RealMe

Identity Attribute Provider Web Service Specification

Version: 0.5 – DRAFT

Author:

Date: 28th November 2012

Document control

Revisions

Version	Date	Owner	Changes
0.1	10/09/2012	Venkat Maddali	Initial draft
0.2	11/09/2012	Venkat Maddali	Few minor updates based on feedback from business analysts
0.3	14/11/2012	Venkat Maddali	Message specification updated with SAMLv2.0 Attribute Query Profile
0.4	23/11/2012	Venkat Maddali	Few minor content changes based on DIA Authentication Standards Manager feedback
0.5	28/11/2012	Venkat Maddali	Added a section for consent token and validation

Reference Documents

Ref	Document	Version
1	RealMe Solution Principles	1.0
2	RealMe Architectural Artefacts	1.0
3	RealMe Glossary	1.0
4	RealMe Architectural Decisions	1.0
5	RealMe High-Level Architecture	1.1

Contributors

Person	Role
Steffen Sorensen	DIA Enterprise Architect – igovt, DIA
Venkat Maddali	DIA Solution Architect – igovt, DIA
Richard Bergquist	Primary Vendor Architect – Datacom
Mick Clarke	RealMe Architect – NZ Post

Approvers

The approvers for this and other documentation will be the major members of the *Joint Technology Governance Group* (JTGG), namely:

Name	Role	Approve (Y | N | Condition)	Version	Date
Mick Clarke	RealMe Architect			
Steffen Sorensen	igovt Architect			
Richard Bergquist	Primary Vendor Architect			

Table of Contents

1 Introduction

1.1 Overview

This document specifies RealMe messaging requirements for an integrating Identity Attribute Provider (IAP) to design and build an identity attribute web service(s).

1.2 Document Purpose

The purpose of this document is to describe the messaging interface sufficiently for the identity attribute providers to design and develop a web service for RealMe.

1.3 Audience

The audience for this document is intended to be both IT professionals and business stakeholders from New Zealand Post, DIA and Identity Attribute Provider(s).

1.4 Document references

The following NZ government standard references are used throughout this document:

Reference	Name	Description
NZ SAMS	New Zealand Security Assertion Messaging Standard. (June 2008 version 1.0 - ISBN 978-0-478-30344-5) http://ict.govt.nz/guidance-and-resources/standards-compliance/authentication-standards/new-zealand-security-assertion-messaging-standard	Prescribes messaging standards for communicating a range of security assertions (authentication, identity attributes and authorisation) in New Zealand government online services.
NZCIQ	New Zealand Government OASIS CIQ Profile. http://www.authentication.webstandards.govt.nz/new-zealand-government-oasis-ciq-profile/ (Note: this URL will change in early 2013)	A New Zealand Government Profile of the OASIS CIQ v3 Standard. The OASIS CIQ Standard is an international standard for Customer Information Quality. It deals specifically with data exchange of Customer Information by providing a set of predefined XML Schemas for data exchange structures.
NZISM	The New Zealand Information Security Manual v1.01. http://www.gcsb.govt.nz/newsroom/nzism.html	The New Zealand Information Security Manual (NZISM) provides up-to-date technical policy to assist government departments and agencies in securing information systems and the data stored in those systems

Reference	Name	Description
NZ SWS	The New Zealand Secure Web Services Standards (Note: this will be published in early 2013)	The New Zealand Secure Web Services Standards (NZSWS) provides standards for secure communication between web service providers and web service consumers.

Table 1 - NZ Government Standards

The following RealMe references are used throughout this document:

Reference	Name	Description
icms-msg-spec	Context Mapping Service Message Specification v0.18	Defines the Context Mapping Web Service interface for integrating clients with RealMe.
icms-int-guide	Context Mapping Service Integration Guide	A guide for integrating client to the Context Mapping Service using WS-Trust 1.4. Contains the specific technical steps and requirements for an integrator to follow.

Table 2 – RealMe references

The following third party references are used throughout this document:

Reference	Name	Description
ciq-3.0	Customer Information Quality v3.0 Specifications http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ciq	A XML based standard from OASIS to define a vocabulary to represent customer data, including identity related attributes.
xml-schema-datatypes	XML Schema Part 2: Datatypes. http://www.w3.org/TR/xmlschema-2	XML Schema datatypes from W3C.

Table 3 – third party references

The following SAML v2.0 references are used throughout this document:

Reference	Name	Description
saml-core-2.0-os	Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0	The core SAML v2.0 specification from OASIS.
saml-profiles-2.0-os	Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0	The profiles SAML v2.0 specification from OASIS.
saml-bindings-2.0-os	Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0	The bindings SAML v2.0 specification from OASIS.

Table 4 – SAML references

The following WS-* references are used throughout this document:

Reference	Name	Description
ws-trust-1.3	WS-Trust 1.3	WS-* specification from OASIS defining messaging elements for security token exchange.
ws-trust-1.4	WS-Trust 1.4	WS-* specification from OASIS defining extra messaging elements for security token exchange.
ws-addressing-1.0	WS-Addressing 1.0	WS-* specification from OASIS defining messaging elements for the SOAP header that define senders, recipients, and actions.

Table 5 – WS-* references

1.5 Glossary

Please refer to the RealMe Glossary [3] for a comprehensive glossary for the solution. In order to avoid duplication or errors all terms used in RealMe and igovt documentation are defined within that one document.

1.6 Assumptions

This document assumes the reader has: a working knowledge of the documents listed in section 1.4 and RealMe High Level Architecture (HLA) document.

1.7 Notion

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in IETF RFC 2119 [RFC2119].

1.8 Namespaces

The following table lists XML namespaces that are used in this specification. The choice of any namespace prefix is arbitrary and not semantically significant.

Prefix	**Namespace**	**Specification(s)**
saml	urn:oasis:names:tc:SAML:2.0:assertion	[saml-core-2.0-os]
samlp	urn:oasis:names:tc:SAML:2.0:protocol	[saml-protocols-2.0-os]
xpil	urn:oasis:names:tc:ciq:xpil:3	[ciq-v3.0]
xnl	urn:oasis:names:tc:ciq:xnl:3	[ciq-v3.0]
Ct	urn:oasis:names:tc:ciq:ct:3	[ciq-v3.0]

xal	urn:oasis:names:tc:ciq:xal:3	[ciq-v3.0]

Table 6: Namespace prefix and references

1.9 Conformance and Compliance

For an implementation or deployment to call itself compliant with this specification it MUST satisfy all aspects of this document marked as MUST as well as all conformance and specification requirements of the following specifications that are relevant to the functionality covered in this document:

- SAML v2.0
- WS-Trust 1.4
- Oasis CIQ Specifications Version 3.0

2 Message Pre-requisites

There are certain pre-requisites that MUST be completed prior to integration between RealMe and IAP. The following summary is provided to assist IAP integration with RealMe:

2.1 Mutual SSL certificates

Mutual SSL will be used to convey web service messages between RealMe and IAP. The IAP MUST provide the RealMe Operations Manager with their Mutual SSL certificate and RealMe SHALL provide each IAP with their Mutual SSL certificate for this exchange.

