Internet Engineering Task Force   C. Ma
Internet Draft   J. Chen
Intended status: Informational   X. Fan
Expires: December 21, 2022   M. Chen
                                                          Z. Li
          China Academy of Information and Communications Technology
                                                   June 21, 2021

                 HTTP Usage in the Industrial Internet Identifier Data Access
                              Protocol (IIIDAP)
                       draft-ma-identifier-access-http-03

Status of this Memo

   This Internet-Draft is submitted 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 December 21, 2021.

Copyright Notice

Abstract

This document describes an Extensible Provisioning Protocol (EPP) for the provisioning and management of enterprises and identifiers between the server which is called Business Management System (BMS) and is entitled to manage the identifier top-level node and the client which is also referred to as Second Node Management System (SNMS). Specified in XML, the mapping defines EPP command syntax and semantics as applied to enterprise and identifier management.

Table of Contents

1. **Introduction**

   This document describes the usage of the Hypertext Transfer Protocol
   (HTTP) [RFC7230] for the Industrial Internet Identifier Data Access
   Protocol (IIIDAP). The goal of this document is to tie together
   usage patterns of HTTP into a common profile applicable to the
   various types of directory services serving identifier data using
   practices informed by the Representational State Transfer (REST)
   [REST] architectural style. By giving the various directory services
   common behavior, a single client is better able to retrieve data
   from directory services adhering to this behavior.

   Identifier data expected to be presented by this service is
   Industrial Internet Identifier data -- some of the information that
   identifiers of Second-Level Nodes (SLN) and Enterprise-Level Nodes
   (ELN) contain (see [IDENTIFIER-RESPONSES]). This protocol is
   expected to provide a specification for queries and responses,
   redirection to authoritative sources, and support for localized
   identifier data such as addresses and organization or person names.
   This function is expected to be provided by SLN.

   In designing these common usage patterns, this document introduces
   considerations for a simple use of HTTP. Where complexity may
   reside, it is the goal of this document to place it upon the server
   and to keep the client as simple as possible. A client
   implementation should be possible using common operating system
   scripting tools (e.g., bash and wget).

   This is the basic usage pattern for this protocol:

   1. A client determines an appropriate server to query along with the
      appropriate base Uniform Resource Locator (URL) to use in such
      queries. [IDENTIFIER-AUTHORIZATION] describes one method to
      determine the server and the base URL. See Appendix C for more
      information.

   2. A client issues an HTTP (or HTTPS) query using GET [RFC7231]. As
      an example, a query URL for the enterprise identifier 86.100.1
      might be

        http://example.com/iiidap/identifier/86.100.1

    [IDENTIFIER-QUERY] details the various queries used in IIIDAP.

  3. If the receiving server has the information for the query, it
     examines the Accept header field of the query and returns a 200
     response with a response entity appropriate for the requested
     format. [IDENTIFIER-RESPONSES] details a response in JavaScript
     Object Notation (JSON).

  4. If the receiving server does not have the information for the
     query but does have knowledge of where the information can be
     found, it will return a redirection response (3xx) with the
     Location header field containing an HTTP(S) URL pointing to the
     information or another server known to have knowledge of the
     location of the information. The client is expected to require
     using that HTTP URL.

  5. If the receiving server does not have the information being
     requested and does not have knowledge of where the information
     can be found, it returns a 404 response.

  6. If the receiving server will not answer a request for policy
     reasons, it will return an error response (4xx) indicating the
     reason for giving no answer.

    It is not the intent of this document to redefine the meaning and
    semantics of HTTP. The purpose of this document is to clarify the
    use of standard HTTP mechanisms for this application.

2. Conventions used in this document

    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 RFC 2119 [RFC2119].

