

|  |  |  |
| --- | --- | --- |
| Version: | | – |
| Author: | |  |
| Date: | 15th February 2013 | |



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 |
| 0.6 | 12/02/2013 | Venkat Maddali | Updated based on discussions with Datacom. |
| 0.7 | 15/02/213 | Venkat Maddali | Added metadata section which requires for IAP integration |

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 3](#_Toc348689665)

[1.1 Overview 3](#_Toc348689666)

[1.2 Document Purpose 3](#_Toc348689667)

[1.3 Audience 3](#_Toc348689668)

[1.4 Document references 3](#_Toc348689669)

[1.5 Glossary 5](#_Toc348689670)

[1.6 Assumptions 5](#_Toc348689671)

[1.7 Notion 5](#_Toc348689672)

[1.8 Namespaces 5](#_Toc348689673)

[1.9 Conformance and Compliance 6](#_Toc348689674)

[2 Message Pre-requisites 7](#_Toc348689675)

[2.1 Mutual SSL certificates 7](#_Toc348689676)

[2.2 Signing Certificate 7](#_Toc348689677)

[2.3 Transport 7](#_Toc348689678)

[2.4 Server Synchronisation 7](#_Toc348689679)

[3 Get Identity Attributes Status 9](#_Toc348689680)

[3.1 Message Flow 9](#_Toc348689681)

[3.2 Request Message Elements 10](#_Toc348689682)

[3.3 Response Message Elements 13](#_Toc348689683)

[4 Get Identity Attributes Assertion 16](#_Toc348689684)

[4.1 Message Flow 16](#_Toc348689685)

[4.2 Request Message Elements 19](#_Toc348689686)

[4.3 Response Message Elements 22](#_Toc348689687)

[5 Notify Release Consent 27](#_Toc348689688)

[5.1 Message Flow 27](#_Toc348689689)

[5.2 Request Message Elements 28](#_Toc348689690)

[5.3 Response Message Elements 32](#_Toc348689691)

[6 SAML Assertion common elements 35](#_Toc348689692)

[7 Error Messages 36](#_Toc348689693)

[8 Consent Token 37](#_Toc348689694)

[8.1 Consent Token and IAP Validation 37](#_Toc348689695)

[9 IAP Metadata 39](#_Toc348689696)

[9.1 Elements in Identity Provider Metadata 39](#_Toc348689697)

[Appendix 48](#_Toc348689698)

[Part A: Identity Verification Service - Identity Attributes 48](#_Toc348689699)

[Part B: Address Verification Service - Identity Attributes 51](#_Toc348689700)

# Introduction

## Overview

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

## 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.

## 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).

## 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 |
| 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 - 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 – 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](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 – 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 – 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 – WS-\* references

## 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.

## 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.

## 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].

## 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 : Namespace prefix and references

## 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
* NZ CIQ Profile (<http://ict.govt.nz/resources/standards-compliance/nz-government-ciq-profiles>)
* Oasis CIQ Specifications Version 3.0

# 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:

## 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]](#footnote-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 RSA1.

## 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.

## 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.

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](http://www.e.govt.nz/standards/e-gif/ntp) where UTC ([MSL](http://msl.irl.cri.nz)) is the Stratum One NTP time server, with NTP v4 as the delivery method over the internet.

# 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.

## Message Flow

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



Figure - 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. |
| 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 – Get Identity Attributes Status Flow

## 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. |
| <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. The consent token contains the user consent for RealMe to retrieve identity information from IAP. 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. |
| <samlp:AttributeQuery> | <saml:Attribute> ( for identity status code) | Must be provided  Attribute NameFormat attribute must be provided as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the following format:  ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:{AttributesType}:Status***  **Example**:  <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:Attribute>  An Exception SHALL result if invalid or not provided. Refer to section 7 for error codes. |
| <samlp:AttributeQuery> | <saml:Attribute> (for identity status message) | Must be provided  Attribute NameFormat attribute must be provided as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the following format:  ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:{AttributesType}:StatusMessage***  **Example**:  <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name=" urn:nzl:govt:ict:stds:authn:attribute:igovt:IVS:Identity:StatusMessage "> </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 – Get Identity Attributes Status request message elements

