

Registration Protocols Extensions
Internet-Draft
Updates: [7484](#) (if approved)
Intended status: Best Current Practice
Expires: January 16, 2019

S. Hollenbeck
Verisign Labs
A. Newton
ARIN
July 15, 2018

Registration Data Access Protocol (RDAP) Object Tagging
draft-ietf-regext-rdap-object-tag-04

Abstract

The Registration Data Access Protocol (RDAP) includes a method that can be used to identify the authoritative server for processing domain name, IP address, and autonomous system number queries. The method does not describe how to identify the authoritative server for processing other RDAP query types, such as entity queries. This limitation exists because the identifiers associated with these query types are typically unstructured. This document updates [RFC 7484](#) by describing an operational practice that can be used to add structure to RDAP identifiers that makes it possible to identify the authoritative server for additional RDAP queries.

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). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on January 16, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
2.	Object Naming Practice	3
3.	Bootstrap Service Registry for RDAP Service Providers	8
3.1.	Registration Procedure	9
4.	RDAP Conformance	9
5.	IANA Considerations	10
5.1.	Bootstrap Service Registry for RDAP Service Providers	10
5.2.	RDAP Extensions Registry	10
6.	Implementation Status	10
6.1.	Verisign Labs	11
6.2.	OpenRDAP	11
7.	Security Considerations	12
8.	Acknowledgements	12
9.	References	12
9.1.	Normative References	12
9.2.	Informative References	13
Appendix A.	Change Log	13
	Authors' Addresses	14

[1.](#) Introduction

The Registration Data Access Protocol (RDAP) includes a method ([[RFC7484](#)]) that can be used to identify the authoritative server for processing domain name, IP address, and autonomous system number (ASN) queries. This method works because each of these data elements is structured in a way that facilitates automated parsing of the element and association of the data element with a particular RDAP service provider. For example, domain names include labels (such as "com", "net", and "org") that are associated with specific service providers.

As noted in [Section 9 of RFC 7484](#) [[RFC7484](#)], the method does not describe how to identify the authoritative server for processing entity queries, name server queries, help queries, or queries using certain search patterns. This limitation exists because the identifiers bound to these queries are typically not structured in a way that makes it easy to associate an identifier with a specific service provider. This document describes an operational practice that can be used to add structure to RDAP identifiers that makes it

possible to identify the authoritative server for additional RDAP queries.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all capitals, as shown here.

2. Object Naming Practice

Tagging object identifiers with a service provider tag makes it possible to identify the authoritative server for processing an RDAP query using the method described in [RFC 7484](#) [[RFC7484](#)]. A service provider tag is constructed by prepending the Unicode HYPHEN-MINUS character "-" (U+002D, described as an "unreserved" character in [RFC 3986](#) [[RFC3986](#)]) to an IANA-registered value that represents the service provider. For example, a tag for a service provider identified by the string value "ARIN" is represented as "-ARIN".

Service provider tags are concatenated to the end of RDAP query object identifiers to unambiguously identify the authoritative server for processing an RDAP query. Building on the example from [Section 3.1.5 of RFC 7482](#) [[RFC7482](#)], an RDAP entity handle can be constructed that allows an RDAP client to bootstrap an entity query. The following identifier is used to find information for the entity associated with handle "XXXX" at service provider "ARIN":

XXXX-ARIN

Clients that wish to bootstrap an entity query can parse this identifier into distinct handle and service provider identifier elements. Handles can themselves contain HYPHEN-MINUS characters; the service provider identifier is found following the last HYPHEN-MINUS character in the tagged identifier. The service provider identifier is used to retrieve a base RDAP URL from an IANA registry. The base URL and entity handle are then used to form a complete RDAP query path segment. For example, if the base RDAP URL "https://example.com/rdap/" is associated with service provider "YYYY" in an IANA registry, an RDAP client will parse a tagged entity identifier "XXXX-YYYY" into distinct handle ("XXXX") and service provider ("YYYY") identifiers. The service provider identifier "YYYY" is used to query an IANA registry to retrieve the base RDAP URL "https://example.com/rdap/". The RDAP query URL is formed using the base RDAP URL and entity path segment described in [Section 3.1.5 of RFC 7482](#) [[RFC7482](#)], using "XXXX-YYY" as the value of the handle identifier. The complete RDAP query URL becomes "https://example.com/rdap/entity/XXXX-YYYY".

