REGEXT Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 7, 2020

S. Hollenbeck
Verisign Labs
A. Newton
AWS

May 6, 2020

Registration Data Access Protocol (RDAP) Query Format
draft-hollenbeck-regext-rfc7482bis-05

Abstract

This document describes uniform patterns to construct HTTP URLs that may be used to retrieve registration information from registries (including both Regional Internet Registries (RIRs) and Domain Name Registries (DNRs)) using "RESTful" web access patterns. These uniform patterns define the query syntax for the Registration Data Access Protocol (RDAP).

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 November 7, 2020.

Copyright Notice

Copyright (c) 2020 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. Conventions Used in This Document 4
2.1. Acronyms and Abbreviations 4
3. Path Segment Specification 5
3.1. Lookup Path Segment Specification 5
3.1.1. IP Network Path Segment Specification 6
3.1.2. Autonomous System Path Segment Specification 7
3.1.3. Domain Path Segment Specification 7
3.1.4. Nameserver Path Segment Specification 8
3.1.5. Entity Path Segment Specification 9
3.1.6. Help Path Segment Specification 9
3.2. Search Path Segment Specification 9
3.2.1. Domain Search 10
3.2.2. Nameserver Search 11
3.2.3. Entity Search 12
4. Query Processing 12
4.1. Partial String Searching 13
4.2. Associated Records 14
5. Extensibility 14
6. Internationalization Considerations 15
6.1. Character Encoding Considerations 15
7. Implementation Status 16
7.1. Viagenie 16
7.2. ARIN 17
7.3. NicInfo 18
7.4. LACNIC 18
7.5. ICANN 19
8. IANA Considerations 20
9. Security Considerations 20
10. References 20
10.1. Normative References 20
10.2. Informative References 23
Acknowledgements 24
Changes from RFC 7482 24
Authors' Addresses 25

1. Introduction

This document describes a specification for querying registration data using a RESTful web service and uniform query patterns. The service is implemented using the Hypertext Transfer Protocol (HTTP) [RFC7230] and the conventions described in [RFC7480]. These uniform

patterns define the query syntax for the Registration Data Access Protocol (RDAP).

The protocol described in this specification is intended to address deficiencies with the WHOIS protocol [RFC3912] that have been identified over time, including:

- lack of standardized command structures;
- lack of standardized output and error structures;
- lack of support for internationalization and localization; and
- lack of support for user identification, authentication, and access control.

The patterns described in this document purposefully do not encompass all of the methods employed in the WHOIS and other RESTful web services used by the RIRs and DNRs. The intent of the patterns described here are to enable queries of:

- networks by IP address;
- Autonomous System (AS) numbers by number;
- reverse DNS metadata by domain;
- nameservers by name; and
- entities (such as registrars and contacts) by identifier.

Server implementations are free to support only a subset of these features depending on local requirements. Servers MUST return an HTTP 501 (Not Implemented) [RFC7231] response to inform clients of unsupported query types. It is also envisioned that each registry will continue to maintain WHOIS and/or other RESTful web services specific to their needs and those of their constituencies, and the information retrieved through the patterns described here may reference such services.

Likewise, future IETF standards may add additional patterns for additional query types. A simple pattern namespacing scheme is described in Section 5 to accommodate custom extensions that will not interfere with the patterns defined in this document or patterns defined in future IETF standards.

WHOIS services, in general, are read-only services. Therefore, URL [RFC3986] patterns specified in this document are only applicable to the HTTP [RFC7231] GET and HEAD methods.

This document does not describe the results or entities returned from issuing the described URLs with an HTTP GET. The specification of these entities is described in [I-D.hollenbeck-regext-rfc7483bis].

Additionally, resource management, provisioning, and update functions are out of scope for this document. Registries have various and divergent methods covering these functions, and it is unlikely a uniform approach is needed for interoperability.

HTTP contains mechanisms for servers to authenticate clients and for clients to authenticate servers (from which authorization schemes may be built), so such mechanisms are not described in this document. Policy, provisioning, and processing of authentication and authorization are out of scope for this document as deployments will have to make choices based on local criteria. Supported authentication mechanisms are described in [RFC7481].

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 [RFC2119].

2.1. Acronyms and Abbreviations

IDN: Internationalized Domain Name, a fully-qualified domain name containing one or more labels that are intended to include one or more Unicode code points outside the ASCII range (cf. "domain name", "fully-qualified domain name" and "internationalized domain name" in RFC 8499 [RFC8499]).