    In this document, an IIIDAP client is an HTTP user agent performing
    an IIIDAP query, and an IIIDAP server is an HTTP server providing an
    IIIDAP response. IIIDAP query and response formats are described in
    [IDENTIFIER-QUERY] and [IDENTIFIER-RESPONSES], while this document
    describes how IIIDAP clients and servers use HTTP to exchange
    queries and responses. [IDENTIFIER-SECURITY] describes security
    considerations for IIIDAP.

2.1. Acronyms and Abbreviations

    IIIDAP: Industrial Internet Identifier Data Access Protocol

SLN: Second-Level Nodes

ELN: Enterprise-Level Nodes

[3](#). Design Intents

There are a few design criteria this document attempts to meet.

First, each query is meant to require only one path of execution to obtain an answer. A response may contain an answer, no answer, or a redirect, and clients are not expected to fork multiple paths of execution to make a query.

Second, the semantics of the request/response allow for future and/or non-standard response formats. In this document, only a JSON [[RFC8259](#)] response media type is noted, with the response contents to be described separately (see [[IDENTIFIER-RESPONSES](#)]). This document only describes how IIIDAP is transported using HTTP with this format.

Third, this protocol is intended to be able to make use of the range of mechanisms available for use with HTTP. HTTP offers a number of mechanisms not described further in this document. Operators are able to make use of these mechanisms according to their local policy, including cache control, authorization, compression, and redirection. HTTP also benefits from widespread investment in scalability, reliability, and performance, as well as widespread programmer understanding of client behaviors for web services styled after REST [[REST](#)], reducing the cost to deploy Registration Data Directory Services and clients. This protocol is forward compatible with HTTP 2.0.

[4](#). Queries

[4.1](#). HTTP Methods

Clients use the GET method to retrieve a response body and use the HEAD method to determine existence of data on the server. Clients SHOULD use either the HTTP GET or HEAD methods (see [[RFC7231](#)]). Servers are under no obligation to support other HTTP methods; therefore, clients using other methods will likely not interoperate properly.

Clients and servers MUST support HTTPS to support security services.

4.2. Accept Header

   To indicate to servers that an IIIDAP response is desired, clients
   include an Accept header field with an IIIDAP-specific JSON media
   type, the generic JSON media type, or both. Servers receiving an
   IIIDAP request return an entity with a Content-Type header
   containing the IIIDAP-specific JSON media type.

   This specification does not define the responses a server returns to
   a request with any other media types in the Accept header field, or
   with no Accept header field. One possibility would be to return a
   response in a media type suitable for rendering in a web browser.

4.3. Query Parameters

   Servers MUST ignore unknown query parameters. Use of unknown query
   parameters for cache busting is described in Appendix B.

5. Types of HTTP Response

   This section describes the various types of responses a server may
   send to a client. While no standard HTTP response code is forbidden
   in usage, this section defines the minimal set of response codes in
   common use by servers that a client will need to understand. While
   some clients may be constructed with simple tooling that does not
   account for all of these response codes, a more robust client
   accounting for these codes will likely provide a better user
   experience. It is expected that usage of response codes and types
   for this application not defined here will be described in
   subsequent documents.

5.1. Positive Answers

   If a server has the information requested by the client and wishes
   to respond to the client with the information according to its
   policies, it returns that answer in the body of a 200 (OK) response
   (see [RFC7231]).

5.2. Redirects

   If a server wishes to inform a client that the answer to a given
   query can be found elsewhere, it returns either a 301 (Moved
   Permanently) response code to indicate a permanent move or a 302
   (Found), 303 (See Other), or 307 (Temporary Redirect) response code
   to indicate a non-permanent redirection, and it includes an HTTP(S)
   URL in the Location header field (see [RFC7231]). The client is
   expected to issue a subsequent request to satisfy the original query

using the given URL without any processing of the URL. In other
words, the server is to hand back a complete URL, and the client
should not have to transform the URL to follow it. Servers are under
no obligation to return a URL conformant to [IDENTIFIER-QUERY].

For this application, such an example of a permanent move might be a
SLN operator informing a client the information being sought can be
found with another SLN operator (i.e., a query for the identifier
86.100.1 in foo.example is found at
http://foo.example/identifier/86.100.1).

