Abstract

LRDD (pronounced 'lard') provides a process for obtaining information about a resource identified by a URI. The 'information about a resource' - a resource descriptor - provides machine-readable information that aims to increase interoperability and enhance interactions with the resource. LRDD provides a narrow and well-defined set of rules for obtaining and processing link-based descriptors (found in multiple places such as HTTP headers, document markup, and resource descriptors) which are often required for security and consistent client behavior.

Editorial Note (to be removed by RFC Editor)

Please discuss this draft on the apps-discuss@ietf.org [1] mailing list.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on September 4, 2010.
Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the BSD License.

Table of Contents

1. Introduction ................................................. 3  
  1.1. Example ................................................. 3  
  1.2. Notational Conventions ................................. 6  
2. Resource Descriptor ......................................... 6  
  3.1. Source Priority Order .................................. 7  
  3.2. Information Sources .................................... 8  
4. Security Considerations ..................................... 10  
5. IANA Considerations ......................................... 11  
  5.1. The ‘lrdd’ Relation Type ............................... 11  
Appendix A. Acknowledgments ................................. 11  
Appendix B. Document History .................................. 11  
6. References .................................................. 13  
  6.1. Normative References .................................. 13  
  6.2. Informative References ............................... 14  
Author’s Address .............................................. 14
1. Introduction

LRDD defines a simple process for locating resource descriptors for URI-identified resources. Resource descriptors are machine-readable documents which provide information about resources (resource metadata) for the purpose of promoting interoperability and assist in interacting with unknown resources that support known interfaces.

For example, a web page about an upcoming meeting can provide in its descriptor the location of the meeting organizer’s free/busy information to potentially negotiate a different time. A social network profile page descriptor can identify the location of the user’s address book as well as accounts on other sites. A web service implementing an API with optional components can advertise which of these are supported.

Given the wide range of metadata needs, no single descriptor format or retrieval method can adequately accommodate every use case. While there are many methods for obtaining resource descriptors (e.g., links, HTTP headers, WebDAV’s PROPFIND [RFC4918], HTTP OPTIONS, URIQA’s MGET [URIQA]), LRDD utilizes the Web Linking framework defined in [I-D.nottingham-http-link-header]. LRDD defines a narrow profile of Web Linking for obtaining and processing link-based descriptors to accommodate the common discovery needs of many Web protocols.

In LRDD, the resource descriptor is constructed by obtaining and aggregating links and descriptor documents from multiple sources (e.g. HTTP headers, document markup, site-meta). In most cases, clients do not need to construct a complete resource descriptor, but instead process the information in a specific order until the desired information is found.

1.1. Example

An article published by a website allows readers to post comments and provide the web address of their own blog. The article page includes an avatar (a photo of the reader) when displaying comments, which is obtained by performing discovery on the web address provided by the reader (if supported by the reader’s blog).

After receiving a comment from Jane which has her own blog at "http://jane.example.com/blog", the website perform LRDD discovery to try and obtain the Jane’s photo.
First, the website determines the source priority order for Jane’s blog by fetching its host-meta document from "http://jane.example.com/.well-known/host-meta":

```xml
<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'
     xmlns:hm='http://host-meta.net/ns/1.0'>
  <hm:Host>jane.example.com</hm:Host>
  <Property type='http://lrdd.net/priority/resource' />
  <Link rel='lrdd' template='http://jane.example.com?lrdd={uri}'>
</XRD>
```

which indicates the blog is using Resource-priority (giving higher priority to links provided by the resource itself over those defined by the global policy). Since Jane’s blog uses Resource-priority, the website looks for "avatar" links in this order: <Link> elements, HTTP Link headers, and then host-meta link templates.

