WEBDAV Working Group
Internet-Draft
Expires: March 31, 2004

J. Slein
Xerox
J. Whitehead
U.C. Santa Cruz
J. Davis
CourseNet
G. Clemm
Rational
C. Fay
FileNet
J. Crawford
IBM
J. Reschke, Ed.
greenbytes

October 1, 2003

WebDAV Redirect Reference Resources
draft-ietf-webdav-redirectref-protocol-05

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

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 March 31, 2004.

Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

Abstract

This specification defines redirect reference resources. A redirect

reference resource is a resource whose default response is an HTTP/1.1 302 (Found) status code, redirecting the client to a different resource, the target resource. A redirect reference makes it possible to access the target resource indirectly, through any URI mapped to the redirect reference resource. There are no integrity guarantees associated with redirect reference resources.

Distribution of this document is unlimited. Please send comments to the Distributed Authoring and Versioning (WebDAV) working group at w3c-dist-auth@w3.org [1], which may be joined by sending a message with subject "subscribe" to w3c-dist-auth-request@w3.org [2].

Discussions of the WEBDAV working group are archived at URL: http://lists.w3.org/Archives/Public/w3c-dist-auth/.

Table of Contents

1.   Introduction 5
2.   Notational Conventions 7
3.   Terminology 8
4.   Overview of Redirect Reference Resources 9
5.   Creating a Redirect Reference Resource 10
5.1 MKRESOURCE 10
5.2 Example: Creating a Redirect Reference Resource with
     MKRESOURCE 11
6.   Operations on Redirect Reference Resources 13
6.1 Example: GET on a Redirect Reference Resource 13
6.2 Example: PROPPATCH on a Redirect Reference Resource 14
7.   Operations on Collections That Contain Redirect Reference
     Resources 16
7.1 MOVE and DELETE on Collections That Contain Redirect
     References 16
7.2 LOCK on a Collection That Contains Redirect References . . . 17
7.3 Example: PROPFIND on a Collection with Redirect Reference
     Resources 17
7.4 Example: PROPFIND with Apply-To-Redirect-Ref on a
     Collection with Redirect Reference Resources 18
7.5 Example: COPY on a Collection That Contains a Redirect
     Reference Resource 20
7.6 Example: LOCK on a Collection That Contains a Redirect
     Reference Resource 21
8.   Operations on Targets of Redirect Reference Resources . . . 24
9.   Relative URIs in DAV:reftarget 25
9.1 Example: Resolving a Relative URI in a MKRESOURCE Request . 25
9.2 Example: Resolving a Relative URI in a Multi-Status
     Response 26
10. Redirect References to Collections 28
11. Headers 30

```
   11.1 Redirect-Ref Response Header . . . . . . . . . . . . . . . .  30
   11.2 Apply-To-Redirect-Ref Request Header . . . . . . . . . . . .  30
   12.  Properties . . . . . . . . . . . . . . . . . . . . . . . . .  31
   12.1 reftarget Property . . . . . . . . . . . . . . . . . . . . .  31
   12.2 location Pseudo-Property . . . . . . . . . . . . . . . . . .  31
   13.  XML Elements . . . . . . . . . . . . . . . . . . . . . . . .  32
   13.1 redirectref XML Element  . . . . . . . . . . . . . . . . . .  32
   14.  Extensions to the DAV:response XML Element for
        Multi-Status Responses . . . . . . . . . . . . . . . . . . .  33
   15.  Capability Discovery . . . . . . . . . . . . . . . . . . . .  34
   15.1 Example: Discovery of Support for Redirect Reference
        Resources  . . . . . . . . . . . . . . . . . . . . . . . . .  34
   16.  Security Considerations  . . . . . . . . . . . . . . . . . .  35
   16.1 Privacy Concerns . . . . . . . . . . . . . . . . . . . . . .  35
   16.2 Redirect Loops . . . . . . . . . . . . . . . . . . . . . . .  35
   16.3 Redirect Reference Resources and Denial of Service . . . . .  35
   16.4 Revealing Private Locations  . . . . . . . . . . . . . . . .  35
   17.  Internationalization Considerations  . . . . . . . . . . . .  37
   18.  IANA Considerations  . . . . . . . . . . . . . . . . . . . .  38
   19.  Acknowledgements . . . . . . . . . . . . . . . . . . . . . .  39
        Normative References . . . . . . . . . . . . . . . . . . . .  40
        Authors' Addresses . . . . . . . . . . . . . . . . . . . . .  40
   A.   Changes to the WebDAV Document Type Definition . . . . . . .  42
   B.   Change Log (to be removed by RFC Editor before
        publication) . . . . . . . . . . . . . . . . . . . . . . . .  43
   B.1  Since draft-ietf-webdav-redirectref-protocol-02  . . . . . .  43
   B.2  Since draft-ietf-webdav-redirectref-protocol-03  . . . . . .  43
   B.3  Since draft-ietf-webdav-redirectref-protocol-04  . . . . . .  43
   C.   Resolved issues (to be removed by RFC Editor before
        publication) . . . . . . . . . . . . . . . . . . . . . . . .  44
   C.1  lc-56-notjusthttp  . . . . . . . . . . . . . . . . . . . . .  44
   C.2  lc-43-webdav . . . . . . . . . . . . . . . . . . . . . . . .  44
   C.3  lc-04-standard-data-container  . . . . . . . . . . . . . . .  44
   C.4  lc-05-standard-data-container  . . . . . . . . . . . . . . .  44
   C.5  lc-20-intro-mkresource . . . . . . . . . . . . . . . . . . .  45
   C.6  lc-22-coll . . . . . . . . . . . . . . . . . . . . . . . . .  45
   C.7  lc-25-atomic . . . . . . . . . . . . . . . . . . . . . . . .  45
   C.8  lc-42-no-webdav  . . . . . . . . . . . . . . . . . . . . . .  45
   C.9  lc-23-body . . . . . . . . . . . . . . . . . . . . . . . . .  46
   C.10 lc-47-207  . . . . . . . . . . . . . . . . . . . . . . . . .  46
   C.11 lc-49-put  . . . . . . . . . . . . . . . . . . . . . . . . .  47
   C.12 lc-75-ignore . . . . . . . . . . . . . . . . . . . . . . . .  47
   D.   Open issues (to be removed by RFC Editor before
        publication) . . . . . . . . . . . . . . . . . . . . . . . .  48
   D.1  lc-85-301  . . . . . . . . . . . . . . . . . . . . . . . . .  48
   D.2  lc-38-not-hierarchical . . . . . . . . . . . . . . . . . . .  48
   D.3  lc-36-server . . . . . . . . . . . . . . . . . . . . . . . .  48
   D.4  lc-33-forwarding . . . . . . . . . . . . . . . . . . . . . .  48
```

D.5 lc-37-integrity 49
D.6 3-terminology-redirectref 49
D.7 lc-19-direct-ref 49
D.8 lc-41-no-webdav 49
D.9 lc-58-update 50
D.10 lc-24-properties 50
D.11 lc-48-s6 50
D.12 lc-28-lang 51
D.13 lc-29-lang 51
D.14 lc-44-pseudo 51
D.15 lc-61-pseudo 51
D.16 lc-60-ex 52
D.17 lc-62-oldclient 52
D.18 lc-63-move 52
D.19 lc-06-reftarget-relative 53
D.20 lc-57-noautoupdate 53
D.21 lc-71-relative 53
D.22 lc-53-s10 53
D.23 lc-72-trailingslash 54
D.24 lc-50-blindredirect 54
D.25 lc-74-terminology 55
D.26 lc-76-location 55
D.27 lc-79-accesscontrol 55
D.28 lc-80-i18n 55
D.29 lc-55-iana 55
     Intellectual Property and Copyright Statements 57

1. Introduction

This is one of a pair of specifications that extend the WebDAV Distributed Authoring Protocol to enable clients to create new access paths to existing resources. This capability is useful for several reasons:

URIs of WebDAV-compliant resources are hierarchical and correspond to a hierarchy of collections in resource space. The WebDAV Distributed Authoring Protocol makes it possible to organize these resources into hierarchies, placing them into groupings, known as collections, which are more easily browsed and manipulated than a single flat collection. However, hierarchies require categorization decisions that locate resources at a single location in the hierarchy, a drawback when a resource has multiple valid categories. For example, in a hierarchy of vehicle descriptions containing collections for cars and boats, a description of a combination car/boat vehicle could belong in either collection. Ideally, the description should be accessible from both. Allowing clients to create new URIs that access the existing resource lets them put that resource into multiple collections.