For example, if the client uses

        http://serv1.example.com/weirds/identifier/86.100.1

the server redirecting to

        https://serv2.example.net/weirds2/

would set the Location: field to the value

        https://serv2.example.net/weirds2/identifier/86.100.1

5.3. Negative Answers

If a server wishes to respond that it has an empty result set (that
is, no data appropriately satisfying the query), it returns a 404
(Not Found) response code. Optionally, it MAY include additional
information regarding the negative answer in the HTTP entity body.

If a server wishes to inform the client that information about the
query is available, but cannot include the information in the
response to the client for policy reasons, the server MUST respond
with an appropriate response code out of HTTP's 4xx range. A client
MAY retry the query if that is appropriate for the respective
response code.

5.4. Malformed Queries

If a server receives a query that it cannot interpret as an IIIDAP
query, it returns a 400 (Bad Request) response code. Optionally, it
MAY include additional information regarding this negative answer in
the HTTP entity body.

5.5. Rate Limits

Some servers apply rate limits to deter address scraping and other
abuses. When a server declines to answer a query due to rate limits,
it returns a 429 (Too Many Requests) response code as described in
[RFC6585]. A client that receives a 429 response SHOULD decrease its
query rate and honor the Retry-After header field if one is present.
Servers may place stricter limits upon clients that do not honor the
Retry-After header. Optionally, the server MAY include additional
information regarding the rate limiting in the HTTP entity body.

Note that this is not a defense against denial-of-service (DoS)
attacks, since a malicious client could ignore the code and continue
to send queries at a high rate. A server might use another response
code if it did not wish to reveal to a client that rate limiting is
the reason for the denial of a reply.

5.6. Cross-Origin Resource Sharing (CORS)

When responding to queries, it is RECOMMENDED that servers use the
Access-Control-Allow-Origin header field, as specified by [W3C.REC-
cors-20140116]. A value of "*" is suitable when IIIDAP is used for
public resources.

This header (often called the CORS header) helps in-browser web
applications by lifting the "same-origin" restriction (i.e., a
browser may load IIIDAP client code from one web server but query
others for IIIDAP data).

By default, browsers do not send cookies when cross origin requests
are allowed. Setting the Access-Control-Allow-Credentials header
field to "true" will send cookies. Use of the Access-Control-Allow-
Credentials header field is not recommended.

6. Security Considerations

This document does not pose strong security requirements to the
IIIDAP protocol. However, it does not restrict against the use of
security mechanisms offered by the HTTP protocol. It does require
that IIIDAP clients and servers MUST support HTTPS.

This document makes recommendations for server implementations
against DoS (Section 5.5) and interoperability with existing
security mechanisms in HTTP clients (Section 5.6).

Additional security considerations to the IIIDAP protocol are
covered in [IDENTIFIER-SECURITY].

7. Internationalization Considerations

7.1. URIs and IRIs

Clients can use Internationalized Resource Identifiers (IRIs) [RFC3987] for internal use as they see fit but MUST transform them to URIs [RFC3986] for interaction with IIIDAP servers. IIIDAP servers MUST use URIs in all responses, and again clients can transform these URIs to IRIs for internal use as they see fit.

7.2. Language Identifiers in Queries and Responses

Under most scenarios, clients requesting data will not signal that the data be returned in a particular language or script. On the other hand, when servers return data and have knowledge that the data is in a language or script, the data SHOULD be annotated with language identifiers whenever they are available, thus allowing clients to process and display the data accordingly.

[IDENTIFIER-RESPONSES] provides such a mechanism.

7.3. Language Identifiers in HTTP Headers

Given the description of the use of language identifiers in Section 9.2, unless otherwise specified, servers SHOULD ignore the HTTP [RFC7231] Accept-Language header field when formulating HTTP entity responses, so that clients do not conflate the Accept-Language header with the 'lang' values in the entity body.

