Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 30, 2014

M. Blanchet
G. Guillaume
Viagenie

June 28, 2014

Finding the Authoritative Registration Data (RDAP) Service
draft-ietf-weirds-bootstrap-03.txt

Abstract

This document specifies a method to find which Registration Data Access Protocol (RDAP) server is authoritative to answer queries for a requested scope, such as domain names, IP addresses or Autonomous System numbers.

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 http://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 December 30, 2014.

Copyright Notice

Table of Contents

1. Introduction 2
2. Conventions Used In This Document 2
3. Structure of RDAP Bootstrap Registries 3
4. Domain Name RDAP Bootstrap Registry 3
5. Internet Numbers RDAP Bootstrap Registries 4
5.1. IPv4 Address Space RDAP Bootstrap Registry 5
5.2. IPv6 Address Space RDAP Registry 6
5.3. Autonomous Systems RDAP Bootstrap Registry 6
6. Entity 7
7. Non-existent Entries or RDAP URL Values 7
8. Deployment and Implementation Considerations 8
9. Limitations 8
10. Security Considerations 9
11. IANA Considerations 9
12. Acknowledgements 9
13. References 10
13.1. Normative References 10
13.2. Non-Normative References 10
Authors' Addresses 11

1. Introduction

Querying and retrieving registration data from registries are defined in the Registration Data Access Protocol(RDAP)[I-D.ietf-weirds-rdap-query][I-D.ietf-weirds-using-http][I-D.ietf-weirds-json-response]. These documents do not specify where to send the queries. This document specifies a method to find which server is authoritative to answer queries for the requested scope.

The proposed mechanism is based on that allocation data for domain names and IP addresses are maintained by IANA, are publicly available and are in a structured format. The mechanism assumes some data structure within these registries and request IANA to create these registries for the specific purpose of RDAP use, herein named RDAP Bootstrap registries. An RDAP client fetches the RDAP bootstrap registries, extract the data and then do a match with the query data to find the authoritative registration data server and appropriate query base URL.

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].

3. Structure of RDAP Bootstrap Registries

The RDAP Bootstrap Registries are implemented as JSON [RFC7159] objects. A registry starts with metadata such as a version id identified as a timestamp of the publication date of the registry and some defaults values. Then follows an array of arrays. Each second level array lists all the entries available by the same template method. There is no assumption of sorting at the first or second level arrays. An example structure of a JSON RDAP Bootstrap Registry is illustrated:

```
{
"rdap.bootstrap": {
 "version": "1.0",
 "publication": "YYYY-MM-DDTHH:MM:SSZ",
 "scheme": [ https http ],

 "services": [
 ["entry1", "entry2", "entry3"]: {
    "template": "{proto}://registry.example.com/myrdap/{resource}",
    "proto": [ https ],
   },
 ["entry4"]: {
    "template": "{proto}://example.org/{resource}",
   },
 ],
}
}
```

The version corresponds to the format version of the registry. This specification defines "1.0". The syntax of "publication" value conforms to the Internet date/time format [RFC3339]. The "proto" object is an array of transport protocols used to access the resource. The RDAP bootstrap client SHOULD try the transport protocols in the order they are presented in the array. The "proto" object can be overriden in the specific entries. Per [RFC7258], the secure version of the transport protocol SHOULD be first.

Any unknown or unspecified JSON object properties or values should be ignored by implementers.

4. Domain Name RDAP Bootstrap Registry

This registry contains domain labels entries attached to the root, grouped by templates, as shown in this example.

```
{
"rdap.bootstrap": {
 "version": "1.0",
 "publication": "YYYY-MM-DDTHH:MM:SSZ",
 "proto": [ "https", "http" ],

 "services": [
 ["net", "com"]: {
    "template": "https://registry.example.com/myrdap/{resource}",
  },
 ["org", "mytld"]: {
    "template": "{proto}://example.org/{resource}",
  },
 ],
 ["mytld2"]: {
    "template": "{proto}://example.net/rdapmytld2/{resource}",
    "proto": [ "http", "https"],
  },
 ],
}
}
```

The domain names authoritative registration data service is found by doing the longest match of the target domain name with the domain values in the arrays in the IANA Domain Name RDAP Bootstrap Registry. This is a string search of the longest match starting from the end of the target name and the end of each value in the arrays. The value of the corresponding "template" object is the base RDAP URL as described in [I-D.ietf-weirds-rdap-query].

For example, a domain RDAP query for a.b.example.com matches the com entry in one of the arrays of the registry. Following the example above, the base RDAP URL for this query is "https://registry.example.com/myrdap/". The {resource} specified in [I-D.ietf-weirds-rdap-query] is then appended to the base URL to complete the query. The complete query is then "https://registry.example.com/myrdap/domain/a.b.example.com". This example is not normative.

