**XRI Resolution**

**Version 2.0, Working Draft 05, 7 March 2005**

**Document identifier:**
wd-xri-resolution-v2.0-wd05

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

**Editors:**
- Gabe Wachob, Visa International <gwachob@visa.com>
- Drummond Reed, Cordance <drummond.reed@cordance.net>
- 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 Introduction to XRIs 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 v2.0 at @@@[link].

**Status:**
This document is a Working Draft. This document is updated periodically on no particular schedule. Send comments to the editor. Committee members should send comments on this specification to the xri@lists.oasis-open.org list. Others should subscribe to and send comments to the xri-comment@lists.oasis-open.org list. To subscribe, send an email message to xri-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message. 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 XRI TC web page (http://www.oasis-open.org/committees/xri/).

The errata page for this specification is at http://www.oasis-open.org/committees/xxx/yyy.
# Table of Contents

1. Introduction......................................................................................................................... 4
   1.1 XRI Resolution Framework................................................................................................. 4
   1.2 General Format and Reader's Guide ................................................................................... 4
   1.3 Terminology and Notation.................................................................................................... 4
2. Generic Resolution.................................................................................................................. 5
   2.1 Introduction......................................................................................................................... 5
   2.1.1 Assumptions................................................................................................................... 5
   2.1.2 Phases of Resolution...................................................................................................... 5
   2.1.3 XRI vs. IRI Authorities.................................................................................................. 6
   2.1.4 XRI Metadata Reserved for XRI Resolution................................................................. 6
   2.2 XRI Authority Resolution.................................................................................................. 7
   2.2.1 Overview ......................................................................................................................... 7
   2.2.2 XRI Descriptors .............................................................................................................. 8
   2.2.3 Starting the Chain of XRI Descriptors with the Root XRID......................................... 11
   2.2.4 Default HTTP(S)-based Authority Resolution Service................................................ 11
   2.2.5 Examples (Non-Normative) .......................................................................................... 15
   2.2.6 Resolving Cross-References in XRI Authorities........................................................... 19
   2.2.7 XRI Redirects................................................................................................................ 20
   2.3 IRI Authority Resolution................................................................................................... 21
   2.4 Local Access...................................................................................................................... 22
   2.4.1 Local Access Service Types.......................................................................................... 22
   2.4.2 The X2R Local Access Service..................................................................................... 22
   2.5 HTTP Headers.................................................................................................................... 24
   2.5.1 Caching........................................................................................................................ 24
   2.5.2 Location........................................................................................................................ 24
   2.5.3 Content-Type............................................................................................................... 24
   2.6 Other HTTP Features....................................................................................................... 24
   2.7 Caching and Efficiency..................................................................................................... 24
3. Trusted Resolution............................................................................................................... 26
   3.1 Introduction......................................................................................................................... 26
   3.2 Overview and Example (Non-Normative)........................................................................ 26
   3.3 Trusted Resolution Protocol............................................................................................. 32
   3.3.1 XML Elements and Attributes..................................................................................... 32
   3.3.2 Use and Correlation of AuthorityID Elements............................................................. 33
   3.3.3 Client Behavior ............................................................................................................ 33
   3.3.4 Server Behavior .......................................................................................................... 35
   3.3.5 Additional Requirements of Authorities Offering Trusted Resolution.......................... 36
4. Extensibility and Versioning.............................................................................................. 37
   4.1 Extensibility....................................................................................................................... 37
   4.1.1 Specific Points of Extensibility...................................................................................... 37
   4.2 Versioning........................................................................................................................ 38
   4.2.1 Versioning of the XRI Resolution Specification.......................................................... 38
1 Introduction

1.1 XRI Resolution Framework
Extensible Resource Identifiers (XRIs) provide a uniform syntax for abstract identifiers as defined in [XRISyntax]. Because XRIs may be used across a wide variety of communities and applications (as database keys, filenames, directory keys, object IDs, XML IDs etc.), no single resolution mechanism may prove appropriate for all XRIs. However, in the interest of promoting interoperability, this specification defines a standard framework for XRI resolution consisting of two parts:

- **Generic resolution** (section 2) is a simple, flexible resolution protocol for the authority segment of an XRI that relies exclusively on HTTP/HTTPS as a transport.
- **Trusted resolution** (section 3) is an extension of the generic resolution protocol that uses SAML assertions to create a chain of trust between the participating authorities.

Both of these protocols are extensible as described in section 4. In addition, other XRI resolution services or protocols may be defined by future versions of this specification or by other specifications.

1.2 General Format and Reader’s Guide

1.3 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.

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 [SAML] 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 [XMLDSig] Namespace “http://www.w3.org/2000/09/xmldsig#”, and the namespace prefix xrid: stands for the namespace “xri://$res*schema/XRIDescriptor*($v%2F2.0)”, whether or not they are explicitly declared in the example or text. These namespace prefixes are summarized in Table 1.

<table>
<thead>
<tr>
<th>saml</th>
<th>urn:oasis:names:tc:SAML:2.0:assertion</th>
</tr>
</thead>
<tbody>
<tr>
<td>ds</td>
<td><a href="http://www.w3.org/2000/09/xmldsig#">http://www.w3.org/2000/09/xmldsig#</a></td>
</tr>
<tr>
<td>xrid</td>
<td>xri://$res<em>schema/XRIDescriptor</em>($v%2F2.0)</td>
</tr>
</tbody>
</table>

Table 1: XML namespace prefixes used in this specification.

Terms used in this document are defined in the glossary in Appendix @@ of [XRISyntax].

Comment [DSR5]:TODO - Drummond still working on this.
Comment [gmw6]: Insert “How to read” or “Quickstart guide”
Comment [DM7]: Update depending on where the glossary eventually lands.
2 Generic Resolution

2.1 Introduction

Generic XRI resolution is the process of dereferencing an XRI to a network endpoint in order to obtain metadata about or further interact with the resource identified by the XRI. To provide a simple, general-purpose mechanism for accomplishing this task, this specification defines a generic resolution protocol based on HTTP/HTTPS. Other XRI resolution services may be defined by future versions of this specification or by other specifications.

Identifier management policies are defined on a community-by-community basis. With XRIs, the resolution community is specified by the authority segment of the XRI. When a resolution 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

The generic 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 reference, it must be converted into an absolute XRI using the procedure defined 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 specification, however a mechanism for specifying access via URIs using other URI schemes is also defined.
- 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, 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 generic 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

Authority resolution is the process of a) finding the endpoint or endpoints that are authoritative for access to resources under that authority’s control or b) discovering further information about the authority itself. In the case where the desired goal is access to a resource, the result of authority resolution will be a list of local access endpoints, identified by one or more URIs, that support at
least one local access protocol. The calling application may then choose one of these endpoints and access it using the desired local access protocol.

In the case where the goal of resolution is to discover more information about an authority, such as XRI synonyms, public keys, or other XRI resolution metadata, this information will be returned by the authority resolution process itself.

Figure 1 illustrates the two main phases of XRI resolution – authority resolution and local access:

![Diagram of XRI resolution phases](image)

1) What is the identifier authority for example?
2) What is the local access for *home (relative to example)
3) Give me a resource representing example*home/foo.

2.1.3 XRI vs. IRI Authorities

As described in section 2.2.1 of [XRISyntax], XRI authorities and IRI 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. IRI authorities, since they are based on DNS names or IP addresses, are resolved into an authority descriptor through a special HTTP(S) request based on the DNS name or IP address identified by the IRI authority segment.

