XRI 2.0 Resolution

Working Draft 04, 20 February 17, 2005

Document identifier:

wd-xri-resolution-20-04

Location:

http://www.oasis-open.org/spectools/docs/

Editors:

Gabe Wachob, Visa International <gwachob@visa.com>
Drummond Reed, OneName <drummond.reed@onename.com>
Dave McAlpin, Epok <dave.mcalpin@epok.net>
Mike Lindelsee, Visa International <mlindels@visa.com>
Peter Davis, Neustar <peter.davis@neustar.biz>

Contributors:

Chetan Sabnis, Epok <chetan.sabnis@epok.net>

Abstract:

This document defines an HTTP-based resolution mechanism for Extensible Resource Identifiers (XRIs), specifically XRIs conforming to the XRI Generic Syntax Specification v2.0 [link] or higher. For a non-normative introduction to the uses and features of XRIs, see the *"XRI Primer"* at http://www.oasis-open.org/committees/xri/xri-primer-2.0. For the set of XRIs defined by the XRI TC to provide metadata about other XRIs, see the "XRI Metadata Specification" at @@@[link].

Comment [DM1]: Add proper link

Status:

[Describe the status and stability of the specification and where to send comments.] This document is updated periodically on no particular schedule. Send comments to the editor.

[This is boilerplate; to use, fix the hyperlinks:] Committee members should send comments on this specification to the xxx@lists.oasis-open.org list. Others should subscribe to and send comments to the xxx-comment@lists.oasis-open.org list. To subscribe, send an email message to xxx-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

[This is boilerplate; to use, fix the hyperlinks:] For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the XXX TC web page (http://www.oasis-open.org/committees/xxx/).

[If a Committee Specification or OASIS Standard:] The errata page for this specification is at http://www.oasis-open.org/committees/xxx/yyy.

Copyright © OASIS Open 2005. All Rights Reserved.

Table of Contents

Copyright © OASIS Open 2005. All Rights Reserved.

Copyright © OASIS Open 2005. All Rights Reserved.

1 Introduction

1.1 XRI Resolution Framework

Insert introductory content here which is a boiled down version of the introduction for the XRI specification. Talk about how you could have multiple resolution systems.

Talk about the relationship of resolution and trusted resolution.

1.2 Terminology and Notation

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “NOT RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in **[RFC2119]**. When these words are not capitalized in this document, they are meant in their natural language sense.

```
Examples look like this.
```

`XML elements and attributes that appear in text look like this.`

Throughout this document, the XML namespace prefix saml: stands for the Security Assertion Markup Language namespace “urn:oasis:names:tc:SAML:2.0:assertion” whether or not it is explicitly declared in the example or text. Similarly, the XML namespace prefix ds: stands for the W3C Digital Signature Namespace “http://www.w3.org/2000/09/xmldsig#”, and the namespace prefix xri: stands for the XRI Core namespace “xri:$r.s/XRIDescriptor”, whether or not they are explicitly declared in the example or text. These namespace prefixes are summarized in the table below

saml	urn:oasis:names:tc:SAML:2.0:assertion
ds	http://www.w3.org/2000/09/xmldsig#
xri	xri://$res*schema/XRIDescriptor

Terms used in this document are defined in the Glossary section of **[XRISyntax]**.

Copyright © OASIS Open 2005. All Rights Reserved.

2 XRI Resolution

2.1 Introduction

XRI resolution is the process of dereferencing an XRI to a network endpoint in order to obtain metadata about or communicate with the resource identified by the XRI. Because XRIs may be used across a wide variety of communities and applications, including as database keys, filenames, directory keys, object IDs, and XML IDs, no single resolution mechanism may be appropriate for all XRIs. However, in the interest of promoting interoperability, this specification defines a simple, flexible resolution protocol that relies exclusively on HTTP/HTTPS as a transport.

Identifier management policies are defined on a community-by-community basis. With XRIs, the authoritative community is specified by the authority segment of the XRI. When a community chooses to create a new identifier authority, it SHOULD define a policy for assigning and managing identifiers under this authority. Furthermore, it SHOULD define what resolution protocol(s) can be used for resolving identifiers assigned by the authority.

2.1.1 Assumptions

This resolution protocol makes several minimal assumptions about the XRIs being resolved:

- The endpoints representing the top-level authority for any absolute XRI are identified with the authority segment (xri-authority or i-authority productions) of the XRI as defined in section 2.2.1 of **[XRISyntax]**.
- Only absolute XRIs are resolved using this protocol. To resolve a relative XRI, it must be converted into an absolute XRI using the procedure in section 2.4 of **[XRISyntax]**.
- The XRI being resolved has been converted into URI normal form, following the rules in section 2.3.1 of **[XRISyntax]**.
- A resource represented by a single XRI may be accessed by multiple protocols at multiple protocol endpoints. For example, it is possible that a resource represented by a single XRI may be accessed through multiple HTTP URIs, or through both HTTP and another network protocol. Only HTTP access to resources is defined in this document.
- Each network endpoint associated with a resource identified by an XRI may present a different subset, type, or representation of data or metadata associated with the identified resource. For example, two HTTP URIs may be associated with a single XRI for local access – one for data access and one for metadata access. This specification allows XRI authorities to define different types of access using extensible descriptor fields based on content type and the semantics of the interaction.

2.1.2 Phases of Resolution

The XRI resolution protocol is designed to be as simple and flexible as possible given the assumptions above. Based on the structure of XRIs, it consists of two phases:

- Authority resolution
- Local access

Copyright © OASIS Open 2005. All Rights Reserved.

Authority resolution is the process of finding the endpoint or endpoints which authoritatively describe which local access services may be used to access a resource. The result of authority resolution is a list of local access endpoints identified by one or more URIs and supporting at least one local access protocol. Once an XRI resolver has completed authority resolution, it is up to the calling application to choose one of these endpoints and access it using the desired local access service.

Figure 1 illustrates these two phases of XRI resolution:

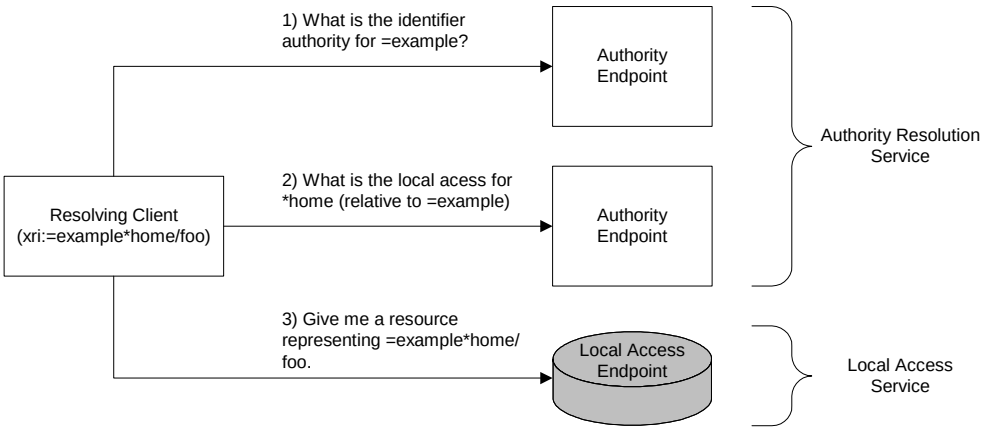


Figure 1: Phases of Resolution

2.1.3 URI vs. XRI Authorities

As described in 2.2.1 of **[XRISyntax]**, URI/IRI and XRI authorities have different syntactic structures, partially due to the higher layer of abstraction represented by XRI authorities. For this reason, XRI authorities are resolved into authority descriptor documents sub-segment by sub-segment as described in section 2.2, while URI/IRI authorities, since they are based on DNS names or IP addresses, are resolved into an authority descriptor through a special HTTP request based on the DNS name or IP address.

2.1.4 XRI Metadata Reserved for XRI Resolution

As defined in section 2.2.1.2 of **[XRISyntax]**, the GCS symbol "$" is reserved for special identifiers assigned by XRI TC specifications, other OASIS specifications, or other standards bodies to describe XRIs, XRI resolution, or other special characteristics of XRI-identified resources. (See also **[XRIMetadata]**.)

Within the "$" namespace, the identifier "$res" is reserved for identifiers assigned by this XRI resolution specification. Table 1 summarizes these identifiers.

Identifier	Use	See Section
xri://$res*schema	XML namespace for XRI resolution schemas	2.2.3
xri://$res*authres	Namespace for authority resolution protocol types	2.2.5

Copyright © OASIS Open 2005. All Rights Reserved.

xri://$res*localaccess	Namespace for local access protocol types	2.4.1
xri://$res*types	Namespace for resource representation types	2.4.2.2
xri://$res*trusted	Namespace for trust mechanisms	2.2.3 and 3

Table 1: Special identifiers reserved for XRI resolution.

2.2 XRI Authority Resolution

2.2.1 Overview

XRI authority resolution is an iterative process that resolves the qualified sub-segments within the XRI authority segment from left to right. A qualified sub-segment is either: a) a global context symbol as defined in section 2.2.1.2 of **[XRISyntax]** or b) a sub-segment as defined in section 2.2.3 of **[XRISyntax]** *together with its preceding syntactic delimiter* ("*" or "!"). Note that in the latter case a qualified sub-segment always includes the syntax delimiter even if it was optionally omitted in the original XRI (see section 2.2.3 of **[XRISyntax]**).

Each qualified sub-segment is resolved in the context of the qualified sub-segment immediately to the left. The first (or leftmost) qualified sub-segment specifies the root of the identifier community. Each XRI community, as identified by a GCS character or top-level cross-reference, provides one or more network endpoints (HTTP or HTTPS URIs) that answer resolution requests at the root level.

This left to right resolution results in a "chain" (an ordered list) of XRI Descriptor documents ("XRIDs"), each one associated with a particular sub-segment of the XRI authority segment of the XRI being resolved. Each HTTP request in XRI authority resolution may resolve one or more sub-segments at a time. Each HTTP response therefore may contain one or more than XRI Descriptor elements in a XML container document. The number of sub-segments resolved depends on:

- Whether the resolution request is done in "proxy" mode
- How many sub-segments the resolving client presents to a responding XRI Authority
- Configuration, policy, and state of the responding XRI Authority (ie previously cached requests).

No matter how the XRI Descriptor documents are retrieved, a successful XRI Authority resolution results in an ordered chain of XRI Descriptor documents.

2.2.2 The Chain of XRI Descriptors

The chain of XRI Descriptors associated with a XRI Authority identifier begins with an XRI Descriptor associated with the community root subsegment ("root XRID"). This subsegment is either a GCS character or a cross-reference. In either case, the root XRID will contain references to one or more Authority resolution services at which the community root authority will resolve one or more sub-segments after the root sub-segment. The resolving client adds to the chain by resolving one or more sub-segments via the community root authority, parses the resulting XRIDs, and proceeds to the next unresolved XRI Authority identifier sub-segment until the entire XRI Authority identifier is resolved. This document specifies one Authority resolution protocol built on HTTP (and HTTPS), and identified by the XRI "xri://$res*authres/XRIA" and defined in section 2.2.5.2. Other authority resolution services may be defined in the future by other documents.

Each XRI Descriptor in the chain contains HTTP or HTTPS URIs (or other URIs, depending on the authority resolution service type) which point to the next authority at which resolution can be performed for the next sub-segment. (Note that this will be true even if the resolving authority performs *lookahead resolution*, where an authority resolves multiple subsegments on on behalf of

Copyright © OASIS Open 2005. All Rights Reserved.

the resolving client, possibly even beyond those for which the authority is actually authoritative. This allows both the client and server the ability to cache intermediate results for future resolution requests.)

The last XRI Descriptor in the chain typically would provide the available local access service protocol(s) as discussed in section 2.4. In addition, this XRI Descriptor can provide a mapping of other XRIs that are synonymous to the resolved XRI authority.

All three options—next authority, local access, or synonyms—may be available at every step of resolution. For example, the XRI authority identifier "@a*b*c" may be the prefix to another XRI authority with the XRI "@a*b*c*d". Or "@a*b*c" may be a local access endpoint itself, in which case its XRI Descriptor will contain references to local access services. Finally, this XRI Descriptor can also assert that the identifier "xri:@a*b*c" maps to the identifier "xri:@!1!2!3" in order to provide resolvers or caches with an equivalent persistent XRI.

Figure 2 below depicts the relationships between authority descriptors, authorities, and authority identifier subsegments.

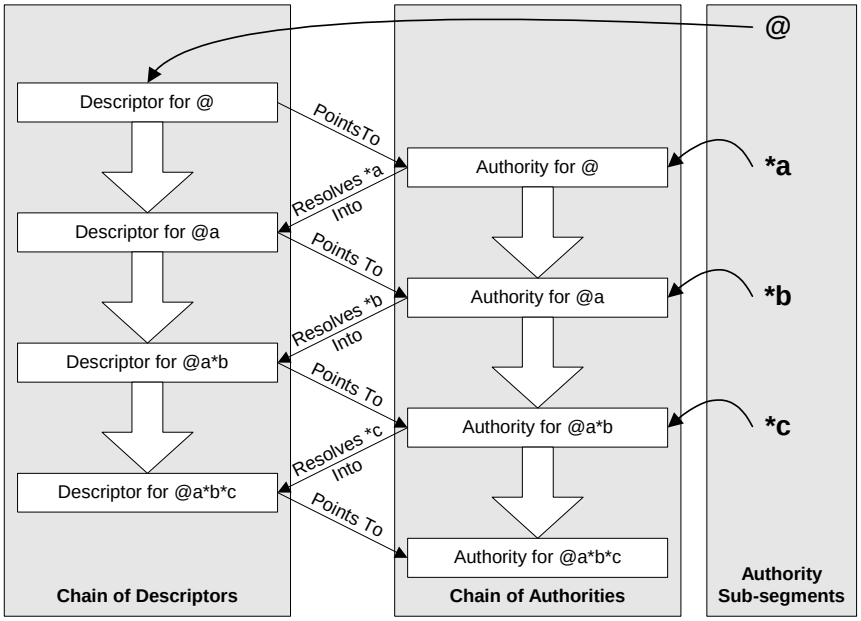


Figure 2: Descriptors, Authorities and Sub-segments for @a*b*c

2.2.3 XRI Descriptors

To provide a straightforward, flexible resolution mechanism, XRI authority endpoints are described using a simple XML element with a very flexible content model. Its only purpose is to provide the data and metadata necessary to support delegated resolution and access of XRI-identified authorities and resources.