The mutual SSL certificate SHOULD be distinct from the message signing certificate. The algorithm used to generate key pairs MUST be RSA[1].

2.2 Signing Certificate

It MAY be possible to have distinct certificates for message signing. This specification SHALL only require a sole signing certificate and this MUST be provided when an IAP is integrated with RealMe.

The algorithm used to generate key pairs SHALL be RSA[1].

2.3 Transport

All messages MUST use SOAP 1.2 over HTTP 1.1 for transport. The HTTP 1.1 transport layer SHALL be encrypted. Both IAP Web Service and RealMe making the request MUST use X.509 certificates to identify themselves to each other. Transport encryption MUST be implemented with one of the following specifications:

- SSL v3
- TLS 1.0
- TLS 1.1
- TLS 1.2

The X.509 certificates used to secure the transport and provide mutual identification of the participating parties MUST be distinct from the SOAP messaging certificates. The X.509 certificates MUST carry RSA public keys. The process of certificate exchange is out of scope for this specification.

2.4 Server Synchronisation

The IAP server's system clock MUST be closely synchronised with a New Zealand Stratum One NTP Time Server.

This is REQURIED in order to:

1. Ensure the limited life time of the sender's message is not exceeded due to system time variations.
2. Ensure messages are honoured within a timeframe that is in common with RealMe and IAP.

Time tolerances in RealMe MAY be subject to change. Indicative tolerance values are +/- 1 minute for item 1 and up to 5 minutes for item 2.

[1] This will conform with [nzism-402].

It is RECOMMENDED that a NTP service be installed locally within an integrators infrastructure to meet this requirement.

Refer to http://www.ntp.org for strategies on how to implement time synchronisation from a NTP time server.

Refer to NZ e-GIF where UTC (MSL) is the Stratum One NTP time server, with NTP v4 as the delivery method over the internet.

3 Get Identity Attributes Status

This IAP web service functionality will provide the customer the ability to view their IAP attributes status through RealMe Dashboard (i.e. RealMe Account UI).

The sequence of messages is driven from the SAMLv2.0 Attribute Query Profile over SOAP binding for the request and response.

3.1 Message Flow

The following sequence diagram depicts the message flow between RealMe and IAP Web service to obtain the customer's identity status.

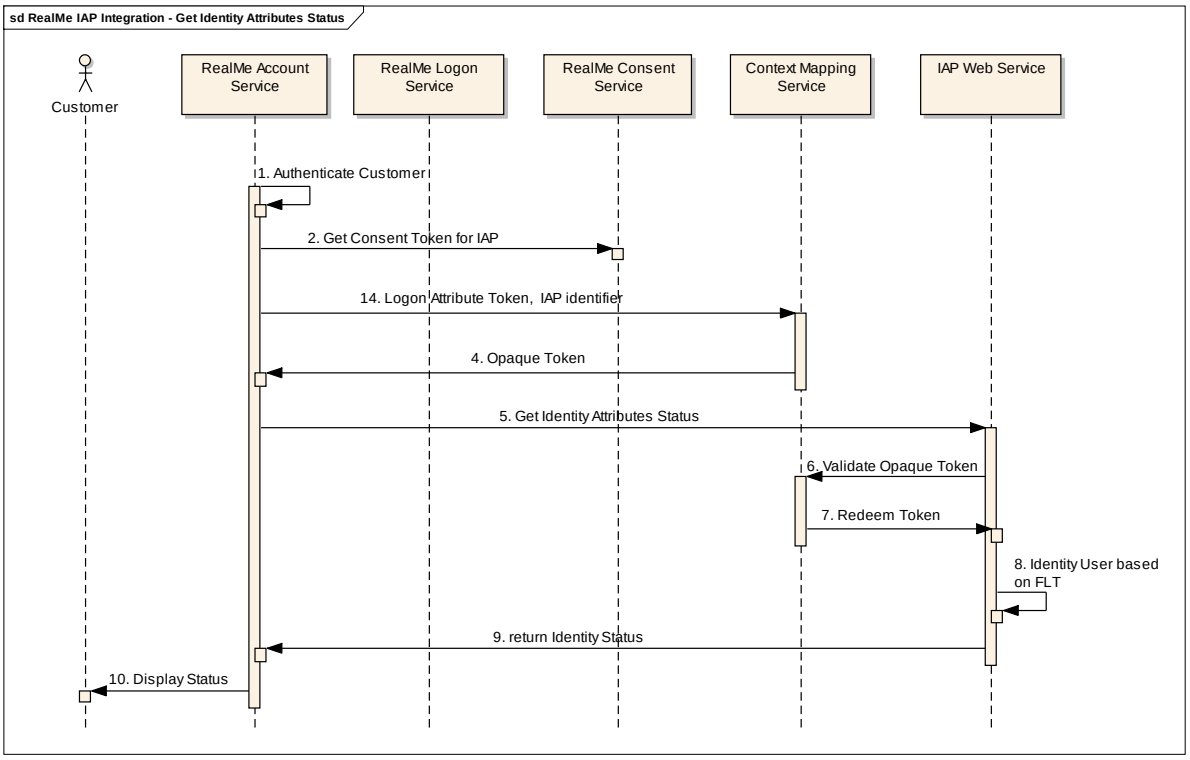

Figure 1 - Get Identity Attributes Status sequence flow

Message	Description
1. Authenticate Customer	The customer interacts with the RealMe account service to request a logon in order to gain access to protected resources. On successful customer authentication, the logon service issues a logon token to the account service. The logon token contains **logon attribute token** as an attribute, will be used for bootstrapping the interactions with IAP web services.
2. Get Consent Token for IAP	The RealMe account service obtains the customer's integration consent token for IAP from RealMe consent service.

Message	Description
3. Get Token for IAP	The RealMe account service invokes context mapping service to obtain a token that can be provided to IAP web service by passing logon attribute token and IAP identifier in the request.
4. Opaque Token	The context mapping service validates the logon attribute token and issues an opaque token for IAP web service.
5. Get Identity Status	The RealMe account service invokes IAP web service for the customer identity status by passing opaque token in the request. The IAP web service validates the request which includes signature verification.
6. Validate Opaque Token	The IAP web service retrieves opaque token from the request and invokes context mapping service with opaque token.
7. Redeem Token	The context mapping service validates opaque token and issues redeem token to IAP which contains FLT of the customer at IAP.
8. identify user based on FLT	The IAP web service checks identity based on FLT and applies required business/translation rules to determine status of the identity.
9. returns identity status	The IAP web service returns status to RealMe account service.
10. Display status	The ReallMe account service displays a page with the returned status to the customer.