2.1.4 XRI Metadata Reserved for XRI Resolution

As defined in section 2.2.1.2 of [XRISyntax], the GCS symbol "$" is reserved for special identifiers assigned by XRI TC specifications, other OASIS specifications, or other standards bodies. (See also [XRIMetadata].) Within the "$" namespace, the identifier "$res" is reserved for identifiers assigned by this XRI resolution specification. Table 2 summarizes these identifiers.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Use</th>
<th>See Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>xri://$res*schema</td>
<td>XML namespace for XRI resolution schema</td>
<td>0</td>
</tr>
<tr>
<td>xri://$res*auth.res</td>
<td>Namespace for authority resolution protocol types</td>
<td>2.2.4</td>
</tr>
<tr>
<td>xri://$res*local.access</td>
<td>Namespace for local access protocol types</td>
<td>2.4.1</td>
</tr>
<tr>
<td>xri://$res*trusted</td>
<td>Namespace for trust mechanisms</td>
<td>0 and 3</td>
</tr>
</tbody>
</table>

Comment [gmw8]: Section references have been mangled somehow – they need to be fixed, which may be a lot of work.
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 by section 2.2.1.2 of [XRISyntax] or b) a sub-segment as defined by the
productions whose names start with "xri-subseg" in section 2.2.3 of [XRISyntax] including the
leading syntactic delimiter ("*" or ")
. Note that in the latter case a qualified sub-segment always
includes the leading syntactic delimiter even if it was optionally omitted in the original XRI (see the
productions names containing "-od" in section 2.2.3 of [XRISyntax]).

Each qualified sub-segment is resolved in the context of the sub-segment immediately to the left,
with the first (or leftmost) sub-segment identifying the root of the identifier community. Each sub-
segment is resolved to a corresponding XRI Descriptor (often abbreviated as "XRID"), an XML
document that specifies one or more network endpoints (HTTP or HTTPS URIs) that answer XRI
resolution requests. As resolution proceeds, the XRI resolver is building a “chain” (an ordered list)
of XRIDs. Resolution is complete when the resolver has completed the chain of XRIDs
for all sub-segments in the XRI authority segment. Figure 2 below depicts this resolution process
for the XRI "@a*b*c".

![Diagram of XRI Descriptors, XRI Authorities and Sub-Segments for @a*b*c](image)

Each HTTP(S) resolution request may ask for resolution of more than one sub-segment at a
time—a process called lookahead resolution. If lookahead resolution is used, each HTTP(S)
response may contain one or more XRIDescriptors inside a XML container document. The
number of sub-segments resolved in one resolution request depends on:
- How many sub-segments the resolving client presents to a responding XRI Authority for
lookahead resolution; and

Comment [gmw9]: Sequence, activity or other UML diagram instead? This has been marked as
“confusing”, so if you agree and have a different way of presenting the
information, PLEASE give ideas!
Each XRI Descriptor in the chain contains one or more of three basic types of information about the XRI authority it describes:

- URIs describing network endpoints for XRI authority resolution services;
- URIs describing network endpoints for local access services;
- XRI synonyms (equivalent XRIs) for the resolved sub-segment.

All three options—authority resolution, local access, or synonyms—may be available at each 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." "@a*b*c" may also be a local access endpoint itself, in which case its XRI Descriptor will contain references to local access services. Finally, "@a*b*c" may have a persistent XRI synonym such as "xri://@$1000213", which may also be included in the XRID in order to provide resolvers or caches with an equivalent persistent XRI.

### 2.2.2 XRI Descriptors

To provide a straightforward, flexible resolution mechanism, XRI authority endpoints are described using a simple, flexible XML document, called an XRI Descriptor (often abbreviated "XRID"). While this specification defines only XRID elements necessary to support delegated resolution and access of XRI-identified authorities and resources, an XRID can easily be extended to publish any form of metadata about the described authority.

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

```xml
<XRIDs
xmlns="xri://$res*schema/XRIDescriptor*($v%2F2.0)"
</XRIDescriptor xrid:id="first">
  <Resolved>*foo</Resolved>
  <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://$res*auth.res/XRIA</Type>
    <URI>http://xri.example.com</URI>
    <URI>https://xri.example.com</URI>
  </Authority>
  <Service>
    <Type>xri://$res*local.access/X2R</Type>
    <URI>http://xri.example.com</URI>
    <MediaType>application/rdf+xml</MediaType>
  </Service>
  <Service>
    <Type>xri://$res*local.access/X2R</Type>
    <URI>http://pictures.xri.example.com</URI>
    <MediaType>image/jpeg</MediaType>
  </Service>
  <Synonyms>
    <Internal>xri://@!1!2!3</Internal>
    <External>xri://@!4!5!6</External>
  </Synonyms>
  <TrustMechanism>xri://$res*trusted/None</TrustMechanism>
</XRIDescriptor>
```
All schema elements in the basic XML Descriptor are in the XML namespace "xri://$res*schema/XRIDescriptor*(v%2F2.0)". Following are the elements and attributes that comprise the XRIDescriptor document type (all XPATHs are relative to the enclosing XRIDescriptors document element):

<xrid:XRIDescriptor>

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

<xrid:XRIDescriptor/xrid:Resolved>

1 per XRIDescriptor. Required. Expresses the qualified sub-segment whose resolution results in this XRIDescriptor element.

<xrid:XRIDescriptor/xrid:AuthorityID>

1 per XRIDescriptor. Required. A unique identifier for the authority that produced this XRI Descriptor, of type xs:anyURI. The value of this element 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 the describing authority (the producer of the current XRID), not the authority described by the XRID. The latter is specified in the xrid:XRIDescriptor/xrid:Authority/xrid:AuthorityID element (see below).

<xrid:XRIDescriptor/xrid:Expires>

0 or 1. The UTC time after which this document cannot be relied upon. A resolver using this XRIDescriptor MUST NOT use the XRIDescriptor after the time stated here. A resolver MAY discard this Descriptor before the time indicated in this result. If the HTTP transport caching semantics specify an expiry time that is earlier than the time expressed in this attribute, then a resolver MUST NOT use this XRIDescriptor after the expiry time declared in the HTTP headers per section 13.2 of [RFC2616].

<xrid:XRIDescriptor/xrid:Authority>

0 or more. Describes an authority resolution service associated with the resolved identifier. If there are additional sub-segments in the authority segment of the XRI being resolve, they can be resolved at this service endpoint.

<xrid:XRIDescriptor/xrid:Authority/xrid:AuthorityID>

1 per Authority element. Required. The unique identifier of the authority described by this xrid:Authority element, of type xs:anyURI. The value of this element 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 xrid:XRIDescriptor/xrid:AuthorityID element (above) corresponding to a resolution result from the described Authority, particularly for trusted resolution (see section 3).

<xrid:XRIDescriptor/xrid:Authority/xrid:Type>

0 or 1 per Authority element. Indicates the type of authority resolution service described by the parent xrid:Authority element. This specification defines one authority resolution service: "xri://$res*auth.res/XRIA" (XRI Authority resolution as described in section 2.2.4). This is the default value if this element is not present.

<xrid:XRIDescriptor/xrid:Authority/xrid: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 versions of this specification or other
specifications may specify other transport protocols. Each URI element has an optional
attribute called “trusted” that indicates whether or not the particular service endpoint
provides trusted resolution (section 3). The trust mechanism is described using the
TrustMechanism element (below).

```xml
<xrid:XRIDescriptor/xrid:Service
  0 or more. Describes a local access service endpoint provided by the described authority.

<xrid:XRIDescriptor/xrid:Service/xrid:Type
  0 or 1 per Service element. Indicates the type of local service being described. This
specification defines one service: “xri://$res*local.access/X2R” (The X2R local access
service as defined in section 2.4.2). This is the default value if this element is not present.

<xrid:XRIDescriptor/xrid:Service/xrid:URI
  1 or more per Service element. Indicates the transport level URI where the service
described may be accessed. For the X2R local access service defined in section 2.4.2,
this URI MUST be an HTTP or HTTPS URI. Other services may use other transport
protocols.

<xrid:XRIDescriptor/xrid:Service/xrid:MediaType
  0 or more per Service element. 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 value of this element must be of the form of a media type defined in
[RFC2046]. This element may appear multiple times to indicate that multiple media types
are available through this local access service.

<xrid:XRIDescriptor/xrid:Synonyms
  0 or 1. Contains statements about the equivalence of the resolved identifier to other XRIs.

<xrid:XRIDescriptor/xrid:Synonyms/xrid:Internal
  0 or more. Represents another XRI assigned to the described authority by the current
describing authority. Must be an absolute XRI (“absolute-xri” in section 2.2 of
[XRISyntax]). An internal synonym 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 being resolved. Both cases may be
particularly useful in populating or querying a cache since resolution of an internal
synonym will typically result in an XRID containing the same information as the current
XRID.

<xrid:XRIDescriptor/xrid:Synonyms/xrid:External
  0 or more. Represents another XRI assigned to the described authority by a different
authority than the current describing authority. Must be an absolute XRI (“absolute-xri” in
section 2.2 of [XRISyntax]). Resolution of an external synonym will typically result in an
XRID containing different information than the current XRID. External synonyms are
used, for example, in XRI redirects, described in Section 2.2.7. They can also be used to
identify alternative sources of local access descriptors if those in the current XRID do not
satisfy the needs of the client.

<xrid:XRIDescriptor/xrid:TrustMechanism
  0 or 1. Identifies the mechanism for trusted resolution associated with this XRID. This
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, the default value is
“xri://$res*trusted/None”.
```
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 resolution schemes, new local access resolution schemes, additional XRI synonym metadata, or other metadata about the described authority. See section 4.1 for more about XRID extensibility.