Hierarchies also make resource sharing more difficult, since resources that have utility across many collections are still forced into a single collection. For example, the mathematics department at one university might create a collection of information on fractals that contains bindings to some local resources, but also provides access to some resources at other universities. For many reasons, it may be undesirable to make physical copies of the shared resources on the local server: to conserve disk space, to respect copyright constraints, or to make any changes in the shared resources visible automatically. Being able to create new access paths to existing resources in other collections or even on other servers is useful for this sort of case.

The redirect reference resources defined here provide a mechanism for creating alternative access paths to existing resources. A redirect reference resource is a resource in one collection whose purpose is to forward requests to another resource (its target), possibly in a different collection. In this way, it allows clients to submit requests to the target resource from another collection. It redirects most requests to the target resource using the HTTP 302 (Found) status code, thereby providing a form of mediated access to the target resource.

A redirect reference is a resource with properties but no body of its own. Properties of a redirect reference resource can contain such information as who created the reference, when, and why. Since

redirect reference resources are implemented using HTTP 302 responses, it generally takes two round trips to submit a request to the intended resource. Servers are not required to enforce the integrity of redirect references. Redirect references work equally well for local resources and for resources that reside on a different server from the reference.

The remainder of this document is structured as follows: Section 3 defines terms that will be used throughout the specification. Section 4 provides an overview of redirect reference resources. Section 5 discusses how to create a redirect reference resource. Section 6 defines the semantics of existing methods when applied to redirect reference resources, and Section 7 discusses their semantics when applied to collections that contain redirect reference resources. Sections 8 through 10 discuss several other issues raised by the existence of redirect reference resources. Sections 11 through 14 define the new headers, properties, and XML elements required to support redirect reference resources. Section 15 discusses capability discovery. Sections 16 through 18 present the security, internationalization, and IANA concerns raised by this specification. The remaining sections provide a variety of supporting information.

2. Notational Conventions

Since this document describes a set of extensions to the WebDAV Distributed Authoring Protocol [RFC2518], itself an extension to the HTTP/1.1 protocol, the augmented BNF used here to describe protocol elements is exactly the same as described in Section 2.1 of [RFC2616]. Since this augmented BNF uses the basic production rules provided in Section 2.2 of [RFC2616], these rules apply to this document as well.

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].

3. Terminology

The terminology used here follows and extends that in the WebDAV Distributed Authoring Protocol specification [RFC2518]. Definitions of the terms resource, Uniform Resource Identifier (URI), and Uniform Resource Locator (URL) are provided in [RFC2396].

Redirect Reference Resource

A resource created to redirect all requests made to it, using 302 (Found), to a defined target resource.

Non-Reference Resource

A resource that is not a reference to another resource.

Target Resource

The resource to which requests are forwarded by a reference resource. A target resource can be anything that can be identified by an absolute URI (see [RFC2396], "absoluteURI").

4. Overview of Redirect Reference Resources

For all operations submitted to a redirect reference resource, the default response is a 302 (Found), accompanied by the Redirect-Ref header (defined in Section 11.1 below) and the Location header set to the URI of the target resource. With this information, the client can resubmit the request to the URI of the target resource.

A redirect reference resource never automatically forwards requests to its target resource. Redirect resources bring the same benefits as links in HTML documents. They can be created and maintained without the involvement or even knowledge of their target resource. This reduces the cost of linking between resources."

If the client is aware that it is operating on a redirect reference resource, it can resolve the reference by retrieving the reference resource's DAV:reftarget property (defined in Section 12.1 below), whose value contains the URI of the target resource. It can then submit requests to the target resource.

A redirect reference resource is a new type of resource. To distinguish redirect reference resources from non-reference resources, a new value of the DAV:resourcetype property (defined in [RFC2518]), DAV:redirectref, is defined in Section 13.1 below.

Since a redirect reference resource is a resource, methods can be applied to the reference resource as well as to its target resource. The Apply-To-Redirect-Ref request header (defined in Section 11.2 below) is provided so that referencing-aware clients can control whether an operation is applied to the redirect reference resource or to its target resource. The Apply-To-Redirect-Ref header can be used with most requests to redirect reference resources. This header is particularly useful with PROPFIND, to retrieve the reference resource's own properties.

5. Creating a Redirect Reference Resource

The new MKRESOURCE method is used to create new redirect reference resources. In order to create a redirect reference resource using MKRESOURCE, the values of two properties must be set in the body of the MKRESOURCE request. The value of DAV:resourcetype MUST be set to DAV:redirectref, a new value of DAV:resourcetype defined in Section 13.1. The value of DAV:reftarget MUST be set to the URI of the target resource.

Used in this way, the MKRESOURCE method creates a redirect reference resource whose target is identified by the DAV:reftarget property.

5.1 MKRESOURCE

The MKRESOURCE method requests the creation of a redirect reference resource and initialization of its properties in one atomic operation.

Preconditions:

A resource MUST NOT exist at the Request-URI.

Request Marshalling:

The location of the new resource to be created is specified by the Request-URI.

The request body of the MKRESOURCE method MUST consist of the DAV:propertyupdate XML element defined in Section 12.13 of [RFC2518], specifying a DAV:resourcetype of "DAV:redirectref".

Postconditions:

If the response status code is 201, a new resource exists at the Request-URI.

The properties of the new resource are as specified by the DAV:propertyupdate request body, using PROPPATCH semantics.

If the response status code is not 201, then a new resource is not created at the Request-URI, and any existing resource at the Request-URI is unaffected.

Response Marshalling:

Responses from a MKRESOURCE request MUST NOT be cached, as MKRESOURCE has non-idempotent semantics.

The following status codes can be expected in responses to MKRESOURCE:

201 (Created): The new resource was successfully created.

403 (Forbidden): The server does not allow the creation of the requested resource type at the requested location, or the parent collection of the Request-URI exists but cannot accept members.

409 (Conflict): A resource cannot be created at the Request-URI because the parent collection for the resource does not exist, or because there is already a resource at that request-URL.

423 (Locked): The Request-URI is locked, and the lock token was not passed with the request.

507 (Insufficient Storage): The server does not have sufficient space to record the state of the resource.

5.2 Example: Creating a Redirect Reference Resource with MKRESOURCE

```
>> Request:

MKRESOURCE /~whitehead/dav/spec08.ref HTTP/1.1
Host: www.ics.uci.edu
Content-Type: text/xml; charset="utf-8"
Content-Length: xxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propertyupdate xmlns:D="DAV:">
  <D:set>
    <D:prop>
      <D:resourcetype><D:redirectref/></D:resourcetype>
      <D:reftarget>
        <D:href>/i-d/draft-webdav-protocol-08.txt</D:href>
      </D:reftarget>
    </D:prop>
  </D:set>
</D:propertyupdate>

 >> Response:

HTTP/1.1 201 Created
```

This request resulted in the creation of a new redirect reference resource at www.ics.uci.edu/~whitehead/dav/spec08.ref, which points to the resource identified by the DAV:reftarget property. In this

example, the target resource is identified by the URI http://
www.ics.uci.edu/i-d/draft-webdav-protocol-08.txt. The redirect
reference resource's DAV:resourcetype property is set to
DAV:redirectref.