Table 7 – Get Identity Attributes Status Flow

3.2 Request Message Elements

The root element contained in the SOAP body of a request message is <**samlp:AttributeQuery**>.
The < **samlp:AttributeQuery>** will contain the following elements and attributes:

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	ID	MUST be provided. An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.
<samlp:AttributeQuery>	Version	MUST be provided. The identifier is “2.0”. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	IssueInstant	MUST be provided. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	Destination	MUST be provided, Identity Attribute Service endpoint location. An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.
<samlp:AttributeQuery>	<saml:Issuer>	MUST be provided. It is REQUIRED and it MUST be in the format of an identity privacy domain of Client (i.e. Agency EntityID). An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.
<samlp:AttributeQuery>	<saml:Consent>	MUST be provided, consent token issued by the consent service. Refer to section 8 for consent token structure and validation. An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.
<samlp:AttributeQuery>	<saml:Subject>	MUST be provided and must be opaque token issued by the igovt context mapping service. An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.
<saml:Subject>	<saml:NameID>	<NameID> Format attribute value MUST be ***urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified***. The <NameID> MUST contain opaque token issued by the igovt context mapping service.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	<saml:Attribute>	Must be provided NameFormat attribute must be provided as ***urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified*** Name attribute must be provided as ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:{AttributesType}:Status*** Examples: <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name="urn:nzl:govt:ict:stds:authn:attribute:igovt:IVS:Identity:Status"> </saml:Attribute> An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.
<samlp:AttributeQuery>	<ds:Signature>	MUST be provided. An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.

Table 8 – Get Identity Attributes Status request message elements

3.3 Response Message Elements

The root element contained in the SOAP body of a successful response message MUST be a single <**samlp**:**Response>**. The <**samlp**:**Response>** MUST be signed with IAP private key. The <**samlp**:**Response>** will contain the following elements and attributes:

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:Response>	ID	SHALL return this attribute.
<samlp:Response>	Version	SHALL return this attribute. The identifier is “2.0”.
<samlp:Response>	IssueInstant	SHALL return this attribute.
<samlp:Response>	Destination	SHALL return this attribute, client identifier.
<samlp:Response>	InResponseTo	SHALL return this attribute, ID attribute of <samlp:AttributeQuery>

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:Response>	<saml:Issuer>	SHALL return this element. It is REQUIRED and MUST be an Identity Attribute Provider's service identifier (i.e. IAP EntityID).
<samlp:Response>	<samlp:Status>	SHALL return this element.
<samlp:Status>	<samlp:StatusCode> (Top Level)	If the request is valid the status code SHALL return `urn:oasis:names:tc:SAML:2.0:status:Success`. In other conditions SHALL return `urn:oasis:names:tc:SAML:2.0:status:Responder`
<samlp:StatusCode>	<samlp:StatusCode> (Second Level)	SHALL return this element if top level StatusCode is other than `urn:oasis:names:tc:SAML:2.0:status:Success`. Refer to section 7 for error codes.
<samlp:Status>	<samlp:StatusMessage>	SHALL return this element if top level StatusCode is other than `urn:oasis:names:tc:SAML:2.0:status:Success`. Refer to section 7 for error codes.
<samlp:Response>	<saml:Assertion>	If the request is valid and SHALL return this element (Status Assertion). The following are few notes re Status Assertion: • Refer to section 6 for Status Assertion common elements. • Subject can be optional • contains only status attributes
<saml:Assertion>	<saml:AttributeStatement>	SHALL return this element.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<saml:AttributeStatement>	<saml:Attribute> (for status Code)	*SHALL return this element.* *NameFormat* attribute SHALL be provided and value as ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic*** *Name* attribute SHALL be as following format ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:{AttributesType}:Status*** SHALL provide status value in <saml:AttributeValue></saml:AttributeValue> Examples: <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name="urn:nzl:govt:ict:stds:authn:attribute:igovt:IVS:Identity:Status"><saml:AttributeValue>VERIFY</saml:AttributeValue></saml:Attribute> Refer to Table 10 for status codes.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<saml:AttributeStatement>	<saml:Attribute> (for status Message)	*SHALL return this element.* *NameFormat* attribute SHALL be provided and value as ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic*** *Name* attribute SHALL be as following format ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:{AttributesType}:StatusMessage*** SHALL provide status value in <saml:AttributeValue></saml:AttributeValue> Examples: <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name="urn:nzl:govt:ict:stds:authn:attribute:igovt:IVS:Identity:StatusMessage"><saml:AttributeValue>Identity is in verified state. Expires on 10/10/2017</saml:AttributeValue></saml:Attribute>

Table 9 – Get Identity Attributes Status response message elements

Status Code	Status Condition
APPLY	IAP returns this status code if no identity exists or previous application is cancelled.
VER	IAP returns this status code if identity exists and verified.
INPRG	IAP returns this status code one of the following conditions: • An application is lodged • Application is in back office verification state
RENEW	IAP returns this status code if identity is either expired or cancelled.
INVALID	IAP returns this status code if an identity is fraud referral or inactive states.
CUST_ACT	IAP returns this status code if the customer is required to perform any action, example photo capture or submit address proof document etc.

Table 10 – IAP Status Codes for RealMe

4 Get Identity Attributes Assertion

RealMe invokes this IAP web service functionality in two contexts.

- **Context 1: View Identity Details at RealMe Dashboard**
  This IAP web service functionality will provide the customer the ability to view their IAP attributes details through RealMe Dashboard (i.e. RealMe Account UI).
- **Context 2: Providing Identity Details to RealMe Client**
  This IAP web service functionality will provide the customer the ability to provide their IAP attributes details to RealMe Client through RealMe Assertion Service.

The sequence of messages is driven from the SAMLv2.0 Attribute Query Profile over SOAP binding for the request and response.

4.1 Message Flow

4.1.1 Context1 - View Identity Details at RealMe Dashboard

The following sequence diagram depicts the message flow between RealMe account service and IAP Web service to obtain the customer's identity details.