IDNA: Internationalized Domain Names in Applications, a protocol for the handling of IDNs. In this document, "IDNA" refers specifically to the version of those specifications known as "IDNA2008" [RFC5890].

DNR: Domain Name Registry or Domain Name Registrar

NFC: Unicode Normalization Form C [Unicode-UAX15]

NFKC: Unicode Normalization Form KC [Unicode-UAX15]

RDAP: Registration Data Access Protocol

REST: Representational State Transfer. The term was first described in a doctoral dissertation [REST].

RESTful: An adjective that describes a service using HTTP and the principles of REST.

RIR: Regional Internet Registry

3. Path Segment Specification

The base URLs used to construct RDAP queries are maintained in an IANA registry described in [RFC7484]. Queries are formed by retrieving an appropriate base URL from the registry and appending a path segment specified in either Sections 3.1 or 3.2. Generally, a registry or other service provider will provide a base URL that identifies the protocol, host, and port, and this will be used as a base URL that the complete URL is resolved against, as per Section 5 of RFC 3986 [RFC3986]. For example, if the base URL is "https://example.com/rdap/", all RDAP query URLs will begin with "https://example.com/rdap/".

The bootstrap registry does not contain information for query objects that are not part of a global namespace, including entities and help. A base URL for an associated object is required to construct a complete query. This limitation can be overcome for entities by using the practice described in RFC 8521 [RFC8521].

For entities, a base URL is retrieved for the service (domain, address, etc.) associated with a given entity. The query URL is constructed by concatenating the base URL to the entity path segment specified in either Sections 3.1.5 or 3.2.3.

For help, a base URL is retrieved for any service (domain, address, etc.) for which additional information is required. The query URL is constructed by concatenating the base URL to the help path segment specified in Section 3.1.6.

3.1. Lookup Path Segment Specification

A simple lookup to determine if an object exists (or not) without returning RDAP-encoded results can be performed using the HTTP HEAD method as described in Section 4.1 of [RFC7480].

The resource type path segments for exact match lookup are:

- 'ip': Used to identify IP networks and associated data referenced using either an IPv4 or IPv6 address.

- 'autnum': Used to identify Autonomous System number registrations and associated data referenced using an asplain Autonomous System number.

- 'domain': Used to identify reverse DNS (RIR) or domain name (DNR) information and associated data referenced using a fully qualified domain name.

- 'nameserver': Used to identify a nameserver information query using a host name.

- 'entity': Used to identify an entity information query using a string identifier.

3.1.1. IP Network Path Segment Specification

Syntax: ip/<IP address> or ip/<CIDR prefix>/<CIDR length>

Queries for information about IP networks are of the form /ip/XXX or /ip/XXX/YY where the path segment following 'ip' is either an IPv4 dotted decimal or IPv6 [RFC5952] address (i.e., XXX) or an IPv4 or IPv6 Classless Inter-domain Routing (CIDR) [RFC4632] notation address block (i.e., XXX/YY). Semantically, the simpler form using the address can be thought of as a CIDR block with a bitmask length of 32 for IPv4 and a bitmask length of 128 for IPv6. A given specific address or CIDR may fall within multiple IP networks in a hierarchy of networks; therefore, this query targets the "most-specific" or smallest IP network that completely encompasses it in a hierarchy of IP networks.

The IPv4 and IPv6 address formats supported in this query are described in Section 3.2.2 of RFC 3986 [RFC3986] as IPv4address and IPv6address ABNF definitions. Any valid IPv6 text address format [RFC4291] can be used. This includes IPv6 addresses written using with or without compressed zeros and IPv6 addresses containing embedded IPv4 addresses. The rules to write a text representation of an IPv6 address [RFC5952] are RECOMMENDED. However, the zone_id [RFC4007] is not appropriate in this context; therefore, the corresponding syntax extension in RFC 6874 [RFC6874] MUST NOT be used, and servers are to ignore it if possible.

For example, the following URL would be used to find information for the most specific network containing 192.0.2.0:

https://example.com/rdap/ip/192.0.2.0

The following URL would be used to find information for the most specific network containing 192.0.2.0/24:

https://example.com/rdap/ip/192.0.2.0/24

The following URL would be used to find information for the most specific network containing 2001:db8::0:

https://example.com/rdap/ip/2001:db8::0

3.1.2. Autonomous System Path Segment Specification

Syntax: autnum/<autonomous system number>

Queries for information regarding Autonomous System number registrations are of the form /autnum/XXX where XXX is an asplain Autonomous System number [RFC5396]. In some registries, registration of Autonomous System numbers is done on an individual number basis, while other registries may register blocks of Autonomous System numbers. The semantics of this query are such that if a number falls within a range of registered blocks, the target of the query is the block registration and that individual number registrations are considered a block of numbers with a size of 1.

