider, for selected content. 9. AzP determines Customer authorization status for requested content and responds to SP. 10. SP verifies that Customer is authorized to access content. 11. SP allows Customer to access/consume selected video content from Content Provider.
Post-Conditions	Subscriber has access to desired video content online.

5.1.4.3 Use Case 3

Name	Customer Accesses Programmer Website With Initial Login No Authorization Attributes Available
Summary	Customer requests access to Online Content Access service at a Programmer website. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume. No authorization attributes per Section 7 are available for filtering.
Actor(s)	Customer plays role of Subscriber Programmer plays role of Service Provider and Content Provider MVPD plays role of Authentication and Authorization Provider
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to Programmer web portal
Terminates	Customer closes session at Programmer web portal
Main Steps	1. Customer navigates to Programmer's web portal, which is acting as the Service Provider. 2. Customer selects "Sign In" and is presented with a list of supported MVPDs. 3. Customer selects the MVPD with which they have a video subscription. 4. The Programmer, as the SP, redirects the Customer's browser to the MVPD (acting as the Authentication Provider) for authentication. 5. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 6. The AnP sends the results of the login to the SP with no authorization attributes included and redirects the Customer's browser back to the SP. 7. SP verifies Customer authentication and displays video content for selection that does not use authorization attributes for filtering. 8. Customer selects video content. 9. SP sends authorization status request to MVPD, now acting as the Authorization Provider, for selected content. 10. AP determines Customer authorization status for requested content and responds to SP. 11. SP verifies that Customer is authorized to access content. 12. SP allows Customer to access/consume selected video content from Content Provider.
Post-Conditions	Subscriber has access to desired video content online.

5.2 Scenario 2

Scenario 2 consists of the Customer visiting an MVPD's web portal with the interest of streaming content. Authentication and authorization credentials are processed directly by the MVPD to verify subscription to the desired content. Upon authentication and authorization, the MVPD allows access to the requested video content.

5.2.1 Actors and Roles

The Customer plays the role of the Subscriber; the MVPD plays the role of a Service Provider and Content Provider, as well as the Authentication and Authorization Providers.

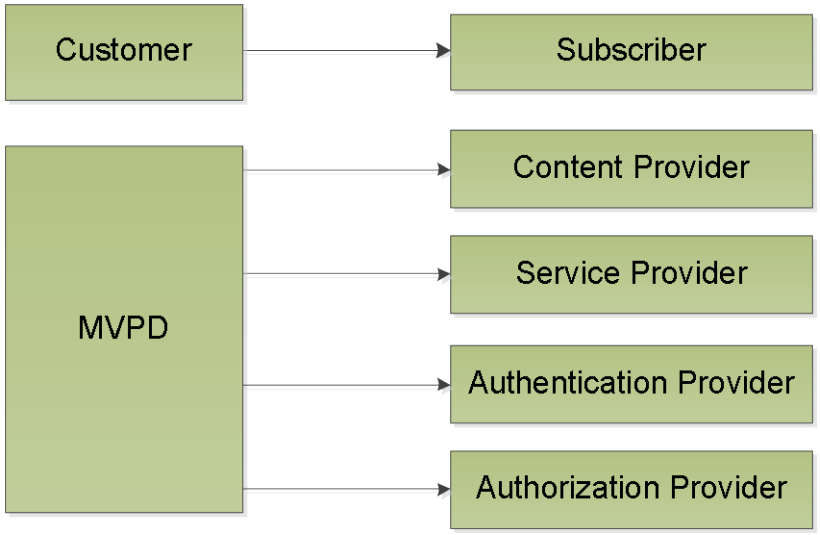

Figure 9 - Actors and Roles for Scenario 2

5.2.2 Architecture

In scenario 2, the MVPD acts as a single unified business entity. As indicated in the previous section, the MVPD plays all of the roles required to enable the Customer access to the subscribed video content at the MVPD's web portal.

Figure 10 provides an architectural diagram that shows where these functional roles are located with respect to the actors. It also indicates the main messaging interfaces.

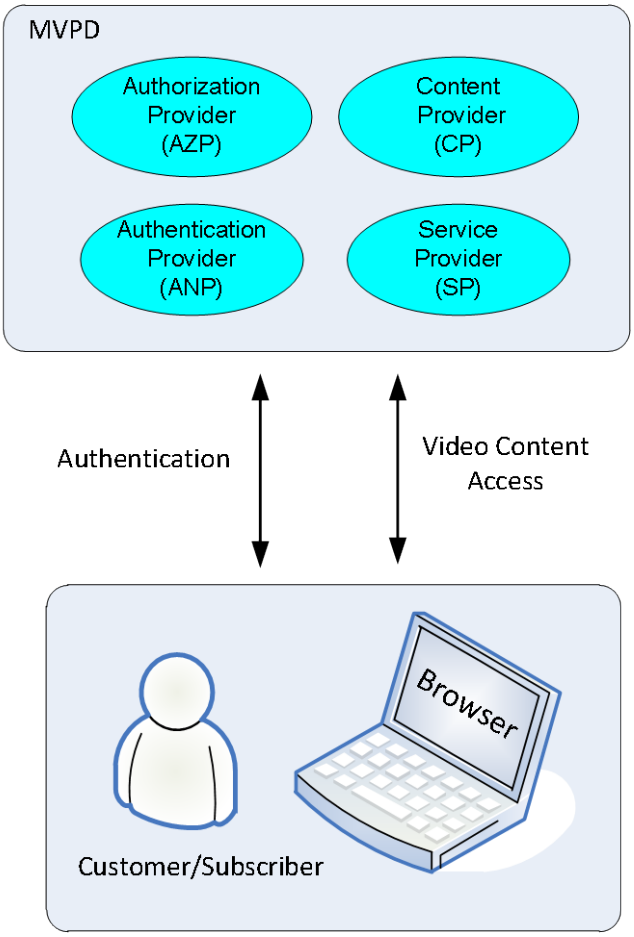

Figure 10 - Scenario 2: System Architecture

Using the web browser, the Customer visits the MVPD web portal to access video content. Normally, the portal web page will have a login button. The SP role controls access to video content and therefore, when the Customer clicks the login button it will initiate authentication of the Customer. The SP redirects the Customer to the AnP at the MVPD site, which prompts for login credentials. The AnP authenticates the Customer using the Subscriber credentials (e.g., username password).

Once the SP verifies Customer authentication it displays available video content for selection. When the Customer selects video content, the SP sends an authorization status request for that content to the AzP. The AzP at the MVPD determines the authorization status based on the Customer's subscription level and responds back to the SP. If the Customer has been authorized to view the content, the SP allows access to the Content Provider, which then streams the selected video content to the Customer.

5.2.3 Pre-conditions

The following pre-conditions exist for all use cases described below.

1. Customer has an active Video subscription with the MVPD.
2. Customer has access to a PC with web browser, video player, and a broadband internet connection.
3. Customer has credentials with an Authentication Provider for authentication.

5.2.4 Use Cases

5.2.4.1 Use Case 1

Name	Customer Accesses MVPD's Website With Initial Login Authorization Attributes Available
Summary	Customer requests access to Online Content Access service at an MVPD website. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume. Authorization attributes are used to filter content.
Actor(s)	Customer plays role of Subscriber MVPD plays role of Service, Content, Authentication and Authorization Providers
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to MVPD's web portal.
Terminates	Customer closes session at MVPD's web portal.
Main Steps	1. Customer navigates to MVPD's web portal, which is acting as the SP. 2. Customer selects "Sign In" and is prompted for login credentials. 3. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 4. The AnP sends the results of the login to the SP along with authorization attributes. 5. SP verifies Customer authentication and displays filtered video content based on the Customer's subscription for selection. 6. Customer selects video content. 7. SP sends authorization status request to MVPD's AzP, for selected content. 8. AzP determines Customer authorization status for requested content and responds to SP. 9. SP verifies that Customer is authorized to access content. 10. SP allows Customer to access/consume selected video content from Content Provider.
Post-Conditions	Subscriber has access to desired video content online.

5.2.4.2 Use Case 2

Name	Customer Accesses MVPD Website Without Initial Login
Summary	Customer selects content from an MVPD website prior to Online Content Access. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume.
Actor(s)	Customer plays role of Subscriber MVPD plays role of Service, Content, Authentication and Authorization Providers
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to MVPD's web portal.
Terminates	Customer closes session at MVPD's web portal.
Main Steps	1. Customer navigates to MVPD's web portal, which is acting as the Service Provider. 2. Customer selects video content. 3. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 4. The AnP sends the results of the login to the SP. 5. SP verifies Customer authentication and sends authorization status request to the MVPD's AzP, for the selected content. 6. AzP determines Customer authorization status for requested content and responds to SP. 7. SP verifies that Customer is authorized to access content. 8. SP allows Customer to access/consume selected video content from Content Provider.
Post-Conditions	Subscriber has access to desired video content online.

5.2.4.3 Use Case 3

Name	Customer Accesses MVPD's Website With Initial Login No Authorization Attributes Available
Summary	Customer requests access to Online Content Access service at an MVPD's website. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume. No authorization attributes are available for filtering.
Actor(s)	Customer plays role of Subscriber MVPD plays role of Service, Content, Authentication and Authorization Providers
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to MVPD's web portal.
Terminates	Customer closes session at MVPD's web portal.
Main Steps	1. Customer navigates to MVPD's web portal, which is acting as the SP. 2. Customer selects "Sign In" and is prompted for login credentials. 3. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 4. The AnP sends the results of the login to the SP without authorization attributes. 5. SP verifies Customer authentication and displays unfiltered video content. 6. Customer selects video content. 7. SP sends authorization status request to MVPD's AzP, for selected content. 8. AzP determines Customer authorization status for requested content and responds to SP. 9. SP verifies that Customer is authorized to access content. 10. SP allows Customer to access/consume selected video content from Content Provider.
Post-Conditions	Subscriber has access to desired video content online.

5.3 Scenario 3

Scenario 3 consists of the Customer visiting an MVPD's web portal with the interest of streaming content provided by a separate Programmer. Authentication and authorization are performed directly by the MVPD. The authentication and authorization status are then passed to the Programmer, which uses the information to make content access decisions.

5.3.1 Actors and Roles

The Customer plays the role of the Subscriber, the Programmer plays the role of a Content Provider, and the MVPD plays the role of the Service, Authentication and Authorization Providers.

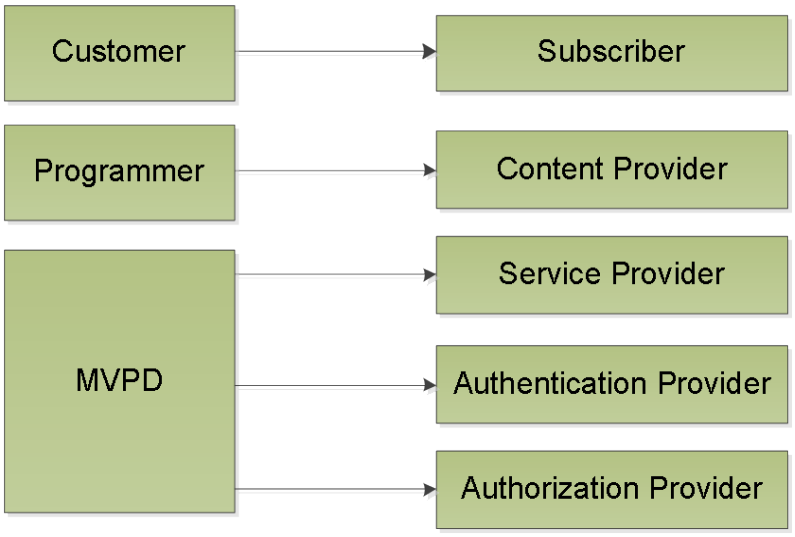

Figure 11 - Actors and Roles for Scenario 3

5.3.2 Architecture

In scenario 3, the MVPD and Programmer actors are separate business entities. As indicated in the previous section, each of them play different roles to enable the MVPD Customer access to their subscribed video content at the MVPD's web portal provided by the Programmer. Figure 12 provides an architectural diagram that shows where these functional roles are located with respect to the actors. It also indicates the main messaging interfaces.

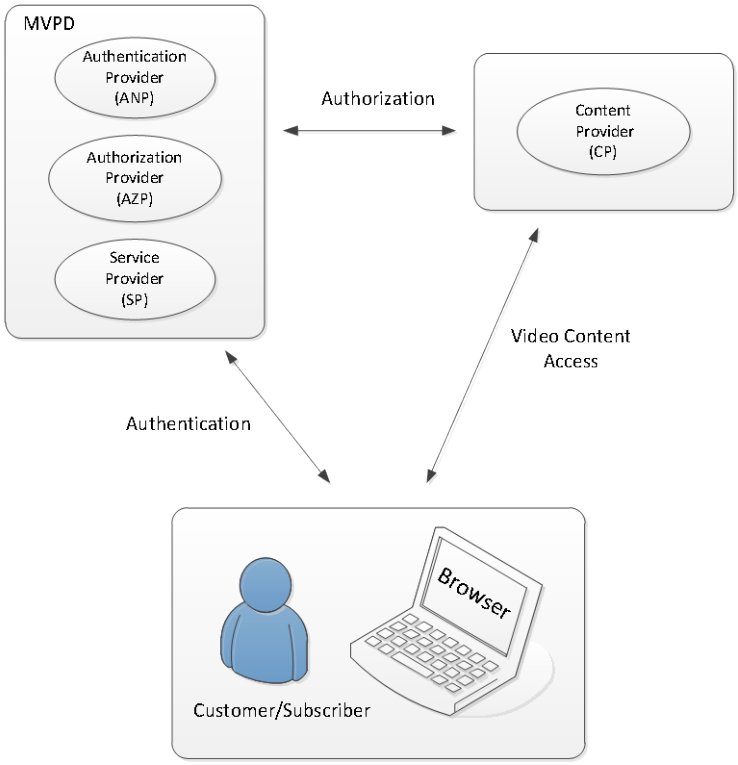

Figure 12 - Scenario 3: System Architecture

Using their web browser, the Customer visits the MVPD web portal to access video content. Normally, the portal web page will have a login button. The SP role controls access to video content and therefore, when the Customer clicks the login button it will want to authenticate the Customer. At this point, the SP will prompt the Customer for the Subscriber credentials (e.g., username password). The MVPD's AnP role authenticates the customer and passes the credentials back to the SP.

Once the SP verifies Customer authentication it displays available video content for selection. When the Customer selects video content, the SP sends an authorization to the MVPD's AzP, which determines the authorization status based on the Customer's subscription level. If the Customer has been authorized to view the content, the SP passes the authorization credentials to the Programmer, acting as the Content Provider, which then streams the selected video content to the Customer.

5.3.3 Pre-conditions

The following pre-conditions exist for all use cases described below.

1. Customer has an active Video subscription with the MVPD.
2. Customer has access to a PC with web browser, video player, and a broadband internet.
3. Customer has credentials with an Authentication Provider for authentication.
4. Customer knows the name of his MVPD of interest.
5. There is an existing business relationship between the Programmer and the MVPD.

5.3.4 Use Cases

5.3.4.1 Use Case 1

Name	Customer Accesses MVPD Website With Initial Login Authorization Attributes Available
Summary	Customer requests access to Online Content Access service at an MVPD website. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume. Authorization attributes are used to filter content.
Actor(s)	Customer plays role of Subscriber Programmer plays role of Content Provider MVPD plays role of Service, Authentication and Authorization Provider
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to MVPD's web portal.
Terminates	Customer closes session at MVPD's web portal.
Main Steps	1. Customer navigates to MVPD's web portal, which is acting as the Service Provider. 2. Customer selects "Sign In" and is prompted for subscription credentials. 3. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 4. The AnP sends the results of the login to the SP along with authorization attributes. 5. SP verifies Customer authentication and displays filtered video content based on the Customer's subscription for selection. 6. Customer selects video content. 7. SP sends authorization status request for selected content to the MVPD's Authorization Provider. 8. AzP determines Customer authorization status for requested content and responds to SP. 9. SP verifies that Customer is authorized to access content and passes the authorization credentials to the Programmer, acting as the Content Provider. 10. CP allows Customer to access/consume selected video content.
Post-Conditions	Subscriber has access to desired video content online.

5.3.4.2 Use Case 2

Name	Customer Accesses MVPD Website Without Initial Login
Summary	Customer selects content from an MVPD's website prior to Online Content Access. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume.
Actor(s)	Customer plays role of Subscriber Programmer plays role of Content Provider MVPD plays role of Service, Authentication and Authorization Provider
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to MVPD's web portal.
Terminates	Customer closes session at MVPD's web portal.
Main Steps	1. Customer navigates to MVPD's web portal, which is acting as the Service Provider. 2. Customer selects video content. 3. MVPD acting as the AnP, prompts the Customer for subscription credentials. 4. The AnP sends the results of the login to the SP which verifies the authentication. 5. SP sends authorization status request for the selected content to the MVPD's Authorization Provider. 6. AzP determines Customer authorization status for requested content and responds to SP. 7. SP verifies that Customer is authorized to access content and sends the authorization credentials to the Programmer acting as the Content provider. 8. CP allows Customer to access/consume selected video content.
Post-Conditions	Subscriber has access to desired video content online.

5.3.4.3 Use Case 3

Name	Customer Accesses MVPD Website With Initial Login No Authorization Attributes Available
Summary	Customer requests access to Online Content Access service at an MVPD website. MVPD authenticates Customer and provides authorization status for video content the Customer wishes to consume. No authorization attributes are available for filtering.
Actor(s)	Customer plays role of Subscriber Programmer plays role of Content Provider MVPD plays role of Service, Authentication and Authorization Provider
Pre-Conditions in Addition to 5.1.3	None
Trigger	Customer navigates to MVPD's web portal.
Terminates	Customer closes session at MVPD's web portal.
Main Steps	1. Customer navigates to MVPD's web portal, which is acting as the Service Provider. 2. Customer selects "Sign In" and is prompted for subscription credentials. 3. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 4. The AnP sends the results of the login to the SP with no authorization attributes included. 5. SP verifies Customer authentication and displays unfiltered video content for selection. 6. Customer selects video content. 7. SP sends authorization status request for selected content to the MVPD's Authorization Provider. 8. AzP determines Customer authorization status for requested content and responds to SP. 9. SP verifies that Customer is authorized to access content and passes the authorization credentials to the Programmer, acting as the Content Provider. 10. CP allows Customer to access/consume selected video content.
Post-Conditions	Subscriber has access to desired video content online.

5.4 Scenario 4

Scenario 4 consists of the Customer visiting a Programmer's web portal from within home or from within MVPD's network. When content is requested for playing, the actual content will be pulled from MVPD's content network. This allows for the views to be counted under Nielsen ratings. Note that, from a Customer's perspective, this scenario is identical to scenario 1. Authentication and Authorization are still performed as indicated in scenario 1.

5.4.1 Actors and Roles

The Customer plays the role of the Subscriber, the Programmer plays the role of a Service Provider, and the MVPD plays the role of the Content, Authentication and Authorization Providers.

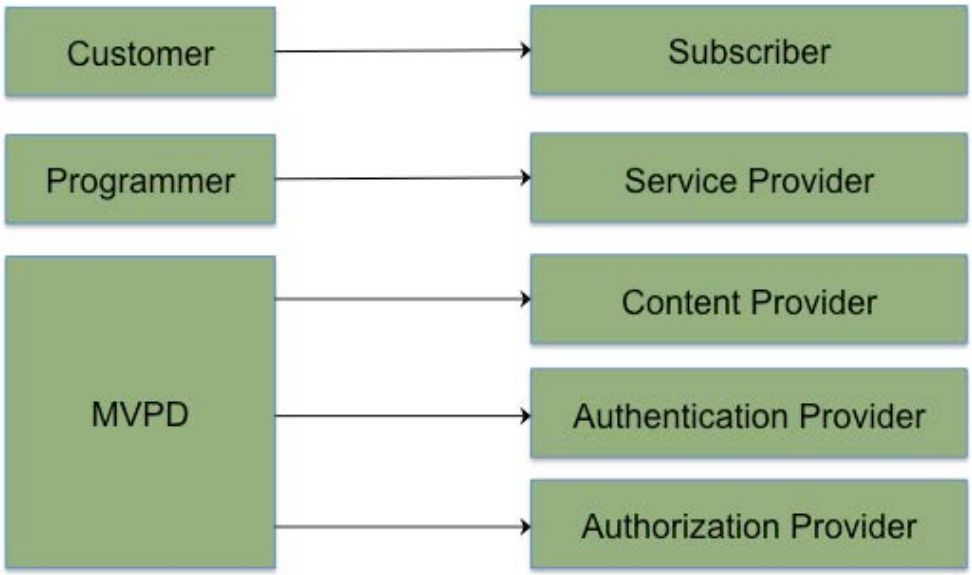

Figure 13 - Actors and Roles for Scenario 4

5.4.2 Architecture

In scenario 4, Customer visits the Programmer's portal, but the Programmer gets from the MVPD a HTML mark-up containing URLs to MVPD's network. This is achieved through the authorization response.

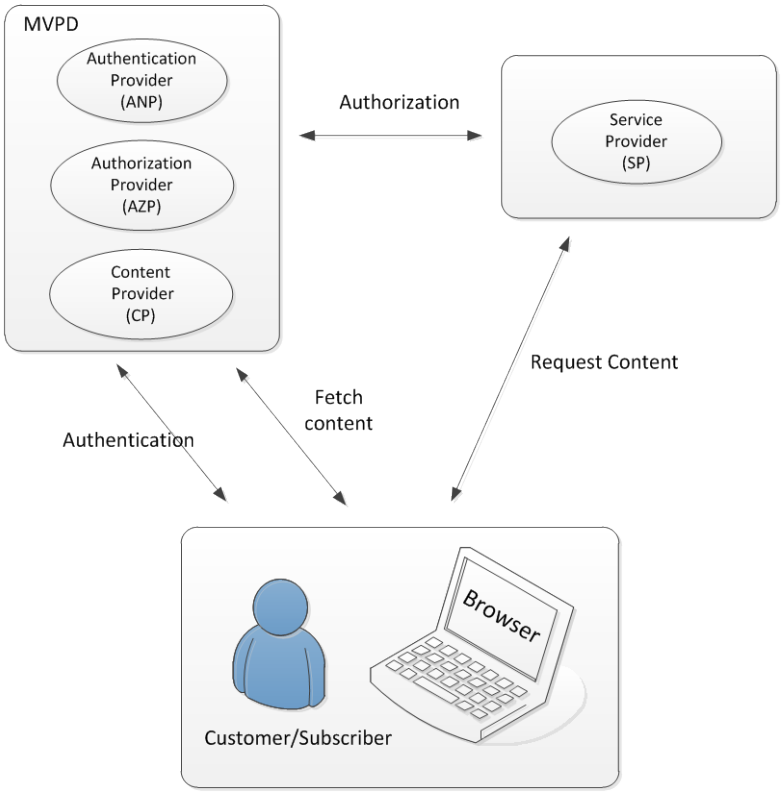

Figure 14 - Scenario 4: System Architecture

Using the web browser, Customer visit's the Programmer's web portal. Programmer performs authentication and authorization of the user as described in scenario 1.

In the authorization response, MVPD indicates that content should be fetched from MVPD's network, and provides the necessary HTML mark-up. Programmer uses that HTML mark-up instead of its own mark-up in the HTML page it sends back to the user. As a result, the user still gets the same experience as before (scenario 1), except that under the hood, the content is being pulled from MVPD's network.

5.4.3 Pre-conditions

The following pre-conditions must be met for all use cases described below.

1. All pre-conditions of scenario 1
2. Customer is on MVPD's network

5.4.4 Use Cases

5.4.4.1 Use Case 1

Name	Customer Accesses Programmer Website From Within MVPD's network
Summary	Customer is connected to Internet through MVPD's network. Customer requests access to Online Content Access service at a Programmer website. MVPD authenticates Customer and includes a flag in the authentication response indicating that the consumer is 'onNet'. If the flag is present, SP uses a back-channel query to determine if the MVPD has content that the Customer wishes to see. If so, SP goes by the response it got for that query. If not, or if the onNet flag was not sent in the authentication response, SP streams content from its own servers, as usual.

Name	Customer Accesses Programmer Website From Within MVPD's network
Actor(s)	Customer plays role of Subscriber Programmer plays role of Service Provider MVPD plays role of Authentication Provider, Authorization Provider and Content Provider
Pre-Conditions in Addition to 5.1.3	Customer is on MVPD's network (connected to internet through MVPD's network).
Trigger	Customer requests content while on MVPD's network.
Terminates	Playing of content is complete.
Main Steps	1. Customer is connected to Internet through MVPD's network. 2. Customer navigates to Programmer's web portal, which is acting as the Service Provider. 3. Customer selects "Sign In" and is presented with a list of supported MVPDs. 4. Customer selects the MVPD with which they have a video subscription. 5. The Programmer, as the SP, redirects the Customer's browser to the MVPD (acting as the Authentication Provider) for authentication. 6. Customer, acting as a Subscriber of the MVPD, uses his credentials to login to the AnP. 7. AnP validates customer's credentials. AnP will also notes that this access is from within MVPD's network. 8. The AnP sends the results of the login to the SP along with authorization attributes and redirects the Customer's browser back to the SP. It includes an 'onNet' flag in the authentication response. 9. SP verifies Customer authentication and displays filtered video content based on the Customer's authorization attributes. 10. Customer selects video content. 11. SP sends authorization status request to MVPD, now acting as the Authorization Provider, for selected content. 12. AzP determines Customer authorization status for requested content. 13. SP verifies that Customer is authorized to access content. 14. Since the onNet flag was sent during authentication, SP sends a content streaming query to the MVPD. Based on the response, SP either streams the content from its own servers, or from MVPD's servers.
Post-Conditions	Video content is shown from MVPD's network.

6 AUTHENTICATION REQUIREMENTS

6.1 Overview

OLCA defines different scenarios for allowing Customers to access their video subscription content online. In each scenario the Customer must be authenticated before the service provider (SP) allows them to access content. This section specifies the necessary requirements to support Customer authentication.

6.2 Authentication Architecture

In each OLCA scenario Actors can take on different functional Roles. The Roles that this section is primarily concerned with are the Authentication Provider (AnP), Service Provider (SP), and Subscriber. In general, authentication messaging and functional requirements will be the same no matter what Role an Actor takes on. Therefore, the authentication architecture is based on these functional Roles.

Authentication messaging between the SP and AnP is supported using the 2.0 version of the Secure Association Markup Language (SAML 2.0). SAML 2.0 defines a number of different application profiles. The profile used to support OLCA Subscriber authentication will be the Web Single Sign-On (SSO) profile. SAML defines three functional entities that map to OLCA Roles. The Identity Provider (IdP) maps to the AnP Role, the Service Provider (SP) maps to the SP Role, and the Subject maps to the Subscriber Role. The implementation requirements in this section are written to apply to these SAML functional entities.

Figure 15 shows the main message flow between the Subject, SP, and IdP.