## 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> |
| <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. * contains only status attributes |
| <saml:Assertion> | <saml:AttributeStatement> | SHALL return this element. |
| <saml:AttributeStatement> | <saml:Attribute> ( requested identity status attribute) | *SHALL return this element.*  *Attribute NameFormat* MUST be same as request  *Attribute Name* MUST be same as request.  Status Value SHALL be provided in <saml:AttributeValue> </saml:AttributeValue> element.  **Example**:  <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. |
| <saml:AttributeStatement> | <saml:Attribute> ( requested identity status message attribute) | *SHALL return this element.*  *Attribute NameFormat* MUST be same as request  *Attribute Name* MUST be same as request.  Status Message Value SHALL be provided in <saml:AttributeValue> </saml:AttributeValue> element.  **Example**:  <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 – 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 – IAP Status Codes for RealMe

# 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.

## Message Flow

### **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 – 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 – View Identity Details at RealMe Dashboard sequence flow

### **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 – 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. |
| 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 – Provide Identity Details to RealMe client flow

## 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. |
| <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. The consent token contains the user consent for RealMe to retrieve identity information from IAP. 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. |
| <samlp:AttributeQuery> | <saml:Attribute> ( for identity attributes) | MUST be provided  Attribute NameID Format MUST be provided as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the following format:  ***urn:nzl:govt:ict:stds:authn:safeb64:attribute:{Provider}:{Service}:{AttributesType}***  **Example**:  <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name=" urn:nzl:govt:ict:stds:authn:safeb64:attribute:igovt:IVS: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:Address"> </saml:Attribute> |
| <samlp:AttributeQuery> | <saml:Attribute> ( for federation identifier)  **Example**: IVS issues user’s Federated Identity Tag (FIT) for the client privacy domain. FIT is unique number for the user at the client privacy domain. | MAY be provided  If provided Attribute NameID Format MUST be as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the following format:  ***urn:nzl:govt:ict:stds:authn:safeb64:attribute:{Provider}:{Service}:{AttributeType}***  **Example**:  <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name=" urn:nzl:govt:ict:stds:authn:safeb64:attribute:igovt:IVS:FIT "> </saml:Attribute> |
| <samlp:AttributeQuery> | <saml:Attribute> ( for transaction identifier)  *Note*: *The transaction identifier will be used for client’s billing. The IAP can retain this identifier if they wish to generate their own billing.*  *This Attribute MUST be passed in the request in identity assert scenario at client.* | MAY be provided  If provided Attribute NameID Format MUST be as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the following format:  ***urn:nzl:govt:ict:stds:authn:safeb64:attribute:{Provider}:{Service}:TransactionID***  **Example**:  <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified" Name=" urn:nzl:govt:ict:stds:authn:safeb64:attribute:igovt:IVS:TransactionID "> </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 - Get Identity Attributes Assertion request message elements

## 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>** MUSTbesigned 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.  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 Assertion common elements. * contains identity attributes |
| <saml:Assertion> | <saml:AttributeStatement> | *SHALL return this element.* |
| <saml:AttributeStatement> | <saml:Attribute>( requested identity attributes) | SHALL return this element.  *Attribute Name* Format MUST be same as in the request.  *Attribute Name* MUST be same as in the request.  Identity Attributes SHALL be provided in <saml:AttributeValue> </saml:AttributeValue> element.  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. |
| <saml:AttributeStatement> | <saml:Attribute> ( requested federated identifier) | SHALL return this element if requested.  *Attribute Name* Format MUST be same as in the request.  *Attribute Name* MUST be same as in the request.  Federated Identifier SHALL be provided in <saml:AttributeValue> </saml:AttributeValue> element.  The value MUST be <saml:NameID> element.  **Example**:  <saml:Attribute  Name=" urn:nzl:govt:ict:stds:authn:safeb64:attribute:igovt:IVS:FIT" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">      <saml:AttributeValue>  <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent" NameQualifier="<https://identity.i.govt.nz/idp/ivs-idp>" SPNameQualifier="<https://sample-agency/realm/app>">WLG3243574587456843573547637 </saml:NameID>  </saml:AttributeValue>     </saml:Attribute> |
| <saml:AttributeStatement> | <saml:Attribute> ( requested transaction identifier) | SHALL return this element if requested.  *Attribute Name* Format MUST be same as in the request.  *Attribute Name* MUST be same as in the request.  Federated Identifier SHALL be provided in <saml:AttributeValue> </saml:AttributeValue> element.  **Example**:  <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name=" urn:nzl:govt:ict:stds:authn:safeb64:attribute:igovt:IVS:TransactionID "> <saml:AttributeValue>TR126789-090-00 </saml:AttributeValue> </saml:Attribute> |