For example, the following URL would be used to find information describing Autonomous System number 12 (a number within a range of registered blocks):

https://example.com/rdap/autnum/12

The following URL would be used to find information describing 4-byte Autonomous System number 65538:

https://example.com/rdap/autnum/65538

3.1.3. Domain Path Segment Specification

Syntax: domain/<domain name>

Queries for domain information are of the form /domain/XXXX, where XXXX is a fully qualified (relative to the root) domain name (as specified in [RFC0952] and [RFC1123]) in either the in-addr.arpa or ip6.arpa zones (for RIRs) or a fully qualified domain name in a zone administered by the server operator (for DNRs). Internationalized Domain Names (IDNs) represented in either A-label or U-label format [RFC5890] are also valid domain names. See Section 6.1 for information on character encoding for the U-label format.

IDNs SHOULD NOT be represented as a mixture of A-labels and U-labels; that is, internationalized labels in an IDN SHOULD be either all A-labels or all U-labels. It is possible for an RDAP client to

assemble a query string from multiple independent data sources. Such a client might not be able to perform conversions between A-labels and U-labels. An RDAP server that receives a query string with a mixture of A-labels and U-labels MAY convert all the U-labels to A-labels, perform IDNA processing, and proceed with exact-match lookup. In such cases, the response to be returned to the query source may not match the input from the query source. Alternatively, the server MAY refuse to process the query.

The server MAY perform the match using either the A-label or U-label form. Using one consistent form for matching every label is likely to be more reliable.

The following URL would be used to find information describing the zone serving the network 192.0.2/24:

https://example.com/rdap/domain/2.0.192.in-addr.arpa

The following URL would be used to find information describing the zone serving the network 2001:db8:1::/48:

https://example.com/rdap/domain/1.0.0.0.8.b.d.0.1.0.0.2.ip6.arpa

The following URL would be used to find information for the blah.example.com domain name:

https://example.com/rdap/domain/blah.example.com

The following URL would be used to find information for the xn--fo-5ja.example IDN:

https://example.com/rdap/domain/xn--fo-5ja.example

3.1.4. Nameserver Path Segment Specification

Syntax: nameserver/<nameserver name>

The <nameserver name> parameter represents a fully qualified host name as specified in [RFC0952] and [RFC1123]. Internationalized names represented in either A-label or U-label format [RFC5890] are also valid nameserver names. IDN processing for nameserver names uses the domain name processing instructions specified in Section 3.1.3. See Section 6.1 for information on character encoding for the U-label format.

The following URL would be used to find information for the ns1.example.com nameserver:

https://example.com/rdap/nameserver/ns1.example.com

The following URL would be used to find information for the ns1.xn--fo-5ja.example nameserver:

https://example.com/rdap/nameserver/ns1.xn--fo-5ja.example

3.1.5. Entity Path Segment Specification

Syntax: entity/<handle>

The <handle> parameter represents an entity (such as a contact, registrant, or registrar) identifier whose syntax is specific to the registration provider. For example, for some DNRs, contact identifiers are specified in [RFC5730] and [RFC5733].

The following URL would be used to find information for the entity associated with handle XXXX:

https://example.com/rdap/entity/XXXX

3.1.6. Help Path Segment Specification

Syntax: help

The help path segment can be used to request helpful information (command syntax, terms of service, privacy policy, rate-limiting policy, supported authentication methods, supported extensions, technical support contact, etc.) from an RDAP server. The response to "help" should provide basic information that a client needs to successfully use the service. The following URL would be used to return "help" information:

https://example.com/rdap/help

3.2. Search Path Segment Specification

Pattern matching semantics are described in Section 4.1. The resource type path segments for search are:

- 'domains': Used to identify a domain name information search using a pattern to match a fully qualified domain name.

- 'nameservers': Used to identify a nameserver information search using a pattern to match a host name.

- 'entities': Used to identify an entity information search using a pattern to match a string identifier.