The formal XML Schema definition of an XRI Descriptors is provided in Appendix B. The following example illustrates the fields defined in this schema:

```
<XRIDescriptors xmlns="xri://$res*schema/XRIDescriptors">
    <XRIDescriptor xrid:id="first">
        <Resolved>*foo</Resolved>
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
        <AuthorityID>urn:uuid:c9f812f3-6544-4e3c-874e-
d3ae79f4ef7b</AuthorityID>
        <Expires> 2002-05-30T09:30:10-06:00</Expires>
        <Authority>
            <AuthorityID>urn:uuid:f0502a17-4503-4463-8516-
f1225b330e4d</AuthorityID>
            <Type>xri://$res*authres/XRIA</Type>
            <URI>http://xri.example.com</URI>
            <URI>https://xri.example.com</URI>
        </Authority>
        <Service>
            <Type>xri://$res*localaccess/X2R</Type>
            <URI>http://xri.example.com</URI>
            <MediaType>application/rdf+xml</MediaType>
        </Service>
        <Service>
            <Type>xri://$res*localaccess/X2R</Type>
            <URI>http://pictures.xri.example.com</URI>
             <MediaType>image/jpeg</MediaType>
        </Service>
        <Synonyms>
            <Internal>xri://@!1!2!3</Local>
            <External>xri://@!4!5!6</External>
        </Synonyms>
        <TrustMechanism>xri://$res*trusted/None</TrustMechanism>
    </XRIDescriptor>
    Other XRIDescriptor elements here
</XRIDescriptors>
```

All schema elements in the basic XML Descriptor are in the XML namespace "xri://$res*schema/XRIDescriptor". Following are the elements and attributes that comprise the XRIDescriptor document type (all XPATHs are relative to the enclosing XRIDescriptors document element):

XRIDescriptor

1 or more within the XRIDescriptors container. Has an "xrid:id" attribute to uniquely identify this element within the containing XRIDescriptors document.

XRIDescriptor/Resolved

0 or more. Expresses the qualified sub-segment whose resolution results in this XRIDescriptor element. This field is used in conjunction with Digital Signatures to provide secure resolution (as defined in section 3). This field may also be useful for debugging or auditing purposes.

XRIDescriptor/AuthorityID

0 or more. A unique identifier for the authority that produced this XRI Descriptor.

XRIDescriptor/Expires

0 or 1. The UTC time at which this document MUST no longer be relied upon. A resolver MAY discard this Descriptor before the time indicated in this result. If the HTTP transport caching semantics specify an expiry time which is earlier than the time expressed in this attribute, then the "XRIDescriptor" document MUST no longer be relied upon after the expiry time declared in the HTTP headers per section 13.2 of **[RFC2616]**.

XRIDescriptor/Authority

Copyright © OASIS Open 2005. All Rights Reserved.

0 or more. Describes an authority resolver service associated with the resolved XRI authority ID. The next next qualified subsegment in the authority can be resolved at this service endpoint.

XRIDescriptor/Authority/AuthorityID

0 or more. A unique identifier for the authority being described in the Authority element.

XRIDescriptor/Authority/Type

0 or more per Authority element. Indicates the type of authority service being described. This specification defines one authority resolution services: “xri://$res*authres/XRIA” (XRI Authority resolution as described in section 2.2.5) which is the default value if this element does not appear.

XRIDescriptor/Authority/URI

1 or more per Authority element. Indicates the transport level URI where the authority resolution service described may be accessed. For the services defined in this document, this URI MUST be an HTTP or HTTPS URI. Future extensions may use other transport protocols. Each URI element has a attribute called “trusted” that indicates whether or not the particular service endpoint provides trusted resolution. The trust mechanism is described in the TrustMechanism element in the XRI Descriptor.

XRIDescriptor/Service

0 or more. Describes a local access service endpoint associated with the resolved XRI.

XRIDescriptor/Service/Type

0 or more per Service element. Indicates the type of local service being described. This specification defines one service: “xri://$res*localaccess/X2R” (The XRI X2R local access resolution service as defined in section 2.4.2). This is the default value if the Service element is not present.

XRIDescriptor/Service/URI

1 or more per Service element. Indicates the transport level URI where the service described may be accessed. For the services defined in this document, this URI MUST be an HTTP or HTTPS URI. Future extensions may use other transport protocols.

XRIDescriptor/Service/MediaType

0 or more. The media type of content available at this service. If this element is not present, then no assumption can be made about the type of data available at this endpoint. The content of this attribute must be of the form of a media type as defined in **[RFC2046]**. This element may appear multiple times to indicate multiple media types available through this local access service.

XRIDescriptor/Synonyms

0 or 1. Contains statements about the relationship of the resolved XRI authority identifier to other XRI authority identifiers.

XRIDescriptor/Synonyms/Internal

0 or more. Represents an XRI which the authority described in the Descriptor may also be known as. Must be an absolute XRI (“absolute-xri” in the ABNF, section 2.2 of **[XRISyntax]**)

Internal synonym XRIs may be used, for example, to assert that a XRI authority known by a reassignable XRI may also be known by one or more persistent XRIs, or by a different reassignable XRI than the one that is being resolved. Both cases may be particularly useful in populating or querying a cache.

Copyright © OASIS Open 2005. All Rights Reserved.

XRIDescriptor/Synonyms/External

0 or more. Represents an alternative XRI that is described by another, external, authority. Must be an absolute XRI ("absolute-xri" in section 2.2 of **[XRISyntax]**) The “external synonym” XRI serves a slightly different purpose than an internal synonym XRI. Resolution of an internal synonym XRI typically results in an XRIDescriptor containing the same information as the one in which the internal synonym element appears. Resolution of an external synonym, on the other hand, typically results in an XRIDescriptor containing different information. External synonyms are used, for example, in XRI redirects, described in Section 3.2.8. It can also be used to identify an alternative source of local access descriptors if those in the current XRIDescriptor do not satisfy the needs of the client.

XRIDescriptor/TrustMechanism

0 or 1. Identifies the mechanism for trusted resolution associated with this Descriptor. The specification defines two values: “xri://$res*trusted/XRITrusted” (for Trusted Resolution as described in section 3) and “xri://$res*trusted/None” (for generic resolution as described here in section 2). If this element does not appear, then the default value is “xri://$res*trusted/None” indicating no specific trust mechanism.

XRI Descriptor documents have an “open schema” that allows other elements and attributes from other namespaces to be added throughout. These points of extensibility can be used to deploy new identifier authority or local access resolution schemes.

The trusted resolution mechanism defined in this document is implemented through additional elements as described in section 3.

2.2.4 Starting the Chain of Descriptors with the Root XRID

With an XRI authority, the first qualified sub-segment corresponding to the community root may be a global context symbol (GCS) or a cross-reference. In either case, the associated community must have published an XRI Descriptor, called the “root XRID”, that contains one or more Authority/Service elements with URIs. Unless there exists a Authority/Service/Type element and it contains a value other than “xri://$res*authres/XRIA” (which is the default authority resolution service type and indicates the use of the authority resolution protocol described in this document), the URI element(s) must contain HTTP or HTTPS URIs declaring the root authority resolvers for the community. The root XRID is known *a priori* and is part of the configuration of a resolver, similar to the configuration of root DNS servers in a DNS resolver.

It is important to note that if the sub-segment following the GCS characterdoes not begin with a exclamation (meaning it is not a persistent identifier), then a asterisk (“*”) is implied, and a asterisk must be added when constructing the qualified sub-segment. Table 2 and Table 3 demonstrate the parsing of such a sub-segment in the case of a GCS character and a cross-reference, respectively.

XRI	xri:@example*internal/foo
XRI Authority	@example*internal
Identifier Community	@
First Qualified Sub-segment Resolved	*example

Table 2: Parsing the first sub-segment of an XRI that begins with a global context symbol character.

Copyright © OASIS Open 2005. All Rights Reserved.

XRI	xri:(http://www.example.com)*internal/foo
XRI Authority	(http://www.example.com)*internal
Identifier Community	(http://www.example.com)
First Qualified Sub-segment Resolved	*internal

Table 3: Parsing the first sub-segment of an XRI that begins with a cross-reference.

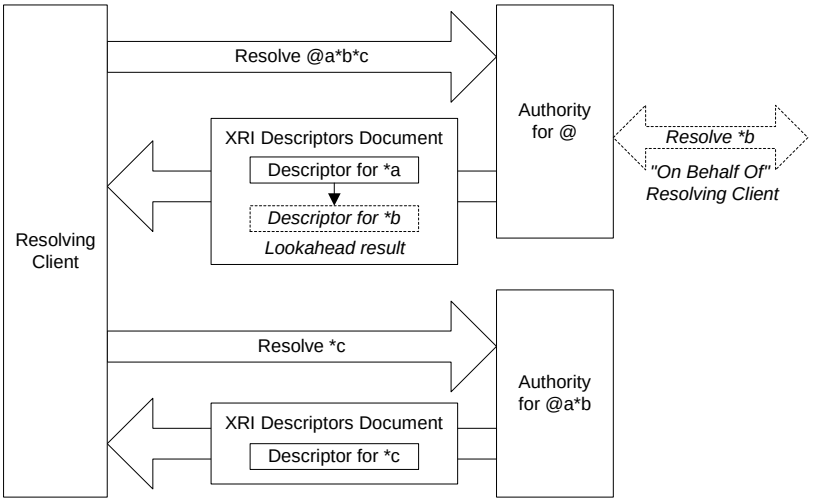
2.2.5 Default HTTP(S)-based Authority Resolution Service

This section defines the default authority resolution service based on HTTP and HTTPS. It is identified in the XML Descriptor Type element by the value “xri://$res*authres/XRIA”.

This service is intiated using the root XRID representing the community root authority as described in the previous section. The process for determining the next authority descriptor (and thus discovering the next authority to communicate with) begins by constructing the “Next Resolution URI”. This URI is used for resolving one or more of the remaining XRI authority identifier sub-segments.

This authority resolution service allows a client to present multiple authority subsegments in one transaction with an authority. If a resolving client presents multiple sub-segments to an authority resolver, the authority resolver may perform the authority resolution steps for the additional sub-segments on behalf of the resolving client. Any resolution beyond the first sub-segment is considered “lookahead” and is performed on behalf of the resolving client. In this case, the authority resolver acts as a resolving client to the authority resolvers corresponding to the “lookahead” segments.

If the authority performs lookahead resolution, the authority resolver will return an ordered list of XRI Descriptor elements in an XRIDescriptors document. Each XRI Descriptor corresponds to the sub-segments the authority resolver resolved, in the same order as they appear in the XRIDescriptors document. It is up to the authority resolver to determine how many sub-segments to resolve on behalf of the resolving client. The authority is under no obligation to resolve more than the first sub-segment (which it must resolve because it is the responsible authority).



Copyright © OASIS Open 2005. All Rights Reserved.

Figure 3: Authority Resolution with Lookahead

Figure 3 demonstrates a resolving client requesting lookahead resolution for the XRI authority "@a*b*c". The "@" authority is willing to resolve "@a*b" on behalf of the resolving client. The authority can accomplish this either by acting as an XRI resolving client itself, or by examining a cache it may have built through previous resolutions. In this example, it is willing or able to resolve only the descriptors for "@a*b". Thus, the resolving client must resolve "*c" itself. The resolving client knows the "@" authority only resolved two segments (*a and *b) because it only returned two XRI Descriptors corresponding to those sub-segments.

If the authority does not resolve the entire XRI Authority identifier presented, the resolving client MUST continue the authority identifier resolution process itself. At any stage, however, the resolving client may request that an authority resolve the entire unresolved portion of the XRI Authority segment. For example, in Figure 3, if the "@" authority had refused to do any lookahead, the resolving client could have asked the "@*a" authority to resolve the unresolved "*b*c" portion of the XRI authority segment.

2.2.5.1 Determining the URI for the Next Resolution Step

Before each authority resolution step is performed, a URI must be constructed at which the default HTTP/HTTPS authority resolution protocol is performed. At each step, this "Next Authority URI" is used to resolve a sub-segment in the current context. Initially the current context is the root authority, and the current context shifts to another authority each time a resolution step is performed. After lookahead resolution the current context is the last authority whose identifying sub-segment was resolved.

The "Next Resolution URI" is constructed from two pieces of information:

- The XRI Authority URI extracted from the XRI Descriptor corresponding to the current context,
- The list of qualified sub-segments that are requested to be resolved by the next authority. Note that this list of sub-segments which always begins with an XRI syntax delimiter ("*" or "!") (see the clarification regarding cross-references in section 2.2.7).

The URI which forms the base of the Next Authority URI is the value of a URI element found at element path **XRIDescriptor/Authority/URI** in the XRI Descriptor. If the path portion of this URI does not end with a "/" character, one must be appended before proceeding. The URI normal form (section 2.3.1 of **[XRISyntax]**) of the qualified sub-segment being resolved is then appended to the path portion of the URI. As noted above, if there is no separator character preceding the sub-segment, a "*" MUST be added when creating the qualified sub-segment.

For example, when resolving the "c" sub-segment of "xri:@a*b*c", if the XRI Authority URI resulting from the resolution of "xri:@a*b" is "http://example.com/xri-authority/", then the Next Authority URI is the concatenation of "http://example.com/xri-authority/" with "*c", yielding "http://example.com/xri-authority/*c". An HTTP GET request is made to this URI, and the next XRI Descriptor for the context "xri:@a*b*c" is retrieved.

As shown in Figure 3, a resolving client may attempt to resolve the entire authority resolution identifier (or any leading subset of it) at once. If the XRI Authority resolver URI for "xri:@" is determined (by configuration) to be "http://at.example.com/xri-authority/", then the Next Authority URI would be the concatenation of "http://at.example.com/xri-authority/" with "*a*b*c", yielding "http://at.example.com/xri-authority/*a*b*c" . As described above, this resolving authority may choose to only provide Descriptors for *a, *a and *a*b, or *a*b*c.