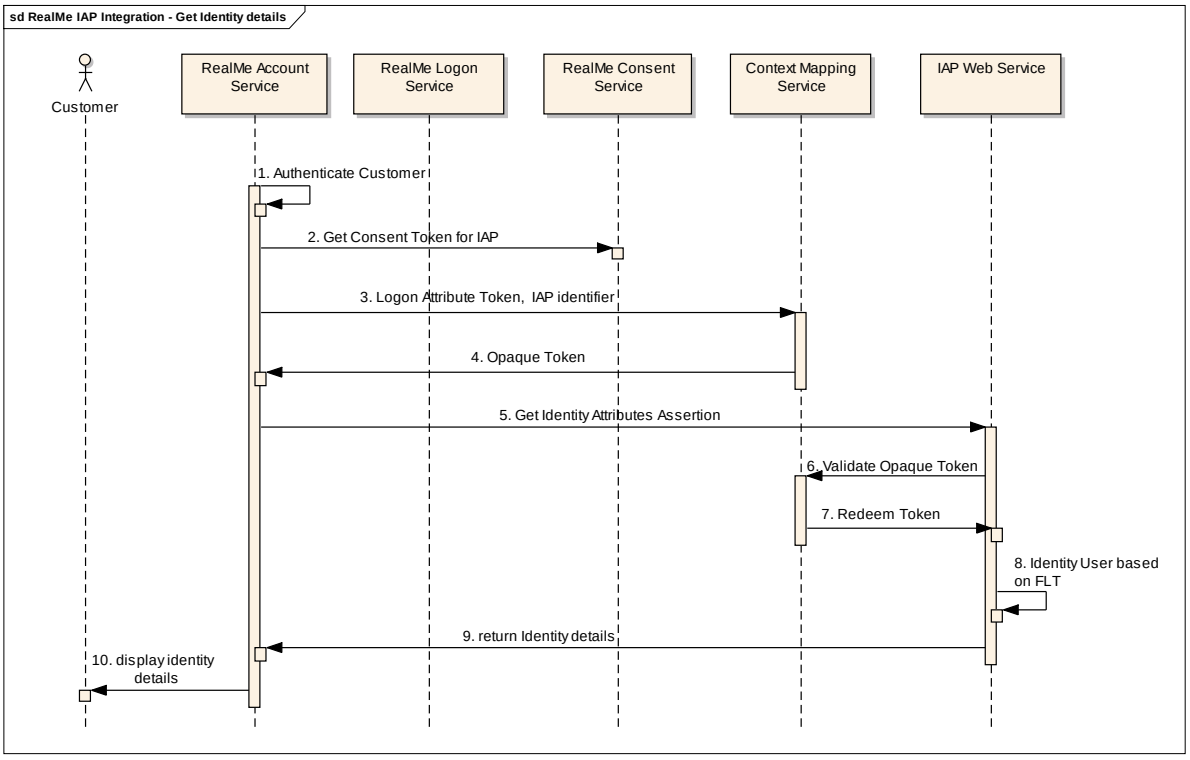

Figure 2 – View Identity Details at RealMe Dashboard sequence flow

Message	Description
1. Authenticate Customer	The customer interacts with the RealMe account service to request a logon in order to gain access to protected resources. On successful customer authentication, the logon service issues a logon token to the account service. The logon token contains logon attribute token as an attribute, will be used for bootstrapping the interactions with IAP web services
2. Get Consent Token for IAP	The RealMe account service obtains the customer's integration consent token for IAP from RealMe consent service.
3. Get Token for IAP	The RealMe account service invokes context mapping service for a token that can be provided to IAP web service by passing logon attribute token and IAP identifier in the request.
4. Opaque Token	The context mapping service validates the logon attribute token and issues an opaque token for IAP web service.
5. Get Identity Attributes Assertion	The RealMe account service invokes IAP web service for the customer identity details by passing opaque token in the request. The IAP web service validates the request which includes signature verification.
6. Validate Opaque Token	The IAP web service retrieves opaque token from the request and invokes context mapping service with opaque token.
7. Redeem Token	The context mapping service validates opaque token and issues redeem token to IAP which contains FLT of the customer at IAP.
8. identify user based on FLT	The IAP web service checks identity based on FLT.
9. returns identity status	The IAP web service returns identity details.
10. Display identity details	The RealMe account service displays a page with the returned identity details to the customer.

Table 11 – View Identity Details at RealMe Dashboard sequence flow

4.1.2 Context2 - Provide Identity Details to RealMe Client

The following sequence diagram depicts the message flow between RealMe assertion service and IAP Web service to allow the customer to assert their identity at the client.

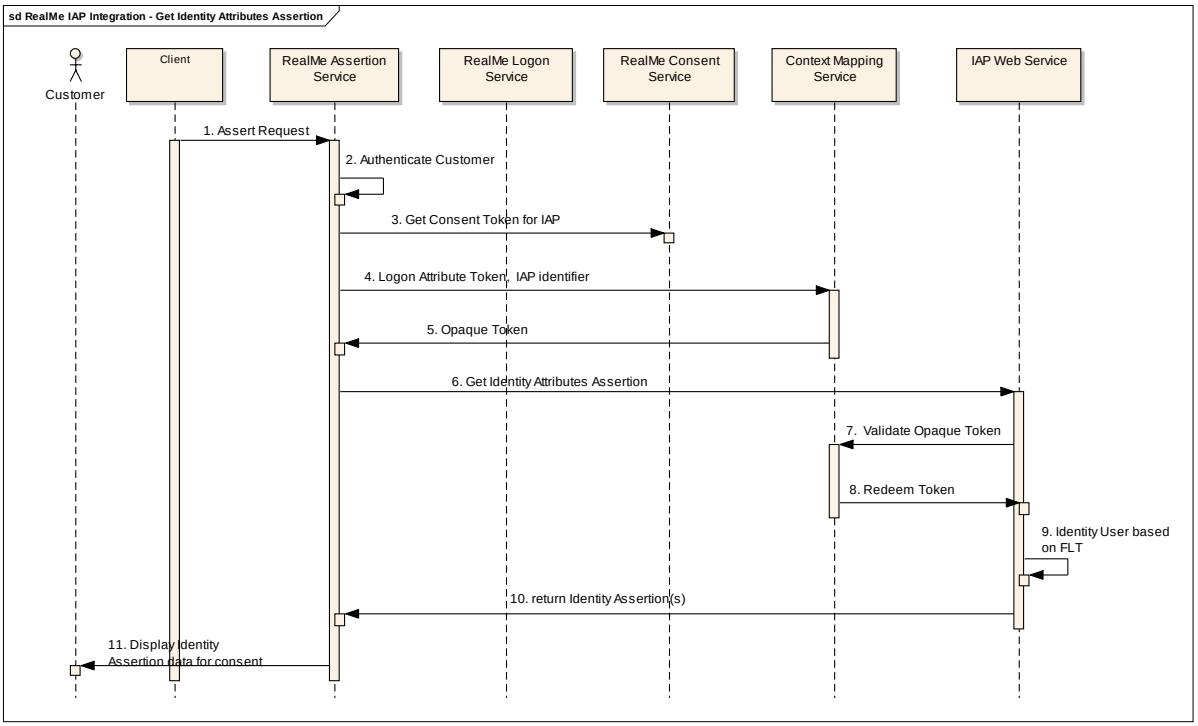

Figure 3 – Provide Identity Details to RealMe client sequence flow

Message	Description
1. Assert Request	The client redirects the customer to the RealMe assertion service for identity assertions.
2. Authenticate Customer	The RealMe assertion service authenticates the customer using logon service. On successful customer authentication, the logon service issues logon token to the account service. The logon token contains logon attribute token as an attribute, will be used for bootstrapping the interaction with IAP web services.
3. Get Consent Token for IAP	The RealMe assertion service obtains the customer's integration consent token for IAP from RealMe consent service.
4. Get Token for IAP	The RealMe assertion service invokes context mapping service to obtain a token that can be provided to IAP web service by passing logon attribute token and IAP identifier in the request.
5. Opaque Token	The context mapping service validates the logon attribute token and issues an opaque token for IAP web service.
6. Get Identity Attributes Assertion	The RealMe assertion service invokes IAP web service for the customer identity details by passing opaque token in the request. The IAP web service validates the request which includes signature verification.
7. Validate Opaque Token	The IAP web service retrieves opaque token from the request and invokes context mapping service with opaque token.
8. Redeem Token	The context mapping service validates opaque token and issues redeem token to IAP which contains FLT of the customer at IAP.