Implementation of this practice requires tagging of unstructured potential query identifiers in RDAP responses. Consider these elided examples ("..." is used to note elided response objects) from [Section 5.3 of RFC 7483](#) [[RFC7483](#)] in which the handle identifiers have been tagged with service provider tags "RIR", "DNR", and "ABC" respectively:

```
{
  "objectClassName" : "domain",
  "handle" : "XXXX-RIR",
  "ldhName" : "0.2.192.in-addr.arpa",
  "nameservers" :
  [
    ...
  ],
  "securedNS":
  {
    ...
  },
  "remarks" :
  [
    ...
  ],
  "links" :
  [
    ...
  ],
  "events" :
  [
    ...
  ],
  "entities" :
  [
    {
      "objectClassName" : "entity",
      "handle" : "XXXX-RIR",
      "vcardArray":
      [
        ...
      ],
      "roles" : [ "registrant" ],
      "remarks" :
      [
        ...
      ],
      "links" :
      [
        ...
      ]
    }
  ]
}
```



```
    ],
    "events" :
    [
        ...
    ]
}
],
"network" :
{
    "objectClassName" : "ip network",
    "handle" : "XXXX-RIR",
    "startAddress" : "192.0.2.0",
    "endAddress" : "192.0.2.255",
    "ipVersion" : "v4",
    "name" : "NET-RTR-1",
    "type" : "DIRECT ALLOCATION",
    "country" : "AU",
    "parentHandle" : "YYYY-RIR",
    "status" : [ "active" ]
}
}
```

Figure 1

```
{
    "objectClassName" : "domain",
    "handle" : "XXXX-DNR",
    "ldhName" : "xn--fo-5ja.example",
    "unicodeName" : "foo.example",
    "variants" :
    [
        ...
    ],
    "status" : [ "locked", "transfer prohibited" ],
    "publicIds":
    [
        ...
    ],
    "nameservers" :
    [
        {
            "objectClassName" : "nameserver",
            "handle" : "XXXX-DNR",
            "ldhName" : "ns1.example.com",
            "status" : [ "active" ],
            "ipAddresses" :
            {
                ...
            }
        }
    ]
}
```



```
    },
    "remarks" :
    [
        ...
    ],
    "links" :
    [
        ...
    ],
    "events" :
    [
        ...
    ]
},
{
    "objectClassName" : "nameserver",
    "handle" : "XXXX-DNR",
    "ldhName" : "ns2.example.com",
    "status" : [ "active" ],
    "ipAddresses" :
    {
        ...
    },
    "remarks" :
    [
        ...
    ],
    "links" :
    [
        ...
    ],
    "events" :
    [
        ...
    ]
}
],
"secureDNS":
{
    ...
},
"remarks" :
[
    ...
],
"links" :
[
    ...
]
```



```
    ],
    "port43" : "whois.example.net",
    "events" :
    [
        ...
    ],
    "entities" :
    [
        {
            "objectClassName" : "entity",
            "handle" : "XXXX-ABC",
            "vcardArray":
            [
                ...
            ],
            "status" : [ "validated", "locked" ],
            "roles" : [ "registrant" ],
            "remarks" :
            [
                ...
            ],
            "links" :
            [
                ...
            ],
            "events" :
            [
                ...
            ]
        }
    ]
}
```

Figure 2

As described in [Section 5 of RFC 7483 \[RFC7483\]](#), RDAP responses can contain "self" links. Service provider tags and self references SHOULD be consistent. If they are inconsistent, the service provider tag is processed with higher priority when using these values to identify a service provider.

There is a risk of unpredictable processing behavior if the HYPHEN-MINUS character is used for naturally occurring, non-separator purposes in an entity handle. This could lead to a client mistakenly assuming that a HYPHEN-MINUS character represents a separator and the text that follows HYPHEN-MINUS is a service provider identifier. A client that queries the IANA registry for what they assume is a valid service provider will likely receive an unexpected, invalid result.

As a consequence, use of the HYPHEN-MINUS character as a service provider tag separator MUST be noted by adding rdapConformance value to query responses as described in [Section 4](#).