RDAP search path segments are formed using a concatenation of the plural form of the object being searched for and an HTTP query string. The HTTP query string is formed using a concatenation of the question mark character ('?', US-ASCII value 0x003F), a noun representing the JSON object property associated with the object being searched for, the equal sign character ('=', US-ASCII value 0x003D), and the search pattern. Search pattern query processing is described more fully in Section 4. For the domain, nameserver, and entity objects described in this document, the plural object forms are "domains", "nameservers", and "entities".

Detailed results can be retrieved using the HTTP GET method and the path segments specified here.

3.2.1. Domain Search

Syntax: domains?name=<domain search pattern>

Syntax: domains?nsLdhName=<nameserver search pattern>

Syntax: domains?nsIp=<IP address search pattern>

Searches for domain information by name are specified using this form:

domains?name=XXXX

XXXX is a search pattern representing a domain name in "letters, digits, hyphen" (LDH) format [RFC5890]. The following URL would be used to find DNR information for domain names matching the "example*.com" pattern:

https://example.com/rdap/domains?name=example*.com

IDNs in U-label format [RFC5890] can also be used as search patterns (see Section 4). Searches for these names are of the form /domains?name=XXXX, where XXXX is a search pattern representing a domain name in U-label format [RFC5890]. See Section 6.1 for information on character encoding for the U-label format.

Searches for domain information by nameserver name are specified using this form:

domains?nsLdhName=YYYY

YYYY is a search pattern representing a host name in "letters, digits, hyphen" format [RFC5890]. The following URL would be used to

search for domains delegated to nameservers matching the "ns1.example*.com" pattern:

https://example.com/rdap/domains?nsLdhName=ns1.example*.com

Searches for domain information by nameserver IP address are specified using this form:

domains?nsIp=ZZZZ

ZZZZ is a search pattern representing an IPv4 [RFC1166] or IPv6 [RFC5952] address. The following URL would be used to search for domains that have been delegated to nameservers that resolve to the "192.0.2.0" address:

https://example.com/rdap/domains?nsIp=192.0.2.0

3.2.2. Nameserver Search

Syntax: nameservers?name=<nameserver search pattern>

Syntax: nameservers?ip=<IP address search pattern>

Searches for nameserver information by nameserver name are specified using this form:

nameservers?name=XXXX

XXXX is a search pattern representing a host name in "letters, digits, hyphen" format [RFC5890]. The following URL would be used to find information for nameserver names matching the "ns1.example*.com" pattern:

https://example.com/rdap/nameservers?name=ns1.example*.com

Internationalized nameserver names in U-label format [RFC5890] can also be used as search patterns (see Section 4). Searches for these names are of the form /nameservers?name=XXXX, where XXXX is a search pattern representing a nameserver name in U-label format [RFC5890]. See Section 6.1 for information on character encoding for the U-label format.

Searches for nameserver information by nameserver IP address are specified using this form:

nameservers?ip=YYYY

YYYY is a search pattern representing an IPv4 [RFC1166] or IPv6 [RFC5952] address. The following URL would be used to search for nameserver names that resolve to the "192.0.2.0" address:

https://example.com/rdap/nameservers?ip=192.0.2.0

3.2.3. Entity Search

Syntax: entities?fn=<entity name search pattern>

Syntax: entities?handle=<entity handle search pattern>

Searches for entity information by name are specified using this form:

entities?fn=XXXX

XXXX is a search pattern representing the "FN" property of an entity (such as a contact, registrant, or registrar) name as specified in Section 5.1 of [I-D.hollenbeck-regext-rfc7483bis]. The following URL would be used to find information for entity names matching the "Bobby Joe*" pattern:

https://example.com/rdap/entities?fn=Bobby%20Joe*

Searches for entity information by handle are specified using this form:

entities?handle=XXXX

XXXX is a search pattern representing an entity (such as a contact, registrant, or registrar) identifier whose syntax is specific to the registration provider. The following URL would be used to find information for entity handles matching the "CID-40*" pattern:

https://example.com/rdap/entities?handle=CID-40*

URLs MUST be properly encoded according to the rules of [RFC3986]. In the example above, "Bobby Joe*" is encoded to "Bobby%20Joe*".

4. Query Processing

Servers indicate the success or failure of query processing by returning an appropriate HTTP response code to the client. Response codes not specifically identified in this document are described in [RFC7480].