See section 3.3.1 for additional XRI Descriptor elements defined for trusted resolution.

2.2.3 Starting the Chain of XRI Descriptors with the Root XRID

With an XRI authority, the first sub-segment corresponding to the community root may be either a global context symbol (GCS) character or top-level cross-reference as specified in section 2.2.1.1 of [XRISyntax]. In either case, the corresponding root XRID (or its equivalent) specifies the top-level authority resolution endpoints for that 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. (Note that it is not strictly necessary to publish this information in an XRID—it may be supplied in any format that enables configuration of the XRI resolvers in the community—but providing an XRID at a known location simplifies the process.)

If the first sub-segment of an XRI authority is a GCS character and the following sub-segment does not begin with a “*” (indicating a reassignable sub-segment) or a “!” (indicating a persistent sub-segment), then a “*” is implied and must be added when constructing the qualified sub-segment. Table 3 and Table 4 illustrate the differences between parsing a reassignable sub-segment following a GCS character and a cross-reference, respectively.

<table>
<thead>
<tr>
<th>XRI</th>
<th>xri://@example*internal/foo</th>
</tr>
</thead>
<tbody>
<tr>
<td>XRI Authority</td>
<td>@example*internal</td>
</tr>
<tr>
<td>Community Root Authority</td>
<td>@</td>
</tr>
<tr>
<td>First Qualified Sub-Segment Resolved</td>
<td>*example</td>
</tr>
</tbody>
</table>

Table 3: Parsing the first sub-segment of an XRI that begins with a global context symbol.

<table>
<thead>
<tr>
<th>XRI</th>
<th>xri://(<a href="http://www.example.com)*internal/foo">http://www.example.com)*internal/foo</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>XRI Authority</td>
<td>(<a href="http://www.example.com)*internal">http://www.example.com)*internal</a></td>
</tr>
<tr>
<td>Community Root Authority</td>
<td>(<a href="http://www.example.com">http://www.example.com</a>)</td>
</tr>
<tr>
<td>First Qualified Sub-Segment Resolved</td>
<td>*internal</td>
</tr>
</tbody>
</table>

Table 4: Parsing the first sub-segment of an XRI that begins with a cross-reference.

2.2.4 Default HTTP(S)-based Authority Resolution Service

This section defines the default authority resolution service for generic XRI resolution. When explicitly declared, it uses the xrid:XRIDescriptor/xrid:Authority/xrid:Type element value “xri://$res*auth.res/XRIA”.

Generic XRI authority resolution service allows a client to request resolution of multiple authority sub-segments in one transaction (“lookahead” resolution). If a client makes such a request, the responding authority MAY perform the additional lookahead resolution steps requested. In this case the responding authority will act as a client to the other authorities that need to be queried for the lookahead segments.
If an authority performs lookahead resolution, it MUST return an ordered list of xs:XRIDescriptor elements in an XRlDescriptors document. Each XRl Descriptor MUST correspond to a sub-segment resolved by the authority on behalf of the resolving client. The list of xs:XRIDescriptor elements in the XRlDescriptors document MUST appear in the same order as the sub-segments in the original request. The responding authority MAY resolve fewer sub-segments than requested by the client. The responding authority is under no obligation to resolve more than the first sub-segment (for which it is authoritative.)

Figure 3 illustrates a resolving client requesting lookahead resolution for the XRl 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 a resolving client itself, or by examining a cache it may have built through previous resolutions. In this example, the "@" authority it is only willing or able to resolve the descriptor for "*a" (for which it is authoritative) plus "@a*b". Therefore the resolving client must resolve "*c" itself. The resolving client will know the "@" authority only resolved two segments (*a and *b) because it only returned two XRl Descriptors corresponding to those two sub-segments.

If the responding authority does not resolve the entire set of sub-segments presented, the resolving client MUST continue the authority resolution process itself. At any stage, however, the resolving client MAY request that the next authority resolve any additional unresolved sub-segments. 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 XRl authority segment.

2.2.4.1 Determining the URI for the Next Resolution Step

Before each authority resolution step is performed, a URI must be constructed for the next HTTP(S) request. This URI establishes the context of that authority. Initially the current context is the root authority, and the current context shifts to a new authority each time a resolution step is performed. After a lookahead resolution request, the current context is the last authority whose sub-segment was resolved by the authority performing the lookahead request.

This "Next Authority URI" is constructed from two strings:

- The contents of the xs:XRIDescriptor/xs:Authority/xs:URI element extracted from the XRl Descriptor corresponding to the current context, and
• The next qualified sub-segment to be resolved. (Note that this sub-segment must begin
with an XRI syntax delimiter, i.e., "*" or "!" —see section 2.2.6.)

If the path portion of the first URI does not end with a "/" character, one must be appended before
proceeding. Then the URI-normal form (section 2.3.1 of [XRISyntax]) of the next qualified sub-
segment being resolved is appended to the path portion of this URI. For example, when resolving
the "c" sub-segment of "xri://@a*b*c*d", 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 XRID for the context "xri://@a*b*c" is retrieved.

In lookahead resolution (Figure 3), any portion of the remaining XRI authority segment may be
appended. For example, if the resolving client wanted to resolve "*c*d", it would append this
entire string to "http://example.com/xri-authority!", yielding "http://example.com/xri-authority/*c*d".