5. Internet Numbers RDAP Bootstrap Registries

This section discusses IPv4 and IPv6 address space and autonomous system numbers.

For IP address space, the authoritative registration data service is found by doing a longest match of the target address with the values of the arrays in the corresponding Address Space RDAP Bootstrap registry. The longest match is done the same way as for routing: the

addresses are converted in binary form and then the binary strings are compared to find the longest match. The value of the template object is the base RDAP url as described in [I-D.ietf-weirds-rdap-query]. The longest match method enables covering prefixes of a larger address space pointing to one RDAP template while more specific prefixes within the covering prefix being served by another RDAP template.

5.1. IPv4 Address Space RDAP Bootstrap Registry

This registry contains IPv4 prefix entries, specified in CIDR format and grouped by templates, as shown in this example.

```
{
"rdap.bootstrap": {
 "version": "1.0",
 "publication": "YYYY-MM-DDTHH:MM:SSZ",
 "proto": [ "https", "http" ],

 "services": [
 ["1.0.0.0/8", "192.0.0.0/8"]: {
    "template": "https://rir1.example.com/myrdap/{resource}",
   },
 ["28.2.0.0/16", "192.0.2.0/24"]: {
    "template": "{proto}://example.org/{resource}",
   },
 ],
 ["28.3.0.0/16"]: {
    "template": "{proto}://example.net/rdaprir2/{resource}",
    "proto": [ "http", "https"],
   },
 ],
}
}
```

For example, a query for "192.0.2.0/24" matches the "192.0.0.0/8" entry and the "192.0.2.0/24" entry in the example registry above. The latter is chosen by the client given the longest match. The base RDAP URL for this query is then taken from the template object and expands to "{proto}://example.org/". The {resource} specified in [I-D.ietf-weirds-rdap-query] is then appended to the base URL to complete the query. The complete query is then "https://example.org/ip/192.0.2.0/24". This example is not normative.

5.2. IPv6 Address Space RDAP Registry

This registry contains IPv6 prefix entries, using [RFC4291] text representation of address prefixes format, grouped by templates, as shown in this example.

```
{
"rdap.bootstrap": {
 "version": "1.0",
 "publication": "YYYY-MM-DDTHH:MM:SSZ",
 "proto": [ "https", "http" ],

 "services": [
 ["2001:0200::/23", "2001:db8::/32"]: {
    "template": "https://rir2.example.com/myrdap/{resource}",
   },
 ["2600::/16", "2100:ffff::/32"]: {
    "template": "{proto}://example.org/{resource}",
   },
 ["2001:0200:1000::/28"]: {
    "template": "{proto}://example.net/rdaprir2/{resource}",
    "proto": [ "http", "https"],
   },
 ],
}
}
```

For example, a query for "2001:0200:1000::/48" matches the "2001:0200::/23" entry and the "2001:0200:1000::/28" entry in the example registry above. The latter is chosen by the client given the longest match. The base RDAP URL for this query is then taken from the template object "{proto}://example.net/rdaprir2/". The {resource} specified in [I-D.ietf-weirds-rdap-query] is then appended to the base URL to complete the query. The complete query is therefore "https://example.net/rdaprir2/ip/2001:0200:1000::/48". This example is not normative.

5.3. Autonomous Systems RDAP Bootstrap Registry

This registry contains Autonomous Systems Number Ranges entries, grouped by templates, as shown in this example. Entries in the arrays are either single AS numbers or ranges of AS numbers where the lower appears first, then the "-" separator and then the upper number. Both 16bit and 32 bit AS numbers are specified in decimal.

```
{
"rdap.bootstrap": {
 "version": "1.0",
 "publication": "YYYY-MM-DDTHH:MM:SSZ",
 "proto": [ "https", "http" ],

 "services": [
 ["2045", "20116-20117"]: {
    "template": "https://rir2.example.com/myrdap/{resource}",
   },
 ["10000-12000", "65900-66000"]: {
    "template": "{proto}://example.org/{resource}",
   },
 ["65512-65534"]: {
    "template": "{proto}://example.net/rdaprir2/{resource}",
    "proto": [ "http", "https"],
   },
 ],
}
}
```

For example, a query for AS 65411 matches the "64512-65534" entry in the example registry above. The base RDAP URL for this query is then taken from the template object "{proto}://example.net/rdaprir2/". The {resource} specified in [I-D.ietf-weirds-rdap-query] is then appended to the base URL to complete the query. The complete query is therefore "https://example.net/rdaprir2/autnum/65411". This example is not normative.

6. Entity

Since there is no global namespace for entities, this document does not describe how to find the authoritative RDAP server for entities. It is possible however that, if the entity identifier was received from a previous query, the same RDAP server could be queried for that entity or the entity identifier itself is a fully referenced URL that can be queried.