Construction of the Next Authority URI is more formally described in this pseudo-code which is resolving a "sub-segment-list" (a list of one or more XRI authority identifier sub-segments) via a HTTP URI called "xa-uri":

Copyright © OASIS Open 2005. All Rights Reserved.

```
xa-uri = xri-authority-uri

if (path portion of xa-uri doesn’t end in “/”):
    append “/” to path portion of xa-uri

if (sub-segment-list isn’t preceded with “*” or “!” separator):
    xa-uri = xa-uri + “*”
else:
     xa-uri  = xa-uri + separator

xa-uri = append uri-escape(sub-segment-list) to path portion of xa-uri
```

2.2.5.2 XRI Authority Identifier Sub-segment Resolution Protocol

Once the Next Authority URI is constructed, an HTTP or HTTPS GET request is made using this URI. Each GET request results in either a 2XX or 304 HTTP response. The HTTP request SHOULD contain an Accept: header with the value of “application/xrid+xml”. See 3.3.3 for a different value which may appear in the Accept header during trusted resolution.

For a successful resolution, the HTTP/HTTPS response MUST either contain either: a) 2XX response with an XRI Descriptors document with a list of one or more XRI Descriptors (corresponding to one or more resolved sub-segments) or, b) 304 response signifying that the cached version on the client is still valid (depending on the client’s HTTP request). HTTP caching semantics should be leveraged as much as possible to support the efficiency and scalability of this HTTP-based resolution system. The recommended use of HTTP caching headers is described in more detail in section 2.5.1.

Any ultimate response besides a HTTP 2XX or 304 should be considered an error in the resolution process. There is no restriction on intermediate redirects (i.e., 3XX result codes) or other result codes (e.g., a 100 HTTP response) that eventually result in a 2XX or 304 response through normal operation of **[RFC2616]**. Ultimately, the content of a successful response will be new XRI Descriptors for the qualified sub-segments being resolved.

If there are no more sub-segments, the final context (as described by the final XRI Descriptor retrieved) can be used for local access services as described in section 2.4.

2.2.5.3 Errors During Proxied and Lookahead Authority Resolution

Proxies and lookahead resolvers SHOULD return any HTTP error codes returned during resolution back to their clients. For example, if during resolution on behalf of a client, a proxy is given back a 404 error code from an authoritative server, it should return that 404 code to its client.

When encountering an HTTP error code, the proxy or lookahead resolver SHOULD return an XRIDescriptors document in the body of the HTTP error response. This XRIDescriptors document SHOULD contain the list of XRI Descriptor elements corresponding to the sub-segments successfully resolved or retrieved from cache. For example, if a proxy is asked to resolve @a*b*c, successfully resolves @a*b, and receives a HTTP 404 on resolving *c, it should return an HTTP 404 response to its client, with XRI Descriptor elements for @, *a, and *b. This will indicate to the resolving client that *c is the sub-segment causing the 404 response.

Resolving clients SHOULD consider the XRI Descriptor elements of a HTTP response to be valid cacheable responses useable in other resolutions (if a client does caching in the first place). All other rules about these Descriptor elements, including those specified in Section 3 below apply to these Descriptors.

Copyright © OASIS Open 2005. All Rights Reserved.

2.2.6 Examples

2.2.6.1 Authority Resolution Without Lookahead

Following is an example of resolving the authority portion of an XRI without lookahead resolution. That is, for each resolution step, the resolving client requests resolution of only one authority sub-segment of the following XRI :

```
xri://=example*home*base/foo*bar
```

Assume that the URI for the "=" global context symbol is "http://equals.example.org/xri-resolve" (found in **XRIDescriptor/Authority/URI** of the XRI Descriptor for this community). As explained in section 2.2.4, this information, which provides a starting point for resolution, is known *a priori* and is part of the configuration of the resolver.

Resolving "=example"

The following HTTP request is made to "equals.example.org":

```
GET /xri-resolve/*example HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:31 GMT
Accept: application/xrid+xml
<other HTTP headers>
```

The following HTTP response is received from "equals.example.org" (the content has changed since "Fri, 31 Oct 2003 19:43:31 GMT"):

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<other HTTP headers>

<XRIDescriptors xmlns="…">
<XRIDescriptor xmlns="…">
<Resolved>*example</Resolved>
<Authority>
<URI>
http://xri.example.com/xri-resolve/
</URI>
</Authority>
<Service>…</Service>
</XRIDescriptor>
</XRIDescriptors>
```

Resolving "=example*home"

Appending the next qualified sub-segment "*home" to the URI "http://xri.example.com/xri-resolve/" yields the URI "http://xri.example.com/xri-resolve/*home", and the following HTTP request is made to xri.example.com:

```
GET /xri-resolve/*home HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:32 GMT
Accept: application/xrid+xml
<other HTTP headers>
```

The following HTTP response is received from xri.example.com:

Copyright © OASIS Open 2005. All Rights Reserved.

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
If-Modified-Since: Fri, 31 Oct 2003 19:43:32 GMT
<other HTTP headers>

<XRIDescriptors xmlns="…">
<XRIDescriptor xmlns="…">
<Resolved>*home</Resolved>
<Authority>
<URI>
http://xri.othersite.com/xri-resolve/*home/
</URI>
</Authority>
<Service>…</Service>
…
</XRIDescriptor>
</XRIDescriptors>
```

Resolving "=example*home*base"

Appending the next qualified sub-segment "*base" to the URI "http://xri.othersite.com/xri-resolve/*home/" gives the URI "http://xri.othersite.com/xri-resolve/*home/*base":

```
GET /xri-resolve/*home/*base HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:32 GMT
Accept: application/xrid+xml
<other HTTP headers>
```

The following HTTP response is received from xri.othersite.com:

```
200 OK HTTP/1.1
Content-type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:33 GMT
<other HTTP headers>

<XRIDescriptors xmlns="…'>
<XRIDescriptor xmlns="…">
<Resolved>*base</Resolved>
<Service>
<Type>
xri://$res*localaccess/X2R
</Type>
<URI>
http://xri.othersite.com/xri-local/base/
</URI>
<URI>
https://xri.othersite.com/xri-local/base/
</URI>
</Service>
…
</XRIDescriptor>
</XRIDescriptors>
```

The result of the final XRI authority resolution step is the set of HTTP and HTTPS URIs shown in the "Service" element above that can be used for local access services (specifically, the X2R local access service as identified by the `xri://$res*localaccess/X2R` type).

Copyright © OASIS Open 2005. All Rights Reserved.

2.2.6.2 Authority Resolution with Lookahead

The following example shows the interaction between a client and server when resolving the authority identifier for the following XRI using lookahead resolution:

```
xri://=example*home*base/foo*bar
```

Assume that the URI for the "=" global context symbol is "http://equals.example.org/xri-resolve" (found in **XRIDescriptor/Authority/URI** of the XRI Descriptor for this community). As explained in section 2.2.4, this information, which provides a starting point for resolution, is known *a priori* and is part of the configuration of the resolver.

In this example, the client will request lookahead resolution of all unresolved authority sub-segments at each authority.

Resolving "=example*home*base"

The following HTTP request is made to "equals.example.org":

```
GET /xri-resolve/*example*home*base HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:31 GMT
Accept: application/xrid+xml
<other HTTP headers>
```

The following HTTP response is received from "equals.example.org" (the content has changed since "Fri, 31 Oct 2003 19:43:31 GMT"). This assumes that the "equals.example.org" authority has cached or performs its own resolution to retrieve the descriptor for =example*home:

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<other HTTP headers>

<XRIDescriptors xmlns="…">
<XRIDescriptor xmlns="…">
<Resolved>*example</Resolved>
<Authority>
<URI>
http://xri.example.com/xri-resolve/
</URI>
</Authority>
<Service>…</Service>
</XRIDescriptor>
<XRIDescriptor xmlns="…">
<Resolved>*home</Resolved>
<Authority>
<URI>
http://xri.othersite.com/xri-resolve/*home/
</URI>
</Authority>
<Service>…</Service>
</XRIDescriptor>
</XRIDescriptors>
```

Note that the order of XRI Descriptor elements is significant. The second XRI Descriptor is understood to apply only in the context of the "=example" authority. That is, the second Descriptor describes the authority identified "*home" within the "=example" namespace.

The resolving client, assuming it trusts the resolver's response (see section 3 for more details on trusted resolution), resolves the "*base" authority sub-segment at the authority URI "http://xri.othersite.com/xri-resolve/*home/" as identified in the last Descriptor of the previous resolution result. The following HTTP request is made to "xri.othersite.com":

Copyright © OASIS Open 2005. All Rights Reserved.

```
GET /xri-resolve/*home/*base HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:31 GMT
Accept: application/xrid+xml

<other HTTP headers>
```

The following HTTP response is received from xri.othersite.com:

```
200 OK HTTP/1.1
Content-type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:33 GMT
<other HTTP headers>

<XRIDescriptors xmlns="…'>
<XRIDescriptor xmlns="…">
<Resolved>*base</Resolved>
<Service>
<Type>
xri://$res*localaccess/X2R
</Type>
<URI>
http://xri.othersite.com/xri-local/base/
</URI>
<URI>
https://xri.othersite.com/xri-local/base/
</URI>
</Service>
…
</XRIDescriptor>
</XRIDescriptors>
```

Note that the resulting three XRI Descriptor elements (two from the first HTTP resolution at equals.example.org and the one from xri.othersite.com) are the exact same three XRI Descriptors as those retrieved from the separate resolution requests showed in section 2.2.6.1.

2.2.7 Resolving Cross-References in XRI Authorities

A sub-segment within an XRI authority segment may be a cross-reference. Resolving a cross-reference is identical to resolving any other sub-segment because, from the standpoint of generic XRI resolution, the cross-reference is considered opaque. In other words, the value of the cross-reference (including the parentheses) is the literal value of the sub-segment for the purpose of authority resolution.

An exception to the above is a cross-reference that begins with the GCS symbol for annotations ("!"). Such a cross-reference and the delimiter that precedes it MUST be ignored entirely during resolution.

Table 4 provides several examples using the default authority resolution service definition. In each of these examples, sub-segment "b" resolves to an XRI Authority URI of "http://example.com/xri-authority/".

Cross-reference type	Example XRI	Next Resolution URI after resolving "xri://@:a:b"
Absolute XRI	xri://@!a!b!(@!1!2!3)*e/f	http://example.com/xri-authority/!(@!1!2!3)
Absolute URI	xri://@!a!b.(mailto:jd@example.com)*e/f	http://example.com/xri-

Copyright © OASIS Open 2005. All Rights Reserved.

		authority/*(mailto:jd@example.com)
Relative XRI	xri://@!a!b!(c*d)*e/f	http://example.com/xri-authority/!(c*d)

Table 4: Examples of the Next Authority URIs constructed using different types of cross-references.

Note that specific identifier communities may specify special resolution rules for specific types of cross-references, but such extensions are out of scope for this specification.

2.2.8 XRI Redirects

It is possible for an XRIDescriptor to contain an XRIDescriptor/Synonyms/External element but to lack expected authority resolution or local access services descriptions. In other words, it is possible to have insufficient information to continue resolution, but to have an alternative XRI for the current XRI authority in an XRIDescriptor/Synonyms/External element. This is called an "XRI Redirect" - the XRIDescriptor is effectively redirecting to a new XRI Authority. In this case, the unresolved portion of the original XRI (i.e. the XRI being resolved) is added to contents of the /XRIDescriptor/Synonyms/External element to create a new XRI. This new XRI is then resolved as described in Section 2.2.2 of this document.

The example in Section 2.2.6 demonstrates the resolution of xri://=example*home*base/foo*bar. The first request is to "equals.example.org". If the response had been as follows, a new XRI would be constructed as xri://=example2*home*base/foo*bar and the resolution process would start again with this new XRI.

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<other HTTP headers>

<XRIDescriptors xmlns="...">
<XRIDescriptor>
  <Resolved>*example</Resolved>
  <Synonyms>
     <External>
        xri://@example2
     </External>
   </Synonym>
</XRIDescriptor>
</XRIDescriptors>
```

If the original XRI has additional sub-segments in the XRI Authority component and the XRIDescriptor/Synonyms/External element contains a local- path component, the client SHOULD consider this an error condition and fail. In the example above, if the response had been as follows, the resulting XRI would be xri://@example2/path*home*base/foo. Unless the client application has specific reasons to believe otherwise, this is an error.

Copyright © OASIS Open 2005. All Rights Reserved.

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<other HTTP headers>

<XRIDescriptors xmlns="...">
<XRIDescriptor>
  <Resolved>*example</Resolved>
  <Synonyms>
    <External>
      xri://@example2/path
    </External>
   </Synonym>
   …
</XRIDescriptor>
</XRIDescriptors>
```

2.2.9 Proxied Resolution

In many cases, it is desirable for a XRI authority resolver to act on behalf of other systems and provide complete authority resolution for those other systems. Proxied resolution defines a service that allows a client to provide an entire XRI authority identifier and result in an XRI Descriptor for the authority identified by that XRI authority identifier. The client must still parse the XRI Descriptor and perform local access. The resolving client must rely on the proxy's application of trusted resolution policies in deciding whether or not to use an XRI Descriptor during the proxy's resolution process. However, the resolving client can still verify the results of the XRI Descriptors returned by the proxy resolver when the proxy resolver returns the XRIDescriptors document. However, no behavior is defined if the proxy returns XRI Descriptors that are invalidly signed or otherwise untrustworthy according to the resolving client's trust policies.

Note that the proxy resolution service does not provide a complete XRI-to-resource mapping service. Because XRI resolution does not define a single local access protocol, there is no single proxy resolution service that encompass both authority and local access resolution in one step. Such a proxy resolution service could be defined however, on a per-local-access basis.

The authority proxy resolution service is simply an HTTP GET performed on a URL constructed by concatenating the base proxy URL and the URI-escaped XRI authority identifier. As with local access, if the base proxy URL does not contain a trailing slash, one is inserted between the base URL and the URI-escaped authority identifier. The proxy answering this request must perform XRI authority resolution and return with an XRI Descriptors document containing the entire chain of XRI Descriptors for the authority, as described in section 2.2.2.

The following example assumes a URL for a local proxy as "http://proxy.corporate.com/xri-proxy" and demonstrates the proxied authority resolution for "xri://=example*home*base". An HTTP GET request is made to "proxy.corporate.com":

```
GET /xri-proxy/=example*home*base HTTP/1.1
<other HTTP headers>
```

The proxy resolver then performs authority resolution, behaving as a resolving client as described in section 1. After completion of this resolution process, the proxy resolver might produce the following HTTP response:

Copyright © OASIS Open 2005. All Rights Reserved.

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<other HTTP headers>

<XRIDescriptors xmlns="...">
<XRIDescriptor>
  <Resolved>=</Resolved>
  …
</XRIDescriptor>
<XRIDescriptor>
  <Resolved>*example</Resolved>
  …
</XRIDescriptor>
<XRIDescriptor>
  <Resolved>*home</Resolved>
  …
</XRIDescriptor>
<XRIDescriptor>
  <Resolved>*base</Resolved>
  <Service>
    <Type>
      xri://$res*localaccess/X2R
    </Type>
    <URI>
      http://xri.othersite.com/xri-local/base/
    </URI>
    <URI>
      https://xri.othersite.com/xri-local/base/
    </URI>
  </Service>
  …
</XRIDescriptor>

</XRIDescriptors>
```

The resolving client can then parse this XRI Descriptor and extract the Local Access element from the last XRI Descriptor element. If the resolving proxy cannot resolve the entire authority identifier, it MUST return a 404 “Not found” error.