4.1. Partial String Searching

Partial string searching uses the asterisk ('*', US-ASCII value 0x002A) character to match zero or more trailing characters. A character string representing a domain label suffix MAY be concatenated to the end of the search pattern to limit the scope of the search. For example, the search pattern "exam*" will match "example.com" and "example.net". The search pattern "exam*.com" will match "example.com". If an asterisk appears in a search string, any label that contains the non-asterisk characters in sequence plus zero or more characters in sequence in place of the asterisk would match. Only a single asterisk is allowed for a partial string search. Additional pattern matching processing is beyond the scope of this specification.

If a server receives a search request but cannot process the request because it does not support a particular style of partial match searching, it SHOULD return an HTTP 422 (Unprocessable Entity) [RFC4918] response. When returning a 422 error, the server MAY also return an error response body as specified in Section 6 of [I-D.hollenbeck-regext-rfc7483bis] if the requested media type is one that is specified in [RFC7480].

Partial matching is not feasible across combinations of Unicode characters because Unicode characters can be combined with each other. Servers SHOULD NOT partially match combinations of Unicode characters where a legal combination is possible. It should be noted, though, that it may not always be possible to detect cases where a character could have been combined with another character, but was not, because characters can be combined in many different ways.

Clients should avoid submitting a partial match search of Unicode characters where a Unicode character may be legally combined with another Unicode character or characters. Partial match searches with incomplete combinations of characters where a character must be combined with another character or characters are invalid. Partial match searches with characters that may be combined with another character or characters are to be considered non-combined characters (that is, if character x may be combined with character y but character y is not submitted in the search string, then character x is a complete character and no combinations of character x are to be searched).

4.2. Associated Records

Conceptually, any query-matching record in a server's database might be a member of a set of related records, related in some fashion as defined by the server -- for example, variants of an IDN. The entire set ought to be considered as candidates for inclusion when constructing the response. However, the construction of the final response needs to be mindful of privacy and other data-releasing policies when assembling the RDAP response set.

Note too that due to the nature of searching, there may be a list of query-matching records. Each one of those is subject to being a member of a set as described in the previous paragraph. What is ultimately returned in a response will be the union of all the sets that has been filtered by whatever policies are in place.

Note that this model includes arrangements for associated names, including those that are linked by policy mechanisms and names bound together for some other purposes. Note also that returning information that was not explicitly selected by an exact-match lookup, including additional names that match a relatively fuzzy search as well as lists of names that are linked together, may cause privacy issues.

Note that there might not be a single, static information return policy that applies to all clients equally. Client identity and associated authorizations can be a relevant factor in determining how broad the response set will be for any particular query.

5. Extensibility

This document describes path segment specifications for a limited number of objects commonly registered in both RIRs and DNRs. It does not attempt to describe path segments for all of the objects registered in all registries. Custom path segments can be created for objects not specified here using the process described in Section 6 of "HTTP Usage in the Registration Data Access Protocol (RDAP)" [RFC7480].

Custom path segments can be created by prefixing the segment with a unique identifier followed by an underscore character (0x5F). For example, a custom entity path segment could be created by prefixing "entity" with "custom_", producing "custom_entity". Servers MUST return an appropriate failure status code for a request with an unrecognized path segment.

6. Internationalization Considerations

There is value in supporting the ability to submit either a U-label (Unicode form of an IDN label) or an A-label (US-ASCII form of an IDN label) as a query argument to an RDAP service. Clients capable of processing non-US-ASCII characters may prefer a U-label since this is more visually recognizable and familiar than A-label strings, but clients using programmatic interfaces might find it easier to submit and display A-labels if they are unable to input U-labels with their keyboard configuration. Both query forms are acceptable.

Internationalized domain and nameserver names can contain character variants and variant labels as described in [RFC4290]. Clients that support queries for internationalized domain and nameserver names MUST accept service provider responses that describe variants as specified in "JSON Responses for the Registration Data Access Protocol (RDAP)" [I-D.hollenbeck-regext-rfc7483bis].

6.1. Character Encoding Considerations

Servers can expect to receive search patterns from clients that contain character strings encoded in different forms supported by HTTP. It is entirely possible to apply filters and normalization rules to search patterns prior to making character comparisons, but this type of processing is more typically needed to determine the validity of registered strings than to match patterns.

An RDAP client submitting a query string containing non-US-ASCII characters converts such strings into Unicode in UTF-8 encoding. It then performs any local case mapping deemed necessary. Strings are normalized using Normalization Form C (NFC) [Unicode-UAX15]; note that clients might not be able to do this reliably. UTF-8 encoded strings are then appropriately percent-encoded [RFC3986] in the query URL.