7. Non-existent Entries or RDAP URL Values

The registries may not contain the requested value or the RDAP URL value may be empty. In these cases, there is no known RDAP server for that requested value and the client SHOULD provide an appropriate error message to the user.

8. Deployment and Implementation Considerations

This method relies on the fact that RDAP clients are fetching the IANA registries to then find the servers locally. Clients SHOULD not fetch every time the registry. Clients SHOULD cache the registry, but use underlying protocol signalling, such as HTTP Expires header field [RFC7234], to identify when it is time to refresh the cached registry.

If the query data does not match any entry in the client cached registry, then the client may implement various methods, such as the following:

- In the case of a domain object to be RDAP queried, the client may first query the DNS to see if the respective entry has been delegated or if it is a mistyped information by the user. The DNS query could be to fetch the NS records for the TLD domain. If the DNS answer is negative, then there is no need to fetch the new version of the registry. However, if the DNS answer is positive, this may mean that the currently cached registry is no more current. The client could then fetch the registry, parse and then do the normal matching as specified above. This method may not work for all types of RDAP objects.

- If the client knows the existence of a RDAP aggregator or redirector and trust that service, then it could send the query to the redirector, which would redirect the client if it knows the authoritative server that client has not found.

IANA should make sure that the service of those registries is able to cope with a larger demand and should take appropriate measures such as caching and load balancing.

This specification does not assume while not prohibiting how some authorities of registration data may work together on sharing their information for a common service, including mutual redirection[I-D.ietf-weirds-redirects].

9. Limitations

This method does not provide a direct way to find authoritative RDAP servers:

- for entities

- for queries using search patterns that do not contain a terminating string that matches some entries in the registries

10. Security Considerations

By providing a bootstrap method to find RDAP servers, this document helps making sure that the end-users will get the RDAP data from authoritative source, instead of from rogue sources. The method itself has the same security properties as the RDAP protocols themselves. The transport used to access the registries could be more secure by using TLS [RFC5246] if IANA supports it.

11. IANA Considerations

IANA is requested to do the following:

- Create a new registry "IPv4 Address Space RDAP Bootstrap Service" in the JSON format, as shown above.
- Create a new registry "IPv6 Address Space RDAP Bootstrap Service" in the JSON format, as shown above.
- Create a new registry "Autonomous System Number Space RDAP Bootstrap Service" in the JSON format, as shown above.
- Create a new registry "Domain Name Space RDAP Bootstrap Service" in the JSON format, as shown above.

It is envisionned that these new registries will have similar entries than the corresponding IANA allocation registries, such as [ipv4reg],[ipv6reg],[asreg], [domainreg], and possibly similar registration policies. However, the registration policies for the new registries of this document are left to IANA.

The registries may be maintained in IANA own format, such as XML. However, the registry should be available in the JSON format, and optionally in other formats such as XML.

12. Acknowledgements

The weirds working group had multiple discussions on this topic, including a session during IETF 84, where various methods such as in-DNS and others were debated. The idea of using IANA registries was discovered by the editor during discussions with his colleagues as well as by a comment from Andy Newton. All the people involved in these discussions are herein acknowledged. Linlin Zhou, Jean-Philippe Dionne, John Levine, Kim Davies, Ernie Dainow, Scott Hollenbeck, Arturo Servin, Andy Newton, Murray Kucherawy, Tom Harrison, Naoki Kambe have provided input and suggestions to this document.

13. References

13.1. Normative References

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

[RFC3339] Klyne, G., Ed. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC7159] Bray, T., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, March 2014.

13.2. Non-Normative References

[I-D.ietf-weirds-json-response]
Newton, A. and S. Hollenbeck, "JSON Responses for the Registration Data Access Protocol (RDAP)", draft-ietf-weirds-json-response-07 (work in progress), April 2014.

[I-D.ietf-weirds-rdap-query]
Newton, A. and S. Hollenbeck, "Registration Data Access Protocol Query Format", draft-ietf-weirds-rdap-query-10 (work in progress), February 2014.

[I-D.ietf-weirds-redirects]
Martinez, C., Zhou, L., and G. Rada, "Redirection Service for Registration Data Access Protocol", draft-ietf-weirds-redirects-03 (work in progress), February 2014.

[I-D.ietf-weirds-using-http]
Newton, A., Ellacott, B., and N. Kong, "HTTP usage in the Registration Data Access Protocol (RDAP)", draft-ietf-weirds-using-http-08 (work in progress), February 2014.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC7234] Fielding, R., Nottingham, M., and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, June 2014.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, May 2014.

[asreg] Internet Assigned Numbers Authority(IANA), , "Autonomous System (