Table - Get Identity Attributes Assertion response message elements

# 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.

## 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 in Section ).



Figure – 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. |
| 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 – Notify Release Consent sequence flow

## Request Message Elements

The root element contained in the SOAP body of a request message is <**samlp:AttributeQuery**>. The <**samlp**:**Response>** MUSTbesigned 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. |
| <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. The consent token contains the user consent for RealMe to retrieve identity information from IAP. 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. |
| <samlp:AttributeQuery> | <saml:Attribute> ( for Transaction ID) | Must be provided  Attribute NameID Format MUST be as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the following format:  ***urn:nzl:govt:ict:stds:authn:safeb64:attribute:{Provider}:{Service}:TransactionID***  Must provide ID value in <saml:AttributeValue></saml:AttributeValue> element.  **Example**:  <saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name=" urn:nzl:govt:ict:stds:authn:safeb64:attribute:igovt:IVS:TransactionID "> <saml:AttributeValue>TR126789-090-00 </saml:AttributeValue> </saml:Attribute>  An Exception SHALL result if invalid or not provided. . Refer to section 7 for error codes. |
| <samlp:AttributeQuery> | <saml:Attribute> ( for Assertion Consent) | Must be provided  Attribute NameID Format MUST be as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the 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> element.  Refer to section 8 for consent token structure and validation.  **Example**:  <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> | <saml:Attribute> ( for Notification status) | Must be provided  Attribute NameID Format MUST be as  ***urn:oasis:names:tc:SAML:2.0:attrname-format:basic***  Attribute Name must be provided in the following format:  ***urn:nzl:govt:ict:stds:authn:attribute:{Provider}:{Service}:Identity:Notify:Status***  **Example**:  <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: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 – Notify Release Consent request message elements

## 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. |
| <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 Assertion common elements. * Returns notify status attribute |
| <saml:Assertion> | <saml:AttributeStatement> | SHALL return this element. |
| <saml:AttributeStatement> | <saml:Attribute> ( for status Code) | *SHALL return this element.*  *Attribute Name* Format MUST be same as in the request.  *Attribute Name* MUST be same as in the request.  SHALL provide status value in <saml:AttributeValue></saml:AttributeValue> element  **Example**:  <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>success </saml:AttributeValue> </saml:Attribute> |

Table – Notify Release Consent response message elements

# 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> | MUST be provided and MUST be same value as in the request (i.e. opaque token) and also NameID Format MUST be same as in the request (i.e. ***urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified***.). |
| <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 – IAP SAML Assertion elements

# 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.

The following table depicts suggested top level and second level error codes that IAP may respond to the invalid <AttributeQuery> request. There is no specific behaviour to these status codes in RealMe. So IAP can respond with other status codes to the error conditions listed in below table. It depends on the IAP product implementation for SAML Attribute Query profile.

| **<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 – IAP web service error scenarios and codes

# Consent Token

## Consent Token and IAP Validation



Figure – 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&****SigAlg****=value9*

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. |
| ConsentCapturedAt | The name of the service or application where the consent is captured at. For RealMe usage, this will always be 'RealMe'. |
| TokenIssuanceDate | The date and time at which the consent token will be issued. |
| TokenExpiryDate | The date and time at which the consent token will be expired. |
| 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. |
| SigAlg | The signature algorithm |

Table – 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.

# IAP Metadata

Each of Identity Attribute Provider (IAP) interface will have its own metadata configuration and the following tables will call out the restrictions placed on IAP metadata. If not noted here the restrictions will be as per [saml-metadata-2.0-os].

## Elements in Identity Provider Metadata

The following restrictions will be placed on the IdP metadata elements and attributes:

### **Root Element**

A SAML v2.0 metadata instance describes either a single entity or multiple entities via use of <EntityDescriptor> or <EntitiesDescriptor> root elements respectively. RealMe supports only the use of single <EntityDescriptor> root elements in a SAML v2.0 metadata file.

| **Attribute / Element** | **RealMe**  **Requirement** | **NZ SAMS 1.0**  **Requirement** | **OASIS SAML v2.0**  **Requirement** |
| --- | --- | --- | --- |
| <EntityDescriptor> | A single <EntityDescriptor> SHALL be provided.   * NZ SAMS | OPTIONAL.  NZ SAMS line 171.   * SAML v2.0 | Must be a root element if <EntitiesDescriptor> is not used.  Ref [saml-metadata-2.0-os] section 2.3, line 307. |