Note that proxy resolvers are uniquely positioned to take advantage of caching and SHOULD maximize the use of caching to shortcut resolution of the same authority sub-segments for multiple clients.

Additionally note that proxied resolution is very similar to lookahead resolution with the default authority resolution service. The differences between the two are:

- The authority for the first subsegment being resolved performs lookahead resolution. A proxy that doesn’t necessarily claim any authority for any of the segments being resolved performs proxy resolution.
- Lookahead resolution is performed on a subset of the list of authority sub-segments. Proxied resolution is performed on the entire authority identifier.
- Lookahead resolution never includes the initial authority sub-segment of an authority identifier (since at least the first authority performing lookahead resolution is identified by the first sub-segment). Proxied resolution always includes the first authority identifier sub-segment since proxied resolution resolves the entire authority identifier.

Copyright © OASIS Open 2005. All Rights Reserved.

2.3 IRI Authority Resolution

An IRI authority segment includes either a DNS name or an IP address that specifies the location of the endpoint with which to perform local access. This section defines the default protocol for retrieving an XRI Descriptor with the IRI authority form of authority.

This process consists of creating a HTTP URL out of the authority segment and performing a HTTP GET request on that HTTP URL which results in an XRIDescriptors document containing one XRI Descriptor for the authority. That XRI Descriptor is then used to retrieve Local Access URIs as in section 2.2.

The HTTP URI constructed from the request is constructed by extracting the entire IRI authority segment, and prepending the string “http://” to the front of it. An HTTP GET is performed with an HTTP Accept header containing only the following:

```
Accept: application/xrid+xml
```

An HTTP server would respond with the XRI Descriptors document for that authority. Trusted resolution is not defined for IRI Authority resolution.

The following example demonstrates how the authority for XRI xri://example.com/local*stuff would be resolved into an XRI Descriptor. The IRI authority would be extracted (“example.com”) and the following HTTP Request would be performed at the server example.com:

```
GET / HTTP/1.1
Accept: application/xrid+xml
<other HTTP headers>
```

The HTTP server acting as the authority might respond with the following HTTP response:

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<other HTTP headers>

<XRIDescriptors xmlns="...">
<XRIDescriptor>
  <Resolved>example.com</Resolved>
  <Synonyms>
    <External>
      xri://@example2/path
    </External>
   </Synonym>
   …
</XRIDescriptor>
</XRIDescriptors>
```

The use of IRI authorities provides backwards compatibility with the large installed base of DNS- and IP-identifiable resources. However because IRI authorities do not support the additional layer of abstraction and extensibility represented by authority syntax in XRI, IRI authorities are not recommended for new deployments of XRI identifiers.

2.4 Local Access

Local access is the process of interacting with a network endpoint to retrieve a representation or interact with a resource identified by an XRI.

2.4.1 Local Access Service Types

Any number of protocols may be used for local access. This specification defines an HTTP/HTTPS local access protocol given the name “X2R”. Other local access services could be defined such as an LDAP or DSML local access protocol that would specify the appropriate

Copyright © OASIS Open 2005. All Rights Reserved.

transformation of the XRI local part into an LDAP distinguished name (including normalization of the XRI local path to the LDAP distinguished name syntax.)

Work on such protocols is left to future specifications. To accommodate such work, this specification reserves a namespace, “$res*localaccess”, for enumerating local access service types.

2.4.2 The X2R Local Access Service

The X2R local access service is derived from the I2R service defined in section 4.3 or **[RFC2483]**. X2R services are available when the associated xri:Descriptor/xri:Service/xri:Type element contains the value “`xri://$res*localaccess/X2R`”.

X2R is essentially defined as the use of HTTP to interact with a resource using the full extent of the HTTP semantics as defined in **[RFC2616]**. Special attention should be paid to the semantics of the four main HTTP verbs: GET, PUT, POST, and DELETE. For example, clients performing local access typically would use GET when wishing to retrieve representations of a resource on the network.

This specification does not impose particular semantics beyond what is defined in **[RFC2616]**, but users of this specification are encouraged to review the **[REST]** architecture when building applications using XRIs. Local access is not limited to the REST model of interaction, however. For example, HTTP local access could be leveraged for the delivery of SOAP messages over HTTP POST, or via use of the GET HTTP verb as a generic read-only resolution infrastructure.

The HTTP/HTTPS local access binding defined in this section is flexible enough to be used for a variety of resources. It makes no assumptions about the type of resource identified by the XRI being resolved. The resource type must be established through the context in which the XRI was originally used (e.g. an XML document) or discovered through use of the HTTP local access protocol (e.g., through the HTTP Content-Type header).

2.4.2.1 Constructing a Local Access HTTP/HTTPS URI

This section defines the construction of URIs for the X2R local access service.

The HTTP/HTTPS URI with which to perform local access is constructed by concatenating the value of xri:Service[Type=’xri://$res*localaccess/X2R’]/URI from the XRI Descriptor (section 2.2.3) which results from the resolution of the authority part of the XRI. This URI is concatenated with the URI normal form of the relative-path of the XRI. If the URI from the XRI Descriptor does not terminate in a “/”, one MUST be inserted before the relative-path.

The following pseudocode describes the process for creating the concrete HTTP/HTTPS URI to which a local access request is made:

```
if (concrete-http-uri does not end in “/”):
   concrete-http-uri = localaccess-uri + “/”
else
   concrete-http-uri = localaccess-uri

concrete-http-uri = concrete-http-uri + uri-escape(relative-path)
```

The verb used in the resulting HTTP/HTTPS request may be any of the verbs defined in **[RFC2616]**, though not all verbs may be supported at every endpoint. All local access endpoints SHOULD support at least the GET verb, and this should return either a representation of the identified resource or metadata about the resource.

The full suite of HTTP content negotiation features is available to clients when performing local access. For example, if the local access service URI is “http://xri.example.com/xri-local”, then the following local access HTTP request for “xri://=example*home/foo*bar” could be made to “xri.example.com”:

Copyright © OASIS Open 2005. All Rights Reserved.

```
GET /xri-local/foo*bar HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:33 GMT
<other HTTP headers>
```

The following HTTP response should then be received from xri.example.com:

```
200 OK HTTP/1.1
Expires: Sat, 1 Nov 2003 19:43:33 GMT
Content-Type: text/plain
<other HTTP headers>

This is the result of a local access request.
```

2.4.2.2 Using a Cross-Reference to Specify a Representation Type

A cross-reference MAY be used to specify a desired resource representation type when performing local access. The namespace "$res*types" is reserved for this purpose. This specification does not enumerate such types; they are further defined in the *"XRI Metadata Specification"* **[XRIMetadata]**.

To specify a particular resource representation type using "$res*types" metadata, a "$res*types" cross-reference is appended to the XRI during a local access request. For example, an RDDL document could be specified by appending the cross-reference "($res*types/RDDL)".

The following example using the X2R local access service illustrates this technique. Assuming the original XRI being resolved is "xri://=example*home/foo*bar" and the local access URI is "http://xri.example.com/xri-local/", the following HTTP request would request the RDDL document describing this resource:

```
GET /xri-local/foo*bar/($res*types%2FRDDL) HTTP/1.1
<other HTTP headers>
```

Note that the cross-reference is escaped per the rules for the URI normal form of an XRI.

The resulting HTTP response might be:

```
200 OK HTTP/1.1
<cache-headers>
<other HTTP headers>

<content of representation of RDDL for xri:=example*home/foo*bar>
```

X2R local access servers MAY return a 404 HTTP status code if they do not have an appropriate representation of the resource, or if they do not recognize the use of the cross-reference to specify a representation type.

2.5 HTTP Headers

2.5.1 Caching

The full caching capabilities of **[RFC2616]** should be leveraged for both the default authority resolution service and the X2R local access service. Specifically, implementations of XRI resolution SHOULD implement the caching model described section 13 of **[RFC2616]**. In particular, the "Expiration Model" of section 13.2 SHOULD be used, as this requires the fewest round-trip network connections.

All servers providing identifier authority lookup responses SHOULD send the Cache-Control or Expires headers per section 13.2 of **[RFC2616]** unless there are overriding security or policy reasons that dictate otherwise.

Note that proxied and lookahead resolution may reduce the amount of http cache hits that occur during resolution. It is expected, however, that the benefit in lookahead and proxied resolution,

Copyright © OASIS Open 2005. All Rights Reserved.

with the reduction of round trip HTTP interactions, will more than compensate for the lack of HTTP caching benefits.

2.5.2 Location

In the default identifier authority resolution HTTP interaction, “Location” headers may be present per **[RFC2616]** (i.e., during 3XX redirects). Redirects SHOULD be made cacheable through appropriate HTTP headers.

During the X2R local access HTTP interaction, redirects may be returned, and the “Location” field may contain an HTTP/HTTPS URI or an XRI in URI normal form. This use of redirects constitutes a mapping facility that allows one XRI to resolve into another during local access. If the local access server is aware of the HTTP/HTTPS URI where the XRI may be accessed, it can provide a “Location” header containing an HTTP/HTTPS URI. In this case, it SHOULD provide an “X-XRI-Canonical” header (see below) to describe the XRI to which the redirection is targeting. If the local access server knows only of the target XRI, then it MUST return a redirection header (3XX code) with the “Location” field containing an XRI.

2.5.3 Content-Location

“Content-Location” may be used during local access where the resource being accessed is an “attribute” or “view” of another resource. This usually would occur in the case where metadata is being accessed using a trailing cross reference to an XRI value under the “$r.t” namespace (see section 2.4.2.2). Such a “Content-Location” header would specify where the resource itself may be accessible (rather than the metadata). This is not required and MUST NOT be required by resolving clients for proper operation. The content-location SHOULD be an HTTP/HTTPS URI if the local access server is aware of the HTTP/HTTPS location, otherwise it MAY be an XRI.

2.5.4 Content-Type

For default authority resolution, the “Content-type” header in the 2XX responses MUST contain the value “application/xrid+xml” or “application/xrid-t+xml”, specifying that the content is an XRI Descriptor (section 2.2.3) or a trusted XRI Descriptor (section 3.3.1).

For X2R local access, clients and servers MAY negotiate content type using standard HTTP content negotiation features. Whether or not this feature is used, however, the server MUST respond with an appropriate media type in the “Content-type” header.

2.5.5 X-XRI-Canonical

This header MAY be present only in HTTP/HTTPS redirects while performing the X2R local access service. Its purpose is to notify a resolving client that the redirect is occurring because the original XRI is a mapping to another XRI. The value of this header is the target XRI in URI normal form. This header MAY be present even when the Location: header is present and contains an XRI. This header SHOULD be present when the Location: header is present and contains a HTTP/HTTPS or other URI.

Form:

```
X-XRI-Canonical: <xri-in-uri-normal-form>
```

2.6 Other HTTP Features

HTTP provides a number of other features including transfer-coding, proxying, validation-model caching, etc. All of these features may be used insofar as they do not conflict with the required uses of HTTP described in this document.

Copyright © OASIS Open 2005. All Rights Reserved.

2.7 Caching and Efficiency

Resolution clients are encouraged to perform caching above the HTTP level in addition to at the HTTP level. For best results, however, resolution clients SHOULD be conservative with caching expiration semantics, including cache expiration dates. This implies that in a series of HTTP redirects, for example, the results of the entire process SHOULD only be cached as long as the shortest period of time allowed by any of the intermediate HTTP responses.

Because not all HTTP client libraries expose caching expiration to applications, identifier authorities and local access servers SHOULD NOT use cacheable redirects with expiration times which are relatively short compared to the expiration times of other HTTP responses in the authority resolution chain or local access interactions. In general, all XRI deployments should be mindful of limitations in current HTTP clients and proxies.

For XRI Descriptors, the cache expiration time may also be shortened by the expiration time provided in the XRI Descriptor at **XRIDescriptor/Expires** (if present). That is, if the expiration time in **XRIDescriptor/Expires** is sooner than the expiration time calculated from the HTTP caching semantics, then the XRI Descriptor SHOULD be discarded before the expiration time in **XRIDescriptor/Expires**. Note also that the SAML assertion present in trusted resolution may cause invalidation of a XRI Descriptor even before HTTP caching semantics or the expires header indicates that a Descriptor is stale.

With both application-level and HTTP-level caching, the resolution process is designed to have minimal overhead. In particular, because each qualified sub-segment of an authority identifier is described by a separate XRI Descriptor, each step of that resolution is an independent and the results are typically completely cacheable. For this reason, resolution of top-level (leftmost) qualified sub-segments, which are common to more identifiers, will naturally result in a greater number of cache hits than resolution of qualified sub-segments further to the right.

2.8 Points of Extensibility

The default authority resolution service and the X2R local access services both use extensible mechanisms such as HTTP and XML to provide maximum flexibility. Specifically, changes or additions can be made at the following points of extensibility:

- Specification of new Authority Resolution Service types
- Specification of new Local Access Service types
- Specification of new TrustMechanism types. A community of interest may have trust due to deployment of a limited number of trusted servers, for example, and may wish to express this explicitly in the TrustMechanism element.
- HTTP negotiation of content types, language, encoding, etc.
- Use of HTTP verbs such as POST, PUT and DELETE during local access.
- Use of HTTP redirects (3XX) or other response codes during identifier authority resolution or X2R local access.
- Insertion of new elements or attributes in the XRI Descriptor.
- Use of cross-references within XRIs, particularly for associating new types of metadata with a resource (see section 2.4.2.2 for an example).

Copyright © OASIS Open 2005. All Rights Reserved.

3 Trusted Resolution

3.1 Introduction

This section defines a method for achieving trusted authority resolution for XRIs with XRI authorities. This method is an extension of, and is compatible with, the resolution protocol defined in section 2 of this document.

This document does not provide a means to encrypt the contents of resolution requests and responses, nor does it provide a means for a responder to provide different responses for different requestors. These services may be provided by other security protocols used in conjunction with this specification, but confidentiality and client-authentication are explicitly out of scope of this document.

This section assumes the reader is familiar, at a minimum, with the ABNF defined in Appendix A of **[XRISyntax]** and the resolution protocol defined in section 2 of this document.

3.2 Overview and Example (Non-normative)

This section gives a brief overview and an example of trusted resolution using HTTP. Specific processing rules are defined in Section 3.3.

The basic approach to trusted XRI Authority resolution is simple. The client application requests resolution of one or more qualified sub-segments in the XRI Authority exactly as described in section 3.2 of **[XRISyntax]** with one change: a content type of “application/xrid-t+xml” is requested using the HTTP “Accept” header mechanism instead of “application/xrid+xml”. The XRI Authority responds with an XRIDescriptor that contains a digitally signed SAML **[SAML]** assertion. If the response does not contain a valid (as defined in section 3.2 of this document), digitally signed SAML assertion, trusted resolution may not proceed.

Section 3.2.5 of **[XRISyntax]** steps through resolution of the authority portion of

```
xri://=example*home*base/foo.bar
```

The example that follows shows the equivalent process using trusted resolution.

As in standard resolution, there is no defined discovery for the URI of the community root – it must be known *a priori* and is part of the configuration of the resolver. A good practice, and one followed by the global communities rooted on @ and =, is to publish an XRI Descriptor containing a valid SAML assertion signed by the community root. For this example, assume that the URI for the “=” global context symbol is http://equals.example.org/xri-resolve, found in the xri:XRIDescriptor/xri:Authority/`xri:URI` element of the XRI Descriptor for the global community rooted on =.

In trusted resolution, each XRI Authority is associated with an additional identifier called an AuthorityID. An AuthorityID is a URI uniquely associated with a particular XRI Authority. Each XRI Authority has one AuthorityID, and no two XRI Authorities have the same AuthorityID. The AuthorityID of the community root, like the community root’s URI, must be known and configured in advance. If the community root publishes an XRI Descriptor, the AuthorityID may be found in the xri:XRIDescriptor/xri:Authority/xri:AuthorityID element of the Descriptor describing the community root. For this example, assume that the AuthorityID for the “=” global context symbol is urn:uuid:498FB006-B9EF-4943-B10A-A71FC2ED1B89. For more information on `xri:AuthorityID`, see Section 3.3.3 below.

Finally, in trusted resolution, each XRI Authority is associated with some key used to verify digital signatures. The key for the community root must be known and configured in advance. If the

Copyright © OASIS Open 2005. All Rights Reserved.

community root publishes a signed XRI Descriptor, information about this key may be found in the xri:XRIDescriptor/xri:Authority/ds:KeyInfo element.

Note that the digital signatures in the following examples are for reference only. The digest values are not valid and the signatures will not verify.

Resolving “=example”

The following HTTP request is made to “equals.example.org”:

```
GET /xri-resolve/*example HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:31 GMT
Accept: application/xrid-t+xml
<other HTTP headers>
```

Example 1 – Request for =example

Notice the use of the Accept header using “application/xrid-t+xml”. The client is requesting a response that contains a signed SAML assertion. Also notice that “equals.example.org” was asked to resolve “*example”. The fact that “example” is part of both the XRI and the domain name in the URI is coincidental; there is no relationship between the two. Additionally, note that if the resolving client would accepted either a trusted or generic resolution, it could have used the following value for the Accept header: “application/xrid-t+xml, application/xrid+xml”.

The following HTTP response is received from “equals.example.org”:

```
200 OK HTTP/1.1
Content-Type: application/xrid-t+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<other HTTP headers>

<XRIDescriptors
  xmlns="xri://$r.s/XRIDescriptor">
<XRIDescriptor
  xrid:id="baec221f3c0f17f53ca6839989632056">
 <Resolved>*example</Resolved>
 <AuthorityID>urn:uuid:498FB006-B9EF-4943-B10A-A71FC2ED1B89
 </AuthorityID>
 <Authority>
  <URI xrid:trusted=”true”>http://xri.example.com/xri-resolve/</URI>
  <AuthorityID>urn:uuid:C5C9EFDF-A3BC-4301-88C6-B1AE0AD6DA77
  </AuthorityID>
  <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  …
  </ds:KeyInfo>
 </Authority>
 <TrustMechanism>xri://$res*trusted/XRITrusted</TrustMechanism>
 <saml:Assertion
   Version="2.0"
   ID="_ad9571ad-cd23-85e2-e928-abba20b6c424"
   IssueInstant="2004-07-01T00:46:02Z"
   xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml:Issuer>urn:uuid:498FB006-B9EF-4943-B10A-
A71FC2ED1B89</saml:Issuer>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <ds:SignedInfo>
     <ds:CanonicalizationMethod
       Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
     <ds:SignatureMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
     <ds:Reference URI="#baec221f3c0f17f53ca6839989632056">
     <ds:Transforms>
      <ds:Transform
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
        Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature" />
      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#">
       <ec:InclusiveNamespaces
         xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
         PrefixList="#default code ds kind rw saml samlp typens" />
      </ds:Transform>
     </ds:Transforms>
     <ds:DigestMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
     <ds:DigestValue>BSsnowZG5DYV0X0C8GAeB1cvLzw=</ds:DigestValue>
    </ds:Reference>
   </ds:SignedInfo>
   <ds:SignatureValue>
kE9p35G4mcombsqEztJMX1R3J26gwc4cbjSz5fUv3aVg3j/iLhrbf0qKywYNMLdQMjBRcCg
5N1l0
Kvv2UrgvQ5kgQ9dm7/563rRzKAaIQwMopZpTFli4eXw+nc8XEH+KnXdu/R9DHOg9k0BKIF6
BGk07
xC6Q9X+byQWenPjAZ1c=
   </ds:SignatureValue>
  </ds:Signature>
  <saml:Subject>
   <saml:NameID NameQualifier="urn:uuid:498FB006-B9EF-4943-B10A-
A71FC2ED1B89">
*example
   </saml:NameID>
  </saml:Subject>
  <saml:Conditions
    NotBefore="2004-06-01T00:00:00Z"
    NotOnOrAfter="2004-09-01T00:00:00Z" />
  <saml:AttributeStatement>
   <saml:Attribute Name="xri://$res*trusted/XRIDescriptor">

<saml:AttributeValue>#baec221f3c0f17f53ca6839989632056</saml:AttributeV
alue>
   </saml:Attribute>
  </saml:AttributeStatement>
 </saml:Assertion>
</XRIDescriptor>
</XRIDescriptors>
```

Example 2 – Response for =example

The response contains an `xri:XRIDescriptor/saml:Assertion` element that provides an assertion about the validity of the XRIDescriptor. For more information about SAML assertions in XRIDescriptors, see section 3.3.3.The response also contains an xri:XRIDescriptor/xri:Authority/ds:KeyInfo element. This required element tells the client that digital signatures by the described XRI Authority are to be verified using the indicated key. Also note that `xri:XRIDescriptor/xri:Resolved`, and an optional element in standard resolution, is required in trusted resolution. Finally, notice that two instances of `xri:AuthorityID` appear in the XRIDescriptor, one as a child of `xri:XRIDescriptor` and one as a child of xri:Authority. The child of `xri:XRIDescriptor` is the AuthorityID of the *describing* authority (the one making publishing this XRI Descriptor) and matches the expected AuthorityID of the community root (urn:uuid:498FB006-B9EF-4943-B10A-A71FC2ED1B89). The child of `xri:Authority` element contains the AuthorityID of the described XRI Authority (the authority being described within the xri:Service element). Responses from that XRI Authority will contain this AuthorityID as a child of `xri:XRIDescriptor`.

The client validates the signed SAML assertion as described in Section 3.2 before continuing.

Copyright © OASIS Open 2005. All Rights Reserved.

Resolving “=example*home”

Appending the next qualified sub-segment “*home” to the URI “http://xri.example.com/xri-resolve/” yields the URI http://xri.example.com/xri-resolve/*home. The Accept header with the value of “application/xrid-t+xml” used and the following HTTP request is made to xri.example.com:

```
GET /xri-resolve/*home HTTP/1.1
Accept: application/xrid-t+xml
<other HTTP headers>
```

Example 3 – Request for *home

The following HTTP response is received from xri.example.com:

```
200 OK HTTP/1.1
Content-Type: application/xrid-t+xml
If-Modified-Since: Fri, 31 Oct 2003 19:43:32 GMT
<other HTTP headers>

<XRIDescriptors
  xmlns="xri://$r.s/XRIDescriptor">
<XRIDescriptor
  xrid:id="1f81b6e0-b64b-1026-f1bc-c0a80b9d3f5b">
 <Resolved>*home</Resolved>
 <AuthorityID>urn:uuid:C5C9EFDF-A3BC-4301-88C6-B1AE0AD6DA77
 </AuthorityID>
 <Authority>
  <AuthorityID>urn:uuid:A9F28515-AB03-4883-8852-8EECB54CE1D5
  </AuthorityID>
  <URI xrid:trusted=”true”>
     http://xri.example.com/xri-resolve/*home/
  </URI>
  <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  …
  </ds:KeyInfo>
 </Authority>
 <Service>…</Service> <!-- Local Access Service -->
 <TrustMechanism>xri://$res*trusted/XRITrusted</TrustMechanism>
 <saml:Assertion
   Version="2.0"
   ID="_66f1f3e0-b64b-1026-34a4-c0a80b9d59c1"
   IssueInstant="2004-05-01T00:46:03Z"
   xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml:Issuer>urn:uuid:C5C9EFDF-A3BC-4301-88C6-
B1AE0AD6DA77</saml:Issuer>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <ds:SignedInfo>
     <ds:CanonicalizationMethod
       Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
     <ds:SignatureMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
     <ds:Reference URI="#1f81b6e0-b64b-1026-f1bc-c0a80b9d3f5b">
     <ds:Transforms>
      <ds:Transform
        Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature" />
      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#">
       <ec:InclusiveNamespaces
         xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
         PrefixList="#default code ds kind rw saml samlp typens" />
      </ds:Transform>
     </ds:Transforms>
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
     <ds:DigestMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
     <ds:DigestValue>BSsnowZG5DYV0X0C8GAeB1cvLzw=</ds:DigestValue>
    </ds:Reference>
   </ds:SignedInfo>
   <ds:SignatureValue>
kE9p35G4mcombsqEztJMX1R3J26gwc4cbjSz5fUv3aVg3j/iLhrbf0qKywYNMLdQMjBRcCg
5N1l0
Kvv2UrgvQ5kgQ9dm7/563rRzKAaIQwMopZpTFli4eXw+nc8XEH+KnXdu/R9DHOg9k0BKIF6
BGk07
xC6Q9X+byQWenPjAZ1c=
   </ds:SignatureValue>
  </ds:Signature>
  <saml:Subject>
   <saml:NameID NameQualifier="urn:uuid:C5C9EFDF-A3BC-4301-88C6-
B1AE0AD6DA77">
*home
   </saml:NameID>
  </saml:Subject>
  <saml:Conditions
    NotBefore="2004-06-01T00:00:00Z"
    NotOnOrAfter="2004-09-01T00:00:00Z" />
  <saml:AttributeStatement>
   <saml:Attribute Name="xri://$res*trusted/XRIDescriptor">

<saml:AttributeValue>#baec221f3c0f17f53ca6839989632056</saml:AttributeV
alue>
   </saml:Attribute>
  </saml:AttributeStatement>
 </saml:Assertion>
</XRIDescriptor>
</XRIDescriptors>
```

Example 4 – Response for *home

The client validates the SAML assertion as described in Section @@@ before continuing.

Resolving "=example*home*base"

Appending the next qualified sub-segment "*base" to the URI "http://xri.example.com/xri-resolve/*home/" gives the URI http://xri.example.com/xri-resolve/*home/*base. The Accept header with the value "application/xrid-t+xml" is used.