However, servers MAY return language identifiers in the Content-Language header field so as to inform clients of the intended language of HTTP layer messages.

8. IANA Considerations

9. References

References to IIIDAP are subject to the latest edition.

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
          Requirement Levels", BCP 14, RFC 2119, March 1997,
          <http://www.rfc-editor.org/info/rfc2119>.

   [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform
             Resource Identifier (URI): Generic Syntax", STD 66, RFC
             3986, January 2005,
             <http://www.rfc-editor.org/info/rfc3986>.

   [RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource
             Identifiers (IRIs)", RFC 3987, January 2005,
             <http://www.rfc-editor.org/info/rfc3987>.

   [RFC6585] Nottingham, M. and R. Fielding, "Additional HTTP Status
             Codes", RFC 6585, April 2012,
             <http://www.rfc-editor.org/info/rfc6585>.

   [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer
             Protocol (HTTP/1.1): Message Syntax and Routing", RFC
             7230, June 2014,
             <http://www.rfc-editor.org/info/rfc7230>.

   [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer
             Protocol (HTTP/1.1): Semantics and Content", RFC 7231,
             June 2014,
             <http://www.rfc-editor.org/info/rfc7231>.

   [W3C.REC-cors-20140116]
             Kesteren, A., "Cross-Origin Resource Sharing", W3C
             Recommendation, REC-cors-20140116, January 2014,
             <http://www.w3.org/TR/2014/REC-cors-20140116/>.

9.2. Informative References

   [REST]   Fielding, R. and R. Taylor, "Principled Design of the
             Modern Web Architecture", ACM Transactions on Internet
             Technology, Vol. 2, No. 2, May 2002.

   [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data
             Interchange Format", STD 90, RFC 8259,
             DOI 10.17487/RFC8259, December 2017,
             <https://www.rfc-editor.org/info/rfc8259>.

   [IDENTIFIER-SECURITY]
             Ma, C., "Security Services for the Industrial Internet
             Identifier Data Access Protocol (IIIDAP)", Work in
             Progress, draft-mcd-identifier-access-security, June 2021.

   [IDENTIFIER-QUERY]
             Ma, C., "Industrial Internet Identifier Data Access
             Protocol (IIIDAP) Query Format", Work in Progress, draft-
             mcd-identifier-access-query, June 2021.

   [IDENTIFIER-RESPONSES]
            Ma, C., "JSON Responses for the Industrial Internet
            Identifier Data Access Protocol (IIIDAP)", Work in
            Progress, draft-mcd-identifier-access-response, June 2021.

   [IDENTIFIER-AUTHORIZATION]
            Ma, C., "Finding the Authoritative Registration Data
            (IIIDAP) Service", Work in Progress, draft-mcd-identifier-
            access-authority, June 2021.

[Appendix A](#).   **Protocol Example**

   To demonstrate typical behavior of an IIIDAP client and server, the
   following is an example of an exchange, including a redirect. The
   data in the response has been elided for brevity, as the data format
   is not described in this document. The media type used here is
   described in [[IDENTIFIER-RESPONSES](#)].

   An example of an IIIDAP client and server exchange:

     Client:
         <TCP connect to iiidap.example.com port 80>
         GET /iiidap/identifier/86.100.1 HTTP/1.1
         Host: iiidap.example.com
         Accept: application/iiidap+json

     iiidap.example.com:
         HTTP/1.1 301 Moved Permanently
         Location: http://iiidap-
         identifier.example.com/iiidap/identifier/86.100.1
         Content-Length: 0
         Content-Type: application/iiidap+json
         <TCP disconnect>

     Client:
         <TCP connect to iiidap-identifier.example.com port 80>
         GET /iiidap/identifier/86.100.1 HTTP/1.1
         Host: iiidap-identifier.example.com
         Accept: application/iiidap+json

     iiidap-identifier.example.com:
         HTTP/1.1 200 OK
         Content-Type: application/iiidap+json
         Content-Length: 9001

         { ... }
         <TCP disconnect>

Appendix B.   Cache Busting

   Some HTTP [RFC7230] cache infrastructures do not adhere to caching
   standards adequately and could cache responses longer than is
   intended by the server. To overcome these issues, clients can use an
   ad hoc and improbably used query parameter with a random value of
   their choosing. As Section 4.3 instructs servers to ignore unknown
   parameters, this is compatible with the IIIDAP definition.