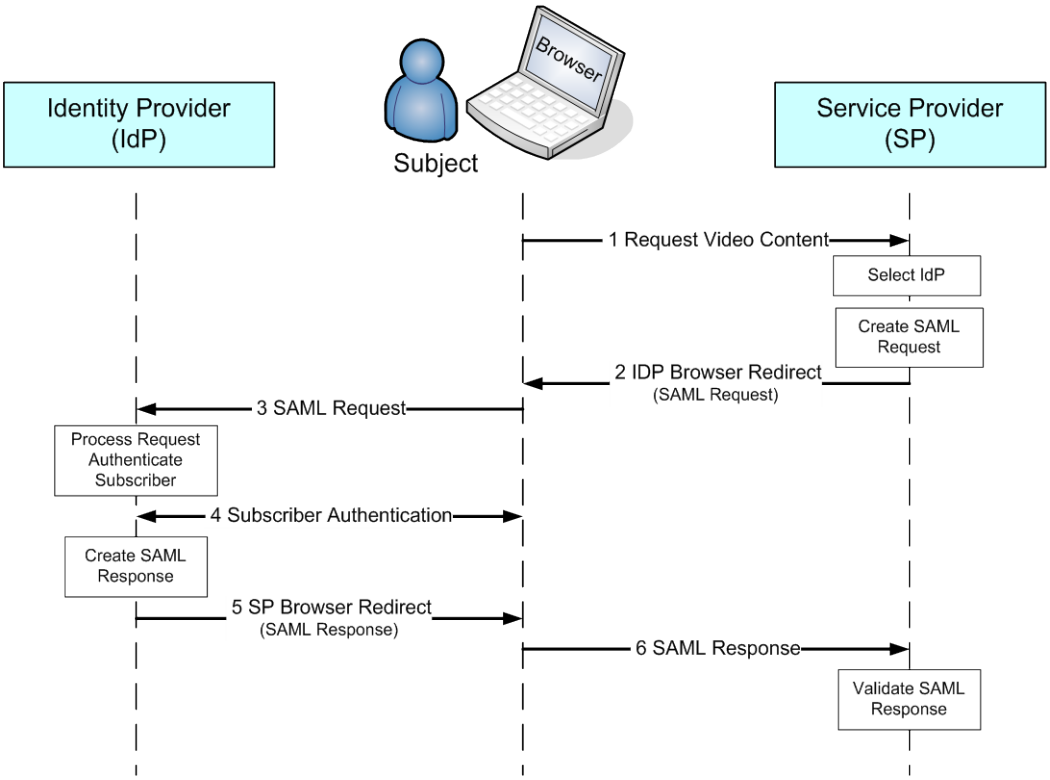

Figure 15 - SAML 2.0 Authentication Flow

When the Subject visits the SP website to access video content, the SP will need to authenticate them using their IdP before allowing them to view content. In order to do this, the SP will need to know which IdP to use for Subscriber authentication. The SP can discover a Subject's IdP by some automated method (see Section 6.12) or it can simply

ask the Subject to select his IdP from an authorized list. The SP can also use some kind of automated IdP discovery method.

Once the SP knows what IdP to use for Subject authentication, a SAML 2.0 Authentication Request is created, using information found in the IdP and SP metadata files (see Section 6.9). The SP MUST create a SAML 2.0 Authentication Request as defined in Section 6.5 and respond with a browser redirect to the selected IdP for Subject Authentication.

The Subject's browser receives the redirect and connects to the IdP site. The IdP extracts the Authentication Request and validates it. The IdP MUST process the Authentication Request as defined in Section 6.5. Once the Authentication request has been processed and validated, the IdP authenticates the Subject. The IdP MUST authenticate the Subject as defined by Section 6.7.

After successful Subject Authentication, the IdP creates an Authentication Response that contains assertion and attribute information about the Subject. The IdP MUST create an Authentication Response as defined in Section 6.8 and respond with a browser redirect containing the Authentication Response to the SP.

The Subject's browser receives the redirect and connects to the SP site. The SP extracts the Authentication Response and validates it. The SP MUST process the Authentication Response as defined in Section 6.8. Once the Authentication request has been successfully processed, the SP can allow the Subject to access video content according to authorization data defined in Section 7.

A number of errors can occur when processing requests and response messages. The SP MUST respond to Authentication Response processing error conditions as defined in Section 6.8. The IdP MUST respond to Authentication Request, Subject Authentication, and Authentication Response processing error conditions as defined in Sections 6.5, 6.7, and 6.8.

For OLCA scenario #2 the IdP and SP are both located at the MVPD. In this case the SP can use SAML messaging as defined above for Subject authentication or some other method supported by the MVPD. Also, the SP should know the IdP to use for Subject authentication and should not need to prompt the Subject for it.

6.3 Assumptions

The following assumptions apply for OLCA authentication:

- The Subject and the IdP have established a relationship and possess some form of mutually understood authentication credential.
- It is assumed that the Subject is using a standard commercial browser and can authenticate to the IdP by some means outside the scope of SAML.
- There is an existing bilateral agreement between the IdP and the SP, which allows a Subject of IdP to access online content through the SP's portal.
- Subject knows the SP URL to access OLCA services.
- SP is providing the video content.
- Automatic discovery of IdP is not required.
- IdP is not required to disclose the identity of the Subject to the SP.

6.4 System Security requirements

As SAML messages are sent between the SP, browser, and IdP, there are security risks that need to be addressed. Appendix VI discusses the various threats that exist for OLCA's application of SAML messaging. System security requirements are necessary to help prevent these threats.

To protect against spoofing and tampering threats when messages are sent across untrusted networks, the SP and IdP MUST support digital signatures for Authentication Request and Authentication Response messages as defined in Sections 6.5 and 6.8. The IdP MUST also use some kind of method to securely authenticate the Subject. This is typically done with a username and password.

IdP SHOULD use a secure channel that provides server authentication and traffic encryption (such as HTTPS) when authenticating the Subscriber.

The SP MUST verify that the received Authentication Response message has not been replayed by using the methods defined in Section 6.8.2.

To help reduce impacts of DoS attacks, the IdP SHOULD implement some kind of DoS detection and response mechanism.

6.5 Subject Identifier

When an IdP issues an assertion it includes an opaque value that is used to identify the Subject. This value can be transient or persistent. Transient values change across user authentication sessions. Persistent values do not change across user authentication sessions.

It is also possible to indicate SP scope for the Subject identifier. If the identifier is scoped for a single SP, then it will change across different SPs. If the identifier is scoped for a group of SPs, then it will not change across different SPs within that group. The transient and persistent meanings still apply within a given scope.

The SAML 2.0 <NameID> element is used to identify the Subject in an assertion. The <NameID> element has transient and persistent format identifiers. SP Scope is indicated by the SPNameQualifier attribute of the <NameID>. The format of the SPNameQualifier attribute string value is a comma separated list of SP identifiers. These identifiers can be the SP Entity Identifier that is used in the <saml:Issuer> element of authentication requests and/or a custom identifier agreed upon between the IdP and SP. The custom identifier may be used to represent a group of SPs.

The SP may request Subject identifier SP scope when sending an authentication request to the IdP. It does this by including a <NameIDPolicy> element that contains an SPNameQualifier attribute in the authentication request message.

The following SPNameQualifier attribute examples demonstrate the value format for a single SP and a group of SPs:

- Single SP
  - SPNameQualifier="SP_entity_identifier"
- Group of SPs
  - SPNameQualifier="SP_entity_identifier_1,SP_entity_identifier_2,SP_entity_identifier_3"
- Group of SPs with custom identifier
  - SPNameQualifier="custom_identifier"
- Combination of single SP and a custom group
  - SPNameQualifier="SP_entity_identifier,custom_identifier"

The SP MUST use the value format described above when including an SPNameQualifier attribute in the authentication request message. The IdP MUST use the value format described above for the SPNameQualifier attribute in the assertion response message.

6.6 Authentication Request

The SAML 2.0 Authentication Request is used by the SP to request Subject authentication at the IdP.

6.6.1 Request Creation at the SP

When an SP creates an Authentication Request, it MUST use an <AuthnRequest> element as defined in the Web Browser SSO profile of the [SAML 2.0 PROFILES] specification. The SP MUST support signing the <AuthnRequest>. The SP MUST use a digital certificate as defined in Section 8 for signing SAML messages. See <AuthnRequest> XML message example in Appendix I.1.

The SP MAY include a <NameIDPolicy> element containing an SPNameQualifier attribute in the <AuthnRequest>.

The SP MUST support both HTTP Redirect and HTTP POST binding as described in the [SAML 2.0 BINDINGS] specification for sending Authentication Request messages to the IdP. Other bindings may also be supported, but [SAML 2.0 CONFORMANCE] requires those two bindings to be supported at a minimum.

The SP SHOULD manage a response time-out function for each Authentication Request that it sends and clear any state data associated with that request if the IdP does not respond within a configured amount of time.

The SP MAY include a ForceAuthn attribute on the <AuthnRequest> to indicate that the IdP MUST revalidate the end user's identity instead of using a cached value. See Section 6.7.4 for more information on Forced Authentication.

The SP MAY include an isPassive attribute on the <AuthnRequest> to indicate whether prompting the end user is forbidden. See Section 6.7.5 for more information on Passive Authentication.

6.6.2 Request Reception at the IdP

The IdP MUST support receiving and processing Authentication Requests from the SP as defined in the Web Browser SSO profile section of the [SAML 2.0 PROFILES] specification, and both the HTTP Redirect and HTTP POST Binding sections of the [SAML 2.0 BINDINGS] specification.

If an Authentication Request is signed, the IdP MUST validate the request as defined in Section 8.

If the Authentication Request contains a <NameIDPolicy> element with an SPNameQualifier attribute, the IdP MUST use the same SPNameQualifier value in the assertion response unless it is not supported. If the IdP does not support the requested SPNameQualifier value, it MUST respond with a <Response> message containing an appropriate error status code or codes.

6.7 Subject Authentication

The IdP is responsible for using a secure method to authenticate the Subject. The method used to authenticate the Subject is not mandated in this specification although some authentication methods are suggested below. The IdP can indicate to the SP how the Subject was authenticated using attributes in the SAML assertion. The IdP MAY maintain authentication state for a Subject using a session token and respond to an Authentication Request without prompting the Subject for authentication credentials (see Section 6.8.1).

6.7.1 Authentication Methods

There are many different methods an IdP can use to authenticate a Subject. Some of the methods an IdP may use are described in the following sections.

6.7.1.1 Username/Password

Username/password credentials are a common method of authenticating Subjects. They are setup when a Subject creates a new account. When Subjects access online services they are authenticated by submitting their username/password.

Each member of a subscriber's household can be identified with username/password credentials. This enables services such as pay-per-view and parental controls.

Unfortunately, username/password credentials are vulnerable to a number of attacks where the password can be exposed or shared in an unauthorized manner. MVPDs using username/password credentials should enforce a strong password policy that is not a major inconvenience to subscribers.

6.7.1.2 Multifactor Authentication

The Subject's authentication assurance level is usually what determines the kind of service he is allowed to access. For example, if the assurance level is considered weak because a simple password is used then low value services may only be offered to that particular Subject.

Increasing the authentication assurance level can be accomplished by using multiple forms of authentication or Multifactor Authentication. This involves having more than one way to authenticate the Subject. For example, when the IdP authenticates the subject it could validate username/password credentials and associate the Subject with an authenticated device such as a modem located in the Subject's home. This form of Multifactor Authentication (two factors) provides a higher assurance level and can enable the Subject to access higher value services.

More authentication factors can be added to further increase the assurance level which can enable access to even higher value content. Additional authentication factors may include location association (how many times a Subject has authenticated at a particular location) or biometrics.

6.7.2 Authentication Context

When an IdP creates an assertion for a Subject it can include how the Subject was authenticated in the assertion, which can help the SP determine the assurance level and what type of services to offer. Stronger Subject authentication, such as multifactor authentication, provides a higher assurance level that the Subject is who they say they are and can result in higher value services being provided.

The <AuthnContext> element in a SAML assertion provides authentication context and can be used to indicate how an IdP authenticated a Subject. An SP can also request authentication context by including the <RequestedAuthnContext> element in an <AuthnRequest> message.

The <RequestedAuthnContext> section of an <AuthnRequest> requires special handling by the IdP. If present, the IdP MUST include an <AuthnContext> in any <Assertion> that is returned. The IdP MUST only use methods that correspond to the requested contexts provided by the SP in the <AuthnRequest>. Likewise it should be remembered that the <RequestedAuthnContext> section is a list and may contain more than one requested context. In such situations, the IdP MUST attempt authentication methods in the order the contexts are defined. For example, an IdP may attempt an authentication method that corresponds to the second requested context only after exhausting all supported methods that correspond to the first requested context without success.

If a the request cannot be completed by the IdP using the contexts that have been requested, then the IdP MUST return a <Response> message with a top-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:Responder*". Additionally, it is recommended that a second-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:NoAuthnContext*" be returned so that the SP understands that the contexts that were requested cannot succeed. The SP may choose to try again using different requested contexts.

6.7.3 Cancel Login

When a Subscriber is at the Authentication Provider login page to authenticate, the Subscriber can decide to cancel the authentication transaction. Canceling the authentication transaction can happen for any reason, and the Authentication Provider MUST send a <Response> as defined in Section 6.8.1 with a top-level status code of "*urn:oasis:names:tc:SAML:2.0:status:Responder*" with a second-level status code of "*urn:cablelabs:olca:1.0:status:AuthnCanceled*" to the Service Provider for notification.

The IdP SHOULD provide a link on the login page to return to the SP, if the person attempting to login is not a Subscriber with that IdP, using a <Response> top-level status code of "*urn:oasis:names:tc:SAML:2.0:status:Responder*" with a second-level status code of "*urn:cablelabs:olca:1.0:status:WrongIdP*".

6.7.4 Forced Authentication

An IdP may cache authentication results in a local session and use them to respond to authentication requests without needing to re-validate the user through credential entry or other means. Such cached results can be problematic if an SP requires a high level of assurance of user identity. If the SP wants to force revalidation, it MUST include a ForceAuthn attribute on the <AuthnRequest> with a value of "true". This will prevent the IdP from using cached results and forces a revalidation of the end user. See Appendix V for best practices on handling user assurance.

6.7.5 Passive Authentication

If the SP wants to avoid user interaction completely, it MUST include an isPassive attribute on the <AuthnRequest> with a value of "true". This will prevent any and all prompting of the end-user for the request according to the handling rules specified in [SAML 2.0 CORE]. If a the request cannot be completed without prompting the end user, then the IdP MUST return a <Response> message with a top-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:Responder*". Additionally, it is strongly recommended that a second-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:NoPassive*" be returned so that the SP understands that prompting will be necessary to complete authentication and can subsequently handle the event accordingly.

Please note that the isPassive attribute is not intended to be used for an SP to indicate a preference for the MVPD to use "friction-free" authentication methods. "Friction-free" authentication methods are typically defined as those that have little or no prompting and result in the least user abandonment. It is the responsibility of the MVPD (and generally best practice) to ALWAYS order the eligible authentication methods for every request in a way that prioritizes "friction-free" authentication methods while falling back to methods with a greater barrier to entry, such as credential prompting. In contrast, the isPassive attribute is only to be used in situations where the flow being executed by the SP cannot support user prompting, such as an attempt to fetch a fresh SAML assertion in the background.

6.7.6 Specific User Authentication

SPs MAY indicate the subject to be authenticated (for example, based on a previous authentication or session that has recently expired). IdPs MAY honor this request for a specific subject to be authenticated (for example, the IdP may pre-populate the username field).

SPs are to use the <Subject> element inside <AuthnRequest> to specify this value, using the <NameID> element. The value of this element MUST exactly match the <NameID> element that SP received in an earlier authentication response from this IdP.

Please note that although a specific user may have been requested by the SP, there is no guarantee that the <Response> returned by the IdP is for the same <Subject> as was requested. It is the responsibility of the SP to always check the <Subject> that is returned in the <Response> and handle the situation appropriately if the <Subject> differs. For example, if an SP only supports single-user sessions, the SP is advised to match the <Subject> received against the <Subject> of the current SP session and, should they differ, terminate the current SP session and start a new SP session for the newly-received <Subject>.

6.7.7 Specifying Device Type

SPs have the option to hint what device type the end user is attempting to authenticate to the IdP. This can be used to implement a different set of business rules, such as extended (or shortened) session TTLs, or be utilized in scenarios where traditional IdP user agent detection is not possible. For example, SPs may have applications running inside devices that need to authenticate using a separate device. During such authentication, the authenticating device will need to indicate to the IdP the device type that the requesting application is running on.

To communicate the deviceType, the SP MAY use the <samlp:Extensions> element inside of the <AuthnRequest> element. The namespace and element schema is given below. Only a single <olca:deviceType> value may be present.

```
<samlp:Extensions xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol">
  <olca:deviceType xmlns:olca="urn:cablelabs:olca">VALUE</olca:deviceType>
</samlp:Extensions>
```

The specific values to be used for deviceType are defined in this specification as follows.

- **urn:cablelabs:olca:1.1:device-type:automotive** – devices integrated into an car or other type of automobile
- **urn:cablelabs:olca:1.1:device-type:e-reader** – a hand-held device whose primary purpose is reading e-books
- **urn:cablelabs:olca:1.1:device-type:game-console** – a non-hand-held device used primarily for playing video games.

- **urn:cablelabs:olca:1.1:device-type:glasses** – a device worn over the eyes
- **urn:cablelabs:olca:1.1:device-type:home-appliance** – a smart, household appliance, such as a refrigerator, washer, dryer, microwave, etc.
- **urn:cablelabs:olca:1.1:device-type:media-player** – a hand-held device used primarily for playback of audio and/or video
- **urn:cablelabs:olca:1.1:device-type:mobile-phone** – a hand-held device used primarily for telephone communication, such as a cellular phone
- **urn:cablelabs:olca:1.1:device-type:pc** – A standalone personal computer, such as a desktop or laptop
- **urn:cablelabs:olca:1.1:device-type:set-top-box** – A console attached to a television monitor used primarily for enabling linear video streams from an MVPD
- **urn:cablelabs:olca:1.1:device-type:tablet** – A touch-screen enabled hand-held device measuring 7 inches or larger diagonally
- **urn:cablelabs:olca:1.1:device-type:tv** – A "smart" television or monitor
- **urn:cablelabs:olca:1.1:device-type:wearable** – A device worn on the user's person, such as a wristwatch

Additional values may ONLY be added through subsequent versions of the OLCA specification. If a more fine-grained device detection is needed, it is suggested that the implementers establish additional, proprietary fields inside the <samlp:Extensions> section.

When the deviceType is present, it must be remembered that the element is used for hinting purposes only. The IdP is not required to utilize the deviceType in any way and may choose to do its own device detection logic. For security reasons, the SP SHOULD sign any <AuthnRequest> that is sent which contains a deviceType so that the IdP can be reasonably sure that such a value was not tampered with while in transit.

6.7.8 Authentication Restrictions

Often times there are certain requirements that an SP will have with regard to the identity that is returned by an IdP for a given authentication attempt. These restrictions can range from the user being in the home to the user being authorized for a specific level of parental controls. This specification defines a mechanism by which these authentication restrictions can be communicated by the SP to the IdP within the <Extensions> section of the <AuthnRequest>.

This specification defines, but does not limit to, the following restrictions:

- **urn:cablelabs:olca:1.1:AuthnRestriction:onNet** – A Boolean value indicating whether the user must be on the MVPD network. A true value indicates the user must be on the network, whereas, a false value indicates the user must not be on the network. When specified as a restriction, a "*urn:cablelabs:olca:1.0:attribute:authz:onNet*" attribute MUST be returned in any successful result.
- **urn:cablelabs:olca:1.1:AuthnRestriction:atHome** – A Boolean value indicating whether the user must be using a device that is physically located in the home (at the service address). A true value indicates the user must be in the home, whereas, a false value indicates the user must be outside the home. When specified as a restriction, a "*urn:cablelabs:olca:1.1:attribute:authz:atHome*" attribute MUST be returned in any successful result.
- **urn:cablelabs:olca:1.1:AuthnRestriction:profileType** – A list which indicates the types of identity profiles that are required for authentication. The current values defined by the specification are, but not limited to: "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:device*", "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group*," and "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:user*". When specified as a restriction, a "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType*" attribute MUST be returned in any successful result.

- **urn:cablelabs:olca:1.1:AuthnRestriction:minMPAA** – A minimum MPAA rating that must be satisfied by the authenticating profile. When specified as a restriction, a "*urn:cablelabs:olca:1.0:attribute:authz:maxMPAA*" attribute MUST be returned in any successful result.
- **urn:cablelabs:olca:1.1:AuthnRestriction:minVCHIP** – A minimum VCHIP rating that must be satisfied by the authenticating profile. When specified as a restriction, a "*urn:cablelabs:olca:1.0:attribute:authz:maxVCHIP*" attribute MUST be returned in any successful result.
- **urn:cablelabs:olca:1.1:AuthnRestriction:country** – A list of countries in which a customer must have service. The countries are a two-letter code following the ISO 3166-1 alpha-2 standard as defined by [ISO 3166]. When specified as a restriction, a "*urn:cablelabs:olca:1.1:attribute:subscriber:country*" attribute MUST be returned in any successful result.
- **urn:cablelabs:olca:1.1:AuthnRestriction:channelID** – A list of channel IDs to which a customer MUST be authorized to view. When specified as a restriction, a "*urn:cablelabs:olca:1.0:attribute:authz:channelID*" attribute MUST be returned in any successful result.
- **urn:cablelabs:olca:1.1:AuthnRestriction:isPrivilegedUser** – A Boolean value requiring that the authenticating profile be a privileged user. A privileged user is usually defined as a profile which has the capability of creating, modifying, or deleting other profiles within the household, as well as managing parental controls for these profiles. When specified as a restriction, a "*urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser*" attribute MUST be returned in any successful result.

Each restriction is defined by an <AuthnRestriction> element in the "*urn:cablelabs:olca*" namespace. The <AuthnRestriction> element MUST contain a "type" attribute with one of the restrictions values listed above or with the unique URI of a restriction defined outside of this specification. Only one <AuthnRestriction> element of a given type is allowed within an <AuthnRequest>.

Each <AuthnRestriction> element MUST contain one or more <RestrictionValue> elements. For restrictions where having multiple <RestrictionValue> elements make sense and are present, the <AuthnRestriction> element MUST contain a "match" attribute, with the value of either "any" or "all" that defines whether at least one <RestrictionValue> matches, or all <RestrictionValue> elements match, respectively. Each <RestrictionValue> SHOULD contain an appropriate xsi:type attribute similarly to how it is used for <AttributeValue> elements as defined in [SAML 2.0 CORE].

The <AuthnRestriction> element MAY also contain an optional "enforcement" attribute that contains either the value "required" or "requested". If not specified, "required" is the default value that is assumed. A value of "required" indicates that the restriction MUST be met and if it is not, the IdP is required to return an error condition to the SP. A value of "requested" indicates that the IdP SHOULD prioritize authentication methods to try to satisfy those restrictions, but is not required to meet those restrictions to return a successful SAML response.

Please note that the intent is for both the IdP and SP to negotiate a standard set of restrictions common to every authentication request as part of establishing their initial integration. These standard restrictions should not be specified in the <AuthnRequest> in order to reduce the size of each <AuthnRequest> message. For example, if a particular SP requires a **urn:cablelabs:olca:1.1:AuthnRestriction:channelID** and/or **urn:cablelabs:olca:1.1:AuthnRestriction:minMPAA** restriction that corresponds to its entire library of protected content, it should be established as a default restriction in the IdP's configuration for that SP and not need to be communicated in every <AuthnRequest>. Restrictions specified in the <AuthnRequest> should only be used when the request is in response to a specific use case requiring that restriction. This limits the size of each <AuthnRequest> by not needing to specify restrictions that are already applied by default.

An example set of restrictions:

```
<samlp:Extensions xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol">
    <olca:AuthnRestriction xmlns:olca="urn:cablelabs:olca"
      type="urn:cablelabs:olca:1.1:AuthnRestriction:atHome">
        <olca:RestrictionValue xsi:type="xsd:boolean">
          true
        </olca:RestrictionValue>
    </olca:AuthnRestriction>
```

```
    <olca:AuthnRestriction xmlns:olca="urn:cablelabs:olca"
      type="urn:cablelabs:olca:1.1:AuthnRestriction:country"
      match="any">
        <olca:RestrictionValue xsi:type="xsd:string">
          US
        </olca:RestrictionValue>
        <olca:RestrictionValue xsi:type="xsd:string">
          PR
        </olca:RestrictionValue>
    </olca:AuthnRestriction>
    <olca:AuthnRestriction xmlns:olca="urn:cablelabs:olca"
      type="urn:cablelabs:olca:1.1:AuthnRestriction:profileType"
      enforcement="requested">
        <olca:RestrictionValue xsi:type="xsd:string">
          urn:cablelabs:olca:1.1:attribute:subscriber:profileType:user
        </olca:RestrictionValue>
    </olca:AuthnRestriction>
</samlp:Extensions>
```

These restrictions require that the end user be in their home (at their service address), which must either be in the US or Puerto Rico. It's requested, but not required, for the IdP to return an identity which corresponds to an individual user instead of a group or household.

If the SP requests a restriction where enforcement is "required" and the restriction is either unknown to the IdP or is something the IdP does not support, the IdP MUST return a <Response> message with a top-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level <StatusCode> of "*urn:cablelabs:olca:1.1:status:UnsupportedAuthnRestriction*". Additionally a third-level status code is to be returned for each restriction that the IdP cannot enforce, where the status code value is the type name of the restriction.