### **Element <EntityDescriptor>**

| **Attribute / Element** | **RealMe**  **Requirement** | **NZ SAMS 1.0**  **Requirement** | **OASIS SAML v2.0**  **Requirement** |
| --- | --- | --- | --- |
| ID | SHALL NOT be provided.   * NZ SAMS | OPTIONAL if the <EntityDescriptor> is not signed.  NZ SAMS line 210.   * SAML v2.0 | Optional. A document-unique identifier for the element, typically used as a reference point when signing.  Ref [saml-metadata-2.0-os] section 2.3.2, line 374. |
| Constraint on NZ SAMS:  Not provided. |
| entityID | SHALL be provided. Contains IAP entityID that SHALL be in an identity privacy domain format in RealMe SAML Message specification.   * NZ SAMS | REQURIED.  NZ SAMS line 209.   * SAML v2.0 | Required attribute. Specifies the unique identifier of the SAML v2.0 entity whose metadata is described by the element's contents.  Ref [saml-metadata-2.0-os] section 2.3.2, line 371. |
| Constraint on NZ SAMS:  Format as per section 7. |
| validUntil | SHALL NOT be provided.  IAP SHALL NOT mandate an expiry time of its metadata to RealMe.   * NZ SAMS | SHOULD be provided for the case of RealMe where the <EntityDescriptor> is the root element.  NZ SAMS line 214.   * SAML v2.0 | Optional attribute indicates the expiration time of the metadata contained in the element and any contained elements.  Ref [saml-metadata-2.0-os] section 2.3.2, line 376. |
| Constraint on NZ SAMS:  Expiry of RealMe IdP metadata not stated. |
| cacheDuration | SHALL NOT be provided.   * NZ SAMS | SHOULD be provided for the case of RealMe where the <EntityDescriptor> is the root element.  NZ SAMS line 214.   * SAML v2.0 | Optional attribute indicates the maximum length of time a consumer should cache the metadata contained in the element and any contained elements.  Ref [saml-metadata-2.0-os] section 2.3.2, line 379. |
| Constraint on NZ SAMS:  CacheDurtion of IdP metadata not stated. |
| <ds:Signature> | SHALL NOT be provided.  IAP SHALL NOT sign metadata.   * NZ SAMS | OPTIONAL   * SAML v2.0 | Optional. An XML signature that authenticates the containing element and its contents.  Ref [saml-metadata-2.0-os] section 2.3.2, line 382. |
| Constraint on NZ SAMS:  Not provided. |
| <Extensions> | SHALL NOT be provided.   * NZ SAMS | NOT RECOMMENDED and if present will be ignored.  NZ SAMS line 222.   * SAML v2.0 | Optional metadata extensions.  Ref [saml-metadata-2.0-os] section 2.3.2, line 385. |
| <AttributeAuthority  Descriptor> | SHALL be provided   * NZ SAMS | For an Attribute Auhority the < AttributeAuthority Descriptor> MUST be provided.   * SAML v2.0 | <RoleDescriptor>, <IDPSSODescriptor>, <SPSSODescriptor>, <AuthnAuthorityDescriptor>, <AttributeAuthorityDescriptor>, <PDPDescriptor> [One or More] OR <AffiliationDescriptor>  Ref [saml-metadata-2.0-os] section 2.3.2, lines 389-395. |
| <Organization> | SHALL be provided.   * NZ SAMS | REQURIED.  NZ SAMS line 232.   * SAML v2.0 | Optional. An element identifying the organization responsible for the SAML entity described by the element.  Ref [saml-metadata-2.0-os] section 2.3.2, line 396. |
| <ContactPerson> | SHALL be provided.   * NZ SAMS | RECOMMENDED that one be included.  NZ SAMS line 235.   * SAML v2.0 | Optional sequence of elements identifying various kinds of contact personnel.  Ref [saml-metadata-2.0-os] section 2.3.2, line 399. |
| <Additional MetadataLocation> | SHALL NOT be provided.   * NZ SAMS | MUST NOT be provided.  NZ SAMS line 236.   * SAML v2.0 | Optional sequence of namespace-qualified locations where additional metadata exists for the SAML entity. This may include metadata in alternate formats or describing adherence to other non-SAML specifications.  Ref [saml-metadata-2.0-os] section 2.3.2, line 401. |