The HYPHEN-MINUS character was chosen as a separator for two reasons: 1) it is a familiar separator character in operational use, and 2) it avoids collision with URI-reserved characters. The list of unreserved characters specified in [Section 2.3 of RFC 3986](#) [[RFC3986](#)] provided multiple options for consideration:

unreserved = ALPHA / DIGIT / "-" / "." / "_" / "~"

ALPHA and DIGIT characters were excluded because they are commonly used in entity handles for non-separator purposes. HYPHEN-MINUS is commonly used as a separator and recognition of this practice will reduce implementation requirements and operational risk. The remaining characters were excluded because they are not broadly used as separators in entity handles.

3. Bootstrap Service Registry for RDAP Service Providers

The bootstrap service registry for the RDAP service provider space is represented using the structure specified in [Section 3 of RFC 7484](#) [[RFC7484](#)]. The JSON output of this registry contains alphanumeric identifiers that identify RDAP service providers, grouped by base RDAP URLs, as shown in this example.


```
{
  "version": "1.0",
  "publication": "YYYY-MM-DDTHH:MM:SSZ",
  "description": "RDAP service provider bootstrap values",
  "services": [
    [
      ["YYYY"],
      [
        "https://example.com/rdap/"
      ]
    ],
    [
      ["ZZ54"],
      [
        "http://rdap.example.org/"
      ]
    ],
    [
      ["1754"],
      [
        "https://example.net/rdap/",
        "http://example.net/rdap/"
      ]
    ]
  ]
}
```

Figure 3

Alphanumeric service provider identifiers conform to the suffix portion ("`\w{1,8}`") of the "roidType" syntax specified in [Section 4.2 of RFC 5730](#) [[RFC5730](#)].

3.1. Registration Procedure

The service provider registry is populated using the "First Come First Served" policy defined in [RFC 8126](#) [[RFC8126](#)]. Provider identifier values can be derived and assigned by IANA on request. Registration requests include the requested service provider identifier (or an indication that IANA should assign an identifier) and one or more base RDAP URLs to be associated with the service provider identifier.

4. RDAP Conformance

RDAP responses that contain values described in this document MUST indicate conformance with this specification by including an `rdapConformance` ([[RFC7483](#)]) value of "rdap_objectTag_level_0". The

information needed to register this value in the RDAP Extensions Registry is described in [Section 5.2](#).

Example rdapConformance structure with extension specified:

```
"rdapConformance" :  
[  
  "rdap_level_0",  
  "rdap_objectTag_level_0"  
]
```

Figure 4

5. IANA Considerations

IANA is requested to create the RDAP Bootstrap Services Registry listed below and make it available as JSON objects. The contents of this registry is described in [Section 3](#), with the formal syntax specified in [Section 10 of RFC 7484](#) [[RFC7484](#)].

5.1. Bootstrap Service Registry for RDAP Service Providers

Entries in this registry contain at least the following:

- o An alphanumeric value that identifies the RDAP service provider being registered.
- o One or more URLs that provide the RDAP service regarding this registration.

5.2. RDAP Extensions Registry

IANA is requested to register the following value in the RDAP Extensions Registry:

```
Extension identifier: rdap_objectTag  
Registry operator: Any  
Published specification: This document.  
Contact: IESG <iesg@ietf.org>  
Intended usage: This extension describes a best practice for  
structuring entity identifiers to enable query bootstrapping.
```

6. Implementation Status

NOTE: Please remove this section and the reference to [RFC 7942](#) prior to publication as an RFC.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this

Internet-Draft, and is based on a proposal described in [RFC 7942](#) [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC 7942](#), "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

[6.1.](#) Verisign Labs

Responsible Organization: Verisign Labs

Location: <https://rdap.verisignlabs.com/>

Description: This implementation includes support for domain registry RDAP queries using live data from the .cc and .tv country code top-level domains. Client authentication is required to receive entity information in query responses.

Level of Maturity: This is a "proof of concept" research implementation.

Coverage: This implementation includes all of the features described in this specification.

Contact Information: Scott Hollenbeck, shollenbeck@verisign.com

[6.2.](#) OpenRDAP

Responsible Organization: OpenRDAP

Location: <https://www.openrdap.org>

Description: RDAP client implementing bootstrapping for entity handles with a service provider tag. A test Bootstrap Services Registry file is currently used in lieu of an official one.