After parsing any percent-encoding, an RDAP server treats each query string as Unicode in UTF-8 encoding. If a string is not valid UTF-8, the server can immediately stop processing the query and return an HTTP 400 (Bad Request) response.

When processing queries, there is a difference in handling DNS names, including those with putative U-labels, and everything else. DNS names are treated according to the DNS matching rules as described in Section 3.1 of RFC 1035 [RFC1035] for Non-Reserved LDH (NR-LDH) labels and the matching rules described in Section 5.4 of RFC 5891 [RFC5891] for U-labels. Matching of DNS names proceeds one label at a time because it is possible for a combination of U-labels and NR-LDH labels to be found in a single domain or host name. The

determination of whether a label is a U-label or an NR-LDH label is based on whether the label contains any characters outside of the US-ASCII letters, digits, or hyphen (the so-called LDH rule).

For everything else, servers map fullwidth and halfwidth characters to their decomposition equivalents. Servers convert strings to the same coded character set of the target data that is to be looked up or searched, and each string is normalized using the same normalization that was used on the target data. In general, storage of strings as Unicode is RECOMMENDED. For the purposes of comparison, Normalization Form KC (NFKC) [Unicode-UAX15] with case folding is used to maximize predictability and the number of matches. Note the use of case-folded NFKC as opposed to NFC in this case.

7. 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".

7.1. Viagenie

Responsible Organization: Viagenie

Location: RDAPBrowser (iOS and Android): https://viagenie.ca/rdapbrowser

Description: Mobile app (iOS and Android) implementing an RDAP client for domains, IP addresses and AS numbers.

Level of Maturity: Production

Coverage: All except for nameserver, entity, help, and search path segments.

Version Compatibility: RFC 7482

Licensing: Proprietary

Implementation Experience: Quite simple and easy to deploy. Responses are much harder to parse because RDAP servers are not compliant.

Contact Information: Marc Blanchet, rdapbrowser@viagenie.ca

Date Last Updated: September 27, 2019

7.2. ARIN

Responsible Organization: ARIN

Location: search.arin.net https://search.arin.net/rdap/

Description: search.arin.net is a public web page getting about 8k queries per day.

Level of Maturity: Production.

Coverage: Search.arin.net supports lookup of entities by handle, search of entities by name, lookup of domain names, lookup of ip networks, lookup of autnums.

Version Compatibility: RFC 7482

Licensing: Search.arin.net is not publicly licensed.

Implementation Experience: The RDAP queries are straightforward for the most part. The vast majority of logic goes into displaying information.

Contact Information: info@arin.net

Date Last Updated: July 2019.

7.3. NicInfo

Responsible Organization: ARIN

Location: NicInfo https://github.com/arineng/nicinfo

Description: NicInfo is a command line client written in Ruby.

Level of Maturity: NicInfo started as a research project, but is known to be used by some organizations in a production capacity.

Version Compatibility: RFC 7482

Licensing: NicInfo is published under the ISC license.

Implementation Experience: The RDAP queries are straightforward for the most part. The vast majority of logic goes into displaying information.

Contact Information: info@arin.net

Date Last Updated: NicInfo was last updated in Feb 2018.

7.4. LACNIC

Responsible Organization: LACNIC

Location: https://github.com/LACNIC/rdap-frontend-angular-dev

Description: The goal of this client is to have an RDAP client that can be easily embedded in web pages. The original request was for a web whois/rdap feature that was to replace a very, very old web whois that just popen'd CLI WHOIS and just copied back the output to html. We decided to implement something that could, in the future, be embedded in any web page and is not tied to our current web portal CMS. The client is implemented in Javascript and AngularJS.

Level of Maturity: We consider the current version production quality, it has been in use in our web portal for more than a year now.

Coverage: The client implements /ip, /autnum, and /entity. The client does not support searches. For these objects the implementation follows the standard closely. There may be a few gaps, but it's mostly aligned to the RFCs.

Version Compatibility: RFC 7482

Licensing: BSD-Style

Implementation Experience: Users of the traditional WHOIS service are a bit confused at first when they realize that an RDAP query does not necessarily return the same information and in some cases they need to "navigate" the RDAP tree to get data that is normally returned in a single WHOIS query. In our experience, this gap in expectations has been one of the most significant hurdles in adoption of RDAP. Our RDAP client makes this "navigation" easier as it presents results in the form of a web page where the "next" necessary RDAP query is a click on a link. On the plus side, the protocol provides all the information needed to present this links and clicks to the user. We have however introduced a few extensions into our RDAP responses to get both services to parity in the information presented in a single query.