### **Element <AttributeAuthorityDescriptor>**

| **Attribute / Element** | **RealMe**  **Requirement** | **NZ SAMS 1.0**  **Requirement** | **OASIS SAML v2.0**  **Requirement** |
| --- | --- | --- | --- |
| ID | SHALL NOT be provided. | Per OASIS.   * SAML v2.0 | Optional. A document-unique identifier for the element, typically used as a reference point when signing.  Ref [saml-metadata-2.0-os] section 2.3.2, line 555. |
| validUntil | SHALL NOT be provided. | One of validUntil or cacheDuration SHOULD be provided.  NZ SAMS line 214.   * SAML v2.0 | Optional attribute indicates the expiration time of the metadata contained in the element and any contained elements.  Ref [saml-metadata-2.0-os] section 2.3.2, line 557. |
| cacheDuration | SHALL NOT be provided. | One of validUntil or cacheDuration SHOULD be provided.  NZ SAMS line 214.   * SAML v2.0 | Optional attribute indicates the maximum length of time a consumer should cache the metadata contained in the element and any contained elements.  Ref [saml-metadata-2.0-os] section 2.3.2, line 560. |
|  |  |  |  |
| protocolSupport Enumeration | SHALL be provided.   * NZ SAMS | Per OASIS.   * SAML v2.0 | Required.  A whitespace-delimited set of URIs that identify the set of protocol specifications supported by the role element. For SAML v2.0 entities, this set MUST include the SAML protocol namespace URI **urn:oasis:names:tc:SAML:2.0:protocol.**  Ref [saml-metadata-2.0-os] section 2.3.2, line 560. |
| errorURL | SHALL NOT be provided.   * NZ SAMS | Per OASIS.   * SAML v2.0 | Optional URI attribute that specifies a location to direct a user for problem resolution and  Additional support related to this role.  Ref [saml-metadata-2.0-os] section 2.3.2, line 560. |
| Constraint on NZ SAMS:  Not provided. |
| <AttributeService> | SHALL be provided. | Per OASIS.   * SAML v2.0 | Optional.  Zero or more elements that describe endpoints that support the profile of the Assertion Request protocol.  Ref [saml-metadata-2.0-os] line 698. |
| <saml:Attribute> | SHALL be provided. One or more elements that identify the SAML attributes supported by the identity attribute provider. | Per OASIS.   * SAML v2.0 | Optional.  Zero or more elements that identify the SAML attributes supported by the identity attribute provider.  Ref [saml-metadata-2.0-os] line 705. |
| <ds:Signature> | SHALL NOT be provided.  RealMe SHALL NOT sign metadata.   * NZ SAMS | OPTIONAL   * SAML v2.0 | Optional. An XML signature that authenticates the containing element and its contents.  Ref [saml-metadata-2.0-os] section 2.3.2, line 572. |
| Constraint on NZ SAMS:  Not provided. |
| <Extensions> | SHALL NOT be provided.   * NZ SAMS | NOT RECOMMENDED and if present will be ignored.  NZ SAMS line 222.   * SAML v2.0 | Optional metadata extensions.  Ref [saml-metadata-2.0-os] section 2.3.2, line 575. |

### **Element <KeyDescriptor>**

| **Attribute / Element** | **RealMe**  **Requirement** | **NZ SAMS 1.0**  **Requirement** | **OASIS SAML v2.0**  **Requirement** |
| --- | --- | --- | --- |
| use | MUST contain the value ‘signing’ to meet the digital signing requirements in section   * NZ SAMS | MUST be used.  NZ SAMS line 246.   * SAML v2.0 | Optional attribute specifying the purpose of the key being described. Values are drawn from the KeyTypes enumeration, and consist of the values ‘encryption’ and ‘signing’.  Ref [saml-metadata-2.0-os] line 615. |
| Constraint on NZ SAMS:  Must specify ‘signing’ only |
| <ds:KeyInfo> | SHALL be provided. SHALL contain RealMe SAML messaging certificate.   * NZ SAMS | Per OASIS.   * SAML v2.0 | Required. The element that identifies the SAML messaging certificates and/or keys.  Ref [saml-metadata-2.0-os] line 618. |
| <EncryptionMethod> | SHALL NOT be provided. RealMe does not perform SAML message encryption.   * NZ SAMS | Per OASIS.   * SAML v2.0 | Optional element specifying an algorithm and algorithm-specific settings supported by the entity. The exact content varies based on the algorithm supported.  Ref [saml-metadata-2.0-os] line 621. |