If, however, all requested restrictions are capable of being enforced by the IdP, but are not satisfied at the time of the authentication request, the IdP MUST return a <Response> message with a top-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level <StatusCode> of "*urn:cablelabs:olca:1.1:status:UnsatisfiedAuthnRestriction*". Additionally, a third-level status code is to be returned for each restriction that is not satisfied, where the status code value is the type of the restriction. For example, if the SP required restrictions of both urn:cablelabs:olca:1.1:AuthnRestriction:atHome and urn:cablelabs:olca:1.1:AuthnRestriction:country, but neither can be satisfied, the following <Response> would be returned by the IdP:

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                ID="_f61dc2a5ed1d4fa59b8aa0cfeab1d24f3128ca377a"
                Version="2.0"
                IssueInstant="2015-05-08T15:10:13Z"
                Destination=" http://sphost.sp.com/spservice"
                InResponseTo="_544d1e06d1ee61084cd84c67d9be29d72b1bed7f27"
                >
    <saml:Issuer>http://idphost.idp.com</saml:Issuer>
    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        …
    </ds:Signature>
    <samlp:Status>
        <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Requester">
            <samlp:StatusCode
Value="urn:cablelabs:olca:1.1:status:UnsatisfiedAuthnRestriction">
                <samlp:StatusCode
Value="urn:cablelabs:olca:1.1:AuthnRestriction:atHome" />
                <samlp:StatusCode
Value="urn:cablelabs:olca:1.1:AuthnRestriction:country" />
            </samlp:StatusCode>
        </samlp:StatusCode>
        <samlp:StatusMessage>One or more requested restrictions could not be
satisfied</samlp:StatusMessage>
```

```
        </samlp:Status>
    </samlp:Response>
```

6.7.9 Requested Attributes

There are times when an SP would like to ensure that the IdP return certain attributes about the user in the response which are normally optional to be included. In those cases, the SP MAY include one or more <RequestedAttribute> elements in the <Extensions> section of the <AuthnRequest>. The <RequestedAttribute> element is of the "urn:cablelabs:olca" namespace. It contains a text section with the name of the requested attribute. For example:

```
<samlp:Extensions xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol">
    <olca:RequestedAttribute xmlns:olca="urn:cablelabs:olca">
        urn:cablelabs:olca:1.1:attribute:authz:onNet
    </olca:RequestedAttribute>
    <olca:RequestedAttribute xmlns:olca="urn:cablelabs:olca">
        urn:cablelabs:olca:1.1:attribute:subscriber:country
    </olca:RequestedAttribute>
    <olca:RequestedAttribute xmlns:olca="urn:cablelabs:olca">
        urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier
    </olca:RequestedAttribute>
</samlp:Extensions>
```

In this example, the SP is requesting to know whether the user is on the MVPD network, the country of the subscriber, and the group identifier of the account with which the user has been authenticated.

Please note that similar to the Authentication Restrictions in Section 6.7.8, the intent is for both the IdP and SP to negotiate a standard set of attributes common to every authentication request as part of establishing their initial integration. Requested attributes should only be specified in the <AuthnRequest> when the request is in response to a specific scenario that requires that attribute. This limits the size of each <AuthnRequest> by not needing to specify a request for attributes that are provided by default.

If a requested attribute is given, but the attribute is unknown to the IdP, the IdP MUST return a <Response> message with a top-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue*". Additionally, a third-level status code should be returned for each invalid attribute, where the status value is the name of the invalid attribute.

6.7.10 PIN authentication

There are specific times when an SP may wish to prompt a subscriber to enter a PIN (personal identification number). One such example is when parental controls restricts playback of content, the SP may prompt the end-user for a PIN to temporarily lift the ratings restrictions and allow playback. To allow for such a mechanism, the "*urn:cablelabs:olca:1.1:ac:classes:PIN*" authentication context class has been created. When specified as the only requested authentication context in the <AuthnRequest>, the IdP MUST restrict authentication to only prompt for a PIN.

When specifying the PIN context, the <AuthnRequest> MUST also contain a "specific user" authentication <Subject> element whose <NameID> element matches the currently authenticated profile. If the <NameID> is not included in the request, IdP MUST return a <Response> message with a top-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level <StatusCode> of "*urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal*". The <NameID> is required so that the IdP knows what subscription account to validate the PIN entries against and is crucial in cases where the IdP maintains multiple authentication sessions or does not maintain any session state.

It is highly recommended that if the PIN context is being used to temporarily override parental controls, that the *urn:cablelabs:olca:1.1:AuthnRestriction:minMPAA* and/or *urn:cablelabs:olca:1.1:AuthnRestriction:minVCHIP* authentication restrictions also be included in the <AuthnRequest> with values that match the parental controls that are being overridden. For example, if the content attempting to be watched carries an MPAA rating of "R", a *urn:cablelabs:olca:1.1:AuthnRestriction:minMPAA* restriction should be included with a value of "R". This way if the IdP happens to have multiple PIN values for an account with different associated parental controls, the IdP can restrict the valid PINs that satisfy the request to only those that have sufficient parental controls to meet the request.

6.8 Authentication Response

When an IdP receives an Authentication Request, it processes that request and sends an appropriate Authentication Response to the SP. If the Subject is authenticated, an assertion is included in the response.

6.8.1 IdP Processing and Response

When an IdP receives an Authentication Request, it MUST support processing and responding to it as defined in the Web Browser SSO profile section of the [SAML 2.0 PROFILES] specification, and the HTTP POST Binding section of the [SAML 2.0 BINDINGS] specification with the following exceptions:

- If the status code is "urn:oasis:names:tc:SAML:2.0:status:success", the <SubjectConfirmationData> element of the assertion MUST contain an Address attribute if configured to do so.
- If the status code is "urn:oasis:names:tc:SAML:2.0:status:success", the <Subject> element MUST contain a <NameID> element with an "urn:oasis:names:tc:SAML:2.0:nameid-format:transient", or an "urn:oasis:names:tc:SAML:2.0:nameid-format:persistent" Format attribute.
- Assertions MUST be signed if the IdP is configured to do so.
- If the status code is "urn:oasis:names:tc:SAML:2.0:status:success", the assertion MUST contain an SPNameQualifier attribute in the <NameID> element.

The IdP MAY support HTTP Artifact Binding as defined in the [SAML 2.0 BINDINGS] specification.

The IdP MUST support signing assertions as defined by [SAML 2.0 CORE]. The IdP MUST support adding an Address attribute to the <SubjectConfirmationData> element of the assertion.

The IdP MUST use a digital certificate as defined in Section 8 for signing SAML messages.

The IdP MAY include an <AuthnContext> inside the <AuthnStatement> with a single <AuthnContextClassRef> value as defined in Section 6.7.2. However, as defined by [SAML 2.0 CORE], this MUST be included if the originating <AuthnRequest> contains a <RequestedAuthnContext> section.

The IdP MUST include an <AuthnInstant> inside the <AuthnStatement> that contains the timestamp of the last time a user's identity was validated (such as through entering credentials, or through another mechanism such as Zero-Auth). If a pre-existing session at the IdP was used to authenticate the user without prompting, the <AuthnInstant> returned MUST reflect the last time the user's identity was verified.

The IdP MAY include <SessionNotOnOrAfter> element to indicate the maximum time the SP is allowed to permit the user to use the SP's service without being reauthenticated. If present, SPs MUST adhere to this maximum session time. If not present, SPs may determine the appropriate session length.

See <Response> XML message example in Appendix I.2.

If the IdP is sending an unsolicited response, then it SHOULD include RelayState parameter in the response. The value of this parameter SHOULD be resolvable to a specific (video) content at the SP. For more details, see the Authorization Requirements in Section 7.

6.8.2 SP Processing

When the SP receives an Authentication Response, it MUST process that request as defined in the Web Browser SSO profile section of the [SAML 2.0 PROFILES] specification. The SP MUST also perform the following checks when processing the Authentication Response:

- Verify that the service provider identifier found in the <AudienceRestriction> element matches the service provider's identifier in the SP metadata.
- If assertions are signed by the IdP, the SP MUST validate the signature as defined in Section 8.

The SP MAY support HTTP Artifact Binding as defined in the [SAML 2.0 BINDINGS] specification.

If the SP is unable to process and validate the Authentication Response according to the requirements above, it MUST NOT allow the Subject access to video content requiring IdP authorization.

The SP SHOULD NOT use the NotOnOrAfter attribute of the assertion as a TTL for sessions it establishes with subjects.

See Section 6.8.1 for use of the <SessionNotOnOrAfter> element for session times.

If the <AuthnStatement> contains an <AuthnContext>, then the SP MUST use the indicated value of the <AuthnContextClassRef> element to process the response.

The SP MUST be able to process unsolicited responses (IdP initiated SSO). For specific requirements on processing content identifiers (sent through RelayState parameter), see the Authorization Requirements Section 7.

If the SP requires a high level of assurance of user identity, it may use the <AuthnInstant> timestamp included in the <AuthnStatement> section to determine when the end-user's identity was last validated. This can be useful if authentication hasn't occurred in a while and the user is entering a sensitive or restricted area (such as an administrative section). If the <AuthnInstant> is too far in the past for the business rules, the SP MAY choose to issue another <AuthnRequest> to the IdP, but one that includes the optional ForceAuthn attribute with a value of true. This will force the IdP to re-validate an end-user's identity instead of re-using an identity that is cached in a pre-existing IdP session. See Appendix V for best practices on handling user assurance.

6.9 Establishing Trust Relationships

SAML requires agreements between system entities regarding identifiers, binding support and endpoints, certificates and keys, and so forth. Metadata is used for describing this configuration information in a standardized way. IdPs and SPs that are within the same circle of trust should exchange metadata files. Metadata exchange SHOULD be done over a secure channel to prevent compromise of security parameters.

The IdP and SP MUST support the SSO Identity Provider and SSO Service Provider Profiles as defined in the SAML-Metadata specification [SAML 2.0 Metadata].

6.10 Error Conditions

The IdP MUST respond to Authentication Request, Subject authentication, and Authentication Response error conditions as defined by the [SAML 2.0 CORE] specification, the HTTP POST Binding section of the [SAML 2.0 BINDINGS] specification, and the Web Browser SSO Profile section of the [SAML 2.0 PROFILES] specification, unless otherwise stated in this specification. An IdP MAY notify the Subscriber of the error condition and provide a link to continue to the SP with the appropriate <Response> error status code.

When the SP receives an error response, it SHOULD log the error message and notify the Subscriber with links to return to the original page or retry.

6.11 Session Management

HTTP cookies may be used to track Subject sessions. After a Subject authenticates at the IdP, using ZSO or with their credentials, the IdP can create an authentication session cookie. Each time an IdP receives an Authentication Request from a Subject's redirected browser it can check for an authentication session cookie. If one exists it can immediately respond with an assertion. If an authentication session cookie does not exist the IdP will need to authenticate the Subject via ZSO or using their credentials before responding with an assertion.

SPs can also use cookies for tracking Subject sessions. After an SP as validated an assertion response from an IdP it can create a session cookie that maintains a security context with the Subject. Using the session cookie the SP can allow the Subject to access different resources on their site without having to perform authentication each time.

Session cookies should not be permanent and have a validity time associated with them. MVPDs and Programmers should carefully set cookie validity time so it provides a good user experience for the Subject, but does not make it easy for hackers to steal service.

6.12 IdP Discovery

To improve the Subscriber's OLCA experience, SPs MAY use a method to discover the Subscriber's IdP without having to prompt them for it. One approach for IdP discovery is defined in [ID-IdP]. This technique uses web browser cookies associated with the IdP's domain, and an iframe and JavaScript solution to reveal the existence of those cookies to SPs.

6.13 Logging Out

6.13.1 Service Provider Log Out

How the Subject logs out from a particular SP's service is unspecified and left up to the SP. Typically, logout will automatically occur when the Subject closes their browser, or if the session times out based on a defined service timeout value.

6.13.2 Single Log Out

The IdP can share a single authentication context with multiple service providers. SAML 2.0 offers a single logout profile, which allows a Subject to easily terminate each session across multiple SPs at once. An SP can initiate a single logout request with the Subject's IdP, which will propagate the logout request to other SPs.

With Single Logout enabled, when a Subject logs out of the SP's site, the SP will send a digitally signed <LogoutRequest> message to the IdP. The IdP will then send <LogoutRequest> messages to all the SPs that the user is logged into. Each SP will respond to the IdP with a <LogoutResponse> message. After the IdP receives <LogoutResponse> messages from each SP, the IdP sends a final <LogoutResponse> message to the SP that initiated the Single Log Out process. The SP then terminates the Subject's session to complete the single logout process.

7 AUTHORIZATION REQUIREMENTS

7.1 Overview

Authorization is the process used to validate that a particular user is indeed a subscriber for a particular content, and that he/she is authorized to view that content online. AzPs make this authorization decision, and SPs will enforce this decision. Many factors (and environmental data) may be taken into consideration to arrive at this decision.

This section covers the architecture necessary to achieve this. A preliminary use case for the architecture, is the SP checking with the AzP for authorization to a particular content. As the Subscriber interacts with the SP's system, it becomes evident that the SP may need to filter content shown for selection. Parental controls, or channels subscribed to, are prime examples. Thus, the architecture also covers use cases where the AzP shares certain attributes about the user with the SP.

NOTE: Attribute information is provided by the AzPs solely for the sake of content selection filtering at the SP. Such sharing does not constitute a decision by the AzP that any or all content satisfying the attributes is permitted for a subscriber to view. SPs are required to get such a decision using back-channel authorization decisions (see Use Case 2 in Section 7.2.1.2).

The term 'back-channel' refers to a direct connection between two entities, not involving the subscriber's browser.

7.1.1 Assumptions (for OLCA Scenario 1)

- SP has successfully authenticated the subscriber with AnP before the authorization process begins.
- Authorization should not require AnP/AzP to share the subscriber identity with the SP. The SP and AzP should be able to use a unique opaque identifier (or other mechanism), shared as part of authentication, to correlate the authorization request to a subscriber.
- The SP and AzP have a pre-established and well-understood mechanism to identify the content (content IDs).

7.2 Authorization Use Cases

7.2.1 Use Cases

Authorization specific use cases are listed below.

7.2.1.1 Use Case 1

Name	Present subscription based content
Summary	SP wants to present to the Subscriber a list of (eligible) content to choose from to view online.
Pre-Conditions	Subscriber is authenticated.
Trigger	Subscriber visits the SP's portal and completes the authentication process.
Terminates	Customer closes session at SP's web portal.
Main Steps	1. Subscriber navigates to SP's web portal. 2. Subscriber completes the authentication process. 3. SP now needs to present (eligible) content to the Subscriber. 4. If no attributes are included in the Authentication response, SP will issue an attribute query to the AzP. 5. AzP responds with values to the requested attributes. 6. SP will use the attribute values to determine which content is eligible to the Subscriber and presents that content.
Post-Conditions	Subscriber is presented only with content that satisfies his/her subscription and other configuration attributes.

7.2.1.2 Use Case 2

Name	Authorize a particular content for viewing online
Summary	SP wants to know if the subscriber is authorized to view a particular content.
Pre-Conditions	Subscriber is authenticated, and presented content to view online.
Trigger	Subscriber selects a content to start watching online.
Terminates	Subscriber is either allowed or denied the content.
Main Steps	1. Subscriber navigates to SP's web portal. 2. Subscriber completes the authentication process. 3. SP presents eligible content to the Subscriber (earlier use case). 4. Subscriber selects a particular content to start watching online. 5. SP queries the AzP to verify if this Subscriber can watch this particular content. 6. AzP responds with a permit or deny decision.
Post-Conditions	Subscriber is either permitted to watch the content or denied access.

7.3 Authorization Architecture

From the use cases identified above, the following are the high-level interactions needed to perform and enforce authorization decisions.

1. Explicit decision request –SP asks AzP for an allow/deny decision on a specific content.
2. Attribute exchange – the AnP/AzP may provide SP with a set of attributes that qualifies the content for a given subscriber. This exchange could be 'explicit' – SP explicitly seeking some attributes from the AzP, or 'implicit' – AnP passing them to SP without an explicit request.

Based on these exchanges, a high-level architecture for authorization will be as follows.

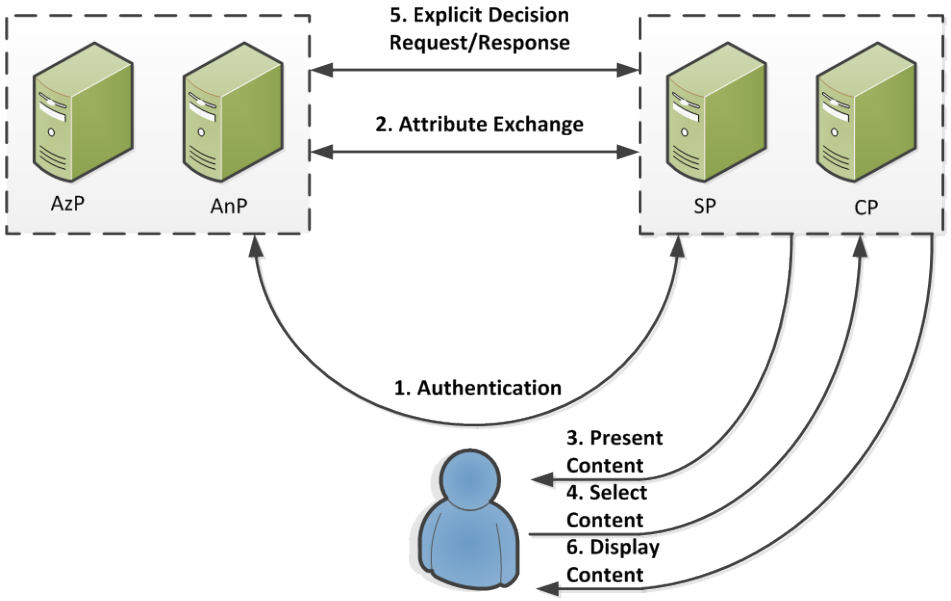

Figure 16 - Authorization Framework

7.3.1 Technology

SAML 2.0 and XACML 2.0 will be the two technologies used to communicate authorization messages between the AA/PDP and the SP.

SAML provides a good vehicle for exchanging attributes – for attribute requests and responses, as well as implicit attribute exchange using attribute statements in the authentication assertion.

XACML provides a clear contract for requesting fine-grained authorization requests and responding with decisions.

Thus, SAML will be used for all attribute exchanges, and XACML will be used for the explicit authorization decision request and response messages. However, while XACML clearly defines the message content necessary for authorization, it does not prescribe the additional constructs necessary to secure the communication. "SAML profile for XACML" bridges that gap (see section 1 of [SAML 2.0 Profile XACML 2.0] for more details). Thus, to be more specific, XACML over SAML is used for the explicit authorization request and response messages (see [SAML 2.0 Profile XACML 2.0] for more details).

7.3.1.1 Terminology

Given the technology choices mentioned above, SAML and XACML, given below is a mapping between SAML and XACML terms and the roles used in this document.

PDP – Policy Decision Point: an entity that accepts decision requests, evaluates policies and responds with a decision. The AzP role provides the PDP functionality.

AA – Attribute Authority: an entity that accepts attribute queries, and responds back with attribute values. The AzP or the AnP roles provide the AA functionality.

PEP – Policy Enforcement Point: an entity that ultimately guards the content, and allows or denies access to content by the subscriber, based on the decision received from the PDP. It will issue decision requests to the PDP, and enforce the received decision. The SP role provides the PEP functionality.

7.3.1.2 Message Data Types

Below are the specific data types used in each type of exchange (as identified earlier).

1. Explicit Attribute exchange – this will be achieved using SAML <AttributeQuery> and SAML <Response> containing an <Assertion> with an <AttributeStatement>.
2. Implicit attribute exchange – this will be communicated as a SAML <AttributeStatement> within a SAML <Assertion> that contains the <AuthenticationStatement>.
3. Explicit Decision request/response – this will be achieved using SAML/XACML <XACMLAuthzDecisionQuery> and SAML <Response> containing an <Assertion> with a <XACMLAuthzDecisionStatement>.

The explicit exchanges are made using SOAP over HTTP. The picture below gives the exchanges using the technology choices identified above.

The AzP is responsible for

- receiving decision requests from the SP,
- evaluating applicable policies, and
- delivering authorization decision to the SP.

The SP is responsible for performing content access control by enforcing the authorization decision received from the AzP.

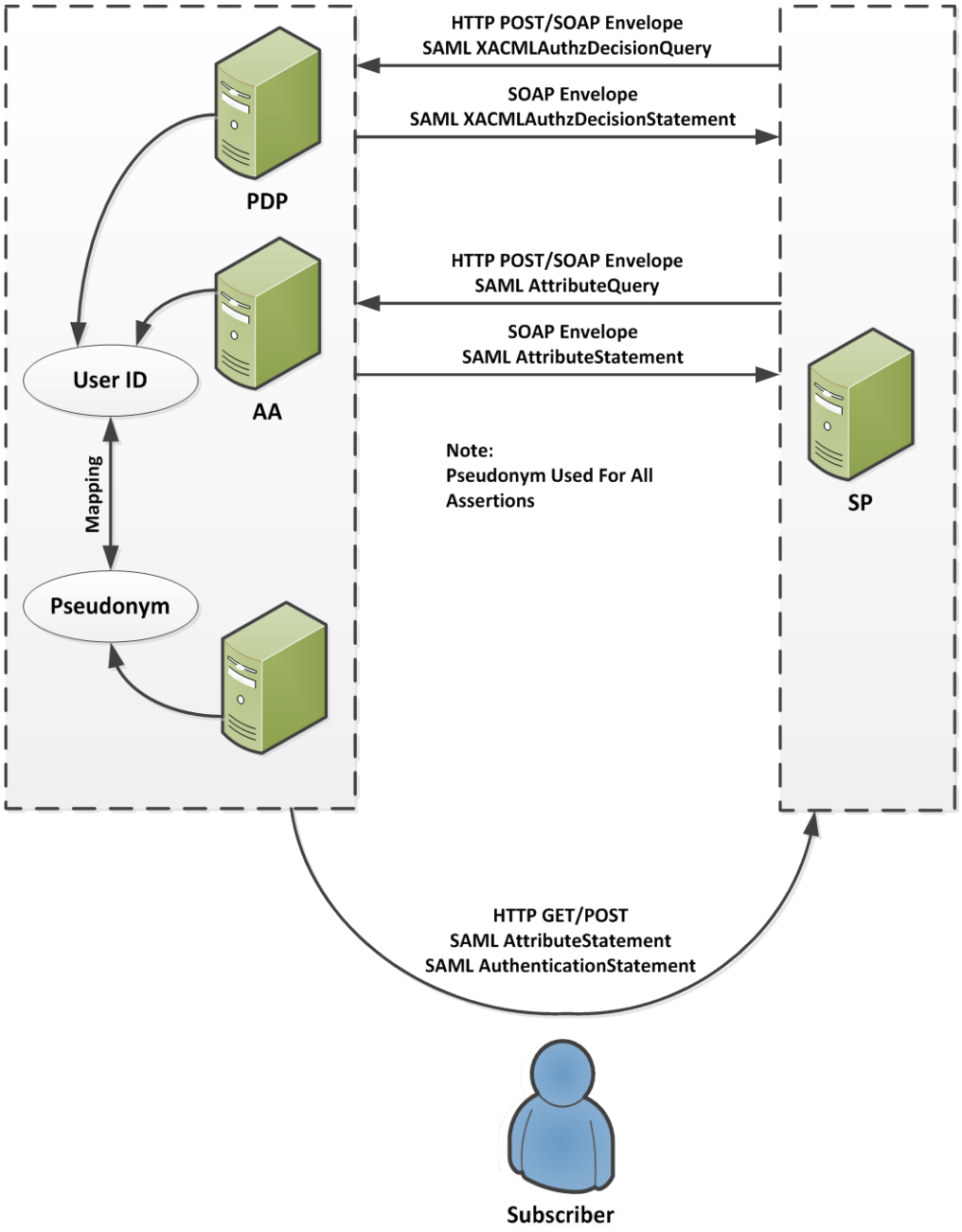

Figure 17 - Authorization Architecture

7.3.1.3 Sample Message Flows

Message flows for two sample scenarios are shown in Figure 18.

NOTE: This is intended for reference only; actual implementations can look different.

Example 1: Subscriber authenticated upfront

In this example, subscriber visits the SP's web site and is authenticated prior to access to any content.

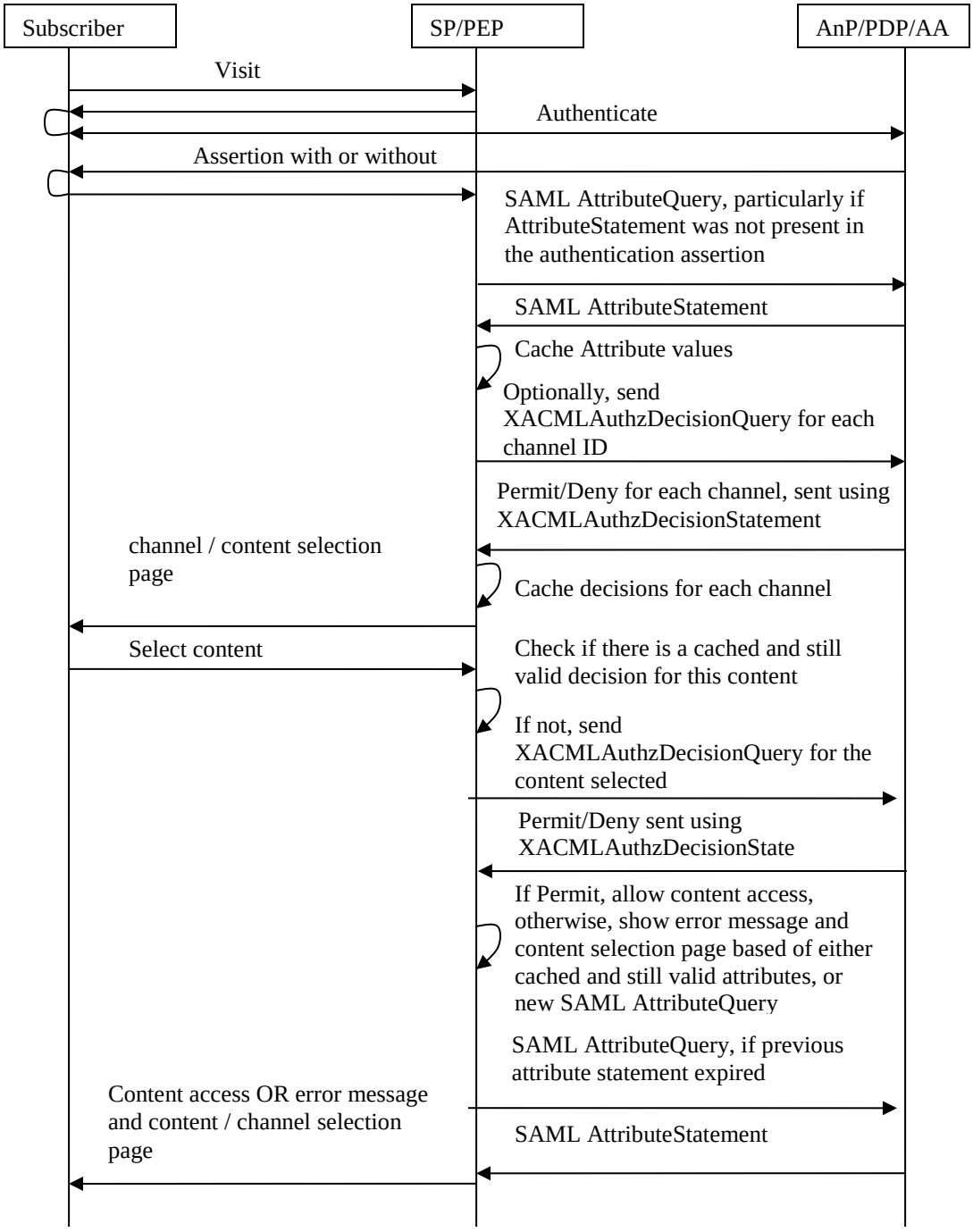

Figure 18 - Scenario 1: Subscriber Authenticated Up-Front

Example 2: Subscriber browses content before authentication

In this example, the subscriber browses through the SP's web site and selects a content to view. Authentication, followed by authorization, occurs at that time.

This scenario also applies to bookmarks – when users either bookmark content, or get a link to content from somewhere else – authentication, followed by authorization, happens at the time of actual access to the content.

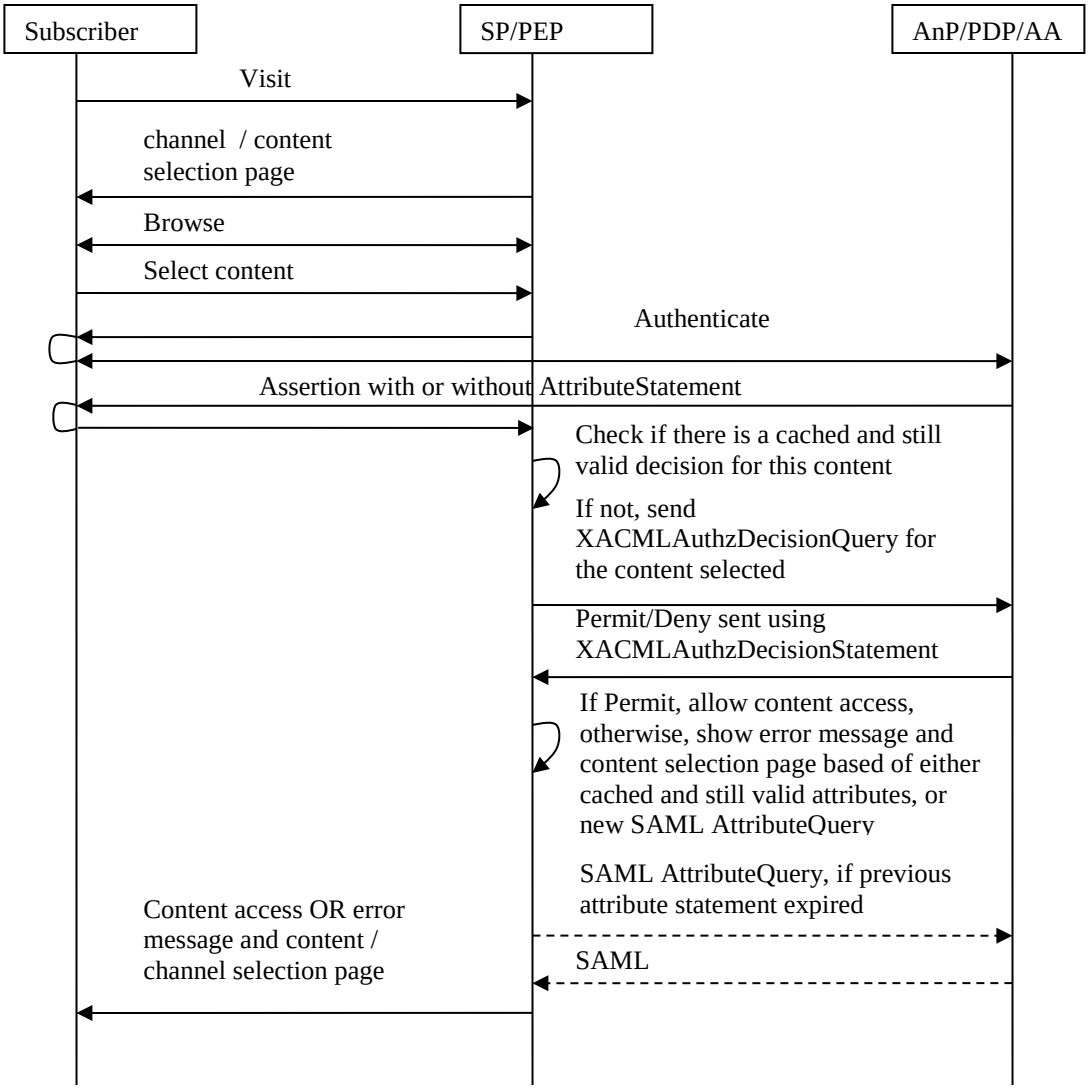

Figure 19 - Scenario 2: Subscriber Authenticated (and Authorized) at Time of Content Access

Requirements

1. AnPs MAY include AttributeStatement in the Assertion sent at the time of authentication.
2. SPs MUST support a PEP role that issues XACMLAuthzDecisionQuery (as described in this document) and processes XACMLAuthzDecisionStatement (as described in this document).
3. SPs MUST honor the decision delivered by the PDP.
4. SPs SHOULD support accepting AttributeStatement in the Assertion, sent at the time of authentication.
5. SPs MUST support SAML AttributeQuery.
6. SPs MUST use the attribute values to filter content in their UI, if provided by the AnP/AA.
7. SPs MUST base their access on explicit XACMLAuthzDecisionStatement for every individual content (not just based on attribute values used to filter content for UI).

NOTE: See Appendix II for sample messages.

7.4 Trust relationships

Each pair of entities communicating over the back channel needs to establish a trust relationships – this relationship will identify the service URLs as well as identities of each entity. These identities will be used for message and / or connection authentication (see Section 7.5.6).

A typical way to establish trust relationships is using the metadata schema provided by the SAML standard. This specification recommends SAML metadata based trust relationships; however, individual implementation may choose other approaches. Also, this specification does not cover exactly how such metadata is exchanged – it is left to individual implementations.

7.4.1 Service URLs from metadata

If using SAML metadata, the following requirements apply.

For attribute queries, data provided under the 'AttributeAuthorityDescriptor' MUST be used. Specifically, 'AttributeService' element under the 'AttributeAuthorityDescriptor' element MUST be used.

For authorization queries, AuthzService element under PDPDescriptor element of AzP's metadata MUST be used.

NOTE: PDPDescriptor is originally intended for SAML AuthzDecisionQuery. XACML has not published (but is currently working on) a metadata schema that will create an XACMLPDPDescriptorType with service URLs specifically for XACML (see http://www.oasis-open.org/committees/download.php/24681/xacml-profile-saml2.0-v2-spec-wd-5-en.pdf). Until the time such an XACML metadata is made a standard, this specification recommends that the SAML PDPDescriptor be used for XACML as well. When the XACML metadata is standardized, this specification will be updated to use it.

PDPs and AAs SHOULD publish SAML metadata. If acting as AA, then AttributeAuthorityDescriptor entry MUST be included. KeyDescriptor with use as 'signing' MUST be included within the AttributeAuthorityDescriptor. If PDP, then PDPDescriptor element MUST be included. KeyDescriptor with use as 'signing' MUST be included within the PDPDescriptor.

7.5 Authorization Message Details

7.5.1 Attribute Statement

An AttributeStatement can contain any number of either Attribute or EncryptedAttribute element.

Attribute element

<Attribute> element identifies the attribute by its name. The name of the attribute is pre-defined, either in this specification or in a bilateral agreement. An optional format for the attribute's name can also be specified using the NameFormat attribute. This specification uses 'URI' as the format - identified by "urn:oasis:names:tc:SAML:2.0:attrname-format:uri" – for all the attributes identified in this specification.

An attribute may have more than one <AttributeValue> child element. The data type for the pre-defined attributes is identified later in this specification.

Below is a sample <AttributeStatement> with one <Attribute> element having two <AttributeValue> elements.