Contact Information: Gerardo Rada (gerardo@lacnic.net), Carlos Martinez (carlos@lacnic.net)

Date Last Updated: This application is currently in maintenance mode. Also, we employ a rolling release update. Latest updates are available in the git log of the repo.

7.5. ICANN

Responsible Organization: Internet Corporation for Assigned Names and Numbers (ICANN)

Location: Domain Name Registration Data Lookup: https://lookup.icann.org/

Description: ICANN created the Domain Name Registration Data Lookup web client as a free public service that gives users the ability to look up and display publicly available registration data related to a domain name using the top level domain's RDAP service location listed in the IANA bootstrap service registry for domain name space (RFC 7484), and the sponsoring Registrar's RDAP server. This web client implementation also supports the specifications defined in the "gTLD RDAP Profile" documents (https://www.icann.org/gtld-rdap-profile).

Level of Maturity: Production.

Coverage: This web client implements RFC 7482 section 3.1.3 "Domain Path Segment Specification" to perform lookups exclusively for the domain object class.

Version Compatibility: RFC 7482

Contact Information: globalSupport@icann.org

Date Last Updated: 07-Oct-2019

8. IANA Considerations

This document has no actions for IANA.

9. Security Considerations

Security services for the operations specified in this document are described in "Security Services for the Registration Data Access Protocol (RDAP)" [RFC7481].

Search functionality typically requires more server resources (such as memory, CPU cycles, and network bandwidth) when compared to basic lookup functionality. This increases the risk of server resource exhaustion and subsequent denial of service due to abuse. This risk can be mitigated by developing and implementing controls to restrict search functionality to identified and authorized clients. If those clients behave badly, their search privileges can be suspended or revoked. Rate limiting as described in Section 5.5 of "HTTP Usage in the Registration Data Access Protocol (RDAP)" [RFC7480] can also be used to control the rate of received search requests. Server operators can also reduce their risk by restricting the amount of information returned in response to a search request.

Search functionality also increases the privacy risk of disclosing object relationships that might not otherwise be obvious. For example, a search that returns IDN variants [RFC6927] that do not explicitly match a client-provided search pattern can disclose information about registered domain names that might not be otherwise available. Implementers need to consider the policy and privacy implications of returning information that was not explicitly requested.

Note that there might not be a single, static information return policy that applies to all clients equally. Client identity and associated authorizations can be a relevant factor in determining how broad the response set will be for any particular query.

10. References

10.1. Normative References

[RFC0952] Harrenstien, K., Stahl, M., and E. Feinler, "DoD Internet host table specification", RFC 952, DOI 10.17487/RFC0952, October 1985, <https://www.rfc-editor.org/info/rfc952>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <https://www.rfc-editor.org/info/rfc1123>.

[RFC1166] Kirkpatrick, S., Stahl, M., and M. Recker, "Internet numbers", RFC 1166, DOI 10.17487/RFC1166, July 1990, <https://www.rfc-editor.org/info/rfc1166>.

[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>.

[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>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, DOI 10.17487/RFC4632, August 2006, <https://www.rfc-editor.org/info/rfc4632>.

[RFC4918] Dusseault, L., Ed., "HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)", RFC 4918, DOI 10.17487/RFC4918, June 2007, <https://www.rfc-editor.org/info/rfc4918>.

[RFC5396] Huston, G. and G. Michaelson, "Textual Representation of Autonomous System (AS) Numbers", RFC 5396, DOI 10.17487/RFC5396, December 2008, <https://www.rfc-editor.org/info/rfc5396>.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.17487/RFC5730, August 2009, <https://www.rfc-editor.org/info/rfc5730>.

[RFC5733] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Contact Mapping", STD 69, RFC 5733, DOI 10.17487/RFC5733, August 2009, <https://www.rfc-editor.org/info/rfc5733>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <https://www.rfc-editor.org/info/rfc5890>.

[RFC5891] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, DOI 10.17487/RFC5891, August 2010, <https://www.rfc-editor.org/info/rfc5891>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/info/rfc5952>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://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, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC7480] Newton, A., Ellacott, B., and N. Kong, "HTTP Usage in the Registration Data Access Protocol (RDAP)", RFC 7480, DOI 10.17487/RFC7480, March 2015, <https://www.rfc-editor.org/info/rfc7480>.