6. Operations on Redirect Reference Resources

Although non-referencing-aware clients cannot create reference resources, they should be able to submit requests through the reference resources created by reference-aware WebDAV clients. They should be able to follow any references to their targets. To make this possible, a server that receives any request made via a redirect reference resource MUST return a 302 (Found) status code, unless the request includes an Apply-To-Redirect-Ref header. The client and server MUST follow [RFC2616] Section 10.3.3 "302 Found," but with these additional rules:

o The Location response header MUST contain an absolute URI that identifies the target of the reference resource.

o The response MUST include the Redirect-Ref header. This header allows reference-aware WebDAV clients to recognize the resource as a reference resource and understand the reason for the redirection.

A reference-aware WebDAV client can act on this response in one of two ways. It can, like a non-referencing client, resubmit the request to the URI in the Location header in order to operate on the target resource. Alternatively, it can resubmit the request to the URI of the redirect reference resource with the Apply-To-Redirect-Ref header in order to operate on the reference resource itself. If the Apply-To-Redirect-Ref header is present, the request MUST be applied to the reference resource itself, and a 302 response MUST NOT be returned.

A reference-aware client may know before submitting its request that the Request-URI identifies a redirect reference resource. In this case, if the client wants to apply the method to the reference resource, it can save the round trip caused by the 302 response by using an Apply-To-Redirect-Ref header in its initial request to the URI.

As redirect references do not have bodies, GET and PUT requests with Apply-To-Redirect-Ref MUST fail with status 403 (forbidden).

6.1 Example: GET on a Redirect Reference Resource

```
 >> Request:

GET /bar.html HTTP/1.1
Host: www.foo.com
```

```
 >> Response:

HTTP/1.1 302 Found
Location: http://www.svr.com/Internet/xxspec08.html
Redirect-Ref:
```

Since /bar.html is a redirect reference resource and the Apply-To-Redirect-Ref header is not included in the request, the response is a 302 (Found). The Redirect-Ref header informs a reference-aware client that this is not an ordinary HTTP 1.1 redirect, but is a redirect reference resource. The URI of the target resource is provided in the Location header so that the client can resubmit the request to the target resource.

6.2 Example: PROPPATCH on a Redirect Reference Resource

```
 >> Request:

PROPPATCH /bar.html HTTP/1.1
Host: www.foo.com
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propertyupdate xmlns:D="DAV:"
xmlns:Z="http://www.w3.com/standards/z39.50/">
  <D:set>
    <D:prop>
      <Z:authors>
        <Z:Author>Jim Whitehead</Z:Author>
        <Z:Author>Roy Fielding</Z:Author>
      </Z:authors>
    </D:prop>
  </D:set>
  <D:remove>
    <D:prop>
      <Z:Copyright-Owner/>
    </D:prop>
  </D:remove>
</D:propertyupdate>

 >> Response:

HTTP/1.1 302 Found
Location: http://www.svr.com/Internet/xxspec08.html
Redirect-Ref:
```

Since /bar.html is a redirect reference resource and the

Apply-To-Redirect-Ref header is not included in the request, the response is a 302 (Found). The Redirect-Ref header informs a reference-aware client that this is not an ordinary HTTP 1.1 redirect, but is a redirect reference resource. The URI of the target resource is provided in the Location header so that the client can resubmit the request to the target resource.

7. Operations on Collections That Contain Redirect Reference Resources

Consistent with the rules in Section 6, the response for each redirect reference encountered while processing a collection MUST be a 302 (Found) unless a Apply-To-Redirect-Ref header is included with the request. The overall response will therefore be a 207 (Multi-Status). Since a Location header and Redirect-Ref header cannot be returned for each redirect reference encountered, the same information is provided using properties in the response elements for those resources. The DAV:location pseudo-property and the DAV:resourcetype property MUST be included with the 302 status code. This necessitates an extension to the syntax of the DAV:response element that was defined in [RFC2518]. The extension is defined in Section 14 below.

A referencing-aware client can tell from the DAV:resourcetype property that the collection contains a redirect reference resource. The DAV:location pseudo-property contains the absolute URI of the target resource. A referencing-aware client can either use the URI value of the DAV:location pseudo-property to resubmit its request to the target resource, or it can submit the request to the redirect reference resource with Apply-To-Redirect-Ref.

It is recommended that future editors of [RFC2518] define the DAV:location pseudo-property in [RFC2518], so that non-referencing clients will also be able to use the response to operate on the target resource. (This will also enable clients to operate on traditional HTTP/1.1 302 responses in Multi-Status responses.) Until then, non-referencing clients will not be able to process 302 responses from redirect reference resources encountered while processing a collection.

The Apply-To-Redirect-Ref header (defined in Section 11.2) MAY be used with any request on a collection. If present, it will be applied to all redirect reference resources encountered while processing the collection.

7.1 MOVE and DELETE on Collections That Contain Redirect References

DELETE removes the binding that corresponds to the Request-URI. MOVE removes that binding and creates a new binding to the same resource. In cases where DELETE and MOVE are applied to a collection, these operations affect all the descendents of the collection, but they do so indirectly. There is no need to visit each descendent in order to process the request. Consequently, even if there are redirect reference resources in a tree that is being deleted or moved, there will be no 302 responses from the redirect reference resources.

7.2 LOCK on a Collection That Contains Redirect References

LOCK poses special problems because it is atomic. An attempt to lock (with Depth: infinity) a collection that contains redirect references will always fail. The Multi-Status response will contain a 302 response for each redirect reference.

Reference-aware clients can lock the collection by using Apply-To-Redirect-Ref, and, if desired, lock the targets of the redirect references individually.

Non-referencing clients must resort to locking each resource individually.

7.3 Example: PROPFIND on a Collection with Redirect Reference Resources

Suppose a PROPFIND request with Depth: infinity is submitted to the following collection, with the members shown here:

```
http://www.svr.com/MyCollection/
     (non-reference resource) diary.html
     (redirect reference resource) nunavut
```

```
 >> Request:

PROPFIND /MyCollection/ HTTP/1.1
Host: www.svr.com
Depth: infinity
Content-Type: text/xml
Content-Length: xxxx

<?xml version="1.0" ?>
<D:propfind xmlns:D="DAV: ">
  <D:prop xmlns:J="http://www.svr.com/jsprops/">
    <D:resourcetype/>
    <J:keywords/>
  </D:prop>
</D:propfind>

 >> Response:

HTTP/1.1 207 Multi-Status
Content-Type: text/xml
Content-Length: xxxx

<?xml version="1.0" ?>
<D:multistatus xmlns:D="DAV:" xmlns:J="http://www.svr.com/jsprops/">
  <D:response>
```

```
       <D:href>http://www.svr.com/MyCollection/</D:href>
       <D:propstat>
         <D:prop>
           <D:resourcetype><D:collection/></D:resourcetype>
           <J:keywords>diary, interests, hobbies</J:keywords>
         </D:prop>
         <D:status>HTTP/1.1 200 OK</D:status>
       </D:propstat>
     </D:response>
     <D:response>
       <D:href>http://www.svr.com/MyCollection/diary.html</D:href>
       <D:propstat>
         <D:prop>
           <D:resourcetype/>
           <J:keywords>diary, travel, family, history</J:keywords>
         </D:prop>
         <D:status>HTTP/1.1 200 OK</D:status>
       </D:propstat>
     </D:response>
     <D:response>
       <D:href>http://www.svr.com/MyCollection/nunavut</D:href>
       <D:status>HTTP/1.1 302 Found</D:status>
       <D:prop>
         <D:location>
           <D:href>http://www.inac.gc.ca/art/inuit/</D:href>
         </D:location>
         <D:resourcetype><D:redirectref/></D:resourcetype>
       </D:prop>
     </D:response>
   </D:multistatus>
```

In this example the Depth header is set to infinity, and the Apply-To-Redirect-Ref header is not used. The collection contains one URI that identifies a redirect reference resource. The response element for the redirect reference resource has a status of 302 (Found), and includes a DAV:prop element with the DAV:location pseudo-property and the DAV:resourcetype property to allow clients to retrieve the properties of its target resource. (The response element for the redirect reference resource does not include the requested properties. The client can submit another PROPFIND request to the URI in the DAV:location pseudo-property to retrieve those properties.)

7.4 Example: PROPFIND with Apply-To-Redirect-Ref on a Collection with Redirect Reference Resources

Suppose a PROPFIND request with Apply-To-Redirect-Ref and Depth: infinity is submitted to the following collection, with the members

shown here:

```
/MyCollection/
     (non-reference resource) diary.html
     (redirect reference resource) nunavut
```

 >> Request:

```
PROPFIND /MyCollection/ HTTP/1.1
Host: www.svr.com
Depth: infinity
Apply-To-Redirect-Ref:
Content-Type: text/xml
Content-Length: xxxx

<?xml version="1.0" ?>
<D:propfind xmlns:D="DAV:">
  <D:prop>
    <D:resourcetype/>
    <D:reftarget/>
  </D:prop>
</D:propfind>
```

 >> Response:

```
HTTP/1.1 207 Multi-Status
Content-Type: text/xml
Content-Length: xxxx

<?xml version="1.0" ?>
<D:multistatus xmlns:D="DAV:">
  <D:response>
    <D:href>http://www.svr.com/MyCollection/</D:href>
    <D:propstat>
      <D:prop>
        <D:resourcetype><D:collection/></D:resourcetype>
      </D:prop>
      <D:status>HTTP/1.1 200 OK</D:status>
    </D:propstat>
    <D:propstat>
      <D:prop><D:reftarget/></D:prop>
      <D:status>HTTP/1.1 404 Not Found</D:status>
    </D:propstat>
  </D:response>
  <D:response>
    <D:href>http://www.svr.com/MyCollection/diary.html</D:href>
    <D:propstat>
      <D:prop>
```

```
          <D:resourcetype/>
        </D:prop>
        <D:status>HTTP/1.1 200 OK</D:status>
      </D:propstat>
      <D:propstat>
        <D:prop><D:reftarget/></D:prop>
        <D:status>HTTP/1.1 404 Not Found</D:status>
      </D:propstat>
    </D:response>
    <D:response>
      <D:href>http://www.svr.com/MyCollection/nunavut</D:href>
      <D:propstat>
        <D:prop>
          <D:resourcetype><D:redirectref/></D:resourcetype>
          <D:reftarget>
            <D:href>http://www.inac.gc.ca/art/inuit/</D:href>
          </D:reftarget>
        </D:prop>
      <D:status>HTTP/1.1 200 OK</D:status>
      </D:propstat>
    </D:response>
  </D:multistatus>
```

Since the Apply-To-Redirect-Ref header is present, the response shows the properties of the redirect reference resource in the collection rather than reporting a 302 status.

7.5 Example: COPY on a Collection That Contains a Redirect Reference Resource

Suppose a COPY request is submitted to the following collection, with the members shown:

```
/MyCollection/
     (non-reference resource) diary.html
     (redirect reference resource) nunavut with target
                                /Someplace/nunavut.map
```

```
 >> Request:

COPY /MyCollection/ HTTP/1.1
Host: www.svr.com
Depth: infinity
Destination: http://www.svr.com/OtherCollection/

 >> Response:

HTTP/1.1 207 Multi-Status
```

```
   Content-Type: text/xml; charset="utf-8"
   Content-Length: xxx

   <?xml version="1.0" encoding="utf-8" ?>
   <D:multistatus xmlns:D="DAV:">
     <D:response>
     <D:href>http://www.svr.com/MyCollection/nunavut</D:href>
     <D:status>HTTP/1.1 302 Found</D:status>
     <D:prop>
       <D:location>
         <D:href>http://www.svr.com//Someplace/nunavut.map</D:href>
       </D:location>
       <D:resourcetype><D:redirectref/></D:resourcetype>
     </D:prop>
     </D:response>
   </D:multistatus>
```

In this case, since /MyCollection/nunavut is a redirect reference resource, the COPY operation was only a partial success. The redirect reference resource was not copied, but a 302 response was returned for it. So the resulting collection is as follows:

```
   /OtherCollection/
         (non-reference resource) diary.html
```

7.6 Example: LOCK on a Collection That Contains a Redirect Reference Resource

Suppose a LOCK request is submitted to the following collection, with the members shown:

```
   /MyCollection/
        (non-reference resource) diary.html
        (redirect reference resource) nunavut
```

>> Request:

```
   LOCK /MyCollection/ HTTP/1.1
   Host: www.svr.com
   Content-Type: text/xml
   Content-Length: nnnn
   Authorizaton: Digest username="jas",
      realm=jas@webdav.sb.aol.com, nonce=". . . ",
      uri="/MyCollection/tuva",
      response=". . . ", opaque=". . . "

   <?xml version="1.0" ?>
```

```
   <D:lockinfo xmlns:D="DAV:">
     <D:lockscope><D:exclusive/></D:lockscope>
     <D:locktype><D:write/></D:locktype>
     <D:owner>
       <D:href>http://www.svr.com/~jas/contact.html</D:href>
     </D:owner>
   </D:lockinfo>

    >> Response:

   HTTP/1.1 207 Multi-Status
   Content-Type: text/xml
   Content-Length: nnnn

   <?xml version="1.0" ?>
   <D:multistatus xmlns:D="Dav:">
     <D:response>
       <D:href>http://www.svr.com/MyCollection/</D:href>
       <D:propstat>
         <D:prop><D:lockdiscovery/></D:prop>
         <D:status>HTTP/1.1 424 Failed Dependency</D:status>
       </D:propstat>
     </D:response>
     <D:response>
       <D:href>http://www.svr.com/MyCollection/diary.html</D:href>
       <D:status>HTTP/1.1 424 Failed Dependency</D:status>
     </D:response>
     <D:response>
       <D:href>http://www.svr.com/MyCollection/nunavut</D:href>
       <D:status>HTTP/1.1 302 Found</D:status>
       <D:prop>
         <D:location>
           <D:href>http://www.inac.gc.ca/art/inuit/</D:href>
         </D:location>
         <D:resourcetype><D:redirectref/></D:resourcetype>
       </D:prop>
     </D:response>
   </D:multistatus>
```

The server returns a 302 response code for the redirect reference resource in the collection. Consequently, neither the collection nor any of the resources identified by its internal member URIs were locked. A referencing-aware client can submit a separate LOCK request to the URI in the DAV:location pseudo-property returned for the redirect reference resource, and can resubmit the LOCK request with the Apply-To-Redirect-Ref header to the collection. At that point both the reference resource and its target resource will be locked (as well as the collection and all the resources identified by its

other members).

8. Operations on Targets of Redirect Reference Resources

Operations on targets of redirect reference resources have no effect on the reference resource.

9. Relative URIs in DAV:reftarget

The URI in the href in a DAV:reftarget property MAY be a relative URI. In this case, the base URI to be used for resolving the relative URI to absolute form is the URI used in the HTTP message to identify the redirect reference resource to which the DAV:reftarget property belongs.

When DAV:reftarget occurs in the body of a MKRESOURCE request, the base URI is constructed as follows: Its scheme component is "http", its authority component is the value of the Host header in the request, and its path component is the Request-URI in the request. See Section 5 of [RFC2396] for a discussion of relative URI references and how to resolve them.

When DAV:reftarget appears in the context of a Multi-Status response, it is in a DAV:response element that contains a single DAV:href element. The value of this DAV:href element serves as the base URI for resolving a relative URI in DAV:reftarget. The value of DAV:href may itself be relative, in which case it must be resolved first in order to serve as the base URI for the relative URI in DAV:reftarget. If the DAV:href element is relative, its base URI is constructed from the scheme component "http", the value of the Host header in the request, and the request-URI.

9.1 Example: Resolving a Relative URI in a MKRESOURCE Request

```
 >> Request:

MKRESOURCE /north/inuvik HTTP/1.1
Host: www.somehost.edu
Content-Type: text/xml; charset="utf-8"
Content-Length: xxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propertyupdate xmlns:D="DAV:">
  <D:set>
    <D:prop>
      <D:resourcetype><D:redirectref/></D:resourcetype>
      <D:reftarget>
        <D:href>mapcollection/inuvik.gif</D:href>
      </D:reftarget>
    </D:prop>
  </D:set>
</D:propertyupdate>

 >> Response:
```

```
   HTTP/1.1 201 Created
```

In this example, the base URI is http://www.somehost.edu/north/inuvik. Then, following the rules in [RFC2396] Section 5, the relative URI in DAV:reftarget resolves to the absolute URI http://www.somehost.edu/north/mapcollection/inuvik.gif.

9.2 Example: Resolving a Relative URI in a Multi-Status Response

>> Request:

```
   PROPFIND /geog/ HTTP/1.1
   Host: www.xxsvr.com
   Apply-To-Redirect-Ref:
   Depth: 1
   Content-Type: text/xml
   Content-Length: nnn

   <?xml version="1.0" ?>
   <D:propfind xmlns:D="DAV:">
     <D:prop>
       <D:resourcetype/>
       <D:reftarget/>
     </D:prop>
   </D:propfind>
```

>> Response:

```
   HTTP/1.1 207 Multi-Status
   Content-Type: text/xml
   Content-Length: nnn

   <?xml version="1/0" ?>
   <D:multistatus xmlns:D="DAV:">
     <D:response>
       <D:href>/geog/</D:href>
       <D:propstat>
         <D:prop>
           <D:resourcetype><D:collection/></D:resourcetype>
         </D:prop>
         <D:status>HTTP/1.1 200 OK</D:status>
       </D:propstat>
       <D:propstat>
         <D:prop><D:reftarget/></D:prop>
         <D:status>HTTP/1.1 404 Not Found</D:status>
       </D:propstat>
     </D:response>
     <D:response>
```

```
    <D:href>/geog/stats.html</D:href>
    <D:propstat>
      <D:prop>
        <D:resourcetype><D:redirectref/></D:resourcetype>
        <D:reftarget>
          <D:href>statistics/population/1997.html</D:href>
        </D:reftarget>
      </D:prop>
    <D:status>HTTP/1.1 200 OK</D:status>
    </D:propstat>
  </D:response>
</D:multistatus>
```

In this example, the relative URI statistics/population/1997.html is returned as the value of reftarget for the reference resource identified by href /geog/stats.html. The href is itself a relative URI, which resolves to http://www.xxsrv.com/geog/stats.html. This is the base URI for resolving the relative URI in reftarget. The absolute URI of reftarget is http://www.xxsrv.com/geog/statistics/population/1997.html.

10. Redirect References to Collections

In a Request-URI /segment1/segment2/segment3, any of the three segments may identify a redirect reference resource. (See [RFC2396], Section 3.3, for definitions of "path" and "segment".) If any segment in a Request- URI identifies a redirect reference resource, the response is a 302. The value of the Location header in the 302 response is as follows:

The leftmost path segment of the request-URI that identifies a redirect reference resource, together with all path segments and separators to the left of it, is replaced by the value of the redirect reference resource's DAV:reftarget property (resolved to an absolute URI). The remainder of the request-URI is concatenated to this path.

Note: If the DAV:reftarget property ends with a "/" and the remainder of the Request-URI is non-empty (and therefore must begin with a "/ "), the final "/" in the DAV:reftarget property is dropped before the remainder of the Request-URI is appended.

Consider Request-URI /x/y/z.html. Suppose that /x/ is a redirect reference resource whose target resource is collection /a/, which contains redirect reference resource y whose target resource is collection /b/, which contains redirect reference resource z.html whose target resource is /c/d.html.

```
/x/y/z.html
    |
    | /x -> /a
    |
    v
/a/y/z.html
    |
    | /a/y -> /b
    |
    v
/b/z.html
    |
    | /b/z.html -> /c/d.html
    |
    v
/c/d.html
```

In this case the client must follow up three separate 302 responses before finally reaching the target resource. The server responds to the initial request with a 302 with Location: /a/y/z.html, and the client resubmits the request to /a/y/z.html. The server responds to

this request with a 302 with Location: /b/z.html, and the client resubmits the request to /b/z.html. The server responds to this request with a 302 with Location: /c/d.html, and the client resubmits the request to /c/d.html. This final request succeeds.

11. Headers

11.1 Redirect-Ref Response Header

```
Redirect-Ref = "Redirect-Ref:"
```

The Redirect-Ref header is used in all 302 responses from redirect reference resources. Its presence informs reference-aware clients that the response is not a plain HTTP/1.1 redirect, but is a response from a redirect reference resource.

11.2 Apply-To-Redirect-Ref Request Header

```
Apply-To-Redirect-Ref = "Apply-To-Redirect-Ref" ":"
```

The optional Apply-To-Redirect-Ref header can be used on any request to a redirect reference resource. When it is used, the request MUST be applied to the reference resource itself, and a 302 response MUST NOT be returned.

If the Apply-To-Redirect-Ref header is used on a request to any other sort of resource besides a redirect reference resource, the server MUST ignore it.

12. Properties

12.1 reftarget Property

Name: reftarget

Namespace: DAV:

Purpose: A property of redirect reference resources that provides an efficient way for clients to discover the URI of the target resource. This is a read-only property after its initial creation. Its value can only be set in a MKRESOURCE request.

Value: href containing the URI of the target resource. This value MAY be a relative URI. The reftarget property can occur in the entity bodies of MKRESOURCE requests and of responses to PROPFIND requests.

```
<!ELEMENT reftarget href >
```

12.2 location Pseudo-Property

Name: location

Namespace: DAV:

Purpose: For use with 302 (Found) response codes in Multi-Status responses. It contains the absolute URI of the temporary location of the resource. In the context of redirect reference resources, this value is the absolute URI of the target resource. It is analogous to the Location header in HTTP 302 responses defined in [RFC2616] Section 10.3.3 "302 Found." Including the location pseudo-property in a Multi- Status response requires an extension to the syntax of the DAV:response element defined in [RFC2518], which is defined in Section 14 below. This pseudo-property is not expected to be stored on the reference resource. It is modeled as a property only so that it can be returned inside a DAV:prop element in a Multi-Status response.

Value: href containing the absolute URI of the target resource.

```
<!ELEMENT location href >
```

13. XML Elements

13.1 redirectref XML Element

Name: redirectref

Namespace: DAV:

Purpose: Used as the value of the DAV:resourcetype property to specify that the resource type is a redirect reference resource.

```
<!ELEMENT redirectref EMPTY >
```

14. Extensions to the DAV:response XML Element for Multi-Status Responses

As described in Section 7, the DAV:location pseudo-property and the DAV:resourcetype property may be returned in the DAV:response element of a 207 Multi-Status response, to allow clients to resubmit their requests to the target resource of a redirect reference resource.

Whenever these properties are included in a Multi-Status response, they are placed in a DAV:prop element associated with the href to which they apply. This structure provides a framework for future extensions by other standards that may need to include additional properties in their responses.

Consequently, the definition of the DAV:response XML element changes to the following:

```
<!ELEMENT response (href, ((href*, status, prop?) | (propstat+)),
responsedescription?) >
```

15. Capability Discovery

Sections 9.1 and 15 of [RFC2518] describe the use of compliance classes with the DAV header in responses to OPTIONS, to indicate which parts of the WebDAV Distributed Authoring protocols the resource supports. This specification defines an OPTIONAL extension to [RFC2518]. It defines a new compliance class, called redirectrefs, for use with the DAV header in responses to OPTIONS requests. If a resource does support redirect references, its response to an OPTIONS request may indicate that it does, by listing the new redirectrefs compliance class in the DAV headerand by listing the MKRESOURCE method as one it supports.

When responding to an OPTIONS request, any type of resource can include redirectrefs in the value of the DAV header. Doing so indicates that the server permits a redirect reference resource at the request URI.

15.1 Example: Discovery of Support for Redirect Reference Resources

```
 >> Request:

OPTIONS /somecollection/someresource HTTP/1.1
HOST: somehost.org

 >> Response:

HTTP/1.1 200 OK
Date: Tue, 20 Jan 1998 20:52:29 GMT
Connection: close
Accept-Ranges: none
Allow: OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, COPY, MOVE,
MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK, MKRESOURCE
DAV: 1, 2, redirectrefs
```

The DAV header in the response indicates that the resource / somecollection/someresource is level 1 and level 2 compliant, as defined in [RFC2518]. In addition, /somecollection/someresource supports redirect reference resources. The Allow header indicates that MKRESOURCE requests can be submitted to /somecollection/ someresource. The Public header shows that other Request-URIs on the server support additional methods.

16. Security Considerations

This section is provided to make applications that implement this protocol aware of the security implications of this protocol.

All of the security considerations of HTTP/1.1 and the WebDAV Distributed Authoring Protocol specification also apply to this protocol specification. In addition, redirect reference resources introduce several new security concerns and increase the risk of some existing threats. These issues are detailed below.

16.1 Privacy Concerns

By creating redirect reference resources on a trusted server, it is possible for a hostile agent to induce users to send private information to a target on a different server. This risk is mitigated somewhat, since clients are required to notify the user of the redirection for any request other than GET or HEAD. (See [RFC2616], Section 10.3.3 302 Found.)

16.2 Redirect Loops

Although redirect loops were already possible in HTTP 1.1, the introduction of the MKRESOURCE method creates a new avenue for clients to create loops accidentally or maliciously. If the reference resource and its target are on the same server, the server may be able to detect MKRESOURCE requests that would create loops. See also [RFC2616], Section 10.3 "Redirection 3xx."

16.3 Redirect Reference Resources and Denial of Service

Denial of service attacks were already possible by posting URLs that were intended for limited use at heavily used Web sites. The introduction of MKRESOURCE creates a new avenue for similar denial of service attacks. Clients can now create redirect reference resources at heavily used sites to target locations that were not designed for heavy usage.

16.4 Revealing Private Locations

There are several ways that redirect reference resources may reveal information about collection structures. First, the DAV:reftarget property of every redirect reference resource contains the URI of the target resource. Anyone who has access to the reference resource can discover the collection path that leads to the target resource. The owner of the target resource may have wanted to limit knowledge of this collection structure.

Sufficiently powerful access control mechanisms can control this risk to some extent. Property-level access control could prevent users from examining the DAV:reftarget property. (The Location header returned in responses to requests on redirect reference resources reveals the same information, however.) In some environments, the owner of a resource might be able to use access control to prevent others from creating references to that resource.

This risk is no greater than the similar risk posed by HTML links.

17. Internationalization Considerations

This specification follows the practices of [RFC2518] in encoding all human-readable content using XML [XML] and in the treatment of names. Consequently, this specification complies with the IETF Character Set Policy [RFC2277].

WebDAV applications MUST support the character set tagging, character set encoding, and the language tagging functionality of the XML specification. This constraint ensures that the human-readable content of this specification complies with [RFC2277].

As in [RFC2518], names in this specification fall into three categories: names of protocol elements such as methods and headers, names of XML elements, and names of properties. Naming of protocol elements follows the precedent of HTTP, using English names encoded in USASCII for methods and headers. The names of XML elements used in this specification are English names encoded in UTF-8.

For error reporting, [RFC2518] follows the convention of HTTP/1.1 status codes, including with each status code a short, English description of the code (e.g., 423 Locked). Internationalized applications will ignore this message, and display an appropriate message in the user's language and character set.

This specification introduces no new strings that are displayed to users as part of normal, error-free operation of the protocol.

For rationales for these decisions and advice for application implementors, see [RFC2518].

18. IANA Considerations

All IANA considerations mentioned in [RFC2518] also apply to this document.

19. Acknowledgements

This draft has benefited from thoughtful discussion by Jim Amsden, Peter Carlson, Steve Carter, Tyson Chihaya, Ken Coar, Ellis Cohen, Bruce Cragun, Spencer Dawkins, Mark Day, Rajiv Dulepet, David Durand, Roy Fielding, Yaron Goland, Fred Hitt, Alex Hopmann, James Hunt, Marcus Jager, Chris Kaler, Manoj Kasichainula, Rohit Khare, Daniel LaLiberte, Steve Martin, Larry Masinter, Jeff McAffer, Joe Orton, Surendra Koduru Reddy, Juergen Reuter, Max Rible, Sam Ruby, Bradley Sergeant, Nick Shelness, John Stracke, John Tigue, John Turner, Kevin Wiggen, and others.

Normative References

[RFC2277] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, January 1998.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2396] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[RFC2518] Goland, Y., Whitehead, E., Faizi, A., Carter, S. and D. Jensen, "HTTP Extensions for Distributed Authoring -- WEBDAV", RFC 2518, February 1999.

[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.

[XML]   Bray, T., Paoli, J., Sperberg-McQueen, C. and E. Maler, "Extensible Markup Language (XML) 1.0 (2nd ed)", W3C REC-xml, October 2000, <http://www.w3.org/TR/2000/REC-xml-20001006>.

[1] <mailto:w3c-dist-auth@w3.org>

[2] <mailto:w3c-dist-auth-request@w3.org?subject=subscribe>

[3] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0306.html>

[4] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0294.html>

[5] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0189.html>

[6] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0189.html>

[7] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0266.html>

[8] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0266.html>

[9] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/

0266.html>

[10] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0293.html>

[11] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0266.html>

[12] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0297.html>

[13] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0299.html>

[14] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[15] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0289.html>

[16] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0285.html>

[17] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0284.html>

[18] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0288.html>

[19] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0290.html>

[20] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0266.html>

[21] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0292.html>

[22] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0308.html>

[23] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0266.html>

[24] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0298.html>

[25] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/

0266.html>

[26] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0266.html>

[27] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0302.html>

[28] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[29] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[30] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[31] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[32] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0222.html>

[33] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0307.html>

[34] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[35] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0304.html>

[36] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[37] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0300.html>

[38] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0316.html>

[39] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0359.html>

[40] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0359.html>

[41] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/

0359.html>

[42] <http://lists.w3.org/Archives/Public/w3c-dist-auth/2000JanMar/0305.html>

Authors' Addresses

J. Slein
Xerox Corporation
800 Phillips Road, 105-50C
Webster, NY 14580

EMail: jslein@crt.xerox.com

Jim Whitehead
UC Santa Cruz, Dept. of Computer Science
1156 High Street
Santa Cruz, CA 95064
US

EMail: ejw@cse.ucsc.edu

J. Davis
CourseNet Systems
170 Capp Street
San Francisco, CA 94110

EMail: jrd3@alum.mit.edu

G. Clemm
Rational Software Corporation
20 Maguire Road
Lexington, MA 02173-3104

EMail: geoffrey.clemm@us.ibm.com

C. Fay
FileNet Corporation
3565 Harbor Boulevard
Costa Mesa, CA 92626-1420

EMail: cfay@filenet.com

J. Crawford
IBM Research
P.O. Box 704
Yorktown Heights, NY 10598

EMail: ccjason@us.ibm.com

Julian F. Reschke (editor)
greenbytes GmbH
Salzmannstrasse 152
Muenster, NW 48159
Germany

Phone: +49 251 2807760
Fax: +49 251 2807761
EMail: julian.reschke@greenbytes.de
URI: http://greenbytes.de/tech/webdav/

Appendix A. Changes to the WebDAV Document Type Definition

```
   <!-- XML Elements from Section 13 -->
   <!ELEMENT redirectref EMPTY >
   <!--  -->Property Elements from Section 12 -->
   <!ELEMENT reftarget href>
   <!ELEMENT location href>
   <!-- Changes to the DAV:response Element from Section 14 -->
   <!ELEMENT response (href, ((href*, status, prop?) | (propstat+)),
   responsedescription?) >
```

Appendix B. Change Log (to be removed by RFC Editor before publication)

B.1 Since draft-ietf-webdav-redirectref-protocol-02

Julian Reschke takes editorial role (added to authors list). Cleanup XML indentation. Start adding all unresolved last call issues. Update some author's contact information. Update references, split into "normative" and "informational". Remove non-RFC2616 headers ("Public") from examples. Fixed width problems in artwork. Start resolving editorial issues.

B.2 Since draft-ietf-webdav-redirectref-protocol-03

Added Joe Orton and Juergen Reuter to Acknowledgements section. Close more editorial issues. Remove dependencies on BIND spec.

B.3 Since draft-ietf-webdav-redirectref-protocol-04

More editorial fixes. Clarify that MKRESOURCE can only be used to create redirect references (switch to new method in a future draft). Clarify that redirect references do not have bodies.

Appendix C. Resolved issues (to be removed by RFC Editor before publication)

Issues that were either rejected or resolved in this version of this document.

C.1 lc-56-notjusthttp

Type: change

[3]

yarong@Exchange.Microsoft.com (2000-02-11): Make it clear in examples and text that the redirection URI could be non-HTTP.

Resolution: We agree that it is possible to create redirect references to non-HTTP resources. Add example. (actually it was added to the definition of "target resource").

C.2 lc-43-webdav

Type: change

[4]

yarong@Exchange.Microsoft.com (2000-02-11): Get rid of the DAV:reftarget property.

Resolution: DAV:reftarget is readonly and is present only for redirect references that are also WebDAV resources. We'll also have a method for setting target; Redirect-Ref header (returned on all 302 responses) will have the target as its value. See also issue 6, 17, 50.

C.3 lc-04-standard-data-container

Type: change

[5]

joe@orton.demon.co.uk (2000-01-26): "Standard data container" needs to be defined in the context of MKRESOURCE

Resolution: Not relevant once we switch to MKREF.

C.4 lc-05-standard-data-container

Type: change

[6]

joe@orton.demon.co.uk (2000-01-26): Inconsistency about whether a "standard data container" can be created with MKRESOURCE or not.

Resolution: Not relevant once we switch to MKREF.

C.5 lc-20-intro-mkresource

Type: change

[7]

reuterj@ira.uka.de (2000-02-07): Section 5: Start with "The new MKRESOURCE method" to make it clear that it is being introduced for the first time here.

Resolution: Say "The MKREF method defined normatively here . . ."

C.6 lc-22-coll

Type: change

[8]

reuterj@ira.uka.de (2000-02-07): Inconsistency about whether collections can be created with MKRESOURCE.

Resolution: (1) Strip all non-redirect-ref functionality from MKRESOURCE, then (2) later switch to a new method.

C.7 lc-25-atomic

Type: change

[9]

reuterj@ira.uka.de (2000-02-07): Is MKRESOURCE atomic as viewed by a client? Can another client access the new resource's properties before they have been fully initialized? Maybe the MKRESOURCE request should let the client ask for it to be atomic.

Resolution: No longer relevant once we switch to MKREF with no request body. Also, as an intermediate step MKRESOURCE is defined to be atomic.

C.8 lc-42-no-webdav

Type: change

[10]

yarong@Exchange.Microsoft.com (2000-02-11): Use a creation method that creates only redirect references. The MKRESOURCE method hinders experiment because a user of a server who wishes to add support for the creation of a new resource type can't simply throw in another Apache module and allow it to provide the code for the new resource type. They have to find the code used for MKRESOURCE and change it to support the new resource type.

Resolution: We will replace MKRESOURCE with MKREF, which creates only redirect reference resources.

C.9 lc-23-body

Type: change

[11]

reuterj@ira.uka.de (2000-02-07): Section 5.1: Get rid of the statement that the body of the resource is empty (PostConditions). It would be good if the response to GET included a response body that could be shown to a user by a client that doesn't do automatic redirection. There is a related problem in Section 6 on PUT. It is wrong to assume that what is PUT to a resource is what GET will return. In Section 6, say "A PUT with Apply-To-RR MAY contain a request body. The semantics of the request body is out of scope for this specification..." Also fix the discussion of example 6.2.

Resolution: Redirect references cannot have bodies. GET with Apply-To-RR MUST fail with 403. PUT with Apply-To-RR MUST fail with 403. See also issue 1.

C.10 lc-47-207

Type: change

[12]

yarong@Exchange.Microsoft.com (2000-02-11): In line with his wish to get rid of the request message body of MKRESOURCE, 207 would not be an appropriate response code. The description of 409 might lead someone to believe that you can't create redirect references outside of WebDAV namespaces. Suggests a different description.

Resolution: No longer relevant - MKREF can't get a 207 response.

Revise to make it clear that the first condition will only occur in WebDAV-compliant namespaces.

C.11 lc-49-put

Type: change

[13]

yarong@Exchange.Microsoft.com (2000-02-11): Remove the last sentence of Example 6.2, which says that PUT replaces the reference with a different resource.

Resolution: No longer relevant. Deleted this example in response to issue 48.

C.12 lc-75-ignore

Type: change

[14]

reuterj@ira.uka.de (2000-02-14): 11.2: "If the Apply-To-Redirect-Ref header is used on a request to any other sort of resource besides a redirect reference resource, the server SHOULD ignore it." Don't need to say this since HTP already says that any header that is not understood should be ignored.

Resolution: Need to keep this to specify what a server that does support this protocol needs to do when the header appears in a request to a non-redirect-ref resource. However, say "MUST".

Appendix D. Open issues (to be removed by RFC Editor before publication)

D.1 lc-85-301

Type: change

ejw@cse.ucsc.edu (2000-01-03): Support creation of other than 302 redirects, especially 301.

D.2 lc-38-not-hierarchical

Type: change

[15]

yarong@Exchange.Microsoft.com (2000-02-11): Not Hierarchical: The first sentence of the second paragraph of the introduction of the redirect spec asserts that the URIs of WebDAV compliant resources match to collections. The WebDAV standard makes no such requirement. I therefore move that this sentence be stricken.

Resolution: State the more general HTTP rationale first (alternative names for the same resource), then introduce the collection hierarchy rationale, which applies only if you are in a WebDAV-compliant space.

D.3 lc-36-server

Type: change

[16]

yarong@Exchange.Microsoft.com (2000-02-11): Servers: Replace "server" with "unrelated system" throughout.

Resolution: Try replacing "server" with "host" in some contexts, rephrasing in passive voice in others. See also issue 40.

D.4 lc-33-forwarding

Type: change

[17]

yarong@Exchange.Microsoft.com (2000-02-11): Forwarding: Replace "forward" with "redirect" throughout.

Resolution: Use "redirect" for the behavior redirect resources do exhibit. Use "forward" for the contrasting behavior (passing a method

on to the target with no client action needed). Define these two terms. See also issue 40.

D.5 lc-37-integrity

Type: change

[18]

yarong@Exchange.Microsoft.com (2000-02-11): Integrity: Intro, para 7 "Servers are not required to enforce the integrity of redirect references." Integrity is not defined. Replace with something clearer.

Resolution: Rewrite to say that the server MUST NOT update the target See also issue 6.

D.6 3-terminology-redirectref

Type: change

[19]

julian.reschke@greenbytes.de (2003-07-27): Consider global rename of "redirect reference resource" to "redirect resource".

D.7 lc-19-direct-ref

Type: change

[20]

reuterj@ira.uka.de (2000-02-07): Section 4, para 5 and Section 6, para 3 discussions of the Apply-to-Redirect-Ref header make it sound as if we are specifying direct reference behavior.

Resolution: Change these passages so that the contrast is between applying the method to the redirect reference and responding with a 302.

D.8 lc-41-no-webdav

Type: change

[21]

yarong@Exchange.Microsoft.com (2000-02-11): Make redirect references independent of the rest of WebDAV. The creation method for redirect

references shouldn't require an XML request body.

Resolution: We will make redirect references independent of the rest of WebDAV. MKREF will not have an XML request body.

D.9 lc-58-update

Type: change

[22]

yarong@Exchange.Microsoft.com (2000-02-11): There needs to be a way to update the target of a redirect reference.

Resolution: Agreed. See also issues 6, 43.

D.10 lc-24-properties

Type: change

[23]

reuterj@ira.uka.de (2000-02-07): Section 5.1: Replace the sentence "The properties of the new resource are as specified by the DAV:propertyupdate request body, using PROPPATCH semantics" with the following: "The MKRESOURCE request MAY contain a DAV:propertyupdate request body to initialize resource properties. Herein, the semantics is the same as when sending a MKRESOURCE request without a request body, followed by a PROPPATCH with the DAV:propertyupdate request body."

Resolution: No longer relevant once we switch to MKREF with no request body.

D.11 lc-48-s6

Type: change

[24]

yarong@Exchange.Microsoft.com (2000-02-11): Replace all of section 6 with just this: A redirect resource, upon receiving a request without an Apply-To-Redirect-Ref header, MUST respond with a 302 (Found) response. The 302 (Found) response MUST include a location header identifying the target and a Redirect-Ref header. If a redirect resource receives a request with an Apply-To-Redirect-Ref header then the redirect reference resource MUST apply the method to itself rather than blindly returning a 302 (Found) response.

Resolution: Keep a summary along the lines of Yaron's proposal (don't use the word "blindly"). Keep the bullets detailing the headers to be returned. Delete the rest, including the examples. See also issue 28, 29, 30, 31, 32.

D.12 lc-28-lang

Type: edit

[25]

reuterj@ira.uka.de (2000-02-07): Section 6: Get rid of the sentence "A reference-aware WebDAV client can act on this response in one of two ways." A client can act on the response in any way it wants.

Resolution: Agreed. See also issue 48.

D.13 lc-29-lang

Type: edit

[26]

reuterj@ira.uka.de (2000-02-07): Section 6, para 4: Obvious, doesn't need to be stated. Maybe note in an example.

Resolution: Agreed. See also issue 48.

D.14 lc-44-pseudo

Type: change

[27]

yarong@Exchange.Microsoft.com (2000-02-11): Instead of adding an optional prop XML element to the response element in 207 responses, define a new location XML element and a new refresource XML element.

Resolution: Agree to define new XML elements that are not pseudo-properties. Disagreement about whether refresource is needed. See issue 61.

D.15 lc-61-pseudo

Type: change

[28]

reuterj@ira.uka.de (2000-02-14): Section 7: It doesn't make sense to ask future editors of RFC 2518 to define DAV:location with the semantics it has here. RFC 2518 should provide the information in the Location header somehow in multistatus responses, but not by using properties.

Resolution: Define an XML element for location that is not a pseudo-property. We'll keep the recommendation that RFC 2518 add this for 302 responses. See also issue 44.

D.16 lc-60-ex

Type: change

[29]

reuterj@ira.uka.de (2000-02-14): Section 7, para 3: Make it clear that these are just examples of client behavior, and are not meant to limit the client's behavior to these options.

Resolution: Agreed to delete this paragraph. Continue discussion of what information should be returned with 302 in multistatus. Just location? Also redirectref?

D.17 lc-62-oldclient

Type: change

[30]

reuterj@ira.uka.de (2000-02-14): Section 7: It's too strong to claim that non-referencing clients can't process 302 responses occurring in Multi-Status responses. They just have an extra round trip for each 302.

Resolution: Remove last sentence of the paragraph that recommends changes to RFC 2518.

D.18 lc-63-move

Type: change

[31]

reuterj@ira.uka.de (2000-02-14): Section 7.1: Is MOVE atomic from the perspective of a client? Agrees that there should be no 302s for member redirect references, but finds the rationale dubious.

Resolution: Remove 7.1. Reword 7.2 to avoid concerns with "poses special problems" and "due to atomicity".

D.19 lc-06-reftarget-relative

Type: change

[32]

joe@orton.demon.co.uk (2000-01-29): Why does the spec talk about relative URIs in DAV:reftarget in MKRESOURCE requests? Is the server required to resolve the relative URI and store it as absolute? Is the server required to keep DAV:reftarget pointing to the target resource as the reference / target move, or is DAV:reftarget a dead property?

Resolution: DAV:reftarget is readonly and present only on redirect references that are also WebDAV resources. Add a method for setting the target. Change definition of Redirect-Ref header so that it has the target as its value (comes back on all 302 responses). Server MUST store the target exactly as it is set. It MUST NOT resolve relatives to absolutes and MUST NOT update if target resource moves. See also issue 17, 43, 50, 57

D.20 lc-57-noautoupdate

Type: change

[33]

yarong@Exchange.Microsoft.com (2000-02-11): Add language to forbid servers from automatically updating redirect resources when their targets move.

Resolution: Agreed. See also issue 6.

D.21 lc-71-relative

Type: change

[34]

reuterj@ira.uka.de (2000-02-14): Section 9: Base URI should be the Request-URI or href minus its final segment.

Resolution: Fix this.

D.22 lc-53-s10

Type: change

[35]

yarong@Exchange.Microsoft.com (2000-02-11): The behavior described in this section would have a very serious impact on the efficiency of mapping Request-URIs to resources in HTTP request processing. Also specify another type of redirect resource that does not behave as in section 10, but instead would "expose the behavior we see today in various HTTP servers that allow their users to create 300 resources." Be sure we know what behavior will be if the redirect location is not an HTTP URL, but, say ftp.

Resolution: We won't define 2 sorts of redirect references here. Servers SHOULD respond with 302 as described here, but if they can't do that, respond with 404 Not Found. (It's hard to modularize the behavior specified - it impacts processing Not Found cases of all methods, so you can't just add it to an HTTP server in a redirect ref module.)

D.23 lc-72-trailingslash

Type: change

[36]

reuterj@ira.uka.de (2000-02-14): Section 10: Forbid DAV:reftarget from ending in "/"

Resolution: Make the note warn about the possibility of two slashes in a row, recommend against ending target with a slash, since that could result in two slashes in a row.

D.24 lc-50-blindredirect

Type: change

[37]

yarong@Exchange.Microsoft.com (2000-02-11): Replace current language explaining the purpose of the Redirect-Ref header with language that simply states that it marks blind 302 responses from redirect resources. (Section 6.3, 11.1)

Resolution: Section 6.3 was removed in response to issue 48. In 11.1, change the definition of the Redirect-Ref header to have the value of the target (relative URI) as its value. Then we don't need a method for retrieving the target's relative URI. Presence of the

Redirect-Ref header lets the client know that the resource accepts Apply-To-RR header and the new method for updating target. Reject Yaron's suggested language, but make the above changes.

D.25 lc-74-terminology

Type: change

[38]

reuterj@ira.uka.de (2000-02-14): "plain HTTP/1.1 redirect" - find some good name for this an use it consistently

D.26 lc-76-location

Type: change

[39]

reuterj@ira.uka.de (2000-02-22): 12.2: Make DAV:location a real (live) property, get rid of the DAV:reftarget property

D.27 lc-79-accesscontrol

Type: change

[40]

reuterj@ira.uka.de (2000-02-22): Section 16.4: "In some environments, the owner of a resource might be able to use access control to prevent others from creating references to that resource." That would not be consistent with the concept of redirect references as weak links (e.g. think of moving a resource to a different locationo that is already the target of some redirection reference.

D.28 lc-80-i18n

Type: change

[41]

reuterj@ira.uka.de (2000-02-22): Section 17: Could get rid of a lot of this section, since this protocol extends WebDAV. Just reference [WebDAV].

D.29 lc-55-iana

Type: change

[42]

yarong@Exchange.Microsoft.com (2000-02-11): Expand the IANA section to list all methods, headers, XML elements, MIME types, URL schemes, etc., defined by the spec.

Resolution: Agreed.

Intellectual Property Statement

The IETF 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 the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. 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 IETF Secretariat.

The IETF 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 practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2003). 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 may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process 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 the Internet Society or its successors or assignees.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE 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.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.