Message	Description
9. identify user based on FLT	The IAP web service checks identity based on FLT. Creates an assertion with identity details for the client.
10. returns identity assertion(s)	The IAP web service returns identity assertion(s).
11. Display identity assertion details for user consent	The RealMe assertion service displays a page with the returned identity details to the customer for their consent to release the assertions to the client.

Table 12 – Provide Identity Details to RealMe client flow

4.2 Request Message Elements

The root element contained in the SOAP body of a request message is <**samlp:AttributeQuery**>.
The < **samlp:AttributeQuery>** will contain the following elements and attributes:

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	ID	MUST be provided. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	Version	MUST be provided. The identifier is “2.0”. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	IssueInstant	MUST be provided. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	Destination	MUST be provided, Identity Attribute Service endpoint location. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	<saml:Issuer>	MUST be provided. The value MUST be identifier of identity attributes consumer. It is REQUIRED and it MUST be in the format of an identity privacy domain of Client (i.e. Agency EntityID). An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	<saml:Consent>	MUST be provided, consent token issued by the consent service. . Refer to section 8 for consent token structure and validation. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	<saml:Subject>	MUST be provided and must be opaque token issued by the igovt context mapping service. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<saml:Subject>	<saml:NameID>	<NameID> Format attribute value MUST be ***urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified***. The <NameID> MUST contain opaque token issued by the igovt context mapping service.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	<saml:Attribute>	MUST be provided NameFormat attribute MUST be provided as ***urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified*** Name attribute MUST be provided as ***urn:nzl:govt:ict:stds:authn:safeb64:attribute:{Provider}:{Service}:Assertion:{AttributesType}*** Examples: <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name="urn:nzl:govt:ict:stds:authn:safeb64:attribute:igovt:IVS:Assertion:Identity"> </saml:Attribute> <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name="urn:nzl:govt:ict:stds:authn:safeb64:attribute:NZPost:AVS:Assertion:Address"> </saml:Attribute>
<samlp:AttributeQuery>	<ds:Signature>	MUST be provided. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.

Table 13 - Get Identity Attributes Assertion request message elements

4.3 Response Message Elements

The root element contained in the SOAP body of a successful response message MUST be a single <**samlp**:**Response>**. The <**samlp**:**Response>** MUST be signed with IAP private key. The <**samlp**:**Response>** will contain the following elements and attributes:

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:Response>	ID	SHALL return this attribute.
<samlp:Response>	Version	SHALL return this attribute. The identifier is “2.0”.
<samlp:Response>	IssueInstant	SHALL return this attribute.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:Response>	Destination	SHALL return this attribute, client identifier. An Exception SHALL result if invalid or not provided.
<samlp:Response>	InResponseTo	SHALL return this attribute, ID attribute of <samlp:AttributeQuery>
<samlp:Response>	<saml:Issuer>	SHALL return this element. It is REQUIRED and MUST be an Identity Attribute Provider's service identifier (i.e. IAP EntityID).
<samlp:Response>	<samlp:Status>	SHALL return this element.
<samlp:Status>	<samlp:StatusCode>	If Identity attributes exist and verified the status code SHALL return `urn:oasis:names:tc:SAML:2.0:status:Success`. In other conditions SHALL return `urn:oasis:names:tc:SAML:2.0:status:Responder`
<samlp:StatusCode>	<samlp:StatusCode>	SHALL return this element if top level StatusCode is `urn:oasis:names:tc:SAML:2.0:status:Responder`.
<samlp:Status>	<samlp:StatusMessage>	SHALL return this element if top level StatusCode is `urn:oasis:names:tc:SAML:2.0:status:Responder`.
<samlp:Response>	<saml:Assertion>	SHALL return this element if the request is valid. The following are few notes re Identity Attributes Assertion: • Refer to section 6 for Status Assertion common elements. • Subject can be optional • contains identity attributes

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<saml:Assertion>	<saml:AttributeStatement>	The IAP SHALL return one instance of this element with an Name attribute of: Either `urn:nzl:govt:ict:stds:authn:safeb64:attribute:{IAP}:{Service}:Assertion:{AttributesType}` This element contains the identity attributes of the Customer encoded in [nzciq] and [ciq-3.0] XML formats. Only identity attributes that the Client has agreed with RealMe during integration SHALL be passed in the encoded [nzciq] XML. Refer to the Appendix section for individual IAP's identity payload.

Table 14 - Get Identity Attributes Assertion response message elements

5 Notify Release Consent

The operation will provide the ability for the IAP to audit the information that the customer consented to release an IAP identity assertion to a client. This can be an optional functionality for IAP.

The sequence of messages is driven from the SAMLv2.0 Attribute Query Profile over SOAP binding for the request and response.

5.1 Message Flow

The following sequence diagram depicts the message flow between RealMe assertion service and IAP Web service to notify release consent to IAP (extending the message flow depicted in Figure 3 in Section 4.1.2).