Construction of the Next Authority URI is more formally described in this pseudo-code for
resolving a "sub-segment-list" via an HTTP URI called "xa-uri":

```plaintext
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):
    prepend "*" to sub-segment-list

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

### 2.2.4.2 Making HTTP(S) Resolution Requests

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 section 3.3.3 for
a different value that may appear in the Accept header during trusted resolution.

The HTTP/HTTPS response for a successful resolution MUST contain either: a) a 2XX response
with an XRI Descriptors document containing a list of one or more `xrid:XRIDescriptor`
elements, or b) a 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 an 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
an XRI Descriptors document containing one or more `xrid:XRIDescriptor` elements for the
qualified sub-segment(s) being resolved.

If there are no more sub-segments, the final context (as described by the final XRID retrieved)
can be used for local access services as described in section 2.4, or to obtain synonyms or other
metadata about the final authority.

### 2.2.4.3 Proxied Resolution

In some cases it may be desirable for an HTTP(S) server to do fully proxied XRI resolution on
behalf of a client. While this is very similar to lookahead resolution, it differs in that a lookahead
resolution request is always sent to the authority for the first sub-segment in the resolution

Comment [gmw10]: Used to be section 2.2.9 and it has also been modified
request. A proxied resolution request, in contrast, may be sent to any XRI proxy server that will accept the request.

The proxy resolution service is very simple: an HTTP GET is performed on a URI constructed by concatenating the base URI for the proxy resolution service and the XRI authority segment for which proxy resolution is being requested. As with standard resolution: a) this XRI authority segment MUST be in URI-normal form, and b) if the base proxy URI does not contain a trailing forward slash "/", one MUST be inserted between the base URI and the XRI authority segment. The proxy answering this request MUST perform XRI authority resolution as specified in this document and MUST return either: a) an XRI Descriptors document containing a entire chain of xrid:XRIPropertyDescriptor elements for the segments of the authority it resolves, or b) an HTTP error code as described in section 2.2.4.4.

Note that because a proxy is not associated with any specific authority, proxied resolution requests MUST be comprised of authority segments starting with a GCS character or a cross-reference identifying a community root authority. In addition, a proxy resolver MUST return an XR! Descriptor chain that begins with an XR!ID describing the community root authority. If the community root authority does not publish an XR!ID itself, a proxy MUST construct one from the equivalent information published by the community root authority.

The following example illustrates a proxied resolution request for "xri://=example*home*base". It assumes that the URI for a local proxy server is "http://proxy.example.com/xri-proxy". First the following HTTP GET request is made to "proxy.example.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 completing this resolution process, the proxy resolver might produce 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>=</Resolved>
  </XRIDescriptor>
  ...
  <XRIDescriptor>
    <Resolved>*example</Resolved>
  </XRIDescriptor>
  ...
  <XRIDescriptor>
    <Resolved>*home</Resolved>
  </XRIDescriptor>
  ...
  <XRIDescriptor>
    <Resolved>*base</Resolved>
    <Service>
      <Type>xri://$res*local.access/X2R</Type>
      <URI>http://xri.other.example.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.

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

Also note that this proxy resolution service does not provide a complete XRI-to-resource mapping service. For the default X2R local access protocol, such a service could be defined by sending the proxy server the complete XRI in XRI-normal form and having it return an HTTP redirect to the local access URI. Alternatively the proxy server could return the resource directly by performing the local access request itself. Neither method, however, is prescribed or defined by this document.

2.2.4.4 Errors During Proxied and Lookahead Authority Resolution

Proxies and lookahead resolvers SHOULD return any HTTP error codes returned during resolution back to the resolving client. 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, and successfully resolves @a*b, but receives a HTTP 404 on resolving *c, it should return an HTTP 404 response to its client that include xrid:XRIDescriptor elements for @, *a, and *b. This will indicate to the resolving client that *c is the sub-segment causing the 404 response.

This use of error codes, while slightly unusual, conforms to the requirements of [RFC2616], specifically sections 10.4 and 10.5 which state that “the server SHOULD include an entity containing an explanation of the error situation.” The combination of the error code and the list of successfully resolved xrid:XRIDescriptor elements indicates to the client exactly which sub-segment caused the error. This should save both the client and the authority returning the error code an extra HTTP request/response cycle.

Even when given an HTTP error response, resolving clients SHOULD consider the xrid:XRIDescriptor elements returned in the content of the HTTP response as valid cacheable responses (if a client does caching in the first place). All other rules about XRI Descriptors (including those specified in Section 3 for trusted resolution) also apply.

2.2.5 Examples (Non-Normative)

2.2.5.1 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
```
This example assumes that the URI for the “=” global context symbol is http://equals.example.org/xri-resolve, found in xrid:XRIDescriptor/xrid:Authority/xrid:URI of the XRID for this community.

**Resolving “=example”**

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

```
GET /xri-resolve/*example HTTP/1.1
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”, the value specified in the the “If-Modified-Since” header):

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

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

Resolving "*example*home*base"

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

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

200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:33 GMT

<XRIDescriptors xmlns="...">
  <XRIDescriptor>
    <Resolved>*base</Resolved>
    <Service>
      <Type>
        xri://$res*local.access/X2R
      </Type>
      <URI>
        http://xri.other.example.com/xri-local/base/
      </URI>
      <URI>
        https://xri.other.example.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*local.access/X2R type).
2.2.5.2 Authority Resolution with Lookahead

The next example shows the interaction between a client and server using lookahead resolution when resolving the authority portion of the following XRI:

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

Assume as in the previous example that the URI for the "=" global context symbol is "http://equals.example.org/xri-resolve". In this example, the client will always request lookahead resolution of all unresolved authority sub-segments.

Resolving "=example*home*base"

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

```
GET /xri-resolve/*example*home*base HTTP/1.1
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", the value specified in the the "If-Modified-Since" header).

```
The response contains two XRIDescriptors elements, one for "=example" and one for "=home". This indicates to the resolving client that the "equals.example.org" authority has either cached or performed 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>
    <Resolved>=example</Resolved>
    <Authority>
      <URI>
        http://xri.example.com/xri-resolve/
      </URI>
      <Service>...</Service>
    </XRIDescriptor>
  </XRIDescriptor>
  <XRIDescriptor xmlns="...">
    <Resolved>=home</Resolved>
    <Authority>
      <URI>
        http://xri.other.example.com/xri-resolve/*home/
      </URI>
      <Service>...</Service>
    </XRIDescriptor>
  </XRIDescriptor>
</XRIDescriptors>
```

Note that the XRI Descriptor elements must appear in resolution order, i.e. the first XRI Descriptor describes the authority "=example" and the second describes the authority "=home" within the "=example" namespace.
The resolving client, assuming it trusts the resolver’s response (see section 3 for more details on trusted resolution), then resolves the “base” authority sub-segment using the authority URI “http://xri.other.example.com/xri-resolve/"home/" as identified in the last XRI Descriptor above. The following HTTP request is made to “xri.other.example.com”:

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

The following HTTP response is received from xri.other.example.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>
    <Resolved>*base</Resolved>
    <Service>
      <Type>xri://$res*local.access/X2R</Type>
      <URI>http://xri.other.example.com/xri-local/base/</URI>
    </Service>
  <XRIDescriptor>
  <XRIDescriptors>
```

Note that the three XRI Descriptor elements in this example (two from the first HTTP resolution from equals.example.org and the one from xri.other.example.com) are exactly the same three XRI Descriptors retrieved from the separate resolution requests showed in section 2.2.5.1.

2.2.6 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 the cross-reference is considered opaque by generic XRI resolution. 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.

The one exception is the special cross-reference for XRI annotations as defined in the XRI Metadata Specification [XRIMetadata]. This cross-reference begins with the GCS symbol “$” followed by the hyphen character “-”. As specified in [XRIMetadata], such a cross-reference and the delimiter that precedes it MUST be ignored entirely during resolution.

Table 5 provides several examples of resolving cross-references. In each example, sub-segment “!b” resolves to an XRI Authority URI of “http://example.com/xri-authority/”, and lookahead resolution is not being requested.

<table>
<thead>
<tr>
<th>Cross-reference type</th>
<th>Example XRI</th>
<th>Next Resolution URI after resolving “xri://@lab”</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 2.2.7 XRI Redirects

An XRIDescriptor may contain an `<xrid:XRIDescriptor/xrid:Synonyms/xrid:External>` element and not contain any `<xri:XRIDescriptor/xri:Authority>` or `<xri:XRIDescriptor/xri:LocalAccess>` elements. This is called an "XRI redirect" because 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 0 of this document.

The example in Section 2.2.5 demonstrates the resolution of `xri://example*home*base/foo*bar`. The first request is to "equals.example.org". The following XRI redirect could be received as a response.

```
200 OK HTTP/1.1
Content-Type: application/xrid+xml
Expires: Fri, 7 Nov 2003 19:43:31 GMT
<br>other HTTP headers><XRIDescriptors xmlns="...">
  <XRIDescriptor>
    <Resolved>*example</Resolved>
    <Synonyms>
      <External>
        xri://@example2
      </External>
    </Synonyms>
  </XRIDescriptor>
</XRIDescriptors>
```

In this case, a new XRI would be constructed as "xri://example2*home*base/foo*bar" and the resolution process would begin again with this new XRI.

If the original XRI has additional sub-segments in the XRI Authority component and the `<xrid:XRIDescriptor/xrid:Synonyms/xrid:External>` element contains a local-path component, the client SHOULD consider this an error condition and fail. For example, consider if the XRI redirect above had been as follows:
Now 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.

2.3 IRI Authority Resolution

From the standpoint of generic XRI resolution as defined in this specification, an IRI authority segment represents either a DNS name or an IP address at which an XRID for the authority may be retrieved. Requesting the corresponding XRID is a simple matter of making an HTTP(S) GET request using a URI created from by the IRI authority segment. The resulting XRI Descriptor is then used to retrieve Local Access URIs or other XRI authority synonyms or metadata as described in section 2.2.

The HTTP URI is constructed by extracting the entire IRI authority segment and prepending the string "http://". Then an HTTP GET is performed using an HTTP Accept header containing only the following:

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

Additionally, the HTTP GET request MUST have a Host: header (as defined in section 14.23 of [RFC2616]) containing the value of the IRI authority segment. The resolving authority MUST use the value of the Host header to populate the XRIDescriptor/Resolved element in the resulting XRIDescriptors document. For example:

```
Host: example.com
```

An HTTP server acting as an IRI authority SHOULD respond with the XRIDescriptors document for that authority.

Section 3 of this document defines trusted resolution only for XRI authorities. This document does not define trusted resolution for IRI Authorities. If, however, an IRI authority is known to respond to HTTPS requests (by some means not described in this document) then the resolving client MAY use HTTPS as the access protocol for retrieving the authority’s XRID. If the resolving client is satisfied, via transport level security mechanisms, that the response is from the expected IRI authority, then the resolving client may place a higher level of trust on the contents of the XRID than it would have otherwise.

The following example demonstrates how the IRI authority segment of the XRI "xri://example.com/local*path" would be resolved into an XRI Descriptor. First the IRI authority is extracted ("example.com"), then the following HTTP Request is made of the server...
"example.com":

GET / HTTP/1.1
Accept: application/xrid+xml
Host: example.com
<other HTTP headers>

The HTTP server acting as the authority might provide the following HTTP response, using the value of the Host: header to fill the contents of the XRIDescriptor/Resolved element:

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>
    </Synonyms>
  </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 XRI authority syntax, 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 of or metadata about 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(S) local access protocol with the name “X2R”. Other local access services could be defined such as an LDAP or DSML local access protocol that specified the appropriate 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, "$res*local.access”, for enumerating local access service types. The "$res" namespace can also be extended by other authorities besides the XRI Technical Committee. See [XRIMetadata] for more information about extending "$" namespaces.

New local access service types intended for widespread use MUST be identified with XRIs in the $res*local.access namespace. Local access service types defined solely for use within a private or closed community MAY have service types identified by any XRI.

2.4.2 The X2R Local Access Service

The X2R local access service is derived from the I2R service defined in section 4.3 of [RFC2483]. X2R is the default local access service defined in this specification; it is available.
when the associated xrid:Descriptor/xrid:Service/xrid:Type element is not present or when it explicitly contains the value "xri://$res*local.access/X2R".

At a high level, X2R is 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 will use GET 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. HTTP local access could be leveraged for the delivery of SOAP messages over HTTP POST, for example, or via use of the GET HTTP verb as a generic read-only operation.

The HTTP/HTTPS local access binding defined in this section is flexible enough to be used for a variety of resources. By itself it makes no assumptions about the type of resource identified by the XRI being resolved, however such metadata can be supplied using the xrid:XRIDescriptor/xrid:Service/xrid:MediaType element in an XRID. The resource type may also be established through the context in which the XRI was originally used (e.g., an XML document) or discovered through the HTTP Content-Type header.

### 2.4.2.1 Constructing a Local Access HTTP(S) URI

The HTTP(S) URI upon which to perform X2R local access service is constructed by concatenating the value of any xrid:XRIDescriptor/xrid:Service/xrid:URI element in the XRI Descriptor with the URI-normal form of the path portion (matching the xri-path-absolute production described in section 2.2.3 of [XRISyntax]) of the XRI. If the URI from the XRI Descriptor ends in a forward slash "/", then the slash MUST be removed before concatenating the path portion.

The following pseudocode describes the process for creating, from the local access URI in the XRID, the concrete HTTP(S) URI to which a local access request is made:

```plaintext
if (local-access-uri ends in "/"):
  remove trailing "/" in local-access-uri
local-access-uri = local-access-uri + uri-escape(relative-path)
```

The verb used in the resulting HTTP(S) request may be any of the verbs defined in [RFC2616], though not all verbs may be supported at every endpoint. All X2R 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":

```
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 might then be received from xri.example.com:
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 in 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.

Note that HTTP Cache headers SHOULD NOT conflict with expiration information in an XRID. That is, the expiration date specified by HTTP caching headers MUST NOT be later than any of the expiration dates for any of the xrid:XRIDescriptor/xrid:Expires elements returned in the HTTP response. This implies that lookahead and proxy resolvers have to compute the “soonest” expiration date for XRI Descriptors in a resolution chain and ensure that the HTTP caching headers for the HTTP response do not express a later expiration date.

2.5.2 Location

During 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, as specified in section 2.5.1.

2.5.3 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-saml+xml” specifying that the content is an XRI Descriptor (section 0) or a trusted XRI Descriptor (section 3.3.1) respectively.

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 if the resource is found and an appropriate content type is returned.

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.

2.7 Caching and Efficiency

In addition to HTTP-level caching, resolution clients are encouraged to perform caching at the application 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 that 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 xrid:XRIDescriptor/xrid:Expires element (if present). That is, if the expiration time in xrid:XRIDescriptor/xrid:Expires is sooner than the expiration time calculated from the HTTP caching semantics, then the XRI Descriptor MUST be discarded before the expiration time in xrid:XRIDescriptor/xrid:Expires. Note also that the SAML assertion used in trusted resolution (section 3) may cause invalidation of a XRI Descriptor even before HTTP caching semantics or the xrid:XRIDescriptor/xrid:Expires element.

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 independent, and intermediate results can typically be cached in their entirety. For this reason, resolution of higher-level (further left) qualified sub-segments, which are common to more identifiers, will naturally result in a greater number of cache hits than resolution of lower-level sub-segments.
3 Trusted Resolution

3.1 Introduction

This section defines a method for performing trusted XRI authority resolution as an extension of the generic XRI resolution protocol defined in section 2 of this document.

This trusted resolution protocol 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 specification.

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

3.2 Overview and Example (Non-Normative)

Trusted XRI Authority resolution is a straightforward enhancement to generic XRI resolution. The client application requests resolution of one or more qualified sub-segments in the XRI Authority segment exactly as described in section 2 of this document with one exception: instead of using "application/xrid+xml" in the "Accept" header of the HTTP(S) request, a content type of "application/xrid-t-saml+xml" is used. The XRI Authority responds with an XRI Descriptor that contains an additional element - a digitally signed SAML [SAML] assertion that asserts the validity of the containing XRI Descriptor. If the response does not contain a valid, digitally signed SAML assertion (as defined in section 3.2 of this document), trusted resolution may not proceed.

The following example will step through resolution of the authority portion of the same XRI used in Section 2 of this document:

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

As in standard resolution, there is no defined discovery for the trusted resolution URI(s) of the community root – it must be known a priori and is part of the configuration of the resolver. A recommended practice is to publish an XRI Descriptor containing a valid SAML assertion signed by the community root. In this example, assume the xrid:Authority/xrid:URI element of the XRI Descriptor for the global community root “=” specifies that the URI for the “=” global context symbol is “http://equals.example.org/xri-resolve”.

In trusted resolution, each XRI Authority is associated with an identifier called an AuthorityID. An AuthorityID is a URI or XRI in URI-normal form uniquely associated with a particular XRI Authority. Each XRI Authority MUST have at least one AuthorityID, and no two XRI Authorities can have the same AuthorityID. The AuthorityID of the community root, like the community root’s URI, is defined in the xrid:XRIDescriptor/xrid:Authority/xrid:AuthorityID element of the community root’s XRI Descriptor (or its equivalent). For this example, assume the AuthorityID for the “=” global context symbol is “urn:uuid:498FB006-B9EF-4943-B10A-A71FC2ED1B89”. For more information on xrid:XRIDescriptor/xrid: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 an XRI Descriptor is used to describe the community root, information about this key may be found in the xrid:XRIDescriptor/xrid:Authority/ds:KeyInfo element of that document.

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
Accept: application/xrid-t-saml+xml

Example 1: Request for =example

Notice the use of the Accept header with the value "application/xrid-t-saml+xml". The client is requesting a response that contains a signed SAML assertion. If the resolving client will accept either trusted or generic resolution, it could have used the value "application/xrid-t-saml+xml, application/xrid+xml" for the Accept header.