To obtain a markup representation of Jane’s blog, the website makes an HTTP "GET" request to "http://jane.example.com/blog":

```
GET /blog HTTP/1.1
Host: jane.example.com
Accept: text/html
```

And receives back (HTML simplified for display purposes):

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Link: <http://jane.example.com/author>; rel="author"

<html>
  <head>
    <link href='http://jane.example.com/image' rel='avatar' />
  </head>
  <body>
    <h1>Jane’s Blog</h1>
  </body>
</html>
```
The HTML markup includes the desired link. The website can fetch Jane’s photo from "http://jane.example.com/image".

Now that it has Jane’s photo, the website is looking for a short description of Jane to include with her comment. It performs another LRDD discovery, this time looking for an "about" link. Repeating the same process, the website looks for qualified <Link> elements in Jane's blog HTML markup and finds none. It then looks for a LRDD document - a descriptor document containing additional information about the resource - which uses the "lrdd" link relation. It finds none in the HTML representation.

In Resource-priority, the next source is HTTP header. The HTTP header (shown above with the HTML response) includes a qualified link to Jane’s author page.

Before it can display Jane’s photo and description, the website needs to find the copyright license used by Jane’s blog. It performs another LRDD discovery looking for a link with a "copyright" relation type. It fails to find such link in the HTML markup as well as a link to a LRDD document. It then tries and fails to find a "copyright" link in the HTTP header or a LRDD document linked from the header.

The website then proceeds to the host-meta source, looking for link templates. It fails to find a link to a copyright statement, but it does find a link to a LRDD document. It obtains the LRDD document for Jane’s blog by applying the blog’s URI to the template:

\[http://jane.example.com?lrdd=http%3A%2F%2Fjane.example.com%2Fblog\]

and obtains the LRDD document for Jane’s blog:

```xml
<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'>
  <Subject>http://jane.example.com/blog</Subject>
  <Link rel='copyright' href='http://jane.example.com/copyright'/>
</XRD>
```

The LRDD document provided by the host-meta link template includes a link to the copyright statement. The website now has all the information it needs to display Jane’s comment along with her photo.
1.2. Notational Conventions

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

2. Resource Descriptor

The LRDD resource descriptor is the aggregation of links and descriptor documents obtained from four sources (when applicable and available):

- **host-meta** - links generated by applying the resource URI to the link templates provided by the host’s host-meta document as defined in [I-D.hammer-hostmeta].

- **HTTP Link headers** - links included in the HTTP response header to an HTTP [RFC2616] "HEAD" or "GET" request for the resource.

- **<Link> elements** - links included in the resource representation markup.

- **LRDD documents** - links and other metadata contained in XRD [OASIS.XRD-1.0] documents linked to from any of the previous three link sources using the "lrdd" relation type.

LRDD clients usually look for a link with a specific relation type or other well-typed metadata. In such cases, the client does not need to construct the entire resource descriptor, but instead, process the various sources in their prescribed order until the desired information is found.

3. Discovery Process

The LRDD process begins with the URI of the resource being discovered and the type of information being sought:

- **Link of a given relation type** - the client is looking for a link with a specific relation type and other link attributes such as media type or relation-specific attributes.

- **Metadata contained within a LRDD-linked XRD document** - the client is seeking information other than a link to another resource, such as resource properties or other structured metadata.
The client MUST first determine the host source priority order as described in Section 3.1. Once the order is determined, the client process each of the sources listed in Section 3.2 as follows:

1. If the information being sought is a link: find a link matching the desired criteria by processing the links in order. If a qualified link is found, the process ends successfully.

2. Find the first link with the "lrdd" relation type. The link’s media type attribute MUST be set to "application/xrd+xml" if present. The link’s target is the location of the LRDD document. Any "lrdd" links other than the first MUST be ignored. If no qualified link is found, continue with the next source.

3. Obtain the LRDD document by following the scheme-specific rules for the LRDD document’s URI. If the document’s URI scheme is "http" or "https", the document is obtained via an HTTP "GET" request to the identified URI. The client MUST obey HTTP redirections (3xx). The document is considered valid only if retrieved with a successful HTTP response status (2xx) and is a valid XRD document per [OASIS.XRD-1.0]. If no valid document is found, continue with the next source.

4. If the information being sought is a link: find a link matching the desired criteria by processing the LRDD document as defined by [OASIS.XRD-1.0] section 4. If a qualified link is found, the process ends successfully.

5. If the information being sought is something other than a link: parse the LRDD document as defined by [OASIS.XRD-1.0], looking for the desired metadata. If the desired information is found, the process ends successfully.

6. If the information being sought is not found, continue with the next source. If all the sources have been exhausted, the process ends unsuccessfully.

3.1. Source Priority Order

LRDD descriptors include information contained or linked to from three sources: host-meta link templates, HTTP Link headers, and <Link> elements. To ensure consistent client behavior and to enable hosts to set their own security policy with regard to metadata authority, LRDD provides two processing profiles:

- Host-priority - Priority is given to links set by the host policy (via host-meta and HTTP Link headers) over those set by each individual resources (via <Link> elements). Clients MUST process
the three sources in the following order: host-meta link templates, HTTP Link headers, and <Link> elements.

- Resource-priority - Priority is given to the individual resource over the policies of the host. Clients MUST process the three sources in the following order: <Link> elements, HTTP Link headers, and host-meta link templates.

Host-priority is the default source priority order. Hosts that wish to use Resource-priority MUST declare it by setting the LRDD priority property in their host-meta document: "http://lrdd.net/priority/resource". The priority property does not have a value.

For example:

```xml
<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'
     xmlns:hm='http://host-meta.net/ns/1.0'>
  <hm:Host>example.com</hm:Host>
  <Property type='http://lrdd.net/priority/resource' />  
</XRD>
```

3.2. Information Sources

Each of the information sources supported by LRDD presents a different set of requirements and benefits. The criteria used to determine which sources a server MAY support are based on a combination of factors:

- The ability to offer and obtain a representation of the resource by dereferencing its URI.

- The availability of a representation supporting <Link> markup compatible with [I-D.nottingham-http-link-header].

- The availability of an HTTP representation of the resource and the ability to provide and access link information in its response header.

3.2.1. host-meta Link Templates

The host-meta document source is available for any resource identified by a URI with an authority that supports the host-meta document as defined in [I-D.hammer-hostmeta]. This method does not
require obtaining any representation of the resource, and operates solely using the resource URI.

Links between the resource URI and other resources are expressed by link templates as defined by [I-D.hammer-hostmeta] section 3.2. By applying the host-wide templates to an individual resource URI, a resource-specific link is constructed which can be used to express links without the need to access or provide a representation for the resource.

Clients MUST process the host-meta document, looking for link templates and applying the resource URI to produce a list of links to be used in the discovery process. Clients MUST retain the order of the links as present in the host-meta document. Any links contained in the host-meta document not using the "template" attribute MUST be ignored and excluded from LRDD processing, but are still valid for other purposes.

For example, the resource URI "http://example.com/x" and the following host-meta link template:

   <Link rel='author' template="http://example.com?author={uri}" />

generates an "author" link between "http://example.com/x" and "http://example.com?author=http%3A%2F%2Fexample.com%2Fx".

3.2.2. HTTP Link Headers

The HTTP Link header source is limited to resources for which an HTTP "GET" or "HEAD" request returns a 2xx, 3xx, or 4xx HTTP response per [RFC2616]. This method uses the Link header defined in [I-D.nottingham-http-link-header] and requires the retrieval of a resource representation header.

For example:

   Link: <http://example.com?author=http%3A%2F%2Fexample.com%2Fx>; rel="author"

Clients obtain HTTP Link header links by making an HTTP (or HTTPS as required) "GET" or "HEAD" request to the resource URI to obtain a valid response header. If the HTTP response carries a status code other than successful (2xx), redirection (3xx), or client error (4xx), the method fails.
Link headers can include multiple relation types in a single "rel" attribute (for example "rel="license copyright")"). Clients MUST properly process such multiple relation "rel" attributes as defined by [I-D.nottingham-http-link-header].

3.2.3. <Link> Elements

The <Link> element source is limited to resources with an available markup representation that supports typed-relations using the <Link> element, such as HTML [W3C.REC-html401-19991224], XHTML [W3C.REC-xhtml1-20020801], and Atom [RFC4287]. Other markup formats are permitted as long as the semantics of their <Link> elements are fully compatible with the link framework defined in [I-D.nottingham-http-link-header]. This source requires the retrieval of a resource representation. While HTTP is the most common transport for such documents, this source is transport independent.

For example:

   <LINK href='http://example.com?author=http%3A%2F%2Fexample.com%2Fx' 
       rel='author' />

Clients obtain <Link> element links by retrieving a representation of the resource using the applicable transport for that resource URI. If the markup document is obtained using HTTP, it MUST only be used by the client if the document is a valid representation of the resource identified by the HTTP request URI, typically received with a successful (2xx) response code. If no such valid representation of the request URI is found, the source MUST NOT be used.

The client MUST obey the document markup schema and ignore any invalid elements (such as <Link> elements outside the <Head> section of an HTML document). This is done to avoid unintentional markup from other parts of the document to be used for discovery purposes, which can have vast impact on usability and security.