### **Element <AttributeService>**

| **Attribute / Element** | **RealMe**  **Requirement** | **NZ SAMS 1.0**  **Requirement** | **OASIS SAML v2.0**  **Requirement** |
| --- | --- | --- | --- |
| Binding | SHALL use the value: urn:oasis:names:tc:  SAML:2.0:bindings:SOAP   * NZ SAMS | Per OASIS.   * SAML v2.0 | MUST be provided.  Ref [saml-metadata-2.0-os] line 229. |
| Constraint on NZ SAMS:  Must specify SOAP only. |
| Location | MUST be provided.   * NZ SAMS | Per OASIS.   * SAML v2.0 | MUST be provided.  A required URI attribute that specifies the location of the endpoint.  Ref [saml-metadata-2.0-os] line 232. |
| ResponseLocation | MUST NOT be provided.   * NZ SAMS | Per OASIS.   * SAML v2.0 | Optional. Used only for protocols or profiles that has more than one type of request or response message.  Ref [saml-metadata-2.0-os] line 235. |

The following is sample IAP metadata in the form of XML.

<md:EntityDescriptor entityID="http://identity.govt.nz/IAP/AttributeService" xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:query="urn:oasis:names:tc:SAML:metadata:ext:query" xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">

<md:AttributeAuthorityDescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">

<md:KeyDescriptor use="signing">

<ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

<ds:X509Data>

<ds:X509Certificate>

MIICBDCCAW0CBEnn9rYwDQYJKoZIhvcNAQEFBQAwSTELMAkGA1UEBhMCTloxEjAQBgNVBAoMCXNh

bXBsZV9zYTEOMAwGA1UECxMFbG9nb24xFjAUBgNVBAMTDXNhbXBsZSBhZ2VuY3kwHhcNMDkwNDE3

MDMyNTQyWhcNMTQwNDI0MDMyNTQyWjBJMQswCQYDVQQGEwJOWjESMBAGA1UECgwJc2FtcGxlX3Nh

MQ4wDAYDVQQLEwVsb2dvbjEWMBQGA1UEAxMNc2FtcGxlIGFnZW5jeTCBnzANBgkqhkiG9w0BAQEF

AAOBjQAwgYkCgYEAjaPQMCe6O24eYG3B00bUwC10Bgggn6T1eom2pQoMp+dPYEeSorzh6Li6tf/5KEg

1iGX/NdppK55CR+Z68A7bJthf3/X6Kpqq+Cy9MFqkB3GiLAPuyX+bzkmd0AK0sSEb19cxaeJkhYT

yhN8QdkTZORJePwxND7eX/rJlx0eGmlNOz8CAwEAATANBgkqhkiG9w0BAQUFAAOBgQB5peuOsRq8

RYar8So2ckrUtptaDSR4/27qR0dpsxQUSSIEWuLzD9Ubm9Z6HytIOvrY4d3g+mzq0goG1XIO6DtS

SJGsSMrl9xMIHpx2MQ3JgZUT3eIcIjysIMfSSfa8nQ6DStjgXsEXc48RIbV9hJfL0EnzP0lBa1nz

OdtmQCBSZw==

</ds:X509Certificate>

</ds:X509Data>

</ds:KeyInfo>

</md:KeyDescriptor>

<md:AttributeService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP" Location="https://iap.govt.nz:456/AttributeServiceSoap/IAPService"/>

<saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name="xxx" FriendlyName="xxx"></saml:Attribute>

<saml:Attribute NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" Name="TransactionID" FriendlyName="TransactionID"></saml:Attribute>

</md:AttributeAuthorityDescriptor>

<md:Organization>

<md:OrganizationName xml:lang="en-us">Sample IAP</md:OrganizationName>

<md:OrganizationDisplayName xml:lang="en-us">Sample IAP</md:OrganizationDisplayName>

<md:OrganizationURL xml:lang="en-us">https://iap.govt.nz</md:OrganizationURL>

</md:Organization>

<md:ContactPerson contactType="technical">

<md:Company> </md:Company>

<md:GivenName> </md:GivenName>

<md:SurName> </md:SurName>

</md:ContactPerson>

</md:EntityDescriptor>

# 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”. |
| <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 – 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 – 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 – AVS address attributes data elements

1. This will conform with [nzism-402]. [↑](#footnote-ref-2)