```
   <saml:AttributeStatement>
     <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:uri"        Name="urn:cablelabs:olca:1.0:attribute:authz:channelID">
       <saml:AttributeValue xsi:type="xs:string">Channel-1-unique-
ID</saml:AttributeValue>
       <saml:AttributeValue xsi:type="xs:string">Channel-2-unique-
ID</saml:AttributeValue>
     </saml:Attribute>
   </saml:AttributeStatement>
```

EncryptedAttribute element

It should be noted that the AnP (who issues the assertions with the AttributeStatements) might not really know if the channel between the subscriber and the SP (over which the assertion is transported) is secured or not. Thus, it is advised that any attribute whose contents are to be protected from a potential eavesdropper be encrypted.

The actual content is symmetrically encrypted (with the algorithm identified within the contents of EncryptedAttribute element) with a random key. The key itself is encrypted with the public key of the SP. The SP's public key is obtained from the <KeyDescriptor> entry of the <SPSSODescriptor> element of SP's metadata. The 'use' on the <KeyDescriptor> must be 'encryption'.

For the rest of the details on <EncryptedAttribute>, please see [SAML 2.0 CORE] document.

Below is a sample <AttributeStatement> with <EncryptedAttribute> element entry.

```
<saml:AttributeStatement>
   <saml:EncryptedAttribute>
      <xenc:EncryptedData Id="_encryptedAttr1"
Type="http://www.w3.org/2001/04/xmlenc#Element"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
         <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
         <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:RetrievalMethod
Type="http://www.w3.org/2001/04/xmlenc#EncryptedKey"... URI="#_encryptedKey"/>
         </ds:KeyInfo>

<xenc:CipherData><xenc:CipherValue>A1B2C3...</xenc:CipherValue></xenc:CipherDat
a>
      </xenc:EncryptedData>

      <xenc:EncryptedKey Id="_encryptedKey"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
         <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"/>
         <xenc:CipherData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
            <xenc:CipherValue
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">encrypted key value
here</xenc:CipherValue>
         </xenc:CipherData>
         <xenc:ReferenceList>
            <xenc:DataReference URI="#_encryptedAttr1"/>
         </xenc:ReferenceList>
      </xenc:EncryptedKey>
   </saml:EncryptedAttribute>
</saml:AttributeStatement>
```

NOTE: **To AnPs/AAs** - AnPs/AAs must ensure that the values they are including for attributes are not only subscriber specific, but also SP specific. For example, an attribute called channelID may have only values that are subscribed by the subscriber *and* are within the domain of the SP in context.

Requirements

1. AnPs, AAs, and SPs MUST conform to SAML 2.0 specification for schema and processing instructions for <AttributeStatement>, including <Attribute> and <EncryptedAttribute> elements.
2. Implementers (SP, AnP/AA) MUST support the attribute names as published in this specification. Implementers (SP, AnP/AA) MAY support additional attributes based on bilateral agreements.
3. The NameFormat MUST be "urn:oasis:names:tc:SAML:2.0:attrname-format:uri" for all attributes identified in this specification.

4. The type of <AttributeValue> MUST be according to this specification for attributes identified in this specification.
5. An <Attribute> may have more than one <AttributeValue> child element.
6. For <EncryptedAttribute>, the key MUST be encrypted with "encryption" usage certificate from the SP's metadata element <SPSSODescriptor>.
7. SP/PEP MAY cache these attributes for the period that the enclosing assertion is valid. After that period, SP/PEP MUST discard these values and issue new attribute query.

7.5.2 Attribute Statement within authentication assertion

Since the attribute statement within the authentication response is of an unsolicited nature, it is important to set expectations on which attributes may be present in such an attribute statement. This specification identifies a set of attributes as a mandatory/optional set that will/can be included in the attribute statement (see the Requirements section below).

A sample message and its contents are identified in Figure 20. Note that the namespaces are not included; real messages will use namespaces as defined in the SAML and XACML schemas.

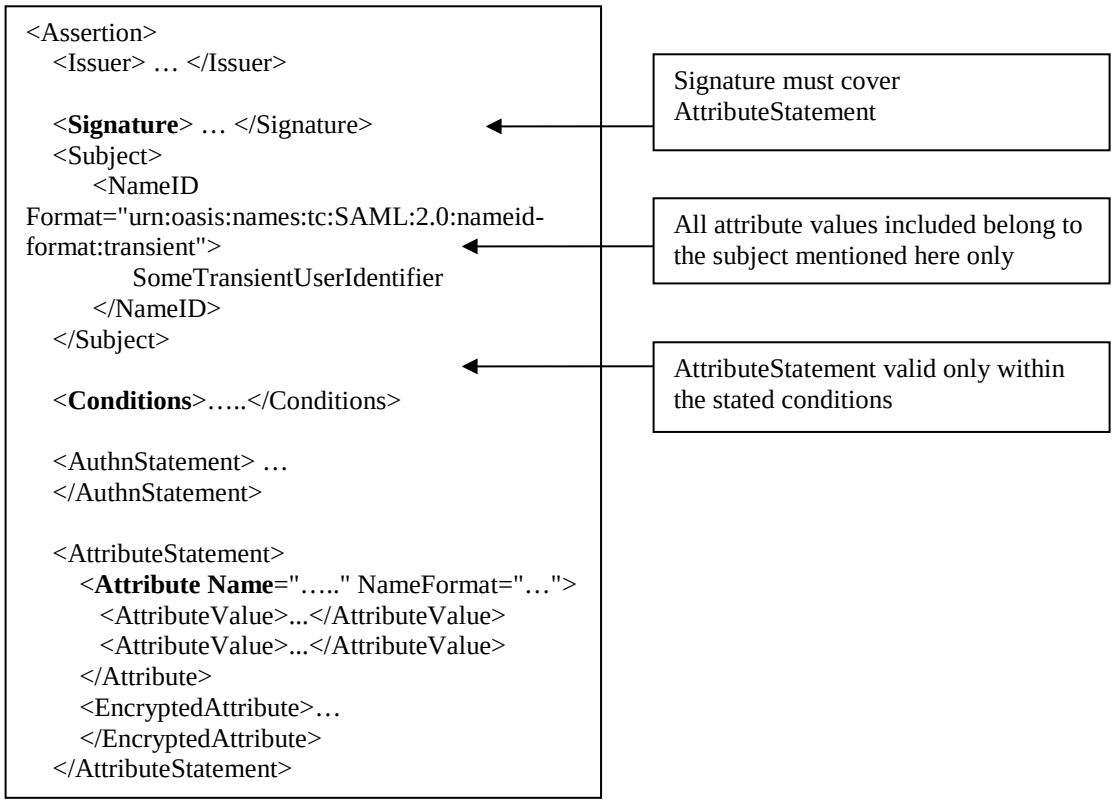

Figure 20 - Sample Attribute Statement Within Authentication Assertion

Requirements

1. Signature MUST cover <AttributeStatement> (if signature covers the entire assertion or SAML response, then it implicitly covers AttributeStatement as well).
2. <AttributeStatement> contents are subject to the entries in the <Conditions> element. Please see the Authentication section (Section 6) of this specification for details on <Conditions> element. After the conditions expire, SP MUST renew the values by issuing a SAML <AttributeQuery> (see Appendix II).
3. AnP MUST include the following attributes (identified by Name of the attribute):

a. "urn:cablelabs:olca:1.0:attribute:subscriber:identifier" – used to uniquely identify this subscriber at the AnP/PDP. SP will use this in all back channel requests. This attribute may be persistent or transient.

4. AnP MAY include the following attributes (identified by Name of the attribute):

   a. "urn:cablelabs:olca:1.0:attribute:authz:channelID" – a list of channel IDs that the subscriber is authorized to view

   b. "urn:cablelabs:olca:1.0:attribute:authz:maxMPAA" – the maximum MPAA rating the subscriber is authorized to view

   c. "urn:cablelabs:olca:1.0:attribute:authz:maxVCHIP" – the maximum VChip rating the subscribed is authorized to view

   d. "urn:cablelabs:olca:1.0:attribute:authz:devicePermission" – valid values are GRANTED and DENIED. "DENIED" implies that the subscriber is authorized for this SP, but the device used by the user is not authorized.

   e. "urn:cablelabs:olca:1.0:attribute:authz:deviceMessage" – a message to be shown to subscriber, related to devicePermission (above).

   f. "urn:cablelabs:olca:1.0:attribute:authz:deviceID" – an opaque identifier for the device used by the subscriber.

   g. "urn:cablelabs:olca:1.0:attribute:authz:deviceType" – an opaque type of the device used by the subscriber.

   h. "urn:cablelabs:olca:1.0:attribute:authz:onNet" – a Boolean indicating that the subscriber is on MVPD's network.

   i. "urn:cablelabs:olca:1.1:attribute:authz:atHome" – a Boolean indicating whether the user was at home when this authentication took place. A false value indicates that a user is known to be outside the home. This attribute MUST be omitted if it is unknown whether or not the user is in the home.

   j. "urn:cablelabs:olca:1.1:attribute:subscriber:profileType" – indicates what the provided "*urn:cablelabs:olca:1.0:attribute:subscriber:identifier*" refers to. Values include, but are not limited to, "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:device*" "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group*" and "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:user*". If the value is either "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:device*" or "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group*", then the value of "*urn:cablelabs:olca:1.0:attribute:subscriber:identifier*" must match the value of "*urn:cablelabs:olca:1.0:attribute:authz:deviceID*" or one of the identifiers specified within the "*urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier*" section, respectively.

   k. "urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier" – one or more persistent values used to uniquely identify the subscriber's distinguishing group(s) at the AnP/PDP. When provided, a custom "*groupType*" attribute may also be provided in the <AttributeValue> tag for each value. If so, the namespace of the groupType must be "*urn:cablelabs:olca:1.1:attribute:subscriber"*. The value of the groupType may be one of:
   "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:household*",
   "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:building*"
   "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:campus*",
   "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:municipalArea*", or any other unique URI. When multiple values are provided, the groupType is required. The SP may use any of these values to link users together for the purposes of implementing parental controls, counting active sessions, publishing social sharing / activity, or providing any other functionality that is relevant to the provided groupType.

Example:

```
<saml:Attribute
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
   Name="urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier">
   <saml:AttributeValue
      xsi:type="xs:string"
      xmlns:olcasub="urn:cablelabs:olca:1.1:attribute:subscriber"
      olcasub:groupType=
"urn:cablelabs:olca:1.1:attribute:subscriber:groupType:household">
         household_id
   </saml:AttributeValue>
   <saml:AttributeValue
      xsi:type="xs:string"
      xmlns:olcasub="urn:cablelabs:olca:1.1:attribute:subscriber"
      olcasub:groupType=
"urn:cablelabs:olca:1.1:attribute:subscriber:groupType:building">
         building_id
   </saml:AttributeValue>
```

For primary-account and sub-account relationship examples, see Appendix II.5 - Privileged-Account, Unprivileged-Account and Group-Account Representation.

l. "urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser" – A Boolean value indicating whether the authenticated profiled represents a "privileged user". A privileged user is a head of household, primary, or admin account of the household and is usually defined as a profile which has the capability of creating, modifying, or deleting other profiles within the household, as well as managing parental controls for these profiles.

m. "urn:cablelabs:olca:1.1:attribute:subscriber:country" – indicates the country of the user's home address. The format of the country is a two-letter code following the ISO 3166-1 alpha-2 standard as defined by [ISO 3166].

n. "urn:cablelabs:olca:1.1:attribute:subscriber:postalCode" – indicates the postal code of the user's home address. When used, the "urn:cablelabs:olca:1.1:attribute:subscriber:country" attribute must also be present. The format of the postal code is specific to the country of origin and outside the scope of this document.

If the attribute "urn:cablelabs:olca:1.0:attribute:authz:devicePermission" is present and its value is DENIED, SP MUST deny access to its content. If the attribute "urn:cablelabs:olca:1.0:attribute:authz:deviceMessage" is present, then SP MUST show that message to the user.

If the attribute "urn:cablelabs:olca:1.0:attribute:authz:deviceID" is present, SP MUST persist it along with other user attributes, and send it in the Authorization query (see below). If the attribute "urn:cablelabs:olca:1.0:attribute:authz:deviceType" is present, then SP MUST persist it along with other user attributes, and send it in the Authorization query (see below).

7.5.3 SAML Attribute Query and Response

7.5.3.1 Request

Attribute queries are used by SP to obtain specific attribute information from the AnP or AA. These requests are sent directly from the SP to the AA using a back channel – not through subscriber's browser channel.

SAML Attribute queries and responses are transported over HTTP/S within a SOAP Envelope.

Requirements

1. The implementations MUST adhere to requirements from section 6, Assertion Query/Request Profile, of [SAML 2.0 PROFILES].

2. SAML SOAP Binding MUST be used for attribute query and response. Implementations MUST adhere to section 3.2, SAML SOAP Binding, of [SAML 2.0 BINDINGS].

Figure 21 is a sample message with the important data highlighted.

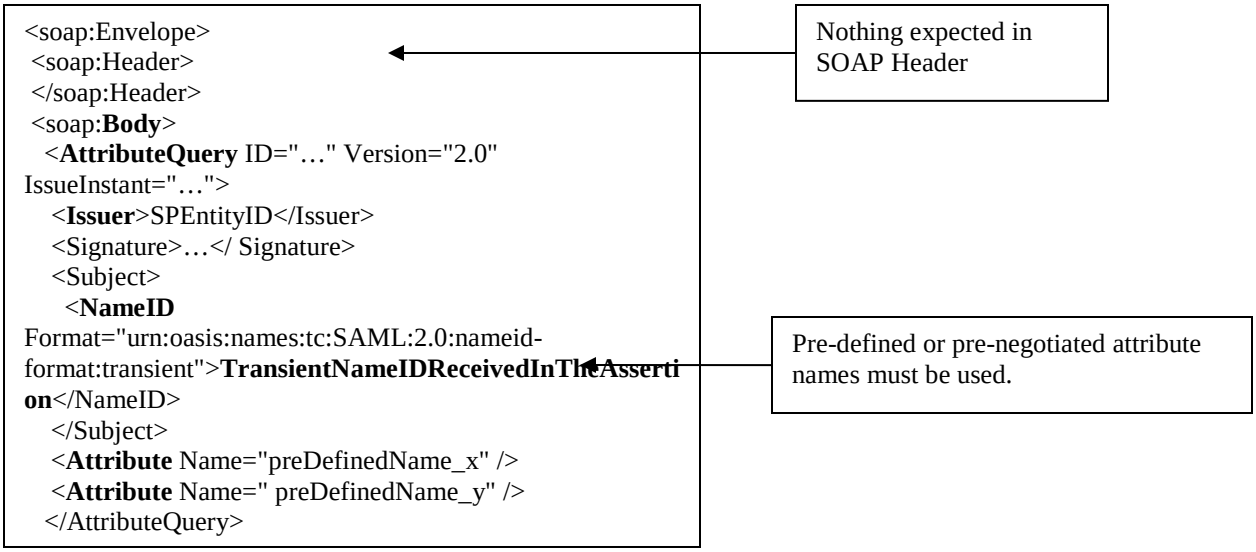

Figure 21 - Sample Attribute Query Message

Though the SAML schema allows more than one <Subject> to be given in the <AttributeQuery>, in the context of this specification, only one <Subject> entry is expected.

AA is expected to be able to map from the <Subject>'s <NameID> provided in this request to the actual user record – as this <NameID> value equals the one sent by the AnP during authentication.

Multiple attributes MAY be mentioned within a single <AttributeQuery>. AA MUST either respond with values for all attributes or an error message (see Section 7.5.7) with no AttributeStatement.

Requirements

1. <Issuer> MUST be present and equal the entity ID of the SP.
2. The <Subject>'s <NameID> element MUST be equal to value of the "urn:cablelabs:olca:1.0:attribute:subscriber:identifier" attribute from the authentication assertion (see below).
3. <Attribute> names MUST be either as defined in this specification, or pre-negotiated using a bilateral agreement.
4. <Attribute> entries MUST not contain any <AttributeValue> child elements.

Mapping from subject identifier attribute to Subject/NameID

Figure 22 outlines how the Subject/NameID in the AttributeQuery needs to be populated from the "urn:cablelabs:olca:1.0:attribute:subscriber:identifier" attribute from the authentication assertion.

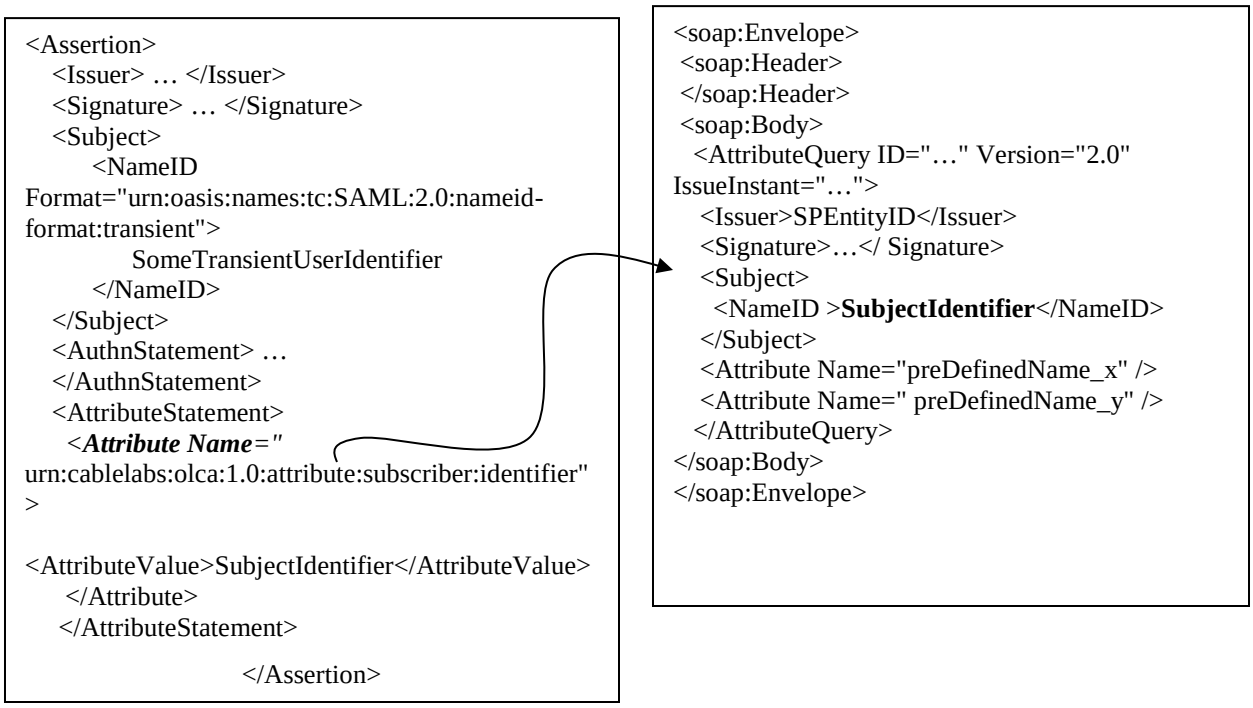

Figure 22 - Mapping Subject Identifier from Authentication Assertion to Subject/NameID in Attribute Query

7.5.3.2 Response

The response is sent using the same connection used for sending the request.

Requirements

1. AAs SHOULD tailor the response attribute value based on the subject in the request *and* the SP issuing the request.

Figure 23 is a sample response message with the important data highlighted.

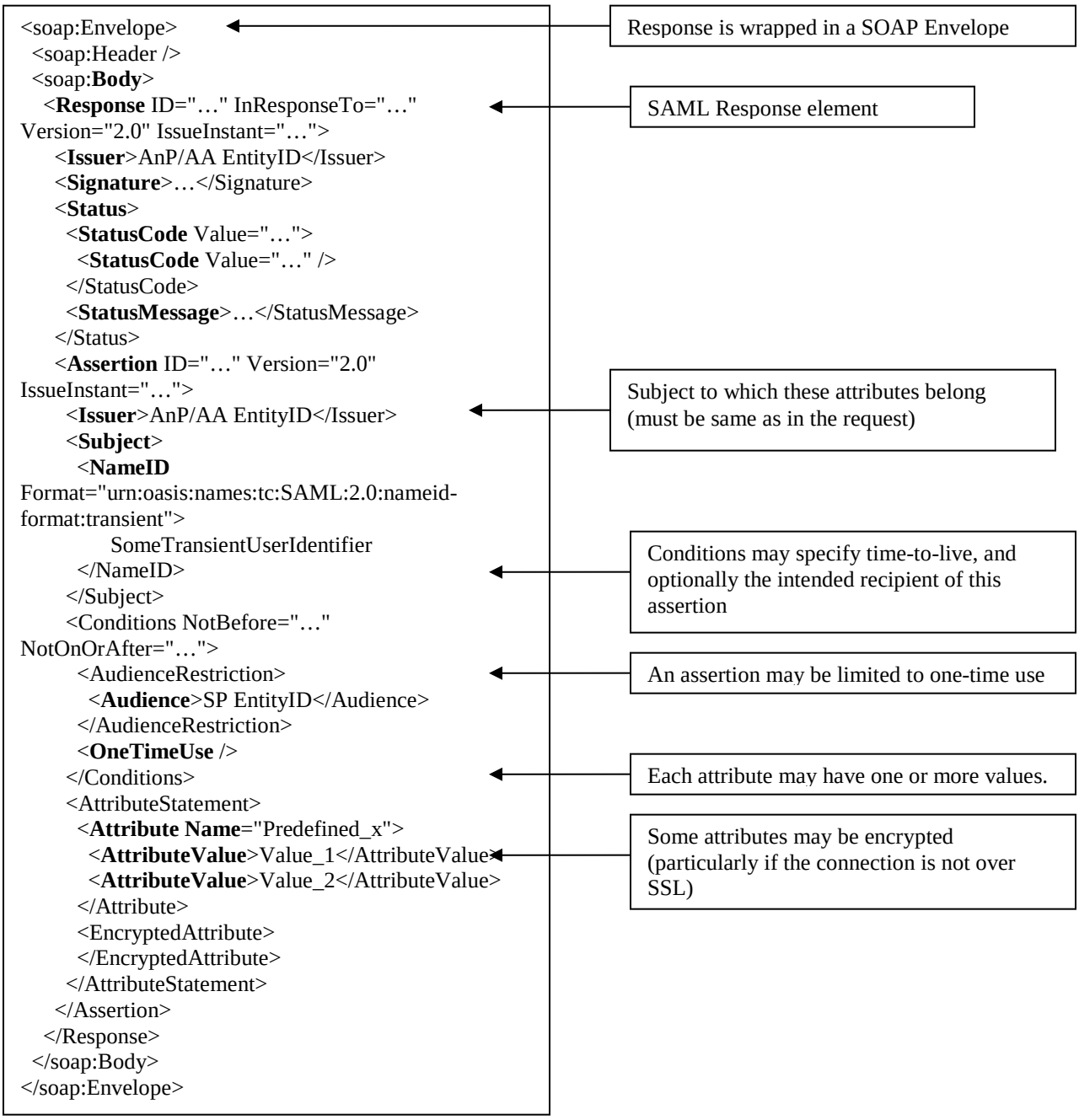

Figure 23 - Sample Attribute Response Message

Requirements

1. SP MUST validate that the <Issuer> of the <Response> is the same entity to which it originally sent the request. This can be done by validating the signature (see Appendix VI).

2. <StatusCode> values MUST be according to section 3.2.2.2 of [SAML 2.0 CORE]. Values relevant to this specification are identified below. AA and SP MUST support these values. AA and SP MUST adhere to the interpretation as given in section 3.2.2.2 of [SAML 2.0 CORE]. The top-level status code must be one of:

    a. urn:oasis:names:tc:SAML:2.0:status:Success

    b. urn:oasis:names:tc:SAML:2.0:status:Requester

    c. urn:oasis:names:tc:SAML:2.0:status:Responder

    d. urn:oasis:names:tc:SAML:2.0:status:VersionMismatch

    Optionally an additional second-level <StatusCode> MAY be present. The following second-level values are used in this specification:

    a. urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue

    b. urn:oasis:names:tc:SAML:2.0:status:NoAuthnContext

    c. urn:oasis:names:tc:SAML:2.0:status:NoPassive

    d. urn:oasis:names:tc:SAML:2.0:status:RequestDenied

    e. urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported

    f. urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal

    g. urn:cablelabs:olca:1.0:status:AuthnCanceled

    h. urn:cablelabs:olca:1.0:status:WrongIdP

    i. urn:cablelabs:olca:1.1:status:UnsupportedAuthnRestriction

    j. urn:cablelabs:olca:1.1:status:UnsatisfiedAuthnRestriction

    See Appendix III for more information about error codes.

3. If the top-level <StatusCode> is not "*`urn:oasis:names:tc:SAML:2.0:status:Success`*", then AA MUST NOT include any <Assertion>.

4. AA MAY include a <StatusMessage>. These are considered specific to the AA implementation and are to be used for logging/diagnosis purposes only.

5. If the AA does not support <AttributeQuery>, then it MUST respond with a top-level status code of "*urn:oasis:names:tc:SAML:2.0:status:Responder*" and a second-level <StatusCode> of "*`urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported`*". In this case, SP MAY continue to let the user browse the available content (obviously not filtered as the SP has no way to filter). After the user selects to view a particular content, SP MUST base its access control on the XACML request/response.

6. If the second-level <StatusCode> is "*`urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal`*", then SP MUST invalidate the subscriber session and initiate the authentication process.

7. If the <StatusCode> is not "*urn:oasis:names:tc:SAML:2.0:status:Success*", then SP MUST show an error message to the subscriber and deny access to the content or provide the appropriate remediation flow based upon the status codes returned and attempt the request again. The SP MAY log the status code(s) and status messages in the response for alert generation and diagnosis.

8. AA MAY include <Conditions> element within the Assertion. If included, SP MUST process them according to [SAML 2.0 CORE]. In particular,

    a. If NotBefore is mentioned, then SP MUST not use the attribute values before such a time.

    b. If NotOnOrAfter is mentioned, then SP MAY cache the results until such a time. The SP MUST discard the values and issue a fresh AttributeQuery after that time.

    c. If <OneTimeUse> is included, then SP MUST NOT cache the attribute values ever, and issue a fresh AttributeQuery the next time the values are needed.

9. AA MUST include all attributes from the request in the response. If any attributes cannot be included for any reason, then no assertion MUST be included in the response. Also, the status code MUST NOT be "*urn:oasis:names:tc:SAML:2.0:status:Success*".
10. There can be more than one AttributeValue for each Attribute.
11. Some attributes MAY be encrypted (see Security requirements below). SPs MUST be capable of decrypting such encrypted attributes.

7.5.4 XACML Authorization Query and Response

7.5.4.1 Authorization Query

XACML authorization queries are used to request subscriber authorization to view a specific content. More than one content ID can be included in the request and the response can state permit/deny decisions individually or collectively. Environmental data may also be included in the request. This specification identifies some environmental attributes, others may be established through bilateral agreements.

PDP will evaluate the contents of the request (subject, content IDs and environmental data) against its stored policies and respond with a decision of permit or deny. The PEP MUST enforce the decision that it received from the PDP.

The implementations MUST adhere to SOAP binding as given in [SAML 2.0 BINDINGS] (section 3.2 for SOAP Binding), SAML profile for [SAML 2.0 Profile XACML 2.0], and [XACML 2.0 Spec Core].

Figure 24 is a sample message with important data highlighted.

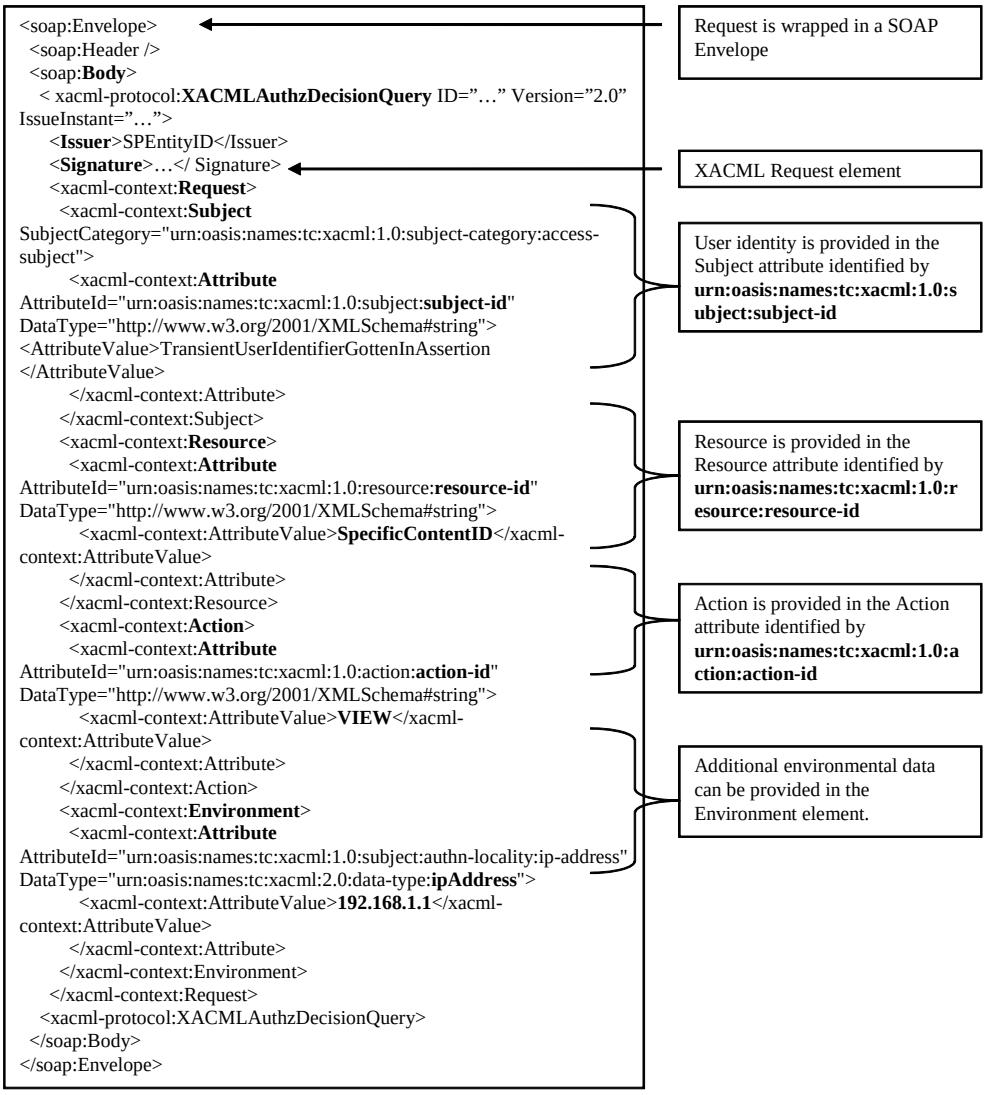

Figure 24 - Sample XACMLAuthzDecisionQuery Message

Subject element: Mapping from SAML assertion to XACML request

Figure 25 shows the mapping from the subject-identifier attribute from authentication assertion to the Subject element in XACML request.

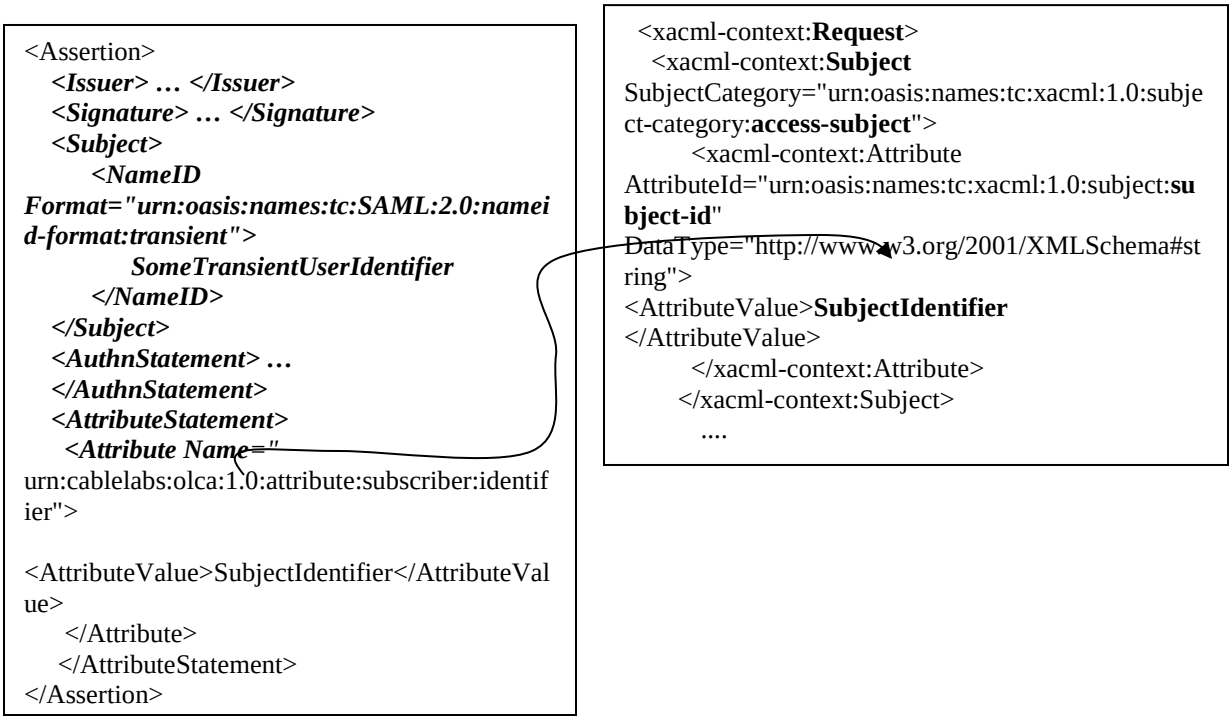

Figure 25 - Mapping from SAML Subject to XACML Subject

The following rules apply:

1. There MUST be only one <Subject> element in the request. Its <SubjectCategory> MUST be "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject".
2. There MUST be only one <Attribute> within an access-subject <Subject>. Its AttributeID for that attribute MUST be "urn:oasis:names:tc:xacml:1.0:subject:subject-id".
3. There MUST be only one <AttributeValue> present for the subject-id <Attribute>. Its value MUST match the value of the " urn:cablelabs:olca:1.0:attribute:subscriber:identifier" attribute from the authentication assertion.

PDP is expected to be able to map from the given Subject's AttributeValue to the actual user record, as this <NameID> used equals the one sent by the AnP during authentication. The state correlation between AnP and PDP instances is internal to implementation.

If the Authentication Assertion contained the attribute "urn:cablelabs:olca:1.0:attribute:authz:deviceID", SP MUST include it in the Authorization query (see below). If the Authentication Assertion contained the attribute "urn:cablelabs:olca:1.0:attribute:authz:deviceType" is present, then SP MUST include it in the Authorization query (see below).

```
<xacml-context:Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-
category:access-subject">
     <xacml-context:Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>…</AttributeValue>
     </xacml-context:Attribute>
     < xacml-context:Attribute
AttributeId="urn:cablelabs:olca:1.0:attribute:authz:deviceID" DataType
="http://www.w3.org/2001/XMLSchema#string">
```

```
          <AttributeValue>deviceID from Authentication
Assertion</AttributeValue>
     </Attribute>
     < xacml-context:Attribute
AttributeId="urn:cablelabs:olca:1.0:attribute:authz:deviceType" DataType
="http://www.w3.org/2001/XMLSchema#string">
          <AttributeValue> deviceType from Authentication Assertion
</AttributeValue>
     </Attribute>
</xacml-context:Subject>
```

Resource element

<Resource> element identifies the content ID under OLCA. Each content ID MUST be provided through an <Attribute> of "urn:oasis:names:tc:xacml:1.0:resource:resource-id". Multiple content IDs MUST be represented by multiple <Resource> elements, not by multiple <Attribute> elements within a single <Resource> element. Thus, the following rules apply.

1. Each <Resource> element MUST have only one <Attribute> element, with an AttributeID of "urn:oasis:names:tc:xacml:1.0:resource:resource-id".
2. The above <Attribute> element MUST have only one <AttributeValue> element whose contents are the content ID.
3. If including multiple content IDs, each content ID MUST be represented by separate <Resource> elements.

In addition, based on a bilateral agreement, the <Resource> element may also contain XACML <ResourceContent> element that contains XML content. Such implementations SHOULD adhere to the OASIS "Hierarchical resource profile for XACML" document (http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-hier-profile-spec-os.pdf).

It is recognized that different schemes could be used to identify the resource to be authorized – for example, through a resource-id attribute or through XML content. To make it easier for the receiving party to know which scheme is being used by the sender, a new 'scheme' attribute is added to the original <Resource> element schema from XACML. Given below is an example usage of this attribute.

```
<Resource scheme="urn:cablelabs:olca:1.0:EIDR">
   <Attribute AttributeId="resource-id" DataType="xsd:String">
       <AttributeValue>a1b2c3</AttributeValue>
    </Attribute>
</Resource>
```

The original XACML schema for the <Resource> element is extended as below

```
<xs:element name="Resource" type="xacml-context:ResourceType"/>
<xs:complexType name="ResourceType">
   <xs:sequence>
      <xs:element ref="xacml-context:ResourceContent" minOccurs="0"/>
      <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute ref="olca:scheme" use="optional"/>
</xs:complexType>
```

See Section 7.7 for the extension attribute definition.

Action element

The only action applicable in OLCA is "VIEW". Thus, the following rules apply:

1. There MUST be only one <Attribute> within the <Action> element. The AttributeID on that <Attribute> element MUST be "urn:oasis:names:tc:xacml:1.0:action:action-id".

2. There MUST be only one <AttributeValue> within the above <Attribute> element. The value of <AttributeValue> MUST be "VIEW".

Environment attributes

This specification recognizes the following attributes that MUST be included in the request.

1. Subscriber's IP address – AttributeId MUST be "urn:oasis:names:tc:xacml:1.0:subject:authn-locality:ip-address". The DataType MUST be "urn:oasis:names:tc:xacml:2.0:data-type:ipAddress".

Requirements

1. Implementations MUST adhere to SOAP binding as given in [SAML 2.0 BINDINGS] (section 3.2 for SOAP Binding), SAML profile for [SAML 2.0 Profile XACML 2.0] and [XACML 2.0 Spec Core].
2. Only one <Subject> element MUST be present in the request (note that XACML schema allows multiple Subjects).
3. Mapping from SAML Assertion <Subject> to XACML request <Subject> MUST be performed according to the rules outlined earlier in this document.

7.5.4.2 Authorization Response

The XACML query response essentially delivers a Permit or Deny decision for the details given in the request. If there are multiple resources given in the request, then there may be multiple decisions, one for each resource. However, a single decision could also be delivered, which implicitly covers all resources in the request.

Figure 26 is a sample response message with the important data highlighted.

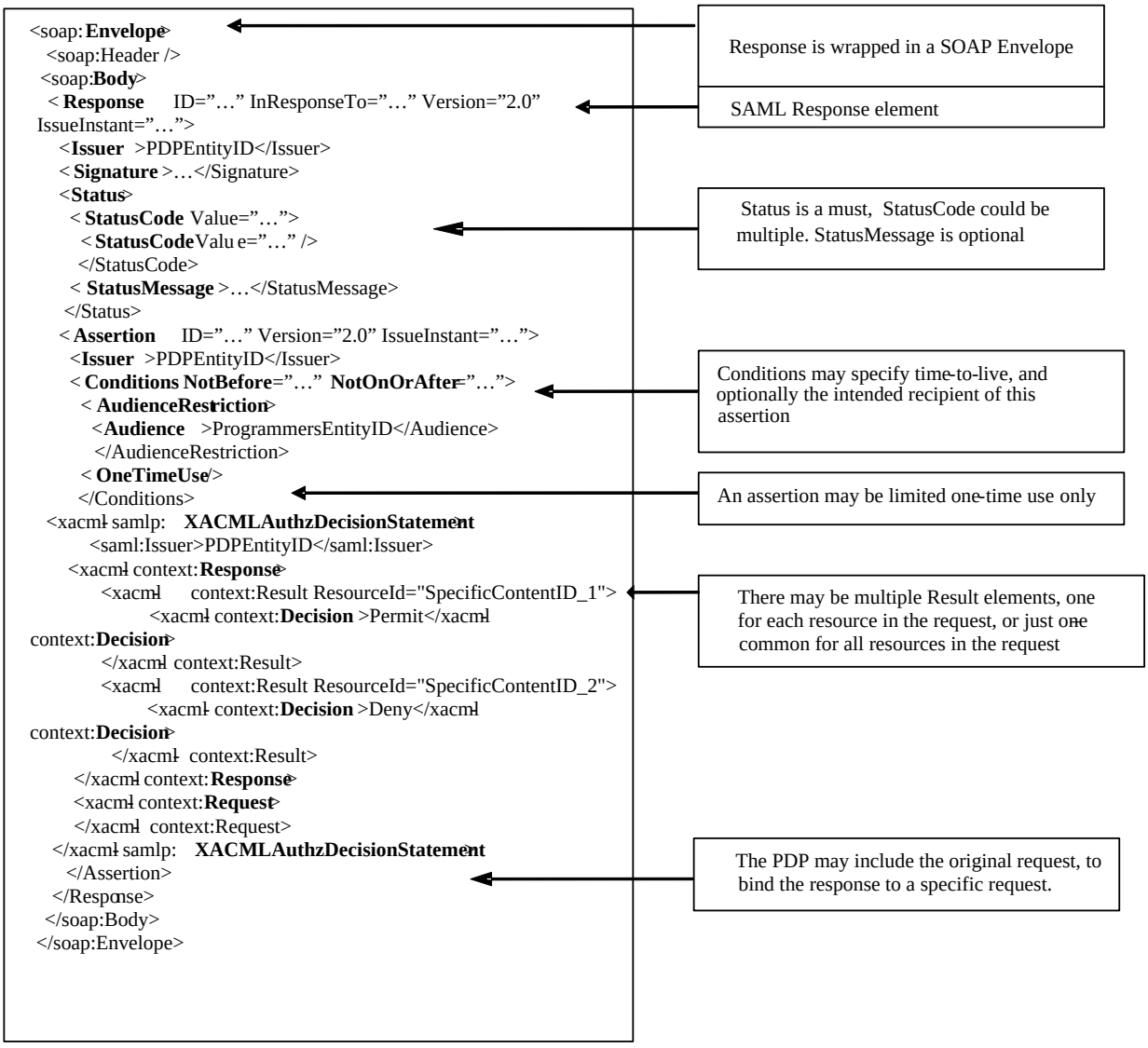

Figure 26 - Sample XACML Response

For details on the schema or explanation of the content, please see [SAML 2.0 BINDINGS], [XACML 2.0 Spec Core] and [SAML 2.0 Profile XACML 2.0].

Multiple <Result> elements may be included – one for each Resource from the Request. Alternately, only one <Result> element may be included – without the ResourceId attribute – that applies to all Resources from the Request.

The PEP MUST adhere to the decisions from the response. The XACMLAuthzDecision assertion may include <Conditions>. The PEP MUST follow the conditions – the description of the expected behavior for each component of Conditions is explained within [SAML 2.0 CORE].

Requirements

1. Implementations MUST adhere to SOAP binding as given in [SAML 2.0 BINDINGS] (section 3.2 for SOAP Binding), SAML profile for [XACML 2.0 Spec Core] and [SAML 2.0 Profile XACML 2.0].

2. If <StatusCode> is "urn:oasis:names:tc:SAML:2.0:status:Success", then <Assertion> MUST be present. <Assertion> MUST have a single XACMLAuthzDecisionStatement. If <StatusCode> is not "urn:oasis:names:tc:SAML:2.0:status:Success", then <Assertion> MUST not be present.
3. Multiple <Result> elements MAY be present. If so, then ResourceId attribute MUST be present for each <Result> element. If only one <Result> element is present, then ResourceId MAY be included, only if the request contained one resource.
4. If there is a problem with any resource, then <Decision> for that Resource MUST be "Indeterminate".
5. <Decision> values can only be "Deny", "Permit", "Indeterminate", or "NotApplicable" (per XACML standard).
    a. "Permit" and "Deny" MUST be treated as per XACML spec.
    b. "Indeterminate" MUST be used to indicate that the PDP encountered problems. SP/PEP MUST treat this as equivalent to "Deny". However, it MUST NOT cache this decision and resend the request when required next time.
    c. "NotApplicable" means that the PDP did not find the resource or any policies corresponding to the resource. This may happen when SP has a content that the PDP is not aware of or not configured for. This decision MUST be treated by SP/PEP as equivalent to "Deny". However, both parties SHOULD also take steps to rectify the configuration issues. SP MAY show a corresponding error message to the subscriber. Again, this decision MUST NOT be cached, and the SP/PEP MUST send another request when required next time.
6. The <Request> element MAY be present in the Response. This is solely for the sake of binding the response to a particular request. PEPs MUST NOT expect this to be present.
7. PDP MAY include <Obligations> in the response. If included, SP MUST perform those obligations before enforcing the decision in the response. This document defines some obligations (see Section 7.6.2). Bilateral agreements can be used to define additional obligations.

7.5.4.3 Caching Authorization Responses

If the PDP determines that a particular decision is going to be valid, not just at the time of making the decision but, into sometime in the future – it may communicate such 'validity period' to the SP. SPs MAY then cache such decision for the given validity period – meaning that they can reuse that decision (within the validity period) without sending another authorization decision to the PDP.

This feature can be used to reduce the load at the PDP, as well as improve the response times for the subscriber.

Note that the core SAML and XACML schemas do not have a provision for conveying such a validity period.

This specification adds a new element – called 'Validity' – to both <XACMLAuthzDecisionStatement> element and to the <Result> element.

The schema for <Validity> element is given in Section 7.7.

The extended schema for <Result> element is given below.