Some <Link> elements allow multiple relation types in a single "rel" attribute (for example "rel='license copyright'`). Clients MUST properly process such multiple relation "rel" attributes as defined by the format specification.

4. Security Considerations

The methods used to perform discovery are not secure, private or
integrity-guaranteed, and due caution should be exercised when using them. Applications that perform LRDD SHOULD consider the attack vectors opened by automatically following, trusting, or otherwise using links gathered from <Link> elements, HTTP Link headers, or host-meta documents.

5. IANA Considerations

5.1. The ’lrdd’ Relation Type

This specification registers the "lrdd" relation type in the Link Relation Type Registry defined by [I-D.nottingham-http-link-header]:

Relation Name: lrdd

Description: Identifies a resource descriptor for the link’s context used by the LRDD protocol.

Reference: [[ This specification ]]

Appendix A. Acknowledgments

Inspiration for this memo derived from previous work on a descriptor format called XRDS-Simple, which in turn derived from another descriptor format, XRDS. Previous discovery workflows include Yadis which is currently used by the OpenID community. While suffering from significant shortcomings, Yadis was a breakthrough approach to performing discovery using extremely restricted hosting environments, and this memo has strived to preserve as much of that spirit as possible.

The author wishes to thanks the OASIS XRI TC and WebFinger communities for their support, encouragement, and enthusiasm for this work. Special thanks go to Phil Archer, Lisa Dusseault, Joseph Holsten, Mark Nottingham, John Panzer, Drummond Reed, and Jonathan Rees for their invaluable feedback.

Appendix B. Document History

[[ to be removed by the RFC editor before publication as an RFC ]]
o Removed analysis appendix and discussion of discovery types and removed informative references.

o Expanded the descriptor definition to include links as well as LRDD documents, moving away from the single-document approach.

o Moved the Link-Pattern field and template syntax to new host-meta draft.

o Updated references.

-03

o Added protocol name LRDD (pronounced ‘lard’).

o Fixed Link-Pattern examples to include missing semicolons.

-02

o Changed focus from an HTTP-based process to Link-based process.

o Completely revised and restructured document for better clarity.

o Realigned the methods to produce consistent results and changed the way redirections and client-errors are handled.

o Updated to use newer version of site-meta, now called host-meta, including a new plaintext-based format to replace the previous XML format.

o Renamed Link-Template to Link-Pattern to avoid future conflict with a previously proposed Link-Template HTTP header.

o Removed support for the "scheme" Link-Template parameter.

o Replaced restrictions with interoperability recommendations.

o Added IANA considerations per new host-meta registry requirements.

-01

o Rename ‘resource discovery’ to ‘descriptor discovery’.

o Added informative reference to Metalink.
- Clarified that the resource descriptor URI can use any URI scheme, not just "http" or "https".

- Removed comment regarding redirects when using <Link> Elements.

- Clarified that HTTPS must be used with "https" URIs for both Link headers and host-meta retrieval.

- Removed DNS verification step for host-meta with schemes other than "http" and "https". Replaced with a general discussion of authority and a security consideration comment.

- Organized host-meta section into another sub-section level.

- Enlarged the template vocabulary from a single "uri" variable to include smaller URI components.

- Added informative reference to RFC 2295 in analysis appendix.

---00

- Initial draft.

6. References

6.1. Normative References

[I-D.hammer-hostmeta]
Hammer-Lahav, E., "host-meta: Web Host Metadata",
draft-hammer-hostmeta-05 (work in progress),
November 2009.

[I-D.nottingham-http-link-header]
Nottingham, M., "Web Linking",
draft-nottingham-http-link-header-08 (work in progress),
March 2010.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H.,
Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1",
RFC 2616, June 1999.

6.2. Informative References

[OASIS.XRD-1.0]


URIs


Author’s Address

Eran Hammer-Lahav
Yahoo!

Email: eran@hueniverse.com
URI: http://hueniverse.com