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

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

<XRIDescriptors

xmlns="xri://$res*schema/XRIDescriptor">

<xRIDescriptor

xrid:id="baec221f3c0f17f53ca6839989632056">

<Resolved>*example</Resolved>

<AuthorityID>urn:uuid:498FB006-B9EF-4943-B10A-A71FC2ED1BB9</AuthorityID>

<Authority>

<AuthorityID>urn:uuid:C5C9EFDF-A3BC-4301-88C6-B1AE0AD6DA77</AuthorityID>

<URI xrid:trusted="true">http://xri.example.com/xri-resolve/</URI>

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

...<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:SignedInfo>
```
Example 2 – Response for =example

The response contains an xrid:XRIDescriptor/saml:Assertion element that provides an assertion about the validity of the XRI Descriptor. For more information about SAML assertions in XRI Descriptors, see section 3.3.3. The response also contains an xrid:XRIDescriptor/xrid: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.

Finally, notice that two instances of xrid:AuthorityID appear in the XRI Descriptor, one as a child of xrid:XRIDescriptor and one as a child of xrid:Authority. The child of xrid:XRIDescriptor is the AuthorityID of the current describing authority (the one publishing this XRI Descriptor) and matches the expected AuthorityID of the community root (urn:uuid:498FB006-B9EF-4943-B10A-A71FC2ED1B89). The child of the xrid:Authority element contains the AuthorityID of the described XRI Authority (the authority being described within the xrid:Authority element). Responses from that XRI Authority will contain this AuthorityID as a child of xrid:XRIDescriptor.

The client validates the signed SAML assertion as described in Section 3.3 before continuing.

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 following HTTP request with an Accept header value of “application/xrid-t-saml+xml” is made to “xri.example.com”:

wd-xri-resolution-v2.0-wd05

Copyright © OASIS Open 2005. All Rights Reserved.
Example 3 – Request for *home

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

```
GET /xri-resolve/*home HTTP/1.1
Accept: application/xrid-t-saml+xml

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-saml+xml
If-Modified-Since: Fri, 31 Oct 2003 19:43:32 GMT