```
<xs:element name="Result" type="xacml-context:ResultType"/>
<xs:complexType name="ResultType">
   <xs:sequence>
      <xs:element ref="olca:Validity"/>
      <xs:element ref="xacml-context:Decision"/>
      <xs:element ref="xacml-context:Status" minOccurs="0"/>
      <xs:element ref="xacml:Obligations" minOccurs="0"/>
   </xs:sequence>
   <xs:attribute name="ResourceId" type="xs:string" use="optional"/>
</xs:complexType>
```

Requirements:

1. <Validity> element is required.

2. If the Validity element contains the <Use> element with a value of 'OneTime', then SPs MUST NOT cache this decision.
3. SPs MAY cache the <Result> contents only for the Validity period mentioned in the <TimeBounds> element. After the validity period ends, if authorization on the same content is required, SP MUST send another authorization request.

7.5.4.4 Hierarchical content treatment

Content could be structured in a hierarchy – for example, a series is composed of seasons. A season itself could be composed of episodes. Further down, a particular content could have manifestations in different format, or different quality/resolution.

From an authorization perspective, hierarchical representation of content allows efficient communication of the decisions. For example, in one shot the PDP can indicate to the SP/PEP that a particular subscriber is allowed to watch all the episodes belonging to all seasons of a series. As another example, the decision can state that the subscriber is allowed to watch episodes belonging to only one season.

What this allows is:

- reduction in traffic (back and forth at each level),
- more intelligent presentation at the SP.

Note that the hierarchy may not necessarily be evident in the 'content ID' – for example, the 'content ID' could be an opaque string, without any hint of who is the parent content ID or what are its child content IDs. On the other hand, an XML representation of the content could provide information regarding the parent or children. Further, the 'content ID' for an XML content could be represented as an XPath (see [XPath]), not as an opaque string. Further, the notion of hierarchy can be anything that is commonly understood between PDP and the PEP.

In this section, however, the 'content ID' will be assumed to be an opaque string. Moreover, 'content ID' will be treated as representing a node in the hierarchy. Also, the hierarchy is assumed to be a tree structure – one parent and many children.

Results on hierarchical content

If the content ID belongs to a hierarchical content, then the meaning of the decisions need to be clearly specified. For example, what is the meaning of a 'Permit' decision on a content ID that is at level-2 of a 5 level tree?

In general, it is assumed that the ultimate viewable content is always the leaf node of the hierarchy. And decisions have the following interpretation.

1. Permit – if given on a leaf node, then the viewable content is allowed. If given on a non-leaf node, then it means that all content IDs underneath this content ID (at all levels below) are allowed.
2. Deny – if a given on a leaf node, then the viewable content is not allowed. If given on a non-leaf node, then its meaning could vary depending on the <Status> element (see later). If the <Status> element within the <Result> contains a <StatusCode> of urn:olca:1.0:status:ContentID_TOO_COARSE_GRAINED then some content underneath this content ID is allowed, but not all the content. The requester will need to make another query for the content underneath. But if either the <Status> element is missing, or has a different <StatusCode>, then it means all the content underneath this content ID is not allowed.
3. Indeterminate – the meaning of this decision does not change from what is stated earlier in this document.
4. NotApplicable – the meaning of this decision does not change from what is stated earlier in this document.

Hierarchical suffixes in Authorization Requests

For non-XPath content IDs, the following suffixes will be used

* – all immediate child nodes

* – all nodes at all levels below the current node

For XPath content IDs, this section does not apply.

These are introduced to allow the requester to specify what level of details they want to include in the response. If '*' is specified, then it means the query is for all immediate children of the given content ID. The responder then

will have to include responses for all immediate children – assuming that the decision for the given content ID is not a 'Permit' (in which case it means all content at all levels under this content ID is allowed), or 'Deny' (which means all content at all levels underneath this content ID is not allowed). Similarly, if the query includes '*', then it means the query is for all children at all levels

Also note that this makes '\' and '*' as reserved characters in content IDs.

This specification addresses the following requirements:

1. ability to query for a decision on 'all' child nodes of a given node,
2. ability to respond with a decision that applies to 'all' child nodes of a given node,
3. ability to query for a decision for 'all' nodes at all levels below the given node,
4. ability to respond with a decision that applies to 'all' nodes at all levels below a given node, and
5. ability to respond with a decision that applies to some, but not all, nodes below a given node.

Requesting decision for 'all immediate child nodes'

The content ID in the 'resource-id' attribute of the <Resource> element can now be appended with a '*', as shown in the example below.

```
         <xacml-context:Resource>
           <xacml-context:Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#string">
             <xacml-context:AttributeValue>SpecificContentID\*</xacml-
context:AttributeValue>
           </xacml-context:Attribute>
         </xacml-context:Resource>
```

The response for such a request could be as earlier - that is a straight-forward 'Permit' or 'Deny', or the PDP could provide responses to specific content at levels below the given content's level. For example, the below example says that all content under 'SpecificContentID/Level-1-1' is allowed, but no content under 'SpecificContentID/Level-1-2' is allowed, and 'some' content under 'SpecificContentID/Level-1-3' is allowed (see later for the status code).

```
        <xacml-context:Response xmlns:xacml-
context="urn:oasis:names:tc:xacml:2.0:context:schema:os">
             <xacml-context:Result ResourceId=" SpecificContentID">
             <xacml-context:Decision>Deny</xacml-context:Decision>
        <xacml-context:Status>
            <xacml-context:StatusCode
Value="urn:olca:1.0:status:ContentID_TOO_COARSE_GRAINED" />
        </xacml-context:Status>
        </xacml-context:Result>
             <xacml-context:Result ResourceId=" Level-1-1_ContentID">
              <xacml-context:Decision>Permit</xacml-context:Decision>
             </xacml-context:Result>
             <xacml-context:Result ResourceId=" Level-1-2_ContentID">
              <xacml-context:Decision>Deny</xacml-context:Decision>
         </xacml-context:Result>
             <xacml-context:Result ResourceId=" Level-1-3_ContentID">
              <xacml-context:Decision>Deny</xacml-context:Decision>
        <xacml-context:Status>
            <xacml-context:StatusCode
Value="urn:olca:1.0:status:ContentID_TOO_COARSE_GRAINED" />
        </xacml-context:Status>
        </xacml-context:Result>
         </xacml-context:Response>
```

Requesting decision for 'all lower-level nodes'

This is done similar to 'all immediate child nodes' case, except that the resource ID in the request will look like 'SpecificContentID*'. The responses will also be similar to the 'all immediate child nodes' case.

Authorizing at a higher-level

Depending on the level of request content within the hierarchy, the PDP may also come back with a decision that applies to a much higher level than the one requested. Consider the following <Resource> element in the request

```
          <xacml-context:Resource>
            <xacml-context:Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#string">
              <xacml-context:AttributeValue>Level-6_ContentID</xacml-
context:AttributeValue>
            </xacml-context:Attribute>
          </xacml-context:Resource>
```

The response for such a request could be

```
        <xacml-context:Response>
             <xacml-context:Result ResourceId="Level-6_ContentID">
         <xacml-context:Decision>Permit</xacml-context:Decision>
             </xacml-context:Result>
             <xacml-context:Result ResourceId="Level-3_ContentID">
         <xacml-context:Decision>Permit</xacml-context:Decision>
             </xacml-context:Result>
         </xacml-context:Response>
```

Note: The response contains <Result>s for both the level 6 content ID and the level 3 content ID.

Error Conditions

The following error conditions are identified:

1. urn:olca:1.0:status:ContentID_HIERARCHY_NOT_SUPPORTED – PDP does not support hierarchical content representation. Thus, the PEP will have to get authorizations without using hierarchical suffixes.
2. urn:olca:1.0:status:ContentID_MISSING_HIERARCHY_DATA – PDP is missing the hierarchy data for this particular resource ID.
3. urn:oasis:names:tc:SAML:2.0:status:ResourceNotRecognized – PDP may respond with this status code if it does not realize that hierarchical suffixes are used along with content ID. The PEP may try the request again without the hierarchical suffixes.
4. urn:olca:1.0:status:ContentID_TOO_COARSE_GRAINED – PDP determines that it cannot give an authorization decision at the current level of the given content ID. It needs more granular content ID.

If the decision is provided at a level other than the one in the request, the PDP will use the following informational codes.

1. urn:olca:1.0:status:AUTHZD_AT_HIGHER_LEVEL – means that the content is actually authorized at a higher level than what is given in the request
2. urn:olca:1.0:status:AUTHZD_AT_LOWER_LEVEL – means that the content is actually authorized at a lower (more detailed) level than what is given in the request

All the codes are to be used in the <Status> element of the <Result> element. Examples are given below.

```
       <xacml-context:Response xmlns:xacml-
context="urn:oasis:names:tc:xacml:2.0:context:schema:os">
<xacml-context:Result ResourceId="Level-1-ContentID">
    <xacml-context:Decision>Deny</xacml-context:Decision>
    <xacml-context:Status>
        <xacml-context:StatusCode
Value="urn:olca:1.0:status:ContentID_TOO_COARSE_GRAINED" />
    </xacml-context:Status>
</xacml-context:Result>

<xacml-context:Result ResourceId="Level-2-ContentID\*">
    <xacml-context:Decision>Permit</xacml-context:Decision>
    <xacml-context:Status>
```

```
        <xacml-context:StatusCode Value="urn:olca:1.0:status:
AUTHZD_AT_HIGHER_LEVEL" />
    </xacml-context:Status>
</xacml-context:Result>
<xacml-context:Result ResourceId="Level-6-ContentID">
    <xacml-context:Decision>Permit</xacml-context:Decision>
    <xacml-context:Status>
        <xacml-context:StatusCode Value="urn:olca:1.0:status:
AUTHZD_AT_LOWER_LEVEL" />
    </xacml-context:Status>
</xacml-context:Result>
        </xacml-context:Response>
```

Note: It is assumed that both the PDP and PEP share the same hierarchy data, and that given two content IDs that are related, each entity can create the path from one to the other.

Requirements

1. PDP implementations MUST recognize hierarchy suffixes. If the PDP does not support hierarchies for evaluating authorization decisions, it MUST respond with a status code of urn:olca:1.0:status:ContentID_HIERARCHY_NOT_SUPPORTED.
2. If the PDP is missing the hierarchy data for the given resource ID, and thus cannot support the request, then the PDP MUST respond with a status code of urn:olca:1.0:status:ContentID_MISSING_HIERARCHY_DATA.
3. If the response contains a status code of urn:olca:1.0:status:ContentID_HIERARCHY_NOT_SUPPORTED or urn:olca:1.0:status:ContentID_MISSING_HIERARCHY_DATA, then the PEP MUST try the request without hierarchical suffixes.
4. If authorizing at a higher level, then the PDP MUST include the information status code of urn:olca:1.0:status:AUTHZD_AT_HIGHER_LEVEL.
5. If authorizing at a higher level, the PDP MUST include <Result>s for both the requested content ID and the higher level content ID.
6. If authorizing at a lower level, then the PDP MUST include the information status code of urn:olca:1.0:status:AUTHZD_AT_LOWER_LEVEL.
7. If authorizing at a lower level, then the PDP MUST include <Result>s for both the requested content ID and the lower level content ID.

7.5.5 Content Streaming Query

This query is to be used by SPs to check if a particular content can be streamed by the MSO. SP may use this query only if the 'onNet' attribute was sent with the authentication response. If the content is available at the MSO, SP must use the HTML mark-up returned for streaming the content. If the MSO cannot stream this content, SP continues to stream the content from their servers.

The request and response use JSON structures. This document uses schema constructs as defined in [RFC 7519].

7.5.5.1 Request

```
The JSON schema for the request is as follows.
{
   "id" : "OLCA_ContentStreamingQueryDef"
   "type" : "object",
   "description" : "Represents a request for querying whether or not the MSO
can stream a particular content.",
   "properties" :  {
      "resourceID" : {"type" : "string", "required" : "true"},
      "resourceScheme" : {"type" : "string",  "required" : "true"}
      "subjectIdentifier" : {"type" : "string",  "required" : "true"}
      "spIdentifier" : {"type" : "string",  "required" : "true"}
      "clientDetails" : {"type" : "object",
"properties" : {
   "ipAddress" : { "type" : "string", "required" : "true" },
   "screenResolution" : { "type" : "string", "required" : "optional" },
```

```
    "mediaResolution" : { "type" : "string", "required" : "optional" },
    "clientPlatform" : { "type" : "string", "required" : "optional" },
    "supportedDRMs" : { "type" : "array", ", "required" : "optional",
  "items" : {
      "drmIdentifier" : { "type" : "string", "required" : "required" }
}
     }
       }
    }
}
```

Table 1 describes the fields in the above request.

Table 1 - Fields in Content Streaming Query

Field Name	Description
resourceID	A string that uniquely identifies the content. The scheme used for the identifier given here is not defined by this specification, but the next field (resourceScheme) can be used to convey that.
resourceScheme	The scheme or namespace that the resourceID belongs to. For example, this could be 'EIDR'.
subjectIdentifier	An identifier for the user that the SP got from the IdP when the user was authenticated
spIdentifier	Service Provider Identifier – as known to the MVPD. This typically is the Entity ID in SAML terms.
ipAddress	Client's IP address
screenResolution	Mostly applicable for mobile devices, it is used to convey the maximum supported resolution by the client device. Value must be in the format of *width*height*. More specifically, the width and height are numeric integers, separated by a '*' (star) character.
mediaResolution	In case the resourceID does not identify the resolution of the content being requested, this filed can be used to communicate the requested content resolution. Valid values are *width*height, 360p, 720p,* and *1080p.*
clientPlatform	A descriptive string that identifies the client platform. Valid values are User-Agent: <<User-Agent string>> iOS Major.Minor Android Major.Minor Major and Minor are integer numbers, together denoting the version of the operating system.
supportedDRMs	An array of DRMs the client is ready to support. Mostly applicable to mobile platforms. Following is a standardized list urn:olca:1.0:drm:adobe urn:olca:1.0:drm:oma urn:olca:1.0:drm:marlin urn:olca:1.0:drm:playready urn:olca:1.0:drm:widevine

7.5.5.2 Response

The response schema is as follows:

```
{
   "id" : "OLCA_ContentStreamingResponseDef"
   "type" : "object",
   "description" : "Represents a response for content streaming query.",
   "properties" :  {
      "status" : {"type" : "string", "required" : "true"},
      "urlInfo" : { "type" : "object", "description" : "Will be included if the
status is URL",
 "required" : "false",
 "properties" : {
      "url" : {"type" : "string",  "required" : "true"},
      "token" : {"type" : "string",  "required" : "false"}
      },
      "htmlMarkup" : {"type" : "string", "required" : "false",
"description" : "Contains XML encoded HTML markup to be included in SP's page"
      }
   }
}
```

Table 2 describes the fields in the above request.

Table 2 - Fields in Content Streaming Response

Field Name	Description
status	Represents status of the response. Valid values are "NA": Content is not available with the MSO "URL": Content URL is provided in the urlInfo field. This implies that the SP can use its own Player and DRM client to access the client. "HTML": Actual HTML markup is included in the response. SP must use this HTML to stream the content. This implies that the MSO could not support the SP's player and DRM client and/or given client characteristics without using its own player and DRM client.
urlInfo	Must be present when the status is 'URL'
url	URL of the content (typically pointing to the Manifest file)
token	A token the SP's player must use when obtaining the license from MSO's license server
htmlMarkup	Must be present when the status is HTML. XML encoded HTML that the SP must include in its response to the user. The MSO must ensure that this markup is sufficient for the content to begin streaming from its own servers

If the status is NA, then SP uses its own content repositories.

If the status is URL, then SP MAY use its own player and DRM client to fetch content from the given URL.

If the status is HTML, SP MUST use the supplied HTML and MUST NOT use its own player or DRM client.

7.5.5.3 onNet flag

Content streaming query is intended primarily when the subscriber is on the MSO's network. To aid the SPs detect this condition, a new attribute is added to the set of attributes sent in the authentication response. The attribute name is

urn:cablelabs:olca:1.0:attribute:authz:onNet

Valid values are 'true' and 'false'. If the attribute is missing in the authentication response, then SP defaults it to 'false'.

SPs should use this flag to determine whether to make the content streaming query or not. If the value of the onNet attribute is set to 'true', then SP MUST make the content streaming query to determine if the MSO can stream the content.

7.5.5.4 Notes

One entity fetching content from another entity is a little challenging considering the DRM and transport protocols involved. A typical scenario includes a custom player and DRM client on the client platform (either browser or mobile devices), and streaming server and license server on the server side. A typical deployment will involve choice of specific technologies for player, DRM client, streaming server and license server – all working in tandem. For one entity to be able to stream content from another entity, the choices made on the client side (for the player and the DRM client) must be compatible with the choices made on the server side.

Content is, typically, made available through either a browser or a custom (SP's) application on a mobile platform. A browser typically can have multiple DRM plugins installed, whereas a mobile application is custom built with a chosen DRM client. In both cases, the player is custom built. Below is a tear down of possible permutations and combinations.

1. Browser
    a. If MSO determines that SP's player and DRM client are compatible with its server technology choices, the MSO can just respond with a URL pointing to the Manifest file (optionally also the token).
    b. If MSO determines that there is no match, it can respond with a complete HTML markup that loads its own player and uses its choice of DRM client.
2. Mobile applications
    a. If MSO determines that SP's player and DRM client are compatible with its server technology choices, the MSO can just respond with a URL pointing to the Manifest file (optionally also the token).
    b. If MSO determines that there is no match, then there is no way for the MSO to stream this content, so it responds with a status of NA.

7.5.6 Security Considerations

Appendix VI of this document covers many of the weaknesses in any back-channel communication – such as MITM, eavesdropping, replays, etc. To counter such threats, it is recommended that either TLS is used or message level signatures/encryption is used.

Requirements (Security)

1. AA/PDPs and SPs MUST support TLS 1.2 based communication as defined by [RFC 5246].

If message-level security is desired, it can be negotiated through a bilateral agreement.

TLS requirements

The following requirements apply if TLS is used.

1. AAs/PDPs MUST support TLS server functionality on port 443.
2. SPs MUST support TLS client functionality.
3. SPs MUST support TLS client authentication using digital certificates.
4. SPs and AAs/PDPs MUST support the certificate profiles and validation methods as defined in Section 8. The AA/PDP certificate used for the TLS connection MUST be the "signing" certificate included in the AttributeAuthorityDescriptor entry of metadata for AAs, and PDPDescriptor entry of metadata for PDPs.

SP MUST ensure that the AA/PDP certificate used in TLS is the same as the one found in the metadata for the entity ID to which it is connecting.

5. The SP certificate used for the TLS connection MUST be the "signing" certificate included in the SPSSODescriptor entry of the SP's metadata. AA/PDP MUST ensure that the SP certificate used in the TLS connection is the same as the one found in the metadata for the entity ID mentioned as issuer in the request.
6. If any of the above checks fail on the incoming request, AA/PDP MAY either drop the connection or respond with a status code of '*urn:oasis:names:tc:SAML:2.0:status:RequestDenied*'. Additional status codes and / or status messages MAY be used to communicate the problem details. (This specification does not define additional status codes or messages – it is up to the AA/PDP implementation.)
7. If any of the above checks fail on the response, SP MUST discard the message SP MUST also show an error message to the subscriber and deny access to content.
8. SP and AA/PDPs MUST support TLS_RSA_WITH_3DES_EDE_CBC_SHA and TLS_RSA_WITH_AES_128_CBC_SHA cipher suites.

Message-level security requirements

The following requirements apply if message level encryption/signature are used.

1. Requests MUST be signed using a digital certificate as defined in Section 8. The signature MUST cover the entire request.
2. AA/PDP MUST verify the signature against the certificate from the metadata of the entity whose ID is mentioned in the Issuer element of the request. The particular certificate used MUST be the 'signing' certificate from the SPSSODescriptor entry of the SP's metadata.
3. AA/PDP SHOULD ensure that IssueInstant value from request is within a valid window with respect to current time (determined at the discretion of individual implementations). Further AA/PDP SHOULD ensure that the ID attribute from the request is not a duplicate within the time window. These requirements ensure replay attack prevention.
4. In the response, AA/PDP MUST encrypt all attribute whose values it wants to protect from eavesdropping. The encryption key itself is encrypted using the public key found in the KeyDescriptor entry with use as 'encryption' under the SPSSODescriptor entry of the SP's metadata.
5. The entire response MUST be signed using a digital certificate as defined in Section 8.
6. SP MUST ensure that the response is signed with the certificate from metadata of the entity identified as the issuer of the response. The particular certificate used MUST be the 'signing' certificate of the AttributeAuthorityDescriptor entry of metadata for AAs and PDPDescriptor entry of metadata for PDPs using a digital certificate as defined in Section 8.
7. SP MUST ensure that the value of the InResponseTo is equal to the ID of the request it issued and is expecting a reply for.
8. SP MUST ensure that the IssueInstant of the response falls within a valid time window (determined at the discretion of individual implementations).
9. If any of the above checks fail on the incoming request, AA/PDP MAY either drop the connection or respond with a top-level status code of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level status code of "*urn:oasis:names:tc:SAML:2.0:status:RequestDenied*". Additional status codes and / or status messages MAY be used to communicate the problem details. (This specification does not define additional status codes or messages - it is up to the AA/PDP implementation.)
10. If any of the above checks fail on the response, SP MUST discard the message. SP MUST also show an error message to the subscriber and deny access to content.

7.5.7 Error Conditions

For any back-channel communication, it is possible that errors are encountered. This specification identifies possible error conditions and the required behavior for each.

1. TLS handshake (certificate validation) errors – AA/ PDP MUST drop the connection; SP MUST drop the connection, and deny access to content. Both parties MAY generate internal alerts to rectify the problem.
2. Invalid signature on the request – AA/ PDP MUST drop the message. AA/ PDP MAY generate a response. The top-level status code MUST be "*urn:oasis:names:tc:SAML:2.0:status:Requester*" with a second-level status code of "*urn:oasis:names:tc:SAML:2.0:status:RequestDenied*".
3. Invalid signature on the response – SP MUST drop the message. SP MUST also show an error message to the user and deny access to the content.
4. AttributeQueries are not supported – the AA MUST respond with a top-level status code of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level status code value of "*urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported*". The SP will have to proceed without attribute information, and rely only on fine-grained authorization through XACML query. Note, this is not the same as receiving an 'error' from the AA, in which case the SP MUST explicitly deny access to the content.
5. Invalid attribute names (either unknown or unsupported attributes) – AA will generate a response with the top-level status code of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level status code of "*urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue*". This second-level status code MUST also have child status elements that indicate which attribute is at fault. For example,

```
<Status xmlns="urn:oasis:names:tc:SAML:2.0:protocol">
    <StatusCode Value=" urn:oasis:names:tc:SAML:2.0:status:Requester">
<StatusCode Value=" urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue">
                <StatusCode Value="attributeName_a" />
          </StatusCode>
    </StatusCode>
</Status>
```

6. Unknown resource – PDP MUST generate a response with the top-level status code of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and a second-level, a second-level status code of "*urn:oasis:names:tc:SAML:2.0:status:ResourceNotRecognized*". The second-level status code MAY also have child status elements that indicate which resource is at fault. For example,

```
<Status xmlns="urn:oasis:names:tc:SAML:2.0:protocol">
    <StatusCode Value=" urn:oasis:names:tc:SAML:2.0:status:Requester">
          <StatusCode Value="
urn:oasis:names:tc:SAML:2.0:status:ResourceNotRecognized">
                <StatusCode Value="contentID_a" />
          </StatusCode>
    </StatusCode>
</Status>
```

7. Subject identifier is not found – the AA/ PDP MUST send a SAML response with the top-level status code value of "*urn:oasis:names:tc:SAML:2.0:status:Requester*" and the second-level status code of "*urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal*". It MAY include a status message. Upon receiving this message, the SP/PEP MUST invalidate his own session, and re-authenticate the subscriber with the AnP.
8. Internal error at the AA/ PDP – AA/ PDP MAY generate a response with the top-level status code of "urn:oasis:names:tc:SAML:2.0:status:Responder". A status message MAY be included, but that will be opaque to the SP. SP MUST deny access in such cases.
9. Request timeout – if the AA/ PDP does not respond (within a certain period of time), SP MUST treat this as an error condition, and MUST show an error message to the subscriber and deny access to the content.

7.5.7.1 Conveying status

A summary of the behavior is captured in this table.

SAML <Response> element provides a <Status> element for conveying status. The implementations MUST use this element to communicate success or error status.

In addition, the HTTP status code of 500 (Internal server error code) MAY be used to convey a system error.

The following table summarizes the different types of errors and how each party MUST behave for each.

Table 3 - Error Codes and Appropriate Response

Error Condition	Error Code in Response/Status/StatusCode	MVPD / SP Behavior
AA/PDP service is unavailable (not responding)	None / Not applicable	SP SHOULD timeout (the actual timeout period is SP defined) For attribute queries, SP MAY limit the UI to public content. For authorization queries, SP MUST deny access.
AA/PDP service encounters errors	urn:oasis:names:tc:SAML:2.0:status:Responder OR HTTP 500 response code	For attribute queries, SP MAY limit the UI to public content. For authorization queries, SP MUST deny access.
Security exceptions: TLS validations		AA/PDP service drops the connection, For attribute queries, SP MAY limit the UI to public content. For authorization queries, SP MUST deny access.
Security exceptions: digital signature validations fail	urn:oasis:names:tc:SAML:2.0:status:Requester (optional)	AA/PDP MAY drop the connection, or respond with the status code For attribute queries, SP MAY limit the UI to public content. For authorization queries, SP MUST deny access.
AA does not support attribute queries	urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported	SP MAY present 'unfiltered' content list to the subscriber.
SP queries for attributes not supported by AA	urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue	SP MAY continue to construct the UI without using the unsupported attribute for filtering
User needs to be re-authenticated	urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal	SP MUST destroy the current session and re-authenticate the user.
PDP does not recognize the content	urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:ResourceNotRecognized	SP MUST deny access to the content.

7.5.8 Unsolicited Authentication Response

IdPs may send SAML Response containing Authentication assertion in an unsolicited fashion. In such cases, IdP may include the RelayState parameter too. The value of this parameter is expected to be a content identifier. Since RelayState is outside of the Assertion that is signed by the IdP, SPs must follow the process to consume unsolicited Authentication responses.

If RelayState parameter is not present, the SP MUST log the user in, but the page it shows to the user is not defined.

If RelayState parameter is present,

a. The SP MUST log the user in.

b. Then the SP MUST verify if it can understand the content identifier (value of the RelayState parameter). If it understands the content identifier, the SP SHOULD verify with MSO's AzP through back-channel authorization that the user has access to that content. If the authorization fails, SP MUST NOT give the user access to the content.

c. If the SP does not understand the content, then the page it shows to the user is not defined.

7.5.8.1 Format of RelayState parameter

This specification RECOMMENDS the following format for RelayState – implementations may choose to use this or some other format.

The format uses XML syntax, contains a 'Resource' element, with a scheme attribute. The value of the 'Resource' element is an XML escaped content identifier.