   An example of using an unknown query parameter to bust caches:

        http://example.com/identifier/86.100.1?__fuhgetaboutit=xyz123

   Use of an unknown parameter to overcome misbehaving caches is not
   part of any specification and is offered here for informational
   purposes.

Appendix C.   **Bootstrapping and Redirection**

   These issues are solved in IIIDAP using HTTP redirects and
   bootstrapping. Bootstrapping is discussed in [IDENTIFIER-
   AUTHORIZATION]. In constrained environments, the processes outlined
   in [IDENTIFIER-AUTHORIZATION] may not be viable, and there may be
   the need for servers acting as a "redirector".

   Redirector servers issue HTTP redirects to clients using a
   redirection table informed by [IDENTIFIER-AUTHORIZATION]. Figure 1
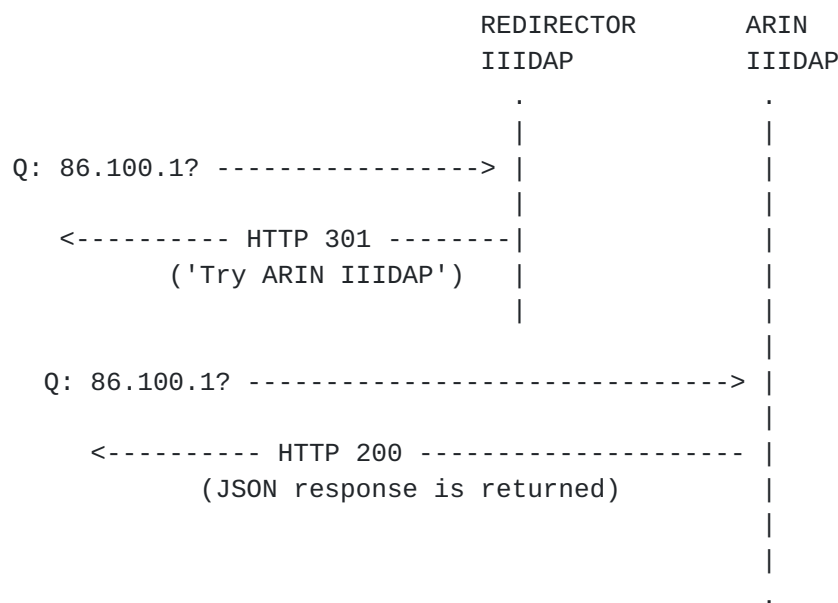   diagrams a client using a redirector for bootstrapping.

```
                                    REDIRECTOR        ARIN
                                    IIIDAP            IIIDAP
                                        .                 .
                                        |                 |
        Q: 86.100.1? ----------------> |                 |
                                        |                 |
           <---------- HTTP 301 --------|                 |
                  ('Try ARIN IIIDAP')   |                 |
                                        |                 |
                                        |                 |
          Q: 86.100.1? -------------------------------> |
                                                          |
             <---------- HTTP 200 -------------------- |
                    (JSON response is returned)         |
                                                          |
                                                          |
                                                          .
```