<XRIDescriptors
xmlns="xri://$res*schema/XRIDescriptor">
<XRIDescriptor
xrid:id="1f81b6e0-b64b-1026-f1bc-c0a80b9d3f5b">
<Resolved>*home</Resolved>
<AuthorityID>urn:uuid:C5C9EFDF-A3BC-4301-88C6-B1AE0A6DA77</AuthorityID>
<Authority>
<AuthorityID>urn:uuid:A9F28515-AB03-4883-8852-8EECB54CE1D5
</AuthorityID>
<URI xrid:trusted="true">
http://xri.example.com/xri-resolve/*home/
</URI>
</Authority>
<Service>…</Service> <!-- Local Access Service -->
<TrustMechanism>xri://$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-B1AE0A6DA77</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>
</ds:Reference>
<ds:Transforms>
<ds:DigestMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
<ds:Reference URI="http://xri.example.com/xri-resolve/*home/"
ID="_66f1f3e0-b64b-1026-34a4-c0a80b9d59c1" />
<ds:DigestValue>BSsnowZG5DYV0X0c0a80b9d3f5b</ds:DigestValue>
</ds:Reference>
</ds:Transforms>
<ds:SignatureValue>
</ds:SignatureValue>
```

wd-xri-resolution-v2.0-wd05  7 March 2005
Copyright © OASIS Open 2005. All Rights Reserved.  Page 29 of 55
Example 4 – Response for *home

The client validates the SAML assertion as described in Section 3.3 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”. This is the target of the next trusted resolution request, again with the Accept header value “application/xrid-t-saml+xml”:

Example 5 – Request for *base

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

200 OK HTTP/1.1
Expires: Fri, 7 Nov 2003 19:43:33 GMT

<xRIDescriptors
xmlns="xri://$res*schema/XRIDescriptor">
hamster<XRIDescriptor
xrid:id="7600e1a0-b64d-1026-ea89-c0a80b9d3814">
<Resolved>*base</Resolved>
<AuthorityID>urn:uuid:A9F28515-AB03-4883-8852-8EECB54CE1D5</AuthorityID>
<Service>
</XRIDescriptor> </XRIDescriptors>

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

<other HTTP headers>

Example 4 – Response for *home

The client validates the SAML assertion as described in Section 3.3 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”. This is the target of the next trusted resolution request, again with the Accept header value “application/xrid-t-saml+xml”:

Example 5 – Request for *base

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

200 OK HTTP/1.1
Expires: Fri, 7 Nov 2003 19:43:33 GMT

<xRIDescriptors
xmlns="xri://$res*schema/XRIDescriptor">
hamster<XRIDescriptor
xrid:id="7600e1a0-b64d-1026-ea89-c0a80b9d3814">
<Resolved>*base</Resolved>
<AuthorityID>urn:uuid:A9F28515-AB03-4883-8852-8EECB54CE1D5</AuthorityID>
<Service>
</XRIDescriptor> </XRIDescriptors>

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

<other HTTP headers>
<URI>http://xri.example.com/xri-local/base/</URI>
<URI>https://xri.example.com/xri-local/base/</URI>

<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: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
            5NI10
            Kvv2UrgyQ5kgQ9dm7/563RzKAAiQwMopSpTF114exw+nc8XEH+KnXdu/R9DH0g9k0BKIF6
            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">
            #baec221f3c0f17f53ca6839989632056</saml:AttributeValue>
        </saml:Attribute>
    </saml:AttributeStatement>
</saml:Assertion>
Example 6 – Response for “base

The SAML assertion is validated as described in Section 3.3 before proceeding. The result of the final XRI Authority resolution step is the set of HTTP and HTTPS URIs shown in the

The SAML assertion is validated as described in Section 3.3 before proceeding. The result of the final XRI Authority resolution step is the set of HTTP and HTTPS URIs shown in the

3.3 Trusted Resolution Protocol

This section normatively defines client and server behavior in trusted resolution. It also defines a new XML element for an XRI Descriptor, called TrustMechanism.

3.3.1 XML Elements and Attributes

Three elements of an XRI Descriptor that were defined in section 2.2.2 have limited usage in generic resolution but play a critical role in trusted resolution.

xrid:XRIDescriptor/xrid:AuthorityID

Always required, but critical in trusted resolution for identification of the current describing authority.

xrid:XRIDescriptor/xrid:Authority/xrid:AuthorityID

Always required, but critical for trusted resolution for identification of the target described authority.

xrid:XRIDescriptor/xrid:TrustMechanism

Required when providing trusted resolution. A URI or XRI in URI-normal form that specifies the mechanism used to provide trusted resolution. The URI for the trust mechanism defined in this specification is “xri://$res*trusted/XRITrusted”.

In addition, one element from the SAML [SAML] namespace is also critical for verifying the results of trusted resolution.

xrid:XRIDescriptor/saml:Assertion

Required when providing trusted resolution. A SAML assertion from the describing Authority (the one providing the XRI Descriptor) that asserts that the describing authority believes the information contained in the enclosing XRI Descriptor is correct. Because the assertion is digitally signed and the digital signature encompasses the containing XRI Descriptor, it also provides a mechanism for the recipient to detect unauthorized changes since the time the XRI Descriptor was published.

Finally, trusted resolution adds several new elements and attributes to XRI Descriptors to assist in verifying XRIDs produced by the next authority in the resolution chain (the authority being described by the xrid:XRIDescriptor/xrid:Authority element of the XRID):

xrid:XRIDescriptor/xrid:Authority/xrid: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 signed XRI Descriptors at this URI.

xrid:XRIDescriptor/xrid:Authority/ds:KeyInfo

Required when providing trusted resolution. Provides the key data needed to validate an XRI Descriptor provided by the described Authority as a result of resolution at the
3.3.2 Use and Correlation of AuthorityID Elements

Each XRI Authority participating in trusted resolution MUST be associated with exactly one AuthorityID and this AuthorityID MUST NOT ever be assigned to any other XRI Authority. In other words, AuthorityID is a persistent 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. To prevent this type of attack, the XRI Descriptor must be explicitly associated with a specific unique XRI Authority, and the client must have some means of verifying this association. The xrid:XRI Descriptor/xrid:AuthorityID element provides this explicit association.

There is no defined discovery for the AuthorityID of the community root; it must be published in the root XRID (or other equivalent description document) and verified independently. The AuthorityID for an XRI Authority other than the community root is described by an xrid:XRI Descriptor/xrid:AuthorityID element in the authority’s XRI Descriptor.

3.3.3 Client Behavior

From a client’s perspective, trusted resolution is identical to the generic resolution protocol described in section 2 of this document with the addition of the following REQUIRED behavior:

- The client MUST indicate to the resolving server that a signed XRI Descriptor is requested. This is expressed by adding an HTTP Accept header with the media type
identifier "application/xrid-t-saml+xml". Clients willing to accept either trusted or untrusted resolution descriptors may use a combination of "application/xrid-t-saml+xml" and "application/xrid+xml" in the Accept header as described in section 14.1 of [RFC2616].

- The client SHOULD NOT request trusted resolution from an authority unless the corresponding xrid:Descriptor/xrid:Authority/xrid: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. When XRIDescriptor elements may come both from freshly-retrieved XRID documents and from a local cache, an implementation MUST ensure that the requirements here are satisfied each time a resolution request is performed.

The client MUST confirm that each xrid: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 xrid: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 id (xrid:id attribute) of the xrid:XRIDescriptor element that is the immediate parent of the signed SAML assertion.
- If the digital signature enveloped by the SAML assertion contains 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 element present in the XRI Descriptor that was used to describe the current authority. For example, if Authority A provides an XRI Descriptor describing Authority B, and this XRID has an xrid:XRIDescriptor/xrid:Authority/xid:KeyInfo element that describes the key used to validate descriptors produced by Authority B, then this is Authority B’s “expected key”. This key should be used when validating XRI Descriptor elements produced by Authority B. Note that for a community root authority, the expected key is known a priori as part of the configuration in the client for that particular community root.
- The client confirms that the value of the xrid:XRIDescriptor/xrid:Resolved element matches the sub-segment whose resolution resulted in the current XRI Descriptor.
- The client confirms that the value of the xrid:XRIDescriptor/xrid:AuthorityID element matches the XRI Authority’s “expected AuthorityID”. As with the key information, the “expected AuthorityID” is the value of xrid:XRIDescriptor/xrid:Authority/xid:AuthorityID in the XRI Descriptor that describes the current Authority. Note that for a community root authority, the XRI Authority’s expected AuthorityID is known a priori and is part of the configuration in the client for that particular community root.
- The client confirms that the value of the xrid:XRIDescriptor/xrid:Resolved element matches the value of both the xrid:XRIDescriptor/saml:Assertion/saml:Issuer element and the NameQualifier attribute of the xrid:XRIDescriptor/saml:Assertion/saml:Subject/saml:NameID element.
- The client confirms that the value of the xrid:XRIDescriptor/xrid:Resolved element matches the value of the xrid:XRIDescriptor/saml:Assertion/saml:Subject/saml:NameID element.
• The client confirms that the value of the
  xrid:XRIDescriptor/xrid: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 a valid trusted resolution response 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 xrid:Authority element and the response from one fails to meet the
requirements above, the client may repeat the current iteration using the second URI. If the
second URI produces a sufficient response, it may be considered a valid trusted resolution
response as defined by this document and trusted resolution may continue.

3.3.4 Server Behavior

From the server’s perspective, trusted resolution is identical to the generic resolution protocol
described in section 2 of this document with the addition of the following behavior. This behavior
is REQUIRED if a resolution client requests trusted resolution as described in section 3.2 and the
server intends to honor the client’s request.

If, during the HTTP(S) request/response interaction, the server agrees to return a trusted
resolution response (indicated by the content type of "application/xrid-t-saml+xml"), the
XRI Descriptor returned by the server must contain a saml:Assertion element as an
immediate child of xrid: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 xrid: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 xrid:XRIDescriptor element that is the immediate parent of the
  signed SAML assertion. The URI reference MUST NOT be empty; it MUST refer to the
  identifier contained in the xrid:id attribute of the xrid:XRIDescriptor element.
• The digital signature enveloped by the SAML assertion MAY contain a ds:KeyInfo
  element. If it is included, it MUST match the
  xrid:Descriptor/xrid:Authority/ds:KeyInfo element in the XRI Descriptor
  that 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 XRID). Because the signing key is known in advance by the resolution
  client, the ds:KeyInfo element SHOULD be omitted from the digital signature.
• The xrid:Resolved element MUST be present, and the value of this field MUST match
  the XRI Authority sub-segment requested by the client.
• The xrid:XRIDescriptor element MUST have an xrid:AuthorityID element as
  an immediate child. The value of the xrid:AuthorityID element MUST be the
  Authority ID, as described in Section 3.2, of the responding XRI Authority.
• The xrid:XRIDescriptor/xrid:TrustMechanism MUST be present and the value
  MUST be "xri://$res*trusted/XRITrusted".

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 3.
However, the server providing lookahead resolution MUST NOT return untrusted XRIDs if the
client requests trusted resolution.
3.3.5 Additional Requirements of Authorities Offering Trusted Resolution

The `xrid:XRIDescriptor/xrid: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 `xrid:XRIDescriptor/xrid:Authority/xrid:URI` element must contain the value "1" or "true".
- The `xrid:XRIDescriptor/xrid: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.
- The `xrid:XRIDescriptor/xrid:Authority` element MUST contain an `xrid:AuthorityID` element as an immediate child. The value of this field MUST be the `AuthorityID` of the described XRI Authority, i.e. the value that will appear in the `xrid:Descriptor/xrid:AuthorityID` element of an XRI Descriptor returned from the described XRI Authority.
- 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.