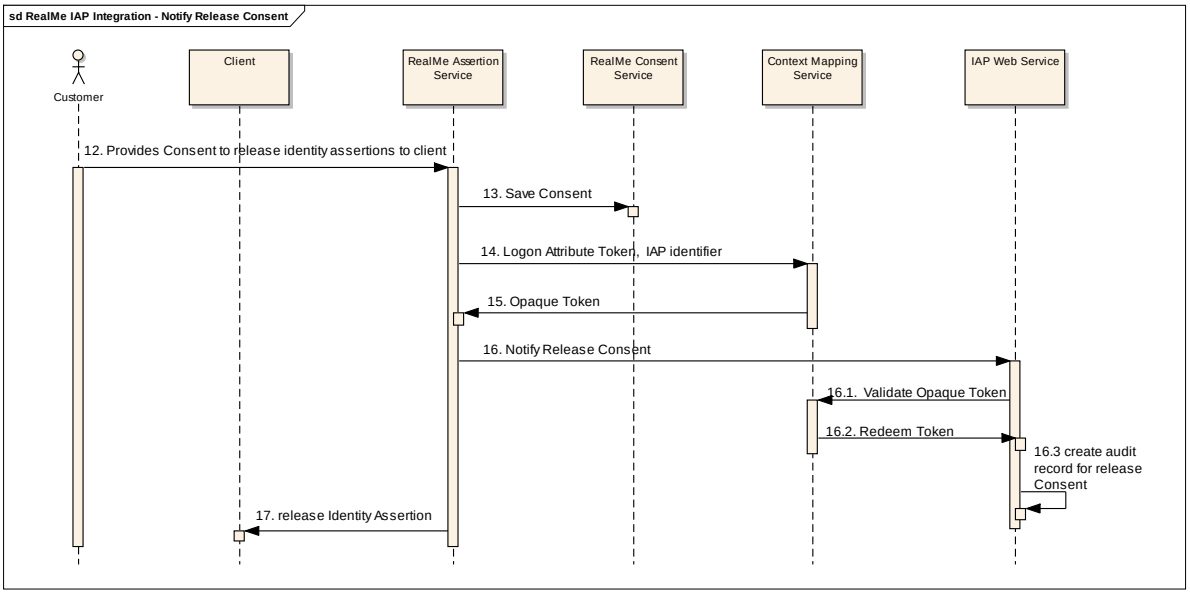

Figure 4 – Notify Release Consent sequence flow

The below sequence of steps are extension to the flow as described in the Table 12.

Message	Description
12. Provides consent	The customer provides consent to release identity assertions to the client.
13. Save Consent	The RealMe assertion service invokes consent service to save user consent on identity assertions.
14. Get Token for IAP	The RealMe assertion service invokes context mapping service for a token that can be provided to IAP web service by passing logon attribute token and IAP identifier in the request.
15. Opaque Token	The context mapping service validates the logon attribute token and issues an opaque token for IAP web service.
16. Notify Release Consent	The RealMe assertion service notifies IAP web service about user's release consent by passing opaque token in the request. The IAP web service validates the request which includes signature verification.

Message	Description
16.1 Validate Opaque Token	The IAP web service retrieves opaque token from the request and invokes context mapping service with opaque token.
16.2 Redeem Token	The context mapping service validates opaque token and issues redeem token to IAP which contains FLT of the customer at IAP.
16.3 Create audit record for release consent	The IAP web service checks identity based on FLT. Creates an audit record for release consent and marks previously issued assertion as billable assertion.
17. release Identity assertions	The RealMe assertion service releases the identity assertion to the client using SAML2.0 websso profile.

Table 15 – Notify Release Consent sequence flow

5.2 Request Message Elements

The root element contained in the SOAP body of a request message is <**samlp:AttributeQuery**>. The <**samlp**:**Response>** MUST be signed with IAP private key. The < **samlp:AttributeQuery>** will contain the following elements and attributes:

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	ID	MUST be provided. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	Version	MUST be provided. The identifier is “2.0”. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	IssueInstant	MUST be provided. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	Destination	MUST be provided, Identity Attribute Service endpoint location. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	<saml:Issuer>	MUST be provided. The value MUST be identifier of identity attributes consumer. It is REQUIRED and it MUST be in the format of an identity privacy domain of Client (i.e. Agency EntityID). An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	<saml:Consent>	MUST be provided, consent token issued by the consent service. . Refer to section 8 for consent token structure and validation. An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes.
<samlp:AttributeQuery>	<saml:Subject>	MUST be provided and must be opaque token issued by the igovt context mapping service. An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<saml:Subject>	<saml:NameID>	<NameID> Format attribute value MUST be ***urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified***. The <NameID> MUST contain opaque token issued by the igovt context mapping service.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	<saml:Attribute> (for Assertion ID)	Must be provided *NameFormat* attribute must be provided and value as ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic*** *Name* attribute must be as following format ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:Assertion:ID*** Must provide ID value in <saml:AttributeValue></saml:AttributeValue> Examples: <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name="urn:nzl:govt:ict:stds:authn:attribute:igovt:IVS:Assertion:ID"><saml:AttributeValue>xxx</saml:AttributeValue></saml:Attribute> An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:AttributeQuery>	<saml:Attribute> (for Assertion Consent)	Must be provided *NameFormat* attribute must be provided and value as ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic*** *Name* attribute must be as following format ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:Assertion:Consent*** Must provide Consent Token issued by the consent service in <saml:AttributeValue></saml:AttributeValue> Refer to section 8 for consent token structure and validation. Examples: <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name="urn:nzl:govt:ict:stds:authn:attribute:igovt:IVS:Assertion:Consent"><saml:AttributeValue>xxx</saml:AttributeValue></saml:Attribute> An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes.
<samlp:AttributeQuery>	<ds:Signature>	MUST be provided.

Table 16 – Notify Release Consent request message elements

5.3 Response Message Elements

The root element contained in the SOAP body of a successful response message MUST be a single <**samlp**:**Response>**. The <**samlp**:**Response>** will contain the following elements and attributes:

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:Response>	ID	SHALL return this attribute.
<samlp:Response>	Version	SHALL return this attribute. The identifier is “2.0”.
<samlp:Response>	IssueInstant	SHALL return this attribute.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<samlp:Response>	Destination	SHALL return this attribute, client identifier. An Exception SHALL result if invalid or not provided.
<samlp:Response>	InResponseTo	SHALL return this attribute, ID attribute of <samlp:AttributeQuery>
<samlp:Response>	<saml:Issuer>	SHALL return this element. It is REQUIRED and MUST be an Identity Attribute Provider's service identifier (i.e. IAP EntityID).
<samlp:Response>	<samlp:Status>	SHALL return this element.
<samlp:Status>	<samlp:StatusCode>	If Identity attributes exist and verified the status code SHALL return `urn:oasis:names:tc:SAML:2.0:status:Success`. In other conditions SHALL return `urn:oasis:names:tc:SAML:2.0:status:Responder`
<samlp:StatusCode>	<samlp:StatusCode>	SHALL return this element if top level StatusCode is other than `urn:oasis:names:tc:SAML:2.0:status:Success`.
<samlp:Status>	<samlp:StatusMessage>	SHALL return this element if top level StatusCode is other than `urn:oasis:names:tc:SAML:2.0:status:Success`.
<samlp:Response>	<saml:Assertion>	If the request is valid and SHALL return this element (Status Assertion). The following are few notes re Status Assertion: • Refer to section 6 for Status Assertion common elements. • Subject can be optional • contains only status attributes
<saml:Assertion>	<saml:AttributeStatement>	SHALL return this element.

Container	Container/ Element/ Attribute	RealMe - IAP web service Requirement
<saml:AttributeStatement>	<saml:Attribute> (for status Code)	*SHALL return this element.* *NameFormat* attribute SHALL be provided and value as ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic*** *Name* attribute SHALL be as following format ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:Identity:Notify:Status*** SHALL provide status value in <saml:AttributeValue></saml:AttributeValue> Examples: <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name="urn:nzl:govt:ict:stds:authn:attribute:igovt:IVS:Identity:Notify:Status"><saml:AttributeValue>SUCESS</saml:AttributeValue></saml:Attribute>

Table 17 – Notify Release Consent response message elements

6 IAP SAML Assertion common elements

For successful requests IAP MUST supply the <saml:Assertion> elements as per below table.

Container	Container / Element / Attribute	Requirement
<saml:Assertion>	ID	MUST be provided
<saml:Assertion>	Version	MUST be provided. The identifier is “2.0”. Ref [saml-core-2.0-os] line 2349, 1471
<saml:Assertion>	IssueInstant	MUST be provided. Ref [saml-core-2.0-os] line 2349, 1474.
<saml:Assertion>	<saml:Issuer>	MUST be provided to identify the IAP.
<saml:Assertion>	<saml:Subject>	MAY be provided.

<saml:Subject> (if provided)	<saml:NameID>	The IAP SHALL return this element.
		The NameID SHALL always be returned with Format attribute with one of the following:
		`urn:oasis:names:tc:SAML:2.0:nameid-format:transient`
		`urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
		IAP SHALL return with NameID format as “*: transient” if the IAP doesn’t create any federation identifiers for the requesters.

<saml:Assertion>	<saml:Conditions>	The IAP SHALL return this element. Any information conveyed by RealMe in the Conditions element MUST be honoured by the SP.
<saml:Assertion>	<ds:Signature>	MUST be provided. IVS must sign an assertion with their signing key.

Table 18 – IAP SAML Assertion elements

7 Error Messages

If any error condition arises while processing an <AttributeQuery> request, the IAP MUST securely communicate the error code to RealMe. The SAML v2.0 Response message contains support for error codes in the <Status> element; see [saml-core-2.0-os].

The <Status> element contains two tiers of <StatusCode> elements. The first tier is reserved for status codes within the SAML v2.0 standard. The second tier is available for system entities to use defined SAML v2.0 status codes and define more specific status codes by defining appropriate URI references.

<StatusCode> Element Values		**Error Condition**
Top Level	**Second Level**	
urn:oasis:names:tc:SAML:2.0:status:Responder	urn:oasis:names:tc:SAML:2.0:status:RequestDenied	<AttributeQuery> attribute IssueInstant not within the accepted time window.
urn:oasis:names:tc:SAML:2.0:status:Responder	urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported	< AttributeQuery > element <Issuer> is not in a format that identifies a privacy domain.
urn:oasis:names:tc:SAML:2.0:status:Responder	urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported	<AttributeQuery> / <Subject> <NameID> attribute Format is not urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified
urn:oasis:names:tc:SAML:2.0:status:Responder	urn:oasis:names:tc:SAML:2.0:status:RequestUnsupported	<AttributeQuery> is provided with unsupported values.
urn:oasis:names:tc:SAML:2.0:status:Responder	urn:nzl:govt:ict:stds:authn:deployment:RealMe:SAML:2.0:status:InternalError	SHALL be returned when there is an internal error in the IAP service.
urn:oasis:names:tc:SAML:2.0:status:Responder	urn:oasis:names:tc:SAML:2.0:status:InvalidSubject	The error code SHALL be returned for one of the following error conditions: • No opaque token in the request. • Opaque token is expired • Opaque token is invalid or unparsable
urn:oasis:names:tc:SAML:2.0:status:Responder	urn:oasis:names:tc:SAML:2.0:status:InvalidSignature	The signature element is either invalid or missing.

Table 19 – IAP web service error scenarios and codes

The table below lists error status codes that are not emitted by normal operation of IAP. They are eligible to be passed as part of IAP support for the wider SAML v2.0 specification. The receipt of these is likely due to incorrect SAML v2.0 messaging or an error in the integration with IAP.

Error URN	**Condition**
urn:oasis:names:tc:SAML:2.0:status:InvalidAttrNameOrValue	Unexpected or invalid content was encountered within a <saml:Attribute> or <saml:AttributeValue> element.

Error URN	Condition
urn:oasis:names:tc:SAML:2.0:status:InvalidNameIDPolicy	The responding provider cannot or will not support the requested name identifier policy.
urn:oasis:names:tc:SAML:2.0:status:RequestVersionDeprecated	The SAML responder cannot process any requests with the protocol version specified in the request.
urn:oasis:names:tc:SAML:2.0:status:RequestVersionTooHigh	The SAML responder cannot process the request because the protocol version specified in the request message is a major upgrade from the highest protocol version supported by the responder.
urn:oasis:names:tc:SAML:2.0:status:RequestVersionTooLow	The SAML responder cannot process the request because the protocol version specified in the request message is too low.
urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal	The responding provider does not recognize the principal specified or implied by the request.

Table 20 – IAP web service standard SAML error codes

8 Consent Token

8.1 Consent Token and IAP Validation

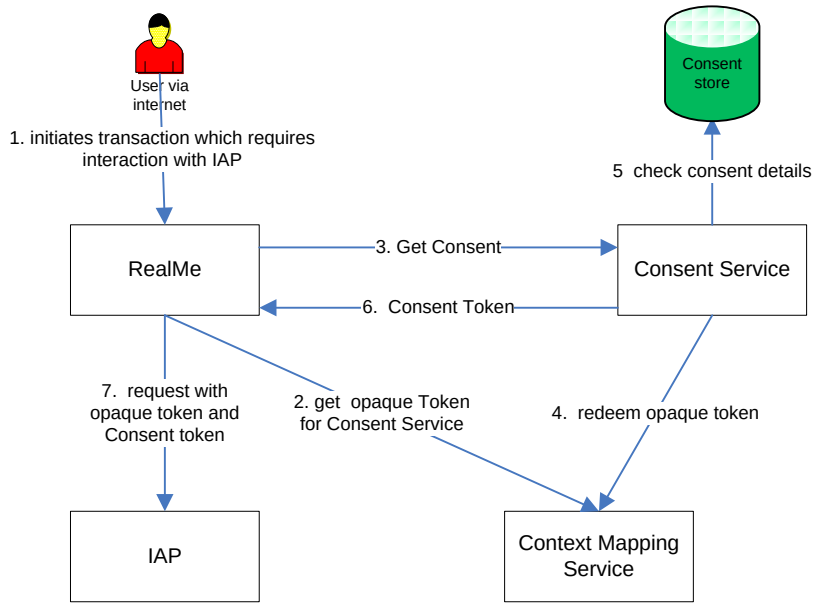

Figure 5 – Consent Token issuance

The consent service issues the consent token to RealMe. The consent token is a string which consists of seven attributes of a consent event in name-value pair format, delimited by ampersand (&) character and the signature value of seven attribute-value pairs. The following is the structure for the consent token:

> ***CosentType=****value1*&***ConsentAttributes=****value2*&***ConsentEventDate=****value3*&***ConsentDecision=****value4*&***ConsentCapturedAt=****value5*&***TokenIssueDate=****value6*&***TokenExpiryDate=****value7*&***Signature=****value8*

The following table describes consent token elements:

Consent Token Element	Description
ConsentType	The type of consent event and the possible values are: 1. Integration Consent 2. Single Transactional Consent (i.e. Release Consent) 3. Multiple Transactional Consent
ConsentAttrubutes	A string value to represent the set of attributes that the user has given consent to.
ConsentEventDate	The instant in date and time the consent was stored.
ConsentDecision	A string value to represent the user's decision for consent event.

Consent Token Element	Description
ConsentCapturedAt	The name of the service or application where the consent was captured at. For RealMe usage, this will always be 'RealMe'.
TokenIssuanceDate	The date and time at which the consent token will issued.
TokenExpiryDate	The date and time at which the consent token will expire.
Signature	The signature value of the first seven parts (i.e. attribute name-value pairs) of consent token. The consent service signs the token with its private key and appends signature value to the consent token.

Table 21 – Consent Token elements

The RealMe account service or assertion service passes consent token in the request to the IAP. The IAP retrieves consent token from <saml:Consent> element in the request. The IAP SHOULD perform the following validations:

- Signature validation: verify signature value of consent token against remaining part of consent token using consent service's public key.
- The expiry date of the token MUST be with in the limits of current time.
- Verify the consent event attribute values, SHOULD match with the request context.

Appendix

Part A: Identity Verification Service - Identity Attributes

The IVS IAP MUST implement the New Zealand Government OASIS CIQ Profile [nzciq] to convey the customer's identity data. This is a standard that references the "Customer Information Quality v3.0 Specifications" from OASIS which is a XML based standard to define a vocabulary to represent customer data, including identity related attributes.

The XML document is constructed as per document reference [nzciq] for the <PartyName> element. The <PersonInfo> and <BirthInfo> elements are not specified in [nzciq], but the parent [ciq-3.0] specification. The XML elements utilised in [nzciq] and [ciq-3.0] to convey identity related data SHALL be constrained as follows.

Container	Container / Element / Attribute	RealMe Constrained Behaviour
<Party>	<PartyName>	As per [nzciq] & [ciq-3.0].
<Party>	<PersonInfo>	As per [nzciq] & [ciq-3.0].
<Party>	<BirthInfo>	As per [nzciq] & [ciq-3.0].
<PartyName>	<PersonName>	As per [nzciq] & [ciq-3.0].
<PersonName	<NameElement>	As per [nzciq] & [ciq-3.0] with the following restrictions: SHALL contain one and only one <NameElement> with ElementType ="LastName". MAY contain one and only one <NameElement> with ElementType = "FirstName". MAY contain one and only one <NameElement> with ElementType = "MiddleName". If present, SHALL NOT be empty or blank.
<PersonInfo>	Gender	As per [ciq-3.0].
<BirthInfo>	<BirthInfoElement>	As per [ciq-3.0] with the following restrictions: SHALL NOT contain a <BirthInfoElement> with Type= "MothersName" SHALL contain one and only one <BirthInfoElement> with Type = "BirthYear". SHALL contain one and only one <BirthInfoElement> with Type = "BirthMonth". SHALL contain one and only one <BirthInfoElement> with Type = "BirthDay". SHALL NOT contain a <BirthInfoElement> with Type="BirthTime".

Container	Container / Element / Attribute	RealMe Constrained Behaviour
<BirthInfo>	<BirthPlaceDetails>	As per [ciq-3.0] with the following restrictions: SHALL contain one <Country> and/or one <Locality> container.
<BirthPlaceDetails>	<Country>	As per [ciq-3.0] with the following restrictions: If present, SHALL contain a NameElement of NameType=”Name”
<BirthPlaceDetails>	<Locality>	As per [ciq-3.0] with the following restrictions: If present, SHALL contain a NameElement of NameType=”Name” If present, SHALL NOT contain a NameElement of NameType=”Type”
<Party>	DateValidFrom	May be provided
<Party>	DateValidTo	May be provided

Table 22 – IVS identity attributes data elements

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns1:Party xmlns:ns1="urn:oasis:names:tc:ciq:xpil:3"
           xmlns:ns2="urn:oasis:names:tc:ciq:xnl:3"
           xmlns:ns3="urn:oasis:names:tc:ciq:ct:3"
           xmlns:ns4="http://www.w3.org/1999/xlink"
           xmlns:ns5="urn:oasis:names:tc:ciq:xal:3">
  <ns1:PartyName>
    <ns2:PersonName>
      <ns2:NameElement ns2:ElementType="FirstName">Amelia</ns2:NameElement>
      <ns2:NameElement ns2:ElementType="MiddleName">Lucy</ns2:NameElement>
      <ns2:NameElement ns2:ElementType="LastName">Macdonald</ns2:NameElement>
    </ns2:PersonName>
  </ns1:PartyName>
  <ns1:PersonInfo ns1:Gender="F"/>
  <ns1:BirthInfo>
    <ns1:BirthInfoElement ns1:Type="BirthYear">1985</ns1:BirthInfoElement>
    <ns1:BirthInfoElement ns1:Type="BirthMonth">06</ns1:BirthInfoElement>
    <ns1:BirthInfoElement ns1:Type="BirthDay">14</ns1:BirthInfoElement>
    <ns1:BirthPlaceDetails>
      <ns5:Country>
        <ns5:NameElement ns5:NameType="Name">New Zealand</ns5:NameElement>
      </ns5:Country>
      <ns5:Locality>
```

```
        <ns5:NameElement ns5:NameType="Name">Wellington</ns5:NameElement>
      </ns5:Locality>
    </ns1:BirthPlaceDetails>
  </ns1:BirthInfo>
</ns1:Party>
```

Figure 6 – Sample IVS identity attributes data

Part B: Address Verification Service - Identity Attributes

The AVS IAP SHOULD implement the New Zealand Government OASIS CIQ Profile [nzciq] to convey the customer's address data. This is a standard that references the "Customer Information Quality v3.0 Specifications" from OASIS which is a XML based standard to define a vocabulary to represent customer data, including identity related attributes.

Container	Container / Element / Attribute	RealMe Constrained Behaviour
<Party>	<Address>	As per [nzciq] & [ciq-3.0].
<Address>	<PersonInfo>	As per [nzciq] & [ciq-3.0].

Table 23 – AVS address attributes data elements