Level of Maturity: Alpha

Coverage: Implements draft 04+, supports the HYPHEN-MINUS separator character only.

Contact Information: Tom Harwood, tfh@skip.org

7. Security Considerations

This practice helps to ensure that end users will get RDAP data from an authoritative source using a bootstrap method to find authoritative RDAP servers, reducing the risk of sending queries to non-authoritative sources. The method has the same security properties as the RDAP protocols themselves. The transport used to access the IANA registries can be more secure by using TLS [[RFC5246](#)], which IANA supports. Additional considerations associated with RDAP are described in [RFC 7481](#) [[RFC7481](#)].

8. Acknowledgements

The author would like to acknowledge the following individuals for their contributions to the development of this document: Tom Harrison, Patrick Mevzek, and Marcos Sanz. In addition, the authors would like to recognize the Regional Internet Registry (RIR) operators (AFRINIC, APNIC, ARIN, LACNIC, and RIPE) that have been implementing and using the practice of tagging handle identifiers for several years. Their experience provided significant inspiration for the development of this document.

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, [RFC 5730](#), DOI 10.17487/RFC5730, August 2009, <<https://www.rfc-editor.org/info/rfc5730>>.
- [RFC7484] Blanchet, M., "Finding the Authoritative Registration Data (RDAP) Service", [RFC 7484](#), DOI 10.17487/RFC7484, March 2015, <<https://www.rfc-editor.org/info/rfc7484>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [RFC 8126](#), DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

9.2. Informative References

- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), DOI 10.17487/RFC3986, January 2005, <<https://www.rfc-editor.org/info/rfc3986>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", [RFC 5246](#), DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC7481] Hollenbeck, S. and N. Kong, "Security Services for the Registration Data Access Protocol (RDAP)", [RFC 7481](#), DOI 10.17487/RFC7481, March 2015, <<https://www.rfc-editor.org/info/rfc7481>>.
- [RFC7482] Newton, A. and S. Hollenbeck, "Registration Data Access Protocol (RDAP) Query Format", [RFC 7482](#), DOI 10.17487/RFC7482, March 2015, <<https://www.rfc-editor.org/info/rfc7482>>.
- [RFC7483] Newton, A. and S. Hollenbeck, "JSON Responses for the Registration Data Access Protocol (RDAP)", [RFC 7483](#), DOI 10.17487/RFC7483, March 2015, <<https://www.rfc-editor.org/info/rfc7483>>.
- [RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", [BCP 205](#), [RFC 7942](#), DOI 10.17487/RFC7942, July 2016, <<https://www.rfc-editor.org/info/rfc7942>>.

Appendix A. Change Log

- 00: Initial version.
- 01: Changed separator character from HYPHEN MINUS to COMMERCIAL AT. Added a recommendation to maintain consistency between service provider tags and "self" links (suggestion received from Tom Harrison). Fixed a spelling error, and corrected the network example in [Section 2](#) (editorial erratum reported for [RFC 7483](#) by Marcos Sanz). Added acknowledgements.
- 02: Changed separator character from COMMERCIAL AT to TILDE. Clarity updates and fixed an example handle. Added text to describe the risk of separator characters appearing naturally in entity handles and being misinterpreted as separator characters.
- 03: Added Implementation Status section ([Section 6](#)).
- 04: Keepalive refresh.
- 05: Added OpenRDAP implementation information to [Section 6](#).

- 00: Initial working group version.
- 01: Added text to describe why the TILDE character was chosen as the separator character.
- 02: Nit fixes. Added rdapConformance text, switched back to HYPHEN MINUS, and added IANA registration instructions per working group last call discussion. Updated suffix syntax reference from the IANA EPPROID registry to [RFC 5730](#) (which is what the IANA registry references).
- 03: Shepherded writeup review updates to explain examples in [Section 2](#).
- 04: AD review update to clarify query path construction.

Authors' Addresses

Scott Hollenbeck
Verisign Labs
12061 Bluemont Way
Reston, VA 20190
USA

Email: shollenbeck@verisign.com
URI: <http://www.verisignlabs.com/>

Andrew Lee Newton
American Registry for Internet Numbers
PO Box 232290
Centreville, VA 20120
US

Email: andy@arin.net
URI: <http://www.arin.net>