```
GET /xri-resolve/*home/*base HTTP/1.1
If-Modified-Since: Fri, 31 Oct 2003 19:43:32 GMT
Accept: application/xrid-t+xml
<other HTTP headers>
```

Example 5 – Request for *base

The following HTTP response is received from xri.example.com:

```
200 OK HTTP/1.1
Content-type: application/xrid-t+xml
Expires: Fri, 7 Nov 2003 19:43:33 GMT
<other HTTP headers>

<XRIDescriptors
  xmlns="xri://$r.s/XRIDescriptor">
<XRIDescriptor
  xrid:id="7600e1a0-b64d-1026-ea89-c0a80b9d3814">
  <Resolved>*base</Resolved>
  <AuthorityID>urn:uuid:A9F28515-AB03-4883-8852-8EECB54CE1D5
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
  </AuthorityID>
  <Service>
    <Type>xri://$res*localaccess/X2R</Type>
    <URI>http://xri.example.com/xri-local/base/</URI>
    <URI>https://xri.example.com/xri-local/base/</URI>
  </Service>
  <TrustMechanism>xri://$res*trusted/XRITrusted</TrustMechanism>
 <saml:Assertion
   Version="2.0"
   ID="_1a6a12d0-b64d-1026-c1ba-c0a80b9db964"
   IssueInstant="2004-06-03T00:46:03Z"
   xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml:Issuer>urn:uuid:A9F28515-AB03-4883-8852-
8EECB54CE1D5</saml:Issuer>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <ds:SignedInfo>
     <ds:CanonicalizationMethod
       Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
     <ds:SignatureMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
     <ds:Reference URI="#7600e1a0-b64d-1026-ea89-c0a80b9d3814">
     <ds:Transforms>
      <ds:Transform
        Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature" />
      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#">
       <ec:InclusiveNamespaces
         xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
         PrefixList="#default code ds kind rw saml samlp typens" />
      </ds:Transform>
     </ds:Transforms>
     <ds:DigestMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
     <ds:DigestValue>BSsnowZG5DYV0X0C8GAeB1cvLzw=</ds:DigestValue>
    </ds:Reference>
   </ds:SignedInfo>
   <ds:SignatureValue>
kE9p35G4mcombsqEztJMX1R3J26gwc4cbjSz5fUv3aVg3j/iLhrbf0qKywYNMLdQMjBRcCg
5N1l0
Kvv2UrgvQ5kgQ9dm7/563rRzKAaIQwMopZpTFli4eXw+nc8XEH+KnXdu/R9DHOg9k0BKIF6
BGk07
xC6Q9X+byQWenPjAZ1c=
   </ds:SignatureValue>
  </ds:Signature>
  <saml:Subject>
   <saml:NameID NameQualifier="urn:uuid:A9F28515-AB03-4883-8852-
8EECB54CE1D5">
*example
   </saml:NameID>
  </saml:Subject>
  <saml:Conditions
    NotBefore="2004-06-03T00:46:03Z"
    NotOnOrAfter="2004-12-01T00:00:00Z" />
  <saml:AttributeStatement>
   <saml:Attribute Name="xri://$res*trusted/XRIDescriptor">

<saml:AttributeValue>#baec221f3c0f17f53ca6839989632056</saml:AttributeV
alue>
   </saml:Attribute>
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
  </saml:AttributeStatement>
 </saml:Assertion>
 …
</XRIDescriptor>
</XRIDescriptors>
```

Example 6 – Response for *base

The SAML assertion is validated as described in Section 3.2 before proceeding. The result of the final XRI Authority resolution step is the set of HTTP and HTTPS URIs shown in the `xri:XRIDescriptor/xri:Service[xri:Type="xri://$res*localaccess/X2R"]` element above that can be used for local access services (in this case, X2R service).

3.3 Trusted Resolution

This section normatively defines client and server behavior in trusted resolution. It also defines two new XML elements, TrustMechanism and AuthorityID.

3.3.1 XML Elements and Attributes

Several elements are either added to XRI Descriptors or required in XRI Descriptors only for trusted resolution. These elements allow resolving clients to verify the Descriptor in which they are contained:

xri:XRIDescriptor/xri:AuthorityID

Required when providing trusted resolution, optional otherwise. A unique identifier for the authority that produced this Descriptor, of type xs:anyURI. If present, the value of this attribute MUST be such that there is negligible probability that the same value will be assigned as an identifier to any other authority. Note that the authority identified by this element is NOT the authority *described* by the XRIDescriptor (in the xri:Authority element). Rather, this AuthorityID element identifies the authoirty that produced this Descriptor which *describes* another authority.

xri:XRIDescriptor/xri:TrustMechanism

Required when providing trusted resolution, optional otherwise. A URI reference that specifies the mechanism used to provide trusted resolution. The URI reference for the trust mechanism defined in this specification is “xri://$res*trusted/XRITrusted”.

xri:XRIDescriptor/saml:Assertion

Required when providing trusted resolution. A SAML assertion from the *describing* Authority (the one providing the XRI Descriptor) that asserts, via a digital signature, that the enclosed XRI Descriptor has not been modified from the time it was published by the *describing* authority, and that the describing authority believes the information to be correct.

Several elements are added to XRI Descriptors to assist in verifying XRI Descriptors produced by the *next* authority in the resolution chain (the authority being described by the xri:Authority element):

xri:XRIDescriptor/xri:Authority/URI/@trusted

Optional. Default value of “false” (or “0”). Indicates whether this service endpoint is capable of returning trusted resolution results. If the value is “1” or “true”, the *described* authority is willing to return trustable XRI Descriptors at this URI.

xri:XRIDescriptor/xri:Authority/xri:AuthorityID

Required when providing trusted resolution. A reference to the unique identifier of the authority which the authority *described* by this xri:Authority element, of type xs:anyURI. If

Copyright © OASIS Open 2005. All Rights Reserved.

present, the value of this attribute MUST be such that there is negligible probability that the same value will be assigned as an identifier to any other authority. This element is correlated to the xri:XRIDescriptor/xri:AuthorityID element corresponding to a resolution result from the *described* Authority.

xri:XRIDescriptor/xri:Authority/ds:KeyInfo

Required when providing trusted resolution. Provides the public key data which must be used to validate any XRI Descriptor provided by the *described* Authority as a result of resolution at the described Authority. This element comprises key distribution method for trusted XRI resolution.

Figure 4 below demonstrates the relationship between these elements for two descriptors in a resolution chain: one describing an authority, and one produced by that authority.

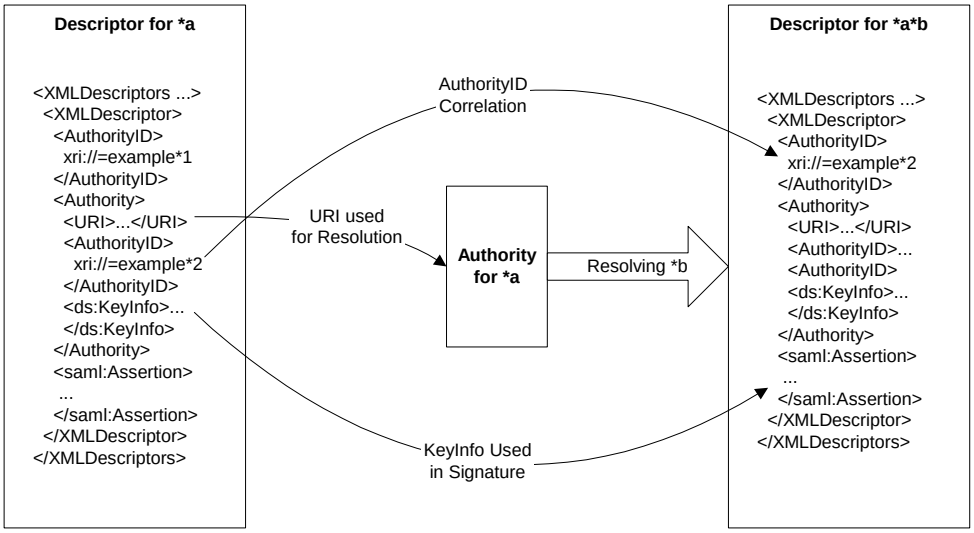


Figure 4: Correlation for Trusted Resolution

3.3.2 Use and Correlation of AuthorityID Elements

Each XRI Authority participating in trusted resolution MUST be associated with one AuthorityID and this AuthorityID MUST never be assigned to any other XRI Authority. In other words, AuthorityID is a permanently unique identifier for a particular XRI Authority.

An AuthorityID may be any valid URI that meets the requirements of permanence and uniqueness described above. Examples of appropriate URIs include URNs as defined by **[RFC2141]** and fully persistent XRIs converted to “URI Normal Form” as defined by **[XRISyntax]**.

Conceptually, AuthorityID assures a resolving client that the returned XRI Descriptor has not been maliciously replaced with a similar XRI Descriptor from a second, and possibly unauthorized, XRI Authority.

There must therefore be a chain of identifiers between describing authority and described authority that is independent of the subsegments being resolved. Consider the following scenario: Imagine that ExampleCorp acts as the global community root and uses the same key pair to sign for both the @ and = namespaces. ExampleCorp’s public key is described in a certificate associated with example.com. ExampleCorp responds to resolution requests in the @

Copyright © OASIS Open 2005. All Rights Reserved.

namespace at http://at.xri.example.com and to resolution requests in the = namespace at http://equals.xri.example.com. A client attempts to resolve xri://@example by sending a request to http://at.xri.example.com. The client receives an XRI Descriptor, properly signed by example.com, with an `xri:Resolved` element of "*example". Although the response appears to be valid, the XRI Descriptor is in fact fraudulent. A malicious party intercepted the request and sent it to http://equals.xri.example.com instead of to the intended http://at.xri.examples.com. The XRI Descriptor describes =example, not @example.

To detect this attack, the XRI Descriptor must be explicitly associated with a particular XRI Authority, and the client must have some means of verifying this association. In trusted resolution as defined by this document, the xri:XRIDescriptor/`xri:AuthorityID` element provides this explicit association. In the example above, the two XRI Authorities responsible for the @ and = namespaces, respectively, each have different AuthorityIDs. Because the client requested resolution in the @ namespace, it knows the AuthorityID associated with the XRI Authority responsible for @. With this knowledge, the client detects that the XRI Descriptor is not provided by @ authority because the value of `xri:XRIDescriptor/xri:AuthorityID` is incorrect.

There is no defined discovery for the AuthorityID of the community root. The AuthorityID for an XRI Authority other than the community root is described by an `xri:AuthorityID` element appearing as a child of the `xri:XRIDescriptor` element in the XRI Authority's XRI Descriptor.

3.3.3 Client Behavior

[@@@ Should we discuss restrictions on HTTP redirects, particularly redirects that contain an XRI?]

From a client's perspective, trusted resolution is identical to the resolution mechanism described in Section 3 of **[XRISyntax]** with the addition of the following REQUIRED behavior:

- The client indicates to the resolving server that a "trusted" XRIDescriptor is requested. In HTTP, this is expressed by adding an "Accept" header with the media type identifier "application/xrid-t+xml". Clients wishing to accept untrusted resolution descriptors may use a combination of "application/xrid-t+xml" and "application/xrid+xml" in the Accept header as described in section 14.1 of **[RFC2616]**.
- The client MUST NOT request trusted resolution from an authority unless the corresponding xri:Descriptor/xri:Authority/xri:URI element has a trusted attribute with the value of "true" or "1".
- For trusted resolution, each XRI Descriptor in a resolution chain MUST be individually validated with the rules described in this section. For complete trusted resolution, each XRI Descriptor in a resolution chain MUST be validated. While XRI Descriptor elements may come from freshly-retrieved XRIDescriptors documents or from local cache, an implementation MUST ensure that the requirements here are satisfied every time a resolution request is performed.

Each confirmation consists checking that each `xri:XRIDescriptor` element contains a `saml:Assertion` element as an immediate child, and that this assertion is valid per the processing rules described by **[SAML]**. In addition, the following requirements must be met:

- The saml:Assertion must contain a valid enveloped digital signature as defined by **[XMLDSig]** and constrained by Section 5.4 of **[SAML]**.
- The signature must apply to the `xri:XRIDescriptor` element that contains the signed SAML assertion. Specifically, the signature must contain a single `ds:SignedInfo/ds:Reference` element, and the URI attribute of this reference

Copyright © OASIS Open 2005. All Rights Reserved.

must refer to the id (xrid:id attribute) of the `xri:XRIDescriptor` element which is the subject of the digital signature.

- If the digital signature enveloped by the SAML assertion may contain a `ds:KeyInfo` element, the client MAY reject the signature if this key does not match the signer's "expected key". The expected key is specified by the ds:KeyInfo present in the XRI Descriptor which was used to describe the current Authority. That is, if Authority A provides an XRI Descriptor describing Authority B, then the keyinfo used to verify descriptors produced by Authority B must be included in the xri:XRIDescriptor/ds:KeyInfo element produced by Authority A.

  For the initial iteration of resolution (e.g. the authority subsegments resolved at a global community root), the signer's expected key is known *a priori* as part of the configuration in the client for that particular authority root.
- The client confirms that the value of the `xri:XRIDescriptor/xri:Resolved` element matches the sub-segment whose resolution resulted in the current XRI Descriptor.
- The client confirms that the value of the `xri:XRIDescriptor/xri:AuthorityID` element matches the XRI Authority's "expected AuthorityID". As with the key information, the "expected AuthorityID" is the value of xri:XRIDescriptor/xri:Authority/xri:AuthorityID in the XRI Descriptor which describes the current Authority. For the initial iteration of resolution (e.g. the authority subsegments resolved at a global community root), the XRI Authority's expected AuthorityID is known *a priori* and is part of the configuration in the client for that particular authority root.
- The client confirms that the value of the `xri:XRIDescriptor/xri:AuthorityID` element, if present,matches the value of both the xri:XRIDescriptor/saml:Assertion/saml:Issuer element and the NameQualifier attribute of the xri:XRIDescriptor/saml:Assertion/saml:Subject/saml:NameID element.
- The client confirms that the value of the `xri:XRIDescriptor/xri:Resolved` element, if present, matches the value of the `xri:XRIDescriptor/saml:Assertion/saml:Subject/saml:NameID` element.
- The client confirms that the value of the `xri:XRIDescriptor/xri:TrustMechanism` is "xri://$res*trusted/XRITrusted".

If any of the above requirements are not met for any XRI Descriptor in the resolution chain, the result MUST NOT be considered "trusted resolution" as defined by this document. Note that this does not preclude a client from attempting an iteration multiple times or from performing an alternate resolution step if the above requirements are not met. For example, if two URIs are listed under an `xri:Authority` element and the response from one fails to meet the requirements above, the client may attempt the current iteration using the second URI. If the second URI produces a sufficient response, resolution may continue and may be considered "trusted" as defined by this document.

3.3.4 Server Behavior

[@@@ Should we discuss restrictions on HTTP redirects, particularly redirects that contain an XRI?]

From the server's perspective, trusted resolution is identical to the resolution mechanism described in Section 3 of **[XRISyntax]** with the addition of the following behavior. This behavior is

Copyright © OASIS Open 2005. All Rights Reserved.

REQUIRED if the client requests trusted resolution as described in section 3.2 and the server intends to honor the client's request.

If, during the HTTP request/response interaction, the server agrees to return a trusted XRI (indicated by the content type of "application/xrid-t+xml"), the XRI Descriptor returned by the server must contain a `saml:Assertion` element as an immediate child of `xri:XRIDescriptor` that is valid per the processing rules described by **[SAML]**. In addition, the following requirements must be met:

- The saml:Assertion must contain a valid enveloped digital signature as defined by **[XMLDSig]** and as constrained by section 5.4 of **[SAML]**.
- The signature must apply to the `xri:XRIDescriptor` element that contains the signed SAML assertion. Specifically, the signature must contain a single `ds:SignedInfo/ds:Reference` element, and the URI attribute of this reference must refer to the `xri:XRIDescriptor` element at the root of the XRI Descriptor being signed. The URI reference should not be empty; it should refer to the identifier contained in the `xrid:id` attribute of the `xri:XRIDescriptor` element.
- The digital signature enveloped by the SAML assertion may contain a `ds:KeyInfo` element. If it is included, it must match the xri:Descriptor/xri:Authority/ds:KeyInfo element in the XRI Descriptor which describes the current Authority, unless the signing XRI Authority is the community root. If the signing XRI Authority is the community root, the `ds:KeyInfo` element must match the well-known signing key for that XRI Authority, which may or may not be published via an XRI Descriptor. Because the signing key is known in advance by the resolution client, the `ds:KeyInfo` element SHOULD typically be omitted from the digital signature.
- The `xri:Resolved` element (optional in **[XRISyntax]**) must be present. The value of this field must match the XRI Authority sub-segment requested by the client.
- The `xri:XRIDescriptor` element may have an `xri:AuthorityID` element as an immediate child. If present, the value of the `xri:AuthorityID` element must be the Authority ID, as described in Section 3.2, of the responding XRI Authority.
- The `xri:XRIDescriptor/xri:TrustMechanism` must be present and the value must be "xri://$res*trusted/XRITrusted".

Also note that if a resolving client requests trusted resolution *and* lookahead resolution, the responding authority SHOULD attempt to perform trusted resolution on behalf of the client as described in section 2.2.5.2. However, the the server providing lookahead resolution MUST NOT return non-trustable XRI Descriptor elements f the client requests trusted resolution.

3.3.5 Additional Requirements

Any server that acts as an XRI Authority as defined by **[XRISyntax]**, with the possible exception of the community root, is described by an `xri: Authority` element within one or more XRI Descriptors. The `xri: Authority` element that describes an authority participating in trusted resolution as defined by this specification ("the described XRI Authority") has the following requirements:

- The trusted attribute of the xri:Descriptor/xri:Authority/xri:URI element must contain the value "1" or "true".
- The `xri:Authority` element MUST contain a `ds:KeyInfo` element as an immediate child. The value of this element MUST be the key that validates digital signatures created by the described XRI Authority. [@@@: XML DigSig has this to say about KeyInfo: "KeyInfo indicates the key to be used to validate the signature.

Copyright © OASIS Open 2005. All Rights Reserved.

Possible forms for identification include certificates, key names, and key agreement algorithms and information – we define only a few. KeyInfo is optional for two reasons. First, the signer may not wish to reveal key information to all document processing parties. Second, the information may be known within the application's context and need not be represented explicitly." Does this argument for making KeyInfo optional apply to us? GMW: I don't think it should be optional.]

- The `xri:Authority` element MAY contain an `xri:AuthorityID` element as an immediate child. If present, the value of this field must be the AuthorityID of the described XRI Authority, i.e. the value that will appear in the xri:Descriptor/`xri:AuthorityID` element of an XRI Descriptor returned from the described XRI Authority. @@@ What happens if the AuthorityID isn't provided? Should this be a MUST?
- In addition, an identifier community SHOULD publish an XRI Descriptor for the community root that meets the requirements listed above and it SHOULD make that XRI Descriptor easily available to relevant parties.

Copyright © OASIS Open 2005. All Rights Reserved.

4 Extensibility and Versioning

4.1 Extensibility

The XRI Descriptors defined in this document are designed to be extended with other metadata. Thus, the XRI Descriptors schema is built on an open content model. In a number of places, extension elements and attributes from namespaces other than “xri://$res*schema/XRIDescriptor” are explicitly allowed. These extension points are designed to make default processing of the XRI Descriptors simple and are based on a “Must Ignore” rule. The base rule is that unrecognized elements and attributes, and the content and child elements of unrecognized elements MUST be ignored. Note that means that even elements that normally recognized by a processor MUST be ignored if they appear as descendants of an unrecognized element.

Extension elements MUST NOT require new interpretation of elements defined in this document. That is, if an extension element is present, a processor can ignore it and still correctly process the Descriptor document.

Extension specifications MAY simulate "Must Understand" behavior by applying an "enclosure" pattern. Elements defined by the XRI-Resolution schema whose meaning or interpretation are to be modified by extension elements can be wrapped in a extension container element, defined by the extension specification. This extension container element would be in the same namespace as the extension elements which must be understood by the consumer of the XRI Descriptor. In doing this, the container and all the elements whose interpretation are modified by the extension are contained in an element (the extension container element) that is ignored by the consumer. In this way, if the consumer is ignorant of the extension, the consumer will be shielded from even being aware that elements in the XRI-Resolution schema are included, and thus they will be ignored.

The following example demonstrates the use of a extension container element from an extension namespace (other:SuperAuthority) and the use of an extension element (other:ExtensionElement) within the container element:

```
<XRIDescriptor>
     <other:SuperAuthority>
        <Authority>
           ...
           <other:ExtensionElement>...</other:ExtensionElement>
        </Authority>
     </other:SuperAuthority>
      <Service>
         ...
      </Service>
</XRIDescriptors>
```

In this example, the other:ExtensionElement modifies the interpretation or processing rules for the authority element. To preserve the correct interpretation of the Authority in this context, the Authority element is "wrapped" so that only consumers which understand elements in the "other" namespace (specifically other:SuperAuthority) will attempt to process the Authority element.

4.2 Versioning

Versioning of the XRI specification set is likely occur rarely, but experience shows that such versioning is inevitable. Thus, guidelines on versioning and the mechanics of expressing and processing version information is described in this section.

Copyright © OASIS Open 2005. All Rights Reserved.

When version information is expressed as both a Major and Minor version, it is expressed in the form *Major.Minor*. The version number $Major_B.Minor_B$ is higher than the version number $Major_A.Minor_A$ if and only if:

$$Major_B > Major_A \text{ OR } ((Major_B = Major_A) \text{ AND } Minor_B > Minor_A)$$

4.2.1 Versioning of the XRI Specification

Each release of the XRI Resolution specification will have a new version number. A new release occurs when there is some non-editorial change to the document specifying a change in behavior or a change in the definition of data on the wire with respect to resolution.

In general, if a change is “backwards compatible”, then the new version should change only the minor version number. Conversely, if the change is “backwards incompatible”, then the major version number should change.

4.2.2 Versioning of XRI Descriptor elements

Both the XRIDescriptors element and the XRIDescriptor element have “version” attributes. This version attribute, while having a fixed value in XRI 2.0, may be used to indicate in future versions of the XRI specification that the relevant element is conformant to a specific version of the XRI specification.

When new versions of the XRI Resolution specification are released, the namespace for the schema may or may not be changed. If there is a major version number change, the namespace for the XRIDescriptors document SHOULD change. If there is only a minor version number change, the namespace for the XRIDescriptors document SHOULD NOT change.

In general, maintaining namespace stability while adding or changing the content of a schema are competing goals. While certain design strategies can facilitate such changes, it is complex to predict how older implementations will react to any given change, making forward compatibility difficult to achieve. Nevertheless, the right to make such changes in minor revisions is reserved, in the interest of namespace stability. Except in special circumstances (for example, to correct major deficiencies or to fix errors), implementations should expect forward-compatible schema changes in minor revisions, allowing new messages to validate against older schemas.

Implementations SHOULD expect and be prepared to deal with new extensions and message types in accordance with the processing rules laid out for those types. Minor revisions MAY introduce new types that leverage the extension facilities described in Section 4.1. Older implementations SHOULD reject such extensions gracefully when they are encountered in contexts that dictate mandatory semantics. @@@ Text taken from SAML Core section 4 – Is it OK to quote verbatim?

Versioning of the namespace identifiers beyond 1.0 SHOULD use the XRI versioning mechanism as described in section @@@ of **[XRIMetadata]**. For example, the 2.0 version of the namespace used for XRI Descriptors documents should be @@@ Example based on XRIMetadata

4.2.3 Versioning of Protocols

Both the local access and authority resolution protocols defined in this document may also be versioned by future versions of the XRI Resolution specification. If these protocols are changed to produce backwards incompatible protocols, they MUST get a new XRI identifying them in an XRI Descriptor. The new identifier SHOULD be based on the original XRI, using the XRI versioning mechanism as described in section @@@ of **[XRIMetadata]**. @@@ Give example

Note that it is possible for version negotiation to happen in the protocol itself. For example, HTTP provides a mechanism to negotiate version of the HTTP protocol being used. When the protocol

Copyright © OASIS Open 2005. All Rights Reserved.

provides its own versioning mechanism, the specification MAY continue to use the same XRI identifying the protocol as used in previous versions of the XRI Resolution specification.

Copyright © OASIS Open 2005. All Rights Reserved.

5 Security Considerations

@@@ peterd to complete this. Should discuss security considerations with and without trusted resolution. Need to discuss the fact that trusted resolution may be useful in a closed world environment. Need to discuss security & trust w/r/t proxy and lookahead as well. Don't forget to discuss caching issues on the client and on servers as well.

This entire document deals with security considerations related to XRI Authority resolution.

Here's some text from DNSSEC that might be a useful model for this section:

This document specifies extensions to the Domain Name System (DNS) protocol to provide data integrity and data origin authentication, public key distribution, and optional transaction and request security.

It should be noted that, at most, these extensions guarantee the validity of resource records, including KEY resource records, retrieved from the DNS. They do not magically solve other security problems. For example, using secure **DNS** you can have high confidence in the IP address you retrieve for a host name; however, this does not stop someone for substituting an unauthorized host at that address or capturing packets sent to that address and falsely responding with packets apparently from that address. Any reasonably complete security system will require the protection of many additional facets of the Internet beyond DNS.

The implementation of NXT RRs as described herein enables a resolver to determine all the names in a zone even if zone transfers are prohibited (section 5.6). This is an active area of work and may change.

A number of precautions in **DNS** implementation have evolved over the years to harden the insecure **DNS** against spoofing. These precautions should not be abandoned but should be considered to provide additional protection in case of key compromise in secure DNS.

Copyright © OASIS Open 2005. All Rights Reserved.

6 Media Type Registration for application/xrid+xml

```
To: ietf-types@iana.org
Subject: Registration of MIME media type application/xrid+xml

MIME media type name: application

MIME subtype name: xrid+xml

Required parameters: None

Optional parameters: None

Encoding considerations: XML

Security considerations:

Interoperability considerations:

Published specification:

Applications which use this media type:

Additional information:

  Magic number(s):
  File extension(s):
  Macintosh File Type Code(s):

Person & email address to contact for further information:

Intended usage: COMMON

Author/Change controller:
```


Copyright © OASIS Open 2005. All Rights Reserved.

7 Media Type Registration for application/xrid-t+xml

```
To: ietf-types@iana.org
Subject: Registration of MIME media type application/xrid-t+xml

MIME media type name: application

MIME subtype name: xrid-t+xml

Required parameters: None

Optional parameters: None

Encoding considerations:

Security considerations:

Interoperability considerations:

Published specification:

Applications which use this media type:

Additional information:

  Magic number(s):
  File extension(s):
  Macintosh File Type Code(s):

Person & email address to contact for further information:

Intended usage: COMMON

Author/Change controller:
```


Copyright © OASIS Open 2005. All Rights Reserved.

8 References

@@@ Be careful with formatting here – it seems to change the entire document when you mess with the indentation. This will need work.

@@@ Do we want to sync these references section with the Core document's references section?

8.1 Normative

[RFC1737] K. Sollins, L. Masinter, *Functional Requirements for Uniform Resource Names*, http://www.ietf.org/rfc/rfc1737.txt, RFC 1737, December 1994.

[RFC2046] N. Borenstein, N. Freed, *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*, http://www.ietf.org/rfc/rfc2046.txt, RFC 2046, November 1996.

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, RFC 2119, March 1997.

[RFC2141] R. Moats, *URN Syntax*, http://www.ietf.org/rfc/rfc2141.txt, IETF RFC 2141, May 1997.

[RFC2234] D. H. Crocker and P. Overell, *Augmented BNF for Syntax Specifications: ABNF*, http://www.ietf.org/rfc/rfc2234.txt, RFC 2234, November 1997.

[RFC2396] T. Berners-Lee, R. Fielding, L. Masinter, *Uniform Resource Identifiers (URI): Generic Syntax,* http://www.ietf.org/rfc/rfc2396.txt, RFC 2396, August 1998.

[RFC2483] M. Meallling, R. Daniel Jr., *URI Resolution Services Necessary for URN Resolution,* http://www.ietf.org/rfc/rfc2483.txt, RFC 2483, January 1999.

[RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, *Hypertext Transfer Protocol -- HTTP/1.1*, http://www.ietf.org/rfc/rfc2616.txt, RFC 2616, June 1999.

[RFC2718] L. Masinter, H. Alvestrand, D. Zigmond, R. Petke, *Guidelines for New URL Schemes*, http://www.ietf.org/rfc/rfc2718.txt, RFC 2718, November 1999.

[RFC2732] R. Hinden, B. Carpenter, L. Masinter, *Format for Literal IPv6 Addresses in URL's,* http://www.ietf.org/rfc/rfc2732.txt, RFC 2732, December, 1999.

[RFC3066] H. Alvestrand, *Tags for the Identification of Languages,* http://www.ietf.org/rfc/rfc3066.txt, RFC 3066, January, 2001.

[RFC3305] M. Mealing, R. Denenberg, *Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations*, http://www.ietf.org/rfc/rfc3305.txt, RFC 3305, August 2002.

[RFC3490] P. Faltstrom, P. Hoffman, A. Costello, *Internationalizing Domain Names in Applications (IDNA)*, http://www.ietf.org/rfc/rfc3490, RFC 3490, March 2003.

[RFC3491] P. Hoffman, M. Blanchet, *Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN),* http://www.ietf.org/rfc/rfc3491, RFC 3491, March 2003.

Copyright © OASIS Open 2005. All Rights Reserved.

[RFC3986] T. Berners-Lee, R. Fielding, L. Masinter, *Uniform Resource Identifier (URI): Generic Syntax,* http://www.ietf.org/rfc/rfc3986, RFC 3986, January 2005.

[UML] Object Management Group, *Unified Modeling Language (UML) Version 1.5*, http://www.omg.org/technology/documents/formal/uml.htm, March 1, 2003.

[Unicode] The Unicode Consortium. The Unicode Standard, Version 4.0.0, defined by: *The Unicode Standard, Version 4.0* (Boston, MA, Addison-Wesley, 2003. ISBN 0-321-18578-1)

[UniXML] Duerst, M. and A. Freytag, *Unicode in XML and other Markup Languages*, Unicode Technical Report #20, World Wide Web Consortium Note, February 2002.

[UTR15] M. Davis, M. Duerst, *Unicode Normalization Forms*, http://www.unicode.org/unicode/reports/tr15/tr15-23.html, April 17, 2003.

[XML] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, *Extensible Markup Language (XML) 1.0 (Second Edition) W3C Recommendation*, http://www.w3.org/TR/REC-xml, October 2000.

[XMLSchema2] P. Biron, A. Malhotra, *XML Schema Part 2: Datatypes W3C Recommendation,* http://www.w3.org/TR/xmlschema-2/, May 2001.

[DNSSEC] D. Eastlake, *Domain Name System Security Extensions*, http://www.ietf.org/rfc/rfc2535, March 1999.

[ExclC14N] J. Boyer, D. Eastlake 3rd, J. Reagle, *Exclusive XML Canonicalization Version 1.0*, World Wide Web Consortium, http://www.w3.org/TR/xml-exc-c14n/, July 18, 2002.

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, RFC 2119, March 1997.

[RFC2141] R. Moats, *URN Syntax*, http://www.ietf.org/rfc/rfc2141.txt, IETF RFC 2141, May 1997.

[SHA-1] U.S. Department of Commerce/National Institute of Standards and Technology, *FIPS PUB 180-1, Secure Hash Standard*, http://csrc.nist.gov/publications/fips/fips180-1/fip180-1.txt, April 17, 1995.

[SAML] TODO: Add ref to SAML 2.0 when it is available

[SSL] Netscape, Inc., *SSL 3.0 Specification*, http://home.netscape.com/eng/ssl3/, November 1996.

[XMLDSig] D. Eastlake, J. Reagle, D. Solo et al., *XML-Signature Syntax and Processing*, World Wide Web Consortium, http://www.w3.org/TR/xmldsig-core/, February 12, 2002.

[XRISyntax] G. Wachob, D. Reed, D. McAlpin, M. Lindelsee, P. Davis, N. Sakimura, *Extensible Resource Identifier (XRI) Generic Syntax and Resolution Specification*, http://www.oasis-open.org/committees/xri/xri-syntax-resolution-1.0-cd, January 12, 2004.

8.2 Informative

[IRI] M. Duerst, M. Suignard, *Internationalized Resource Identifiers (IRIs)*, http://www.ietf.org/internet-drafts/draft-duerst-iri-05.txt, Work-In-Progress, October 2003.

[REST] http://internet.conveyor.com/RESTwiki/moin.cgi/FrontPage

Copyright © OASIS Open 2005. All Rights Reserved.

[RFC2396bis] R. Fielding, *Uniform Resource Identifiers (URI): Generic Syntax*, Internet Draft draft-fielding-uri-rfc2396bis-03, http://www.apache.org/~fielding/uri/rev-2002/rfc2396bis.html, Work-In-Progress, June 2003.

[XRIMetadata] OASIS XRI Technical Committee, *Extensible Resource Identifier (XRI) Metadata Specification*, http://www.oasis-open.org/committees/xri/xri-metadata-1.0, Work-In-Progress, January 2003.

[XRIReqs] G. Wachob, D. Reed, M. Le Maitre, D. McAlpin, D. McPherson, *Extensible Resource Identifier (XRI) Requirements and Glossary v1.0*, http://www.oasis-open.org/apps/org/workgroup/xri/download.php/2523/xri-requirements-and-glossary-v1.0.doc, June 2003.

Copyright © OASIS Open 2005. All Rights Reserved.

1902 # Appendix A. Revision History

Rev	Date	By Whom	What
2.0	2005-01-15	All Editors	Initial document.

1903

Copyright © OASIS Open 2005. All Rights Reserved.

Appendix B. XML Schema for XRI Descriptor (Normative)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="xri://$res*schema/XRIDescriptor"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xrid="xri://$res*schema/XRIDescriptor"
elementFormDefault="qualified">
    <!-- Utility patterns -->
    <xs:attributeGroup name="otherattribute">
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:attributeGroup>
    <xs:group name="otherelement">
        <xs:choice>
            <xs:any namespace="##other" processContents="lax"/>
            <xs:any namespace="##local" processContents="lax"/>
        </xs:choice>
    </xs:group>
    <xs:complexType name="URIpattern">
        <xs:simpleContent>
            <xs:extension base="xs:anyURI">
                <xs:attributeGroup ref="xrid:otherattribute"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="Stringpattern">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attributeGroup ref="xrid:otherattribute"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <!-- Patterns for elements -->
    <xs:element name="XRIDescriptors">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="xrid:XRIDescriptor" maxOccurs="unbounded"/>
                <xs:group ref="xrid:otherelement" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attributeGroup ref="xrid:otherattribute"/>
            <xs:attribute ref="xrid:version"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="XRIDescriptor">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="xrid:Resolved" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element ref="xrid:AuthorityID" minOccurs="0"/>
                <xs:element ref="xrid:Expires" minOccurs="0"/>
                <xs:element ref="xrid:Authority" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element ref="xrid:Service" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element ref="xrid:Synonyms" minOccurs="0"/>
                <xs:element ref="xrid:TrustMechanism" minOccurs="0"/>
                <xs:group ref="xrid:otherelement" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute ref="xrid:id"/>
            <xs:attributeGroup ref="xrid:otherattribute"/>
            <xs:attribute ref="xrid:version"/>
        </xs:complexType>
    </xs:element>
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
    <xs:element name="Resolved" type="xrid:Stringpattern"/>
    <xs:element name="Expires">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:dateTime">
                    <xs:attributeGroup ref="xrid:otherattribute"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="Authority">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="xrid:AuthorityID" minOccurs="0"/>
                <xs:element ref="xrid:Type" minOccurs="0"/>
                <xs:group ref="xrid:TrustableURI" maxOccurs="unbounded"/>
                <xs:group ref="xrid:otherelement" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attributeGroup ref="xrid:otherattribute"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="AuthorityID" type="xrid:URIpattern"/>
    <xs:element name="Type" type="xrid:URIpattern"/>
    <xs:group name="TrustableURI">
        <xs:sequence>
            <xs:element name="URI">
                <xs:complexType>
                    <xs:simpleContent>
                        <xs:extension base="xrid:URIpattern">
                            <xs:attribute ref="xrid:trusted"/>
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:group>
    <xs:element name="Service">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="xrid:Type" minOccurs="0"/>
                <xs:group ref="xrid:URI" maxOccurs="unbounded"/>
                <xs:element ref="xrid:MediaType" minOccurs="0"/>
                <xs:group ref="xrid:otherelement" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attributeGroup ref="xrid:otherattribute"/>
        </xs:complexType>
    </xs:element>
    <xs:group name="URI">
        <xs:sequence>
            <xs:element name="URI" type="xrid:URIpattern"/>
        </xs:sequence>
    </xs:group>
    <xs:element name="MediaType" type="xrid:Stringpattern"/>
    <xs:element name="Synonyms">
        <xs:complexType>
            <xs:sequence>
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                    <xs:element ref="xrid:Internal"/>
                    <xs:element ref="xrid:External"/>
                </xs:choice>
                <xs:group ref="xrid:otherelement" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attributeGroup ref="xrid:otherattribute"/>
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
            </xs:complexType>
        </xs:element>
        <xs:element name="Internal" type="xrid:URIpattern"/>
        <xs:element name="External" type="xrid:URIpattern"/>
        <xs:element name="TrustMechanism" type="xrid:URIpattern"/>
        <xs:attribute name="version" type="xs:string" fixed="1.0"/>
        <xs:attribute name="trusted" type="xs:boolean"/>
        <xs:attribute name="id" type="xs:ID"/>
</xs:schema>
```


Copyright © OASIS Open 2005. All Rights Reserved.

Appendix C. RelaxNG Compact Syntax Schema for XRI Descriptor (Non-normative)

```
namespace xrid="xri://$res*schema/XRIDescriptor"
namespace xml="http://www.w3.org/XML/1998/namespace"

start=XRIDescriptors

# Utility patterns
anything = ( element * {anything} | attribute * {text} | text ) *
otherattribute = attribute *-xrid:* {text}
otherelement = element *-xrid:* {anything}
URIpattern = (xsd:anyURI, otherattribute *)
Stringpattern = (xsd:string, otherattribute *)
versionattribute = attribute xrid:version {text}
idattribute = attribute xrid:id {xsd:ID}

##########################
# XRIDescriptors Container
XRIDescriptors = element xrid:XRIDescriptors {
    versionattribute,
    XRIDescriptor+,
    XRIDescriptors-ex-elem,
    XRIDescriptors-ex-attr
}

# XRIDescriptors Extension
XRIDescriptors-ex-elem = otherelement *
XRIDescriptors-ex-attr = otherattribute *

##########################
# XRIDescriptor Definition
XRIDescriptor = element xrid:XRIDescriptor {
    attribute xrid:id {xsd:ID}?,
    versionattribute,
    Resolved *,
    AuthorityID ?,
    Expires ?,
    Authority *,
    Service *,
    Synonyms ?,
    TrustMechanism ?,
    XRIDescriptor-ex-elem,
    XRIDescriptor-ex-attr
}

# XRIDescriptor Extension
XRIDescriptor-ex-elem = otherelement *
XRIDescriptor-ex-attr = otherattribute *

####################
# Resolved Definition
Resolved = element xrid:Resolved { Resolved-content}

# Resolved Extension
Resolved-content = Stringpattern

####################
# Expires Definition
Expires = element xrid:Expires {
    xsd:dateTime,
    Expires-ex-attr
}
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
# Expires Extension
Expires-ex-attr = otherattribute *

######################
# Authority Definition
Authority = element xrid:Authority {
    AuthorityID?,
    Type?,
    TrustableURI+,
    Authority-ex-attr,
    Authority-ex-elem
}

# Authority Extension
Authority-ex-attr = otherattribute *
Authority-ex-elem = otherelement *

########################
# AuthorityID Definition
AuthorityID = element xrid:AuthorityID { AuthorityID-content}

# AuthorityID extension
AuthorityID-content = URIpattern

#################
# Type Definition
Type = element xrid:Type { Type-content}

# Type Extension
Type-content = URIpattern

################
# Trustable URI Definition
TrustableURI = element xrid:URI { TrustableURI-content }

TrustableURI-content = (
    URIpattern,
    attribute xrid:trusted {xsd:boolean}?
)

####################
# Service Definition
Service = element xrid:Service {
    Type?,
    URI+,
    MediaType?,
    Service-ex-attr,
    Service-ex-elem
}
# Service Extension
Service-ex-attr = otherattribute *
Service-ex-elem = otherelement *

#####################################
# URI Definition (for Service element)
URI = element xrid:URI { URI-content }

# URI Extension
URI-content = URIpattern

######################
# MediaType Definition
MediaType = element xrid:MediaType { MediaType-content }

# MediaType Extension
MediaType-content = URIpattern
```


Copyright © OASIS Open 2005. All Rights Reserved.

```
####################
# Synonyms Definition
Synonyms = element xrid:Synonyms {
    (
        Internal &
        External
    )+,
    Synonyms-ex-attr,
    Synonyms-ex-elem
}

Synonyms-ex-attr = otherattribute *
Synonyms-ex-elem = otherelement *

####################
# Internal Definition
Internal = element xrid:Internal { Internal-content }

# Internal Extension
Internal-content = URIpattern

####################
# External Definition
External = element xrid:External { External-content }

# External Extension
External-content = URIpattern

##########################
# TrustMechanism Definition

TrustMechanism = element xrid:TrustMechanism { TrustMechanism-content }

# TrustMechanism Extension
TrustMechanism-content = URIpattern
```


Copyright © OASIS Open 2005. All Rights Reserved.

Appendix D. Notices

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.

Copyright © OASIS Open 2005. *All Rights Reserved.*

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself does not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an “AS IS” basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

SUBSTANTIAL OPEN QUESTIONS and TODOs:

- Description of use of saml:NameID (is it correct?) Description of the value of the Name attribute on saml:Attribute – In general, do we need **more** description of the use of SAML as applied to XRI Descriptors?
- In XSD, how can you declare anyattribute to be from “local” (ie no namespace) or “other” namespace. There only appears to be a way to declare one or the other but not “both” (ie allowing both). RelaxNG doesn’t have this problem.
- Insert security discussion section
- Address all @@@’s (ref’s to RFCs, and esp. xrefs to other XRI documents, plus 3 or 4 open questions to editors/reviewers)
- Fill out the RFC 2048 media type application templates (sections 6 and 7)

OTHER TODOs:

- Last Minute: Make sure all the examples show XRI’s in URI normal form
- Last Minute: Make sure all XRIs are legal (esp. making them start with “xri://”)
- Last Minute: Confirm editors/contributors list
- Last Minute: Fill out the metadata at top of document
- Last Minute: Update, prune and verify the references, esp to other XRI specs and new RFCs
- Last Minute: Ensure that cross references to other XRI document sections are still correct
- Last Minute: Ensure correct formatting of XRIs, XPATHs, XML examples, etc

Copyright © OASIS Open 2005. All Rights Reserved.