```
<Resource scheme="urn:cablelabs:olca:1.0:EIDR">
"Actual content identifier"
</Resource>
```

The whole string is base-64 encoded before sending to the SP. Upon receiving the RelayState parameter, SP base-64 decodes the string, and then it may use that string to parse into an XML DOM object.

7.6 Standard identifiers

7.6.1 Standard attributes

Given below is a set of attributes and their semantics and usage.

Table 4 - URN Attributes, Interpretation and Support

Attribute URN	Interpretation	Support
urn:cablelabs:olca:1.0:attribute:subscriber:identifier	This attribute is used by the AnP to identify a subscriber's record. **Note**: Subject/NameID in authentication assertion could potentially have a transient value or a distinct value across SPs. Thus, Subject/NameID value may not be the optimal way to identify a subscriber. AnP may choose to use a different way to identify the subscriber than what they need to put in Subject/NameID. This attribute gives the AnPs this flexibility.	AnPs MUST support this attribute. There MUST be only one <AttributeValue> for this attribute. **Note**: AnPs MAY use the same value as that of Subject/NameID, but it is totally up to them. SPs MUST support this attribute. SPs MUST cache this information for the duration of the session and use this in all back-channel requests as detailed in this specification. **Note**: This attribute cannot be queried on. This attribute value is passed only through authentication assertion.
urn:cablelabs:olca:1.0:attribute:authz:channelID	Can contain one or more channel IDs. By providing this information, AnPs convey to SPs that the subscriber is allowed ONLY these channels. SPs MUST limit subscriber access to only these channels. However, if this attribute is not present in the authentication assertion, SP MUST issue a back-channel attribute query for this attribute. If the AnP does not support either attribute queries or this attribute in particular, then SP is allowed to show contents of all channels that SP caters to. However, SP MUST still get individual content authorization (XACML query) before allowing the subscriber to view a particular content.	AnP MAY support this attribute. SPs MUST support this attribute.

Attribute URN	Interpretation	Support
urn:cablelabs:olca:1.0:attribute:authz:maxMPAA	Designates the maximum MPAA rating that this subscriber is allowed to watch. By providing this information, AnPs mandate the SP to show content that has a MPAA rating of this attribute's value, or below. SPs MUST show content to the subscriber that is below or equal to this rating. However, if this attribute is not present in the authentication assertion, SP MUST issue a back-channel attribute query for this attribute. If the AnP does not support either attribute queries, or this attribute in particular, then SP is allowed to show content with any rating. However, SP MUST still get individual content authorization (XACML query) before allowing the subscriber to view a particular content. The allowed values for this attribute are as provided at http://www.mpaa.org/.	AnP MAY support this attribute. SPs MUST support this attribute.
urn:cablelabs:olca:1.0:attribute:authz:maxVCHIP	To be used similar to the MPAA rating attribute, but for the content rated via the VChip format. The allowed values for this attribute are as provided by FCC at http://www.fcc.gov/vchip/.	AnP MAY support this attribute. SPs MUST support this attribute.
urn:cablelabs:olca:1.0:attribute:authz:devicePermission	To be used to indicated whether the device is authorized for the SP. Valid values are "GRANTED" and "DENIED".	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.0:attribute:authz:deviceMessage	A corresponding message to any included urn:cablelabs:olca:1.0:attribute:authz:devicePermission value.	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.0:attribute:authz:deviceID	An opaque identifier for the device used by the subscriber.	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.0:attribute:authz:deviceType	An opaque type of the device used by the subscriber.	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.0:attribute:authz:onNet	A Boolean indicating that the subscriber is on MVPD's network.	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.1:attribute:authz:atHome	A Boolean indicating whether the user was at home when this authentication took place. A false value indicates that a user is known to be outside the home. This attribute MUST be omitted if it is unknown whether or not the user is in the home.	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.1:attribute:subscriber:profileType	Indicates what the provided "*urn:cablelabs:olca:1.0:attribute:subscriber:identifier*" refers to. Values include, but are not limited to, "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:device*" "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group*" and "*urn:cablelabs:olca:1.1:attribute:subscriber:profileType:user*".	AnP MAY support this attribute. SPs MAY support this attribute.

Attribute URN	Interpretation	Support
urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier	One or more persistent values used to uniquely identify the subscriber's distinguishing group(s) at the AnP/PDP. A "*groupType*" attribute may also be provided in the <AttributeValue> tag for each value with a namespace of "*urn:cablelabs:olca:1.1:attribute:subscriber".* The groupType value may be one of: "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:household*", "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:building*" "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:campus*", "*urn:cablelabs:olca:1.1:attribute:subscriber:groupType:municipalArea*", or any other unique URI.	AnP MAY support this attribute. SPs MAY support this attribute. Any support MUST include the possibility of having more than one value returned as well as support for the custom groupType attribute for each value.
urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser	A Boolean value indicating whether the authenticated profiled represents a "privileged user". A privileged user is a head of household, primary, or admin account of the household and is usually defined as a profile which has the capability of creating, modifying, deleting other profiles within the household, as well as managing parental controls for these profiles.	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.1:attribute:subscriber:country	Indicates the country of the user's home address. The format of the country is a two-letter code following the ISO 3166-1 alpha-2 standard as defined by [ISO 3166].	AnP MAY support this attribute. SPs MAY support this attribute.
urn:cablelabs:olca:1.1:attribute:subscriber:postalCode	Indicates the postal code of the user's home address. When used, the "urn:cablelabs:olca:1.1:attribute:subscriber:country" attribute must also be present. The format of the postal code is specific to the country of origin and outside the scope of this document.	AnP MAY support this attribute. SPs MAY support this attribute.

7.6.2 Standard obligation URNs

Table 5 lists a set of obligations indicating if their support is mandatory.

Table 5 - Obligation URNs and SP Support

Obligation URN	Interpretation	SP Support
urn:cablelabs:olca:1.0:obligations:log	The PEP MUST log the time, resource, action and decision before implementing the decision.	MUST
urn:cablelabs:olca:1.0:obligations:reauthn	The PEP MUST re-authenticate the user. After re-authentication, PEP MAY use this decision (instead of sending another XACML request).	MUST

7.7 OLCA Schema

The schema for extension elements defined earlier in this section is given below.

```
<schema targetNamespace="urn:cablelabs:olca:1.0"
xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:olca=
"urn:cablelabs:olca:1.0" version="1.0">

    <element name="Use" type="olca:UseType"/>
    <simpleType name="UseType">
        <restriction base="string">
```

```
            <enumeration value="OneTime"/>
        </restriction>
    </simpleType>

    <element name="TimeBounds" type="olca:TimeBoundsType"/>
    <complexType name="TimeBoundsType">
        <attribute name="NotBefore" type="dateTime" use="optional"/>
        <attribute name="NotOnOrAfter" type="dateTime" use="optional"/>
    </complexType>

    <element name="Validity" type="olca:ValidityType"/>
    <complexType name="ValidityType">
         <choice>
  <element ref="olca:TimeBounds"/>
  <element ref="olca:Use"/>
         </choice>
    </complexType>

   <attribute name="scheme">
      <simpleType>
        <restriction base="string">
            <enumeration value="urn:cablelabs:olca:1.0:EIDR"/>
            <enumeration value="urn:cablelabs:olca:1.0:MRSS"/>
            <enumeration value="urn:cablelabs:olca:1.0:CustomContentGUID"/>
            <enumeration value="urn:cablelabs:olca:1.0:CustomNetworkGUID"/>
        </restriction>
      </simpleType>
   </attribute>
</schema>
```

7.8 Scenario 3

In scenario 3, the MVPD acts as the SP, but the content is screened from the Content Provider (CP) (distinct from MVPD). The subscriber may not leave the SP's page, but the player used to display the content will use a URL pointing to the CP.

A primary requirement from a security perspective is to provide a way for the CP to verify that this is an authorized access (as the content URL could be requested by anyone on the internet). Figure 27 below depicts the flow.

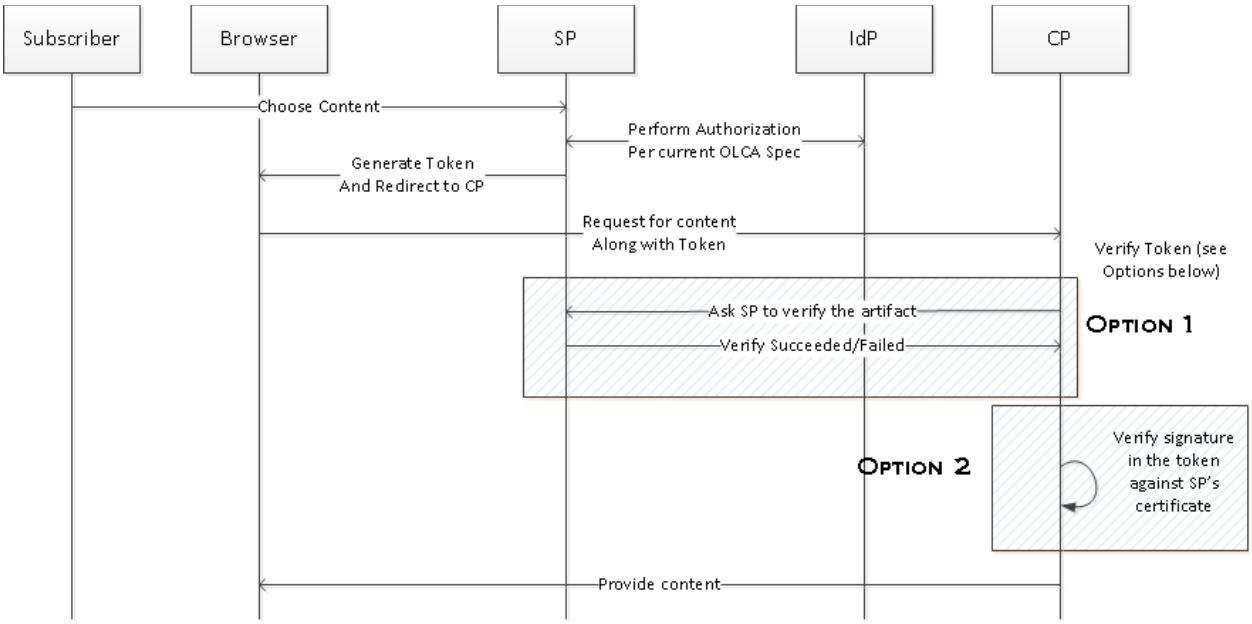

Figure 27 - Scenario 3 Flows

Figure 27 above shows two ways this can be achieved. In option 1, SP attaches an artifact to the URL used to fetch the content. In option 2, SP attaches a token to the content URL – this token is self-contained in the sense that CP can verify the signature on it. This specification supports both options.

7.8.1 Artifact

SP generates a unique artifact and stores it locally. The artifact must be one-time use only and be discarded after the CP queries for it. It is up to the implementations to select the uniqueness criteria, and the window of time within which they will be unique.

The format of the artifact is as per SAML, as stated in section 3.6.4 of [SAML 2.0 BINDINGS]. Implementations MUST follow section 3.6 of [SAML 2.0 BINDINGS]. After receiving the artifact in the request, the CP MUST make a SAML ArtifactResolve request to the issuer (found in the artifact). The response MUST contain an Assertion with XACMLAuthZDecisionStatement (described in Section 7.5.4.2 of this document). The response MUST contain one Result element with one Decision, of either Permit or Deny.

7.8.2 Token

Two token formats are supported – SAML and JWT.

If SAML is used, then HTTP Redirect Binding as described in section 3.4 of [SAML 2.0 BINDINGS], MUST be used. The SAML Assertion MUST contain a XACMLAuthZDecisionStatement with a Result element and a Decision of Permit. The CP MUST validate the incoming signature and the validity period of the assertion to be within allowed variance of the current time.

If JWT is used, then implementations MUST follow section 9.1 in [RFC 7519]. The payload section MUST be as defined below.

{

[M] "iss" : "SP Entity ID",

[M] "issueTime" : "Issued time in milliseconds",

[M] "exp" : "Expires time in milliseconds",

[M] "requestURL" : "URL used to request the content, without this token",

[M] "clientIP" : "IP address of the client device to which this token is issued"

}

[M] – Mandatory field

The complete JWT token is URL encoded and sent as either form encoded POST parameter or as a query parameter on the URL itself. The parameter name used MUST be 'jwt'.

When CP receives this JWT, it verifies that the request URL in the payload JSON matches the content URL (rest of the URL without the 'jwt' parameter) and that the signature is valid against the signing certificate of the 'iss' entity ID.

8 CERTIFICATE PROFILE AND VALIDATION

Digital certificates are used to support TLS mutual authentication for back channel authorization messaging and digital signatures for SAML messaging. Certificate profile and validation requirements in this section apply to the following certificates:

- TLS/SSL server certificates for AA/PDPs
- TLS/SSL client certificates for SPs
- Certificates for signing SAML messages at IdPs and SPs
- Certificate Profile

X.509 digital certificates [RFC 2459] MUST be used for digital signatures on SAML messages and to support TLS mutual authentication. All X.509 certificates MUST be signed by a trusted party. CableLabs operates a digital certificate public key infrastructure (PKI) that can be used for issuing OLCA certificates. The certificates MUST be profiled as described in Table 6.

Table 6 - Certificate Profile

Subject Name Form	C=<Country> O=<Company> CN=<FQDN> Additional fields may be present in the subject name. FQDN is the server's fully qualified domain name (e.g., server.example.com). Only a single FQDN is allowed in the CN field.
Intended Usage	These certificates are used to authenticate TLS handshake exchanges (and encrypt when using RSA key exchange) and digitally sign SAML messages.
Validity Period	Set by operator policy
Modulus Length	2048
Extensions	KeyUsage[c,m](digitalSignature, keyEncipherment) extendedKeyUsage[n,m] (id-kp-serverAuth, id-kp-clientAuth) authorityKeyIdentifier[n,m] (keyIdentifier=<subjectKeyIdentifier value from CA cert>)

8.1 Certificate Validation

Certificates MUST be verified as part of a certificate chain that chains up to a Trusted Root certificate. The chain MAY contain intermediate Certification Authority (CA) certificates. Receiving entities MUST support configuration of a Trusted Root certificate.

To ensure a high degree of trust, all OLCA certificates SHOULD be extended validation (EV) certificates. EV certificates are issued in conformance with the extended validation guidelines defined by the CA/Browser Forum. The extended validation guidelines contain a set of minimum requirements for the operations of certification authorities (CAs) that mostly govern the process of validating the identifying information that is to appear in an EV SSL certificate, but also establish requirements for several other aspects of a CA's operations, including: insurance coverage, revocation services, cryptographic key parameters, personnel qualification, etc.

Usually the first certificate in the chain is not explicitly included in the certificate chain that is sent to the receiving entity. In the cases where the first certificate is explicitly included, it MUST already be known to the verifying party ahead of time; and MUST NOT contain any changes to the certificate, with the possible exception of the certificate serial number, validity period and the value of the signature. If changes other than the certificate serial number, validity period and the value of the signature exist in the root certificate that was received in comparison to the known root certificate, the receiving entity MUST conclude that the certificate verification has failed.

Sending entities MUST include any intermediate CA certificates along with the entity certificate in the message being sent.

Receiving entities MUST build the certificate chain and validate the certificate according to the "Certificate Path Validation" procedures described in [RFC 2459]. In general, X.509 certificates support a liberal set of rules for determining if the issuer name of a certificate matches the subject name of another. The rules are such that two name fields may be declared to match even though a binary comparison of the two name fields does not indicate a match. [RFC 2459] recommends that certificate authorities restrict the encoding of name fields so that an implementation can declare a match or mismatch using simple binary comparison. Accordingly, the DER-encoded tbsCertificate.issuer field of a certificate MUST be an exact match to the DER-encoded tbsCertificate.subject field of its issuer certificate. An implementation MAY compare an issuer name to a subject name by performing a binary comparison of the DER-encoded tbsCertificate.issuer and tbsCertificate.subject fields.

8.2 Certificate Revocation

Certificate Revocation Lists (CRLs) MAY be checked as part of certificate path validation. The CRL profile and how a receiving entity obtains a CRL is not defined.

Appendix I Authentication Prototype

These samples are for conveying the concepts discussed here, they may not be strictly according to the respective schema. Also, these examples are not complete. Many important details, for example the security elements, are left out for clarity.

I.1 Authentication Request XML Example

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
Destination="http://idphost.idp.com/idpservice"
ForceAuthn="false" ID="_a0d0341caf7e80d80abea765" IsPassive="false"
ProtocolBinding="urn:oasis:names:tc:SAML:2.0:protocol"
IssueInstant="2010-06-16T20:35:30.582Z" Version="2.0">
 <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
    http://sphost.sp.com
 </saml:Issuer>
</samlp:AuthnRequest>
```

I.2 Authentication Response XML Example

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
Version="2.0"
ID="s2f6e4a0ae48e634e0b4c60b932ff7dd4773720bd8"
InResponseTo="_a0d0341caf7e80d80abea765"
IssueInstant="2010-06-16T20:35:27Z"
Destination="http://sphost.sp.com/spservice">

 <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
    http://idphost.idp.com
 </saml:Issuer>

 <samlp:Status xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol">
    <samlp:StatusCode  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
       Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
 </samlp:Status>

 <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
 ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600" IssueInstant="2010-06-
16T20:35:27Z" Version="2.0">

    <saml:Issuer>
       http://idphost.idp.com
    </saml: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="#s2cf10a567011a68cc3fc7badfab7a43dbb5335600">
          <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#"/>
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
          <ds:DigestValue>vGIIzERnMYQ339hJXb3m50cQg6c=</ds:DigestValue>
       </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>
       ###DigitalSignatureHere###
```

```
    </ds:SignatureValue>
    <ds:KeyInfo>
       <ds:X509Data>
          <ds:X509Certificate>
             ###CertificateHere###
          </ds:X509Certificate>
    </ds:X509Data>
    </ds:KeyInfo>
    </ds:Signature>

    <saml:Subject>
       <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:transient"
       NameQualifier="http://idphost.idp.com">
          ###transientidentifier###
       </saml:NameID>
       <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
       <saml:SubjectConfirmationData InResponseTo="_a0d0341caf7e80d80abea765"
NotOnOrAfter="2010-06-16T20:45:27Z"
       Recipient="http://sphost.sp.com/spservice"/>
    </saml:SubjectConfirmation>
    </saml:Subject>

    <saml:Conditions NotBefore="2010-06-16T20:25:27Z" NotOnOrAfter="2010-06-
16T20:45:27Z">
       <saml:AudienceRestriction>
          <saml:Audience>
             http://sphost.sp.com
          </saml:Audience>
       </saml:AudienceRestriction>
    </saml:Conditions>

    <saml:AuthnStatement AuthnInstant="2010-06-16T20:35:27Z"
SessionIndex="s2677ff62b195683e63e8ebb0d72093d6bbc71ce01"
SessionNotOnOrAfter="2010-07-16T20:35:27Z">
       <saml:AuthnContext>
          <saml:AuthnContextClassRef>
             ...
          </saml:AuthnContextClassRef>
       </saml:AuthnContext>
    </saml:AuthnStatement>
 </saml:Assertion>
</samlp:Response>
```

Appendix II Sample Authorization Messages

These samples are for conveying the concepts discussed here, they may not be strictly according to the respective schema. Also, these examples are not complete. Many important details, such as the security elements, are left out for clarity.

II.1 Implicit Transfer of Attributes Using Attribute Statement

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
   xmlns:xs="..." xmlns:xsi="..." ID="..." Version="2.0"
   IssueInstant="...">
   <saml:Issuer>https://idp.example.org/SAML2</saml:Issuer>
   <ds:Signature xmlns:ds="...">...</ds:Signature>
   <saml:Subject>...  </saml:Subject>
   <saml:Conditions>   ...  </saml:Conditions>
   <saml:AuthnStatement> ...  </saml:AuthnStatement>
   <saml:AttributeStatement>
     <saml:Attribute
       NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
       Name="urn:cablelabs:olca:1.0:attribute:authz:channelID"
       FriendlyName="allowedChannel">
       <saml:AttributeValue xsi:type="xs:string">
 Channel-1-unique-ID</saml:AttributeValue>
       <saml:AttributeValue xsi:type="xs:string">
 Channel-2-unique-ID</saml:AttributeValue>
     </saml:Attribute>
   </saml:AttributeStatement>
 </saml:Assertion>
```

II.2 Explicit Attribute Request Using SAML AttributeQuery

```
<samlp:AttributeQuery xmlns:saml="..." xmlns:samlp="..."
   ID="..." Version="2.0" IssueInstant="...">
   <saml:Issuer Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:entity">https://sp.example.com/SAML2</saml:Issuer>
   <saml:Subject>...     </saml:Subject>
   <saml:Attribute
     NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
     Name="urn:cablelabs:olca:1.0:attribute:authz:channelID"
FriendlyName="allowedChannel">
   </saml:Attribute>
   <saml:Attribute
     NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
     Name="urn:cablelabs:olca:1.0:attribute:authz:maxMPAA"
FriendlyName="maxRating">
   </saml:Attribute>
 </samlp:AttributeQuery>

Response to Explicit Attribute Request:
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
   xmlns:xs="..." xmlns:xsi="..." ID="..." Version="2.0"
   IssueInstant="...">
   <saml:Issuer>https://idp.example.org/SAML2</saml:Issuer>
   <ds:Signature xmlns:ds="...">...</ds:Signature>
   <saml:Subject>...  </saml:Subject>
   <saml:Conditions ...  </saml:Conditions>
   <saml:AttributeStatement>
     <saml:Attribute
       NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
       Name="urn:cablelabs:olca:1.0:attribute:authz:channelID"
FriendlyName="allowedChannel">
       <saml:AttributeValue xsi:type="xs:string">
```

```
 Channel-1-unique-ID</saml:AttributeValue>
       <saml:AttributeValue xsi:type="xs:string">
 Channel-2-unique-ID</saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute
       NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
       Name="urn:cablelabs:olca:1.0:attribute:authz:maxMPAA"
FriendlyName="maxRating">
       <saml:AttributeValue xsi:type="xs:string">
 PG-13</saml:AttributeValue>
     </saml:Attribute>
   </saml:AttributeStatement>
 </saml:Assertion>
```

II.3 Explicit Decision Query Using SAML/XACML XacmlAuthzDecisionQuery

```
<xacml-samlp:XACMLAuthzDecisionQuery>
   <saml:Issuer>MVPD</saml:Issuer>
   <xacml-context:Request>
 <xacml-context:Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-
category:access-subject">
    <xacml-context:Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
    DataType="http://www.w3.org/2001/XMLSchema#string">
<AttributeValue>TransientUserIdentifierGottenInAssertion </AttributeValue>
    </xacml-context:Attribute>
 </xacml-context:Subject>
 <xacml-context:Resource>
     <xacml-context:Attribute AttributeId="…:resource-id" DataType=" #string">
    <AttributeValue>SpecificContentID</AttributeValue>
     </xacml-context:Attribute>
 </xacml-context:Resource>
 <xacml-context:Action>
     <xacml-context:Attribute AttributeId="…:action-id" DataType="#string">
    <AttributeValue>VIEW</AttributeValue>
     </xacml-context:Attribute>
 </xacml-context:Action>
   </xacml-context:Request>
</xacml-samlp:XACMLAuthzDecisionQuery>
```

II.4 Explicit Decision Response Using SAML/XACML XacmlAuthzDecisionQuery

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
   xmlns:xs="..." xmlns:xsi="..." ID="..." Version="2.0"
   IssueInstant="...">
   <saml:Issuer>https://idp.example.org/SAML2</saml:Issuer>
   <ds:Signature xmlns:ds="...">...</ds:Signature>
   <saml:Subject>...  </saml:Subject>
   <saml:Conditions>   ...  </saml:Conditions>
   <xacml-samlp:XACMLAuthzDecisionStatement>
       <saml:Issuer>MVPD</saml:Issuer>
       <xacml-context:Response>
             <xacml-context:Result ResourceId="SpecificContentID">
        <xacml-context:Decision>Deny</xacml-context:Decision>
 </xacml-context:Result>
       </xacml-context:Response>
    </xacml-samlp:XACMLAuthzDecisionStatement>
</saml:Assertion>
```

II.5 Privileged-Account, Unprivileged-Account and Group-Account Representation

Some AnPs may support multiple user accounts under a single subscription, such as with a household or another business relationship. These accounts are sometimes divided into Primary (head of household) and Sub (child) Accounts. Note that some AnPs may only allow a single Primary Account while others may allow multiple Primary Accounts.

Primary Accounts are usually considered "privileged" accounts, whereas, the Sub-Accounts are usually considered "unprivileged". The *urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser* attribute of the <AttributeStatement> is used to reflect this. The concept of privileged users is defined in Sections 6.7.8 and 7.5.2.

When a user is authenticated by the virtue of user's location, for example at home or on a campus, the AnP cannot accurately identify an individual user profile. Likewise the AnP cannot determine whether or not the request is from an individual who should receive the authority normally granted to a privileged user. In this situation, the AnP can identify the subscriber using the "group-account-id" in the <AttributeStatement>. This "group-account-id" will be provided as the *urn:cablelabs:olca:1.0:attribute:subscriber:identifier* with a *urn:cablelabs:olca:1.1:attribute:subscriber:profileType* of *urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group*. Likewise the "group-account-id" will also be present in the list of *urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier* attributes provided. A "group-account-id", (sometimes referred as a "household-account-id") represents the group of all users in a household, building, campus, or municipal area as indicated in the "urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier" attribute. A "group-account-id" does not represent a specific user. For a household group type, it is usually, but not limited to, a hash of the subscriber-account-number indicated on the billing statements of the subscription.

During the initial establishment of an end-user's subscription, there may not yet be Primary or Sub-Accounts registered. In order to facilitate authentication, the AnP may choose to authenticate the end-user using Home-Based-Authentication while specifying the "group-account-id" as privileged. This situation may persist until the privilege is removed by the end-user through actions, such as establishing:

- a PIN to be used as described in section 6.7.10 - PIN Authentication and Appendix IV
- a Primary Account in the setup for multiple user accounts

This will reduce the friction for on-boarding new subscribers as well as eliminate any extra effort for an end-user who chooses not to utilize privilege restrictions, like parental controls.

Usage of privileged, unprivileged, and group accounts can also aid the following scenarios:

1. PIN Overrides: An unprivileged user that authenticated using home-based authentication or an unprivileged sub-account may seek permission from the privileged account holder to watch privileged content. Using the subject exposed in the <AttributeStatement>, the SP may run authentication flows with PIN overrides, as described in Section 6.7.10 - PIN Authentication and Appendix IV.
2. Analytics and Audience Measurement: When a privileged-account or an unprivileged-account holder is authenticated, the group-account-id exposed in the <AttributeStatement> can be used for linking these accounts together, which may be useful in analytics and audience measurements use-cases.
3. To apply Concurrent-Stream-Limiting Rules: Based on the values of *urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser*, *urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier*, *urn:cablelabs:olca:1.1:attribute:subscriber:groupType*, and subject exposed in the <NameID> Attribute, an SP may apply a certain set of stream limiting rules as per their contractual agreements with the MVPD.

Examples:

Account Identifiers:

Privileged-account: *privileged_account_id*

Unprivileged-account: *unprivileged_account_id*

Group-account: *group_account_id*

Device Identifier: *device_id*

1. Privileged Account Authenticated:

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
 ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600"
 IssueInstant="2010-06-16T20:35:27Z" Version="2.0">
    <saml:Issuer>...</saml:Issuer>
    <ds:Signature>...</ds:Signature>
    <saml:Subject>
          <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">
                privileged_account_id
           </saml:NameID>
    </saml:Subject>
    <saml:Conditions>…..</saml:Conditions>
 <saml:AuthnStatement>...</saml:AuthnStatement>
 <saml:AttributeStatement>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.0:attribute:subscriber:identifier">
       <saml:AttributeValue>
          privileged_account_id
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser">
       <saml:AttributeValue>
          true
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:profileType">
       <saml:AttributeValue>
          urn:cablelabs:olca:1.1:attribute:subscriber:profileType:user
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier">
       <saml: AttributeValue
      xmlns:olcasub="urn:cablelabs:olca:1.1:attribute:subscriber"
 olcasub:groupType="urn:cablelabs:olca:1.1:attribute:subscriber:groupType:house
hold">
          group_account_id
       </saml:AttributeValue>
    </saml:Attribute>
 </saml:AttributeStatement>
<saml:Assertion>
```