[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>.

[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>.

[RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <https://www.rfc-editor.org/info/rfc8499>.

[I-D.hollenbeck-regext-rfc7483bis]
Hollenbeck, S. and A. Newton, "JSON Responses for the Registration Data Access Protocol (RDAP)", draft-hollenbeck-regext-rfc7483bis-00 (work in progress), February 2020.

[Unicode-UAX15]
The Unicode Consortium, "Unicode Standard Annex #15: Unicode Normalization Forms", September 2013, <https://www.unicode.org/reports/tr15/>.

10.2. Informative References

[REST] Fielding, R., "Architectural Styles and the Design of Network-based Software Architectures", Ph.D. Dissertation, University of California, Irvine, 2000, <https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf>.

[RFC3912] Daigle, L., "WHOIS Protocol Specification", RFC 3912, DOI 10.17487/RFC3912, September 2004, <https://www.rfc-editor.org/info/rfc3912>.

[RFC4007] Deering, S., Haberman, B., Jinmei, T., Nordmark, E., and B. Zill, "IPv6 Scoped Address Architecture", RFC 4007, DOI 10.17487/RFC4007, March 2005, <https://www.rfc-editor.org/info/rfc4007>.

[RFC4290] Klensin, J., "Suggested Practices for Registration of Internationalized Domain Names (IDN)", RFC 4290, DOI 10.17487/RFC4290, December 2005, <https://www.rfc-editor.org/info/rfc4290>.

[RFC6874] Carpenter, B., Cheshire, S., and R. Hinden, "Representing IPv6 Zone Identifiers in Address Literals and Uniform Resource Identifiers", RFC 6874, DOI 10.17487/RFC6874, February 2013, <https://www.rfc-editor.org/info/rfc6874>.

[RFC6927] Levine, J. and P. Hoffman, "Variants in Second-Level Names Registered in Top-Level Domains", RFC 6927, DOI 10.17487/RFC6927, May 2013, <https://www.rfc-editor.org/info/rfc6927>.

[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>.

[RFC8521] Hollenbeck, S. and A. Newton, "Registration Data Access Protocol (RDAP) Object Tagging", BCP 221, RFC 8521, DOI 10.17487/RFC8521, November 2018, <https://www.rfc-editor.org/info/rfc8521>.

Acknowledgements

This document is derived from original work on RIR query formats developed by Byron J. Ellacott of APNIC, Arturo L. Servin of LACNIC, Kaveh Ranjbar of the RIPE NCC, and Andrew L. Newton of ARIN. Additionally, this document incorporates DNR query formats originally described by Francisco Arias and Steve Sheng of ICANN and Scott Hollenbeck of Verisign Labs.

The authors would like to acknowledge the following individuals for their contributions to this document: Francisco Arias, Marc Blanchet, Ernie Dainow, Jean-Philippe Dionne, Byron J. Ellacott, Behnam Esfahbod, John Klensin, John Levine, Edward Lewis, Mark Nottingham, Kaveh Ranjbar, Arturo L. Servin, Steve Sheng, and Andrew Sullivan.

Changes from RFC 7482

00: Initial version ported from RFC 7482. Added Implementation Status section. Addressed known errata.

01: Addressed other reported clarifications and corrections: IDN/IDNA definition, note that registrars are entities, definition of "DNR", RFC 8521 to address bootstrap registry limitation, removal of extraneous "...", HTTP query string clarification, search pattern clarification, name server search clarification, domain label suffix and asterisk search clarification.

02: Addressed "The HTTP query string" clarification.

03: Modified co-author address.

04: Updated references to 7483 to 7483bis Internet-Draft. Updated "Change Log" to "Changes from RFC 7482". Added more detail to the changes made in the -01 version.

05: Added an empty IANA Considerations section to satisfy IDNits. Changed references to use HTTPS for targets. Split ARIN and NicInfo implementation status into two sections.

Authors' Addresses

Scott Hollenbeck
Verisign Labs
12061 Bluemont Way
Reston, VA 20190
United States of America

Email: shollenbeck@verisign.com
URI: https://www.verisignlabs.com/

Andy Newton
Amazon Web Services, Inc.
13200 Woodland Park Road
Herndon, VA 20171
United States of America

Email: andy@hxr.us