4 Extensibility and Versioning

4.1 Extensibility

The XRI Descriptors defined in this document are designed to be extended with other metadata. Therefore the XRIDescriptor schema is built on an open content model. In a number of places, extension elements and attributes from namespaces other than "xri://$res*/schema/XRIDescriptor*($v%2F2.0)" are explicitly allowed. These extension points are designed to simplify default processing of XRI Descriptors using 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. This means that even elements that would normally be 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 must be able to 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 Descriptor schema whose meaning or interpretation are to be modified by extension elements can be wrapped in a extension container element that is defined by the extension specification. This extension namespace will be in the same namespace as the extension elements that must be understood by the consumer of the XRI Descriptor. Using this pattern, all elements whose interpretations are modified by the extension will now be contained in an element (the extension container element) that will be ignored by consumers unable to process the extension.

The following example illustrates this pattern using an extension container element from an extension namespace ("other:SuperAuthority") that contains an extension 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 parent xrid:Authority element and therefore must be understood by the consumer for the proper interpretation of the parent xrid:Authority element. To preserve the correct interpretation of the xrid:Authority element in this context, the xrid:Authority element is "wrapped" so only consumers that understand elements in the "other:SuperAuthority" namespace will attempt to process the xrid:Authority element.

4.1.1 Specific Points of Extensibility

The use of HTTP and XML in the design of the generic resolution service, the trusted resolution service, and the X2R local access service provide the following specific points of extensibility:
• Specification of new authority resolution service types (xrid:Authority/xrid:Type in the XRI Descriptor).
• Specification of new local access service types (xrid:Service/xrid:Type in the XRI Descriptor).
• Specification of new trust mechanisms (xrid:TrustMechanism in the XRI Descriptor).

For example, an existing secure, private network in which resolution is intrinsically trustworthy may wish to express its own trust mechanism explicitly.
• 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 defined by [RFC2616] during identifier authority resolution or X2R local access.
• Use of cross-references within XRIs, particularly for associating new types of metadata with a resource.

4.2 Versioning

Versioning of the XRI specification set is expected to occur infrequently, however experience demonstrates such versioning is inevitable. Therefore guidelines on versioning are described in this section.

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

MajorB > MajorA OR ( ( MajorB = MajorA ) AND MinorB > MinorA )

4.2.1 Versioning of the XRI Resolution Specification

New releases of the XRI Resolution specification may specify changes to the resolution protocol and/or to resolution data structures. When changes affect either of these, the resolution specification version number will be changed. Where changes are purely editorial, the version number will not be changed.

In general, if a change is backwards compatible, the new version will be identified using the current major version number and a new minor version number. If the change is not backwards compatible, then the new version will be identified with a new major version number.

4.2.2 Versioning of XRI Descriptor Elements

Both the xrid:XRIDescriptors element and the xrid:XRIDescriptor element have Version attributes. The value of these attributes MUST be the version value of specification to which their containing elements conform.

When new versions of the XRI Resolution specification are released, the namespace for the XRI Descriptor schema may or may not be changed. If there is a major version number change, the namespace for the xrid:XRIDescriptors document is likely to change. If there is only a minor version number change, the namespace for the xrid:XRIDescriptors document may remain unchanged.

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.

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.

4.2.3 Versioning of Protocols

Both the authority resolution and local access protocols defined in this document may also be
versioned by future releases of the XRI Resolution specification. If these protocols are not
backwards compatible with older implementations, they will likely get a new XRI identifying them
in an XRI Descriptor.

Note that it is possible for version negotiation to happen in the protocol itself. For example, HTTP
provides a mechanism to negotiate the version of the HTTP protocol being used. When the XRI
resolution protocol provides its own version negotiation mechanism, the specification is likely to
continue to use the same XRI identifying the protocol as used in previous versions of the XRI
Resolution specification.
5 Security and Data Protection

Significant portions of this specification deal directly with security properties, and shall not be summarized again here. In addition, general security practices and known risks in resolution protocols are well documented in many other specifications. Only security considerations directly relevant to XRI resolution are included here.

5.1 DNS Spoofing

XRI resolution responses that depend on DNS resolution of an XRI authority URI are no more secure than the underlying DNS query. To solve that, either the use of DNSSEC [DNSSEC] for securing and verifying zones, the use of server authenticated HTTPS, or the use of trusted resolution as defined in this specification is recommended.

5.2 HTTP Security

Many of the security considerations set forth in HTTP/1.1 [RFC2616] apply to XRI Resolution protocols defined here. In particular, confidentiality of the communication channel is not provided for by HTTP. Server-authenticated HTTPS should be considered in cases where confidentiality of resolution requests and responses is desired. Special consideration should be given to proxy and caching behaviors to ensure accurate and reliable responses from resolution requests. In particular, for various reasons network topologies increasingly have transparent proxies, some of which may insert VIA and other headers as a consequence, or may even cache content without regard to proper caching policies set for by the HTTP authority for a resource.

5.3 Caching Authorities

In addition to traditional HTTP caching proxies, XRI resolution authority proxies may exist in the resolution topology. These entities should take special precautions against cache poisoning (as described in [RFC....]), as these caching entities may represent trust decision points within a deployment's resolution architecture.

5.4 Lookahead and Proxy Resolution

During proxyhead resolution, some or all of the XRI Authority is provided to the proxy resolver. During lookahead resolution sub-segments may be revealed to the network endpoint providing resolution for which that endpoint is not authoritative. In both cases, privacy considerations should be evaluated before disclosing such information.

5.5 SAML Considerations

For trusted resolution, rules defined by the SAML 2.0 Core Specification must be adhered to. Particularly noteworthy are the XML Transform restrictions on XML Signature defined in SAML and the enforcement of the SAML Conditions element regarding the validity period.

5.6 Community Root Authorities

The XRI Authority information for a community root needs to be well-known to the clients that request resolution within that community. For trusted resolution, this includes the URIs, the AuthorityID, and the ds:KeyInfo information. A suitable way to provide this information is for the community root authority to provide a self-signed XRI Descriptor and publish it to a server-authenticated HTTPS endpoint. Special care should be taken to ensure the correctness of such
an XRID; if this information is incorrect, an attacker may be able to convince a client of an incorrect result during trusted resolution.

5.7 DOS Attacks

XRI Resolution, including trusted resolution, is vulnerable to the typical DOS attacks found in systems relying on DNS and HTTP.

5.8 Limitations of Trusted Resolution

While the trusted resolution mechanism specified in this document provides a way to verify the integrity of a successful XRI resolution, it does not provide a way to verify the integrity of a resolution failure. Reasons for this limitation include non-malicious network failures, DOS attacks, and the ability for a MITM attacker to modify HTTP responses when resolution is not performed over HTTPS.

Additionally, there is no revocation mechanism specified for trusted resolution. Therefore, a signed resolution’s validity period should be limited appropriately to mitigate the risk of an incorrect or invalid resolution.
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:
7 Media Type Registration for application/xrid-t-saml+xml

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

MIME media type name: application
MIME subtype name: xrid-t-saml+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:

Comment [DM21]: Complete this section.
8 References

8.1 Normative


Comment [DM22]: TODO – remove unused references.
8.2 Informative


Comment [DM23]: Fix

Comment [DSR24]: Add Link
G. Wachob, D. Reed, M. Le Maitre, D. McAlpin, D. McPherson,

Extensible Resource Identifier (XRI) Requirements and Glossary v1.0,
## Appendix A. Revision History

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>By Whom</th>
<th>What</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>2005-01-15</td>
<td>All Editors</td>
<td>Initial document.</td>
</tr>
<tr>
<td>3.0</td>
<td>?</td>
<td>Gabe Wachob</td>
<td>Incorporate feedback from other editors.</td>
</tr>
<tr>
<td>4.0</td>
<td>2005-02-17</td>
<td>Gabe Wachob</td>
<td>Incorporate feedback. Add Extensibility and Versioning section.</td>
</tr>
<tr>
<td>5.0</td>
<td></td>
<td>Gabe Wachob</td>
<td>Incorporate feedback and editorial revisions.</td>
</tr>
</tbody>
</table>
Appendix B. XML Schema for XRI Descriptor
(Normative)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="xri://$res*schema/XRIDescriptor*($v%2F2.0)"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xrid="xri://$res*schema/XRIDescriptor*($v%2F2.0)" elementFormDefault="qualified">

<!-- Utility patterns -->
<xs:attributeGroup name="otherattribute">
    <xs:attribute namespace="##other" processContents="lax"/>
</xs:attributeGroup>

<xs:group name="otherelement">
    <xs:choice>
        <xs:attribute namespace="##other" processContents="lax"/>
        <xs:attribute 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" maxOccurs="unbounded"/>
            <xs:element ref="xrid:AuthorityID" maxOccurs="0"/>
            <xs:element ref="xrid:Expires" minOccurs="0" maxOccurs="unbounded"/>
            <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" maxOccurs="unbounded"/>
            <xs:element ref="xrid:TrustMechanism" minOccurs="0" maxOccurs="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>

<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 name="Authority">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="xrid:AuthorityID" minOccurs="0" maxOccurs="unbounded"/>
      <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" maxOccurs="unbounded"/>
      <xs:group ref="xrid:URI" maxOccurs="unbounded"/>
      <xs:element ref="xrid:MediaType" minOccurs="0" maxOccurs="unbounded"/>
      <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:sequence>
    <xs:attributeGroup ref="xrid:otherattribute"/>
  </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="2.0"/>
<xs:attribute name="trusted" type="xs:boolean"/>
<xs:attribute name="id" type="xs:ID"/>
</xs:schema>
Appendix C. RelaxNG Compact Syntax Schema for XRI Descriptor (Non-normative)

```
namespace xrid="xri://$res*schema/XRIDescriptor*($v%2F2.0)"
namespace xml="http://www.w3.org/XML/1998/namespace"
namespace local=""

start=XRIDescriptors

# Utility patterns
anything = ( element * {anything} | attribute * {text} | text ) *
otherattribute = attribute -*{xrid:*|local:*} {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 *, Syonyms ?,
  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
}
```

wd-xri-resolution-v2.0-wd05  7 March 2005
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

# Synonyms Definition
Synonyms = element xrid:Synonyms { Synonyms-content }
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
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.
Appendix E. LAST MINUTE 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: 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 All XML Paths should use XPath style and all element names should be qualified (mostly that’s xrid: for the elements we define)
- Last Minute: What about Acknowledgements section?