2. Unprivileged Account Authenticated

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
 ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600"
 IssueInstant="2010-06-16T20:35:27Z" Version="2.0">
    <saml:Issuer>...</saml:Issuer>
    <ds:Signature>...</ds:Signature>
    <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">
        unprivileged_account_id
      </saml:NameID>
    </saml:Subject>
    <saml:Conditions>…..</saml:Conditions>
```

```
 <saml:AuthnStatement>...</saml:AuthnStatement>
 <saml:AttributeStatement>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.0:attribute:subscriber:identifier">
       <saml:AttributeValue>
          unprivileged_account_id
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name=" urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser">
       <saml:AttributeValue>
          false
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:profileType">
       <saml:AttributeValue>
          urn:cablelabs:olca:1.1:attribute:subscriber:profileType:user
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier">
       <saml: AttributeValue
      xmlns:olcasub="urn:cablelabs:olca:1.1:attribute:subscriber"
 olcasub:groupType="urn:cablelabs:olca:1.1:attribute:subscriber:groupType:household">
          group_account_id
       </saml:AttributeValue>
    </saml:Attribute>
 </saml:AttributeStatement>
<saml:Assertion>
```

3. User Authenticated with Home-Based-Authentication

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
 ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600"
 IssueInstant="2010-06-16T20:35:27Z" Version="2.0">
    <saml:Issuer>...</saml:Issuer>
    <ds:Signature>...</ds:Signature>
    <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">
        group_account_id
      </saml:NameID>
    </saml:Subject>
    <saml:Conditions>…..</saml:Conditions>
 <saml:AuthnStatement>...</saml:AuthnStatement>
 <saml:AttributeStatement>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.0:attribute:subscriber:identifier">
       <saml:AttributeValue>
          group_account_id
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name=" urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser">
       <saml:AttributeValue>
          false
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:profileType">
       <saml:AttributeValue>
```

```
            urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group
          </saml:AttributeValue>
       </saml:Attribute>
       <saml:Attribute
        Name="urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier">
          <saml: AttributeValue
         xmlns:olcasub="urn:cablelabs:olca:1.1:attribute:subscriber"
 olcasub:groupType="urn:cablelabs:olca:1.1:attribute:subscriber:groupType:house
hold">
            group_account_id
          </saml:AttributeValue>
       </saml:Attribute>
 </saml:AttributeStatement>
<saml:Assertion>
```

4. User Authenticated as a result of being present on a College Campus

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
 ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600"
 IssueInstant="2010-06-16T20:35:27Z" Version="2.0">
    <saml:Issuer>...</saml:Issuer>
    <ds:Signature>...</ds:Signature>
    <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">
        group_account_id
      </saml:NameID>
    </saml:Subject>
    <saml:Conditions>…..</saml:Conditions>
 <saml:AuthnStatement>...</saml:AuthnStatement>
 <saml:AttributeStatement>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.0:attribute:subscriber:identifier">
       <saml:AttributeValue>
          group_account_id
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name=" urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser">
       <saml:AttributeValue>
          false
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:profileType">
       <saml:AttributeValue>
          urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group
       </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier">
       <saml: AttributeValue
      xmlns:olcasub="urn:cablelabs:olca:1.1:attribute:subscriber"
 olcasub:groupType="urn:cablelabs:olca:1.1:attribute:subscriber:groupType:campu
s">
          group_account_id
       </saml:AttributeValue>
    </saml:Attribute>
 </saml:AttributeStatement>
<saml:Assertion>
```

5. Device Account Authenticated

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
 ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600"
 IssueInstant="2010-06-16T20:35:27Z" Version="2.0">
    <saml:Issuer>...</saml:Issuer>
    <ds:Signature>...</ds:Signature>
    <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">
        device_id
      </saml:NameID>
    </saml:Subject>
    <saml:Conditions>…..</saml:Conditions>
 <saml:AuthnStatement>...</saml:AuthnStatement>
 <saml:AttributeStatement>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.0:attribute:subscriber:identifier">
      <saml:AttributeValue>
         device_id
      </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name=" urn:cablelabs:olca:1.1:attribute:subscriber:isPrivilegedUser">
      <saml:AttributeValue>
         false
      </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:profileType">
      <saml:AttributeValue>
         urn:cablelabs:olca:1.1:attribute:subscriber:profileType:device
      </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name=" urn:cablelabs:olca:1.0:attribute:authz:deviceType">
      <saml:AttributeValue>
         urn:cablelabs:olca:1.1:device-type:set-top-box
      </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
     Name="urn:cablelabs:olca:1.1:attribute:subscriber:groupIdentifier">
      <saml: AttributeValue
      xmlns:olcasub="urn:cablelabs:olca:1.1:attribute:subscriber"
 olcasub:groupType="urn:cablelabs:olca:1.1:attribute:subscriber:groupType:house
hold">
         group_account_id
      </saml:AttributeValue>
    </saml:Attribute>
 </saml:AttributeStatement>
<saml:Assertion>
```

AnPs that do not support multiple user accounts per household may expose either privileged-account-id or unprivileged-account-id as appropriate in the above occurrences.

Appendix III Sample Error Messages

When an IdP processes a request, it will either return a Response containing an Assertion or an appropriate StatusCode describing the exception. Additionally a second-level StatusCode may be given to further classify the exception so that the SP can invoke an appropriate remediation flow. The following status codes are used within this specification:

StatusCodes	Situation
urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue	The provided attribute or value was invalid. If one or more tertiary StatusCode elements are returned, their values will be the requested attributes that are invalid.
urn:oasis:names:tc:SAML:2.0:status:Responder -> urn:oasis:names:tc:SAML:2.0:status:NoPassive	The isPassive attribute was "true" in the AuthnRequest, however, the IdP needs to prompt the user for something to complete the request. This prompt could be for credentials or could be for other required interaction, such as accepting an updated EULA.
urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:RequestDenied	Validation of the request has failed. Additional status codes and / or status messages may be used to communicate the problem.
urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported	The IdP does not support the request being made.
urn:oasis:names:tc:SAML:2.0:status:Requester -> urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal	The Subject identifier is unknown. The SP/PEP must invalidate its own sessions.
urn:oasis:names:tc:SAML:2.0:status:Responder -> urn:cablelabs:olca:1.0:status:AuthnCanceled	The subscriber has elected to cancel the transaction.
urn:oasis:names:tc:SAML:2.0:status:Responder -> urn:cablelabs:olca:1.0:status:WrongIdP	The subscriber has clicked a link indicating a need to choose a different IdP.
urn:oasis:names:tc:SAML:2.0:status:Responder -> urn:cablelabs:olca:1.1:status:UnsupportedAuthnRestriction	The authentication attempt contains AuthnRestrictions not supported by the IdP. If one or more tertiary StatusCode elements are returned, their values will be the restriction(s) that are unsupported.
urn:oasis:names:tc:SAML:2.0:status:Responder -> urn:cablelabs:olca:1.1:status:UnsatisfiedAuthnRestriction	The authentication attempt contains AuthnRestrictions which cannot be satisfied by any authentication methods available. If one or more tertiary StatusCode elements are returned, their values will be the restriction(s) that could not be satisfied.

Some examples appear below:

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                ID="_f61dc2a5ed1d4fa59b8aa0cfeab1d24f3128ca377a"
                Version="2.0"
                IssueInstant="2015-05-08T15:10:13Z"
                Destination=" http://sphost.sp.com/spservice"
                InResponseTo="_544d1e06d1ee61084cd84c67d9be29d72b1bed7f27"
                >
    <saml:Issuer>http://idphost.idp.com</saml:Issuer>
    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
```

```
        …
    </ds:Signature>
    <samlp:Status>
        <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Responder">
            <samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:NoPassive" />
        </samlp:StatusCode>
        <samlp:StatusMessage>Passive authentication not
possible.</samlp:StatusMessage>
    </samlp:Status>
</samlp:Response>


<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                ID="_f61dc2a5ed1d4fa59b8aa0cfeab1d24f3128ca377a"
                Version="2.0"
                IssueInstant="2015-05-08T15:10:13Z"
                Destination=" http://sphost.sp.com/spservice"
                InResponseTo="_544d1e06d1ee61084cd84c67d9be29d72b1bed7f27"
                >
    <saml:Issuer>http://idphost.idp.com</saml:Issuer>
    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        …
    </ds:Signature>
    <samlp:Status>
        <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Requester">
            <samlp:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue">
                <samlp:StatusCode Value="attributeName_a" />
            </samlp:StatusCode>
        </samlp:StatusCode>
        <samlp:StatusMessage>An invalid attribute was
requested.</samlp:StatusMessage>
    </samlp:Status>
</samlp:Response>
```

Appendix IV Sample Authentication Scenarios

What follows are examples of some common authentication scenarios and their corresponding SAML messages:

IV.1 Authentication Scenario 1

Only authenticate users that are on a device that is physically in the home (at the service address).

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                    Destination="http://idphost.idp.com/idpservice"
                    ForceAuthn="false"
                    IsPassive="false"
                    ID="_a0d0341caf7e80d80abea765"
                    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:protocol"
                    IssueInstant="2010-06-16T20:35:30.582Z"
                    Version="2.0"
                    >
 …
 <saml:Issuer>
    http://sphost.sp.com
 </saml:Issuer>
 <samlp:Extensions>
      <olca:AuthnRestriction xmlns:olca="urn:cablelabs:olca
               type="urn:cablelabs:olca:1.1:AuthnRestriction:atHome">
         <olca:RestrictionValue xsi:type="xsd:boolean">
            true
         </olca:RestrictionValue>
      </olca:AuthnRestriction>
 </samlp:Extensions>
 …
</samlp:AuthnRequest>
```

IV.2 Authentication Scenario 2

Only allow a user to authenticate using a PIN that can override a rated R MPAA rating.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                    Destination="http://idphost.idp.com/idpservice"
                    ForceAuthn="false"
                    IsPassive="false"
                    ID="_a0d0341caf7e80d80abea765"
                    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:protocol"
                    IssueInstant="2010-06-16T20:35:30.582Z"
                    Version="2.0"
                    >
 …
 <saml:Issuer>
    http://sphost.sp.com
 </saml:Issuer>
 <saml:Subject>
    <saml:NameID
      SPNameQualifier="http://sphost.sp.com"
      Format="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent">
       2B8B407A-2CB7-11E5-B6C4-2741588FF78C
    </saml:NameID>
 </saml:Subject>
 <samlp:RequestedAuthnContext>
    <saml:AuthnContextClassRef>
       urn:cablelabs:olca:1.1:ac:classes:PIN
```

```
      </saml:AuthnContextClassRef>
   </saml:RequestedAuthnContext>
   <samlp:Extensions>
        <olca:AuthnRestriction xmlns:olca="urn:cablelabs:olca
                 type="urn:cablelabs:olca:1.1:AuthnRestriction:minMPAA">
           <olca:RestrictionValue xsi:type="xsd:string">
              R
           </olca:RestrictionValue>
        </olca:AuthnRestriction>
   </samlp:Extensions>
   …
</samlp:AuthnRequest>
```

IV.3 Authentication Scenario 3

Request the attributes to know both whether the user is at home as well as what the user's postal code is.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                    Destination="http://idphost.idp.com/idpservice"
                    ForceAuthn="false"
                    IsPassive="false"
                    ID="_a0d0341caf7e80d80abea765"
                    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:protocol"
                    IssueInstant="2010-06-16T20:35:30.582Z"
                    Version="2.0"
                    >
   …
   <saml:Issuer>
      http://sphost.sp.com
   </saml:Issuer>
   <samlp:Extensions>
        <olca:RequestedAttribute xmlns:olca="urn:cablelabs:olca">
           urn:cablelabs:olca:1.1:attribute:authz:atHome
        </olca:RequestedAttribute>
        <olca:RequestedAttribute xmlns:olca="urn:cablelabs:olca">
           urn:cablelabs:olca:1.1:attribute:subscriber:postalCode
        </olca:RequestedAttribute>
   </samlp:Extensions>
   …
</samlp:AuthnRequest>
```

IV.4 Authentication Scenario 4

The user was authenticated using a household account and is at home.

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
Version="2.0"
ID="s2f6e4a0ae48e634e0b4c60b932ff7dd4773720bd8"
InResponseTo="_a0d0341caf7e80d80abea765"
IssueInstant="2010-06-16T20:35:27Z"
Destination="http://sphost.sp.com/spservice">
   …
   <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
   ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600" IssueInstant="2010-06-
16T20:35:27Z" Version="2.0">
      …
      <saml:AttributeStatement>
         <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:uri"
Name="urn:cablelabs:olca:1.1:attribute:subscriber:profileType">
           <saml:AttributeValue xsi:type="xs:string">
      urn:cablelabs:olca:1.1:attribute:subscriber:profileType:group
```

```
         </saml:AttributeValue>
       </saml:Attribute>
       <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:uri"
Name="urn:cablelabs:olca:1.1:attribute:subscriber:groupType">
         <saml:AttributeValue xsi:type="xs:string">
  urn:cablelabs:olca:1.1:attribute:subscriber:groupType:household
         </saml:AttributeValue>
       </saml:Attribute>
       <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:uri"        Name="urn:cablelabs:olca:1.1:attribute:authz:atHome">
         <saml:AttributeValue xsi:type="xs:boolean">
            true
         </saml:AttributeValue>
       </saml:Attribute>
       …
    </saml:AttributeStatement>
  </saml:Assertion>
</samlp:Response>
```

Appendix V Best Practices for Handling User Assurance

User assurance refers to the ability to determine the end-user's identity with confidence. After authentication occurs, user assurance will degrade over time since it becomes less likely that the user who authenticated is the same user who is currently engaging the product. In its simplest form, managing user assurance is typically achieved by adjusting session lengths; however, doing so does not always provide the best user experience. This section provides an alternative view into managing user assurance.

In the context of a SAML federation, not all SPs will have the same business rules surrounding session lengths, nor will every SP be a Content Provider. Ultimately the SP is in the best position to determine the business rules surrounding the level of assurance required to access its own content. Likewise some business rules would be difficult or impossible for the IdP to manage, such as managing session lengths through user activity/inactivity. For example, an MVPD bill payment SP that is part of the federation may require a user to have authenticated within the last 20 minutes and continue to be active (performing clicks in the bill payment site) during that time to remain authenticated. Yet, a TVE Content Provider may require that a user have authenticated within 24 hours (for a Zero-Auth/Home Based Authentication) or within 90 days (if having used username and password credentials). Even within an SP's product, some individual site sections may require a higher level of assurance than others, such as an SP-controlled administrative section for parental control management or a payment/subscription management section for à la carte purchases. Because of this, the SP is generally the best party for managing these business rules and the SAML protocol provides an effective way for it to do so.

When an IdP handles an authentication request successfully, the SAML specification requires it to return an AuthnInstant attribute within the <AuthnStatement> section of the SAML response. This attribute contains the timestamp of the last time the end-user's identity was verified. This verification typically involves credential prompting, IP lookup, or other methods that do not rely on cached data to make an identity determination.

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
Version="2.0"
ID="s2f6e4a0ae48e634e0b4c60b932ff7dd4773720bd8"
InResponseTo="_a0d0341caf7e80d80abea765"
IssueInstant="2010-06-16T20:35:27Z"
Destination="http://sphost.sp.com/spservice">
  …
  <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="s2cf10a567011a68cc3fc7badfab7a43dbb5335600" IssueInstant="2010-06-
16T20:35:27Z" Version="2.0">
     …
     <saml:AuthnStatement AuthnInstant="2010-06-16T20:35:27Z"
SessionIndex="s2677ff62b195683e63e8ebb0d72093d6bbc71ce01"
SessionNotOnOrAfter="2010-07-16T20:35:27Z">
        …
     </saml:AuthnStatement>
  </saml:Assertion>
</samlp:Response>
```

Unless explicitly requested in the <AuthnRequest> through using a ForceAuthn="true" attribute, an IdP will typically give a response that contains a cached assertion from the last time the IdP verified the identity. When such a cached assertion is sent, it still contains an AuthnInstant timestamp of when the identity verification last occurred. This timestamp should be leveraged by the SP to manage the business rules surrounding user assurance.

Scenario: An end user **with an existing SP session navigates to an administrative SP site section that requires the user to have authenticated within the last 20 minutes.**

When the SP established its session initially, the SP must have stored the AuthnInstant that it received within its SP session. When the user accesses the administrative section, the SP will need to recall the AuthnInstant from session storage and compare the timestamp against the current time. If the difference is less than the required 20 minute freshness, the user is allowed to proceed into the administrative section unimpeded. If, however, the time difference is greater than 20 minutes, the SP will need to initiate a flow to re-validate the end user's identity.

The first step in the flow is to issue a passive <AuthnRequest> to the IdP. This is done by setting an isPassive="true" attribute on the <AuthnRequest> element. The purpose of issuing this passive request is to obtain the most recently

cached assertion from the IdP. It's possible that the end-user has re-authenticated between the last time the SP received an assertion and the current moment. If so, the <Assertion> returned by the IdP will have an updated AuthnInstant.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                    Destination="http://idphost.idp.com/idpservice"
                    ForceAuthn="false"
                    IsPassive="true"
                    ID="_a0d0341caf7e80d80abea765"
                    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:protocol"
                    IssueInstant="2010-06-16T20:35:30.582Z"
                    Version="2.0"
                    >
    …
</samlp:AuthnRequest>
```

The SP is directed to process any newly-received assertion as defined in Section 6.8.2. This includes storing the AuthnInstant that was received. Please note that if the SP only supports sessions where only a single user may be authenticated on a device at a time, special care must be taken to verify that the <Subject> present in the new assertion matches the <Subject> of the currently authenticated session. If they differ, the SP in instructed to terminate the original session and establish a new session using the <Subject> from the newly-received <Assertion>. Also note that the current session length may have to be updated if a SessionNotOnOrAfter timestamp is present in the newly-received <Assertion>. This MUST be done regardless of whether this new timestamp extends or even shortens the existing session.

If after issuing the passive request, an <Exception> is returned or the AuthnInstant is still not within the 20 minute limit, the SP is directed to issue another <AuthnRequest>. Instead of setting an isPassive="true" attribute, the SP will now set a ForceAuthn="true" attribute. This will instruct the IdP to attempt authentication without using any cached <Assertion> data. The result is that the end-user's identity will be re-verified using credential prompting or through some other means.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
                    Destination="http://idphost.idp.com/idpservice"
                    ForceAuthn="true"
                    IsPassive="false"
                    ID="_a0d0341caf7e80d80abea765"
                    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:protocol"
                    IssueInstant="2010-06-16T20:35:30.582Z"
                    Version="2.0"
                    >
    …
</samlp:AuthnRequest>
```

When the process is complete, the <Assertion> returned by the IdP will have an updated AuthnInstant. The SP is instructed to process this <Assertion> as previously noted and again with the same special care regarding the <Subject> and SessionNotOnOrAfter. The AuthnInstant is now within the required 20 minute freshness and the SP can direct the user into the administrative section.

Appendix VI Security Considerations

OLCA defines the architecture, messaging interface, and functionality to support Subscriber authentication and authorization for consuming online video content at different Service Providers (SPs). While this provides an interoperable and scalable system there are messaging interface security threats that should be considered and addressed with proper mitigation techniques.

The following sections discuss threats and proposed mitigation techniques for each of the main OLCA messaging interfaces and technologies.

VI.1 SAML Security Threats

OLCA uses the SAML 2.0 Web Browser SSO Profile to provide federated Subscriber authentication between Authentication Providers (AnPs) and SPs. Security threats associated with this application of SAML and mitigation techniques are discussed in the Security and Privacy Considerations for SAML 2.0 document [SAML 2.0 Security].

VI.2 SAML Security Features

To help protect SAML messaging SAML 2.0 defines optional requirements for digital signatures and encryption. These features support message integrity verification and message confidentiality. To properly address the threats related to the SAML authentication-messaging interface SAML messaging security features along with other security technologies, such as SSL/TLS may be needed.

VI.3 Subscriber Authentication Threats and Recommendations

In this section, we list the threats related to the authentication of the Subscriber by the Authentication Provider (AnP). The majority of these threats if not addressed would lead to theft of the Subscriber's identity and consequently theft and abuse of content across the ecosystem.

- **Theft/Compromise of Subscriber's authentication credentials:** Weak credentials such as username & passwords may be compromised or stolen in a variety of ways - through password guessing attacks (e.g., dictionary or brute force attacks), social engineering attacks, phishing and pharming attacks or through malware on the Subscriber's machine (e.g., key loggers). If the Subscriber's credentials are compromised or stolen then the malicious user(s) may have access to unauthorized content.
- **Sharing of authentication credentials:** the legitimate Subscriber may knowingly share their authentication credentials with other non-subscriber users such as their family and friends. While some amount of sharing may be acceptable (e.g., within a household), widespread sharing of credentials (e.g., in a college dormitory) would be unacceptable.
- **Subscriber Impersonation:** Both the above threats enable the malicious user to impersonate the legitimate Subscriber and access unauthorized content. Additionally, there are other more sophisticated techniques that could be used to impersonate the user. These include- man-in-the-middle (MITM) attacks, session and cookie hijacking, etc.
- **Device Impersonation:** Some AnPs may authenticate the Subscriber's device in addition to the user-level credentials. For example, they may allow the Subscriber to register a specific number of devices and then ensure that the user is accessing from a registered device in addition to verifying the user's credential such as a username/password. The AnP can potentially use several methods to identify the user's device including storing cookies, installing device certificates or other proprietary techniques that 'fingerprint' the user's device. Some of the above techniques may be more susceptible to device impersonation attacks.
- **Network Impersonation (Zero-Sign On):** In some cases the MVPD may authenticate the Subscriber based on the fact that the user is accessing the content from within the MVPD's access network (Zero Sign On). The determination that the user is accessing the service from within the MVPD's network would typically be made by matching the IP-address range among other things. In this category of attacks, a non-subscriber may be able to impersonate that he is accessing the service from within the MVPD's network and thus gain access to unauthorized content. For example, the malicious user may be able to take advantage of an unsecured Wi-Fi

network, setup a web-proxy on the user's home network and proxy authentication requests to the AnP through that web-proxy.

Even though the specification does not include any specific or minimum requirements on how the AnP should authenticate the Subscriber, we list some recommendations that the AnP should use to mitigate the threats identified above and protect content across the OLCA ecosystem, while enabling access to legitimate users.

- **Strong Authentication (2-Factor):** We recommend that the AnP can use 2-factor authentication such as one-time password credentials, PKI certificates, etc. in addition to username/passwords to authenticate the users. This will help mitigate the threats around theft and sharing of user credentials.
- **Device Registration:** This is an alternative approach that the AnP may use to mitigate against threats such as theft/compromise or willful sharing of user credentials. The AnP can allow the legitimate user to register a fixed number devices (e.g., up to 5). For authenticating the user, the AnP will not only verify the user's credentials such as username/password but also verify that they are accessing the content from a previously registered device. This model is currently being used by several content platforms to enable legitimate users access to content across all the devices in the household.

  As described above, we recommend that AnP use a technology that can mitigate against device impersonation attacks for the purpose of device identity and registration.
- **Fraud detection technologies:** In addition to above mitigating approaches, the AnP should implement means to detect anomalous or fraudulent patterns of usage. If a potential fraud is detected, then the user can be prompted for additional authentication. Some examples of fraudulent patterns are –
  - Same user logs in from distant geographical areas in a short amount of time – this would indicate a possible scenario that the user has shared their credentials with friends/family living in a different area.
  - Same user logs in from several different IP addresses in the short amount of time – this would indicate a possible scenario where the user's credentials are compromised.
  - High volume of devices being registered and unregistered against the user's account – this would again indicate that there is some out of ordinary sharing of user's credentials.
- **Other mitigating techniques:** We also recommend that AnP should consider implementing these additional techniques to prevent the malicious user from gaining access to the content.
  - **Limiting the simultaneous number of sessions:** this could be difficult to enforce across the ecosystem, so the AnP could look at limiting the number of authentication requests for each user in a given time period.
  - **Throttling unsuccessful authentication attempts:** In order to prevent dictionary and password guessing attacks, the AnP should limit the number of bad authentication attempts, either using throttling or by using lockouts.
  - **Blacklists:** Additionally, the AnP may want to check authentication attempts against blacklists of known bad IP-addresses, devices or users; that have been associated with prior fraudulent or malicious activities. Furthermore, there may be value in sharing the blacklists across the ecosystem, since the malicious users typically try to compromise multiple systems at the same time.

VI.4 Authorization Messaging Security Threats

Section 7 defines how an SP communicates with the Policy Decision Point (PDP) to determine if a Subscriber is authorized to access content. Authorization messaging occurs directly between the SP and PDP. This interface is sometimes referred to as the back channel, as it does not involve redirecting the Subscriber's browser (front channel).

Normally, authorization messaging traverses untrusted networks, such as the Internet, and is exposed to a number of potential attacks. The following threats exist for the back channel authorization-messaging interface:

Tampering. Tampering involves unauthorized modification of the message. This occurs when a message is intercepted, changed, and then forwarded to the receiving end. Authorization messages that are tampered with can cause the SP to display and allow access to content that is not authorized for a given Subscriber.

Technologies that verify the integrity of messages, such as digital signatures and message authentication (HMAC), can be used to prevent this threat.

Information Disclosure. Hackers can snoop network traffic and glean information from messages that enable them to attack the system, steal service, or obtain sensitive information about the sender or receiver. Information obtained from snooping authorization messages can be used to determine personal information about a subscriber, such as age, location and what the subscriber has been watching. This is an invasion of privacy. Encryption helps protect the confidentiality of network messaging.

Spoofing. Spoofing occurs when the sender or receiver pretends to be something they are not. For example, if a hacker pretends to be a trusted SP and requests authorization information for a Subscriber, he could determine personal information about a Subscriber such as age, location and what they have been watching. This is an invasion of privacy. Authentication technologies such as a public key infrastructure (PKI) or pre-shared keys can help prevent spoofing.

Denial of Service. Denial of Service (DoS) is an attempt to make services unavailable to intended users. This is typically done by overwhelming a server with requests so it cannot properly respond to normal traffic. This could include messages that are replayed by an unauthorized source. DoS attacks can occur on authorization web service interfaces, particularly at the AnP. One method to counter these types of attacks is to monitor traffic patterns and reject anything that does not look like normal traffic.

There are number of security technologies that can help prevent the attacks mentioned previously. These include TLS, IPsec, and message level security, which provide authentication, confidentiality, and integrity security services.

Appendix VII Acknowledgements

On behalf of the cable industry and our member companies, CableLabs would like to thank the following organizations for their participation in the development of this document:

Contributor	Company Affiliation
Anand Phatak	Adobe
Stuart Hoggan	CableLabs
Peter Rosenberg	NBC Universal
Timothy Ace	Synacor, Inc.
Brian Brinkmann	Turner

Since this specification is based on the [AUTH1.0] specification, the following individuals and organizations who participated in its development are also acknowledged.

Contributor	Company Affiliation
Oscar Marcia, Seetharama Rao Durbha, Stuart Hoggan, Dave Belt	CableLabs
Greg Ayre, Richard Woundy	Comcast
Hubert Le Van Gong	Neustar
Eric Fazinden	Ping Identity
Siddharth Bajaj	Symantec
Ty Akadiri	Time Warner Cable