              Figure 1: Querying IIIDAP Data for 86.100.1

   In some cases, multiple HTTP redirects will be issued. Figure 2
   shows such a scenario.

```
                          REDIRECTOR  LACNIC            ARIN
                          IIIDAP      IIIDAP            IIIDAP
                              .           .                 .
         Q: 86.100.1? ---->  |           |                 |
                              |           |                 |
            <-- HTTP 301 ---  |           |                 |
             ('Try LACNIC')   |           |                 |
                              |           |                 |
                              |           |                 |
         Q: 86.100.1? ----------------->  |                 |
                              |           |                 |
            <---------- HTTP 301 --------|                 |
```

```
                ('Try ARIN IIIDAP')   |                 |
                                      |                 |
                                                        |
           Q: 86.100.1? -------------------------------> |
                                                        |
             <---------- HTTP 200 -------------------- |
                    (JSON response is returned)         |
                                                        |
                                                        |
                                                        .
```
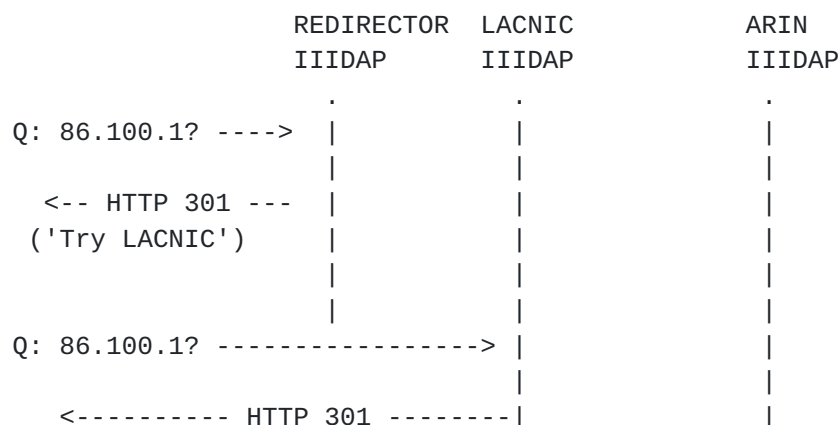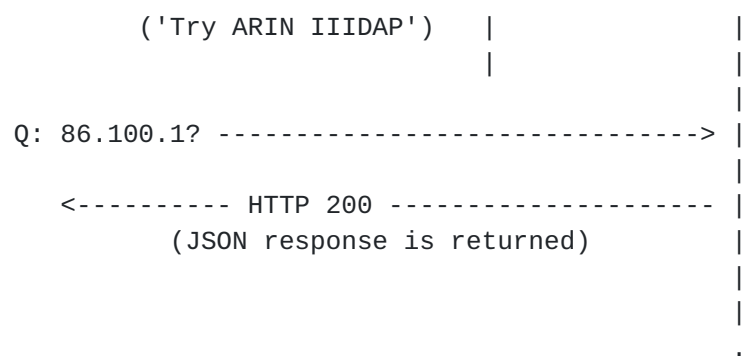
     Figure 2: Querying IIIDAP Data for Data That Has Been Transferred

Authors' Addresses

   Chendi Ma
   CAICT
   No.52 Huayuan North Road, Haidian District
   Beijing, Beijing, 100191
   China

   Phone: +86 177 1090 9864
   Email: machendi@caict.ac.cn

   Chen Jian
   CAICT
   No.52 Huayuan North Road, Haidian District
   Beijing, Beijing, 100191
   China

   Phone: +86 138 1103 3332
   Email: chenjian3@caict.ac.cn

   Xiaotian Fan
   CAICT
   No.52 Huayuan North Road, Haidian District
   Beijing, Beijing, 100191
   China

   Phone: +86 134 0108 6945
   Email: fanxiaotian@caict.ac.cn

   Meilan Chen
   CAICT
   No.52 Huayuan North Road, Haidian District
   Beijing, Beijing, 100191
   China

   Phone: +86 139 1143 7301
   Email: chenmeilan@caict.ac.cn

Zhiping Li
CAICT
No.52 Huayuan North Road, Haidian District
Beijing, Beijing, 100191
China

Phone: +86 185 1107 1386
Email: lizhiping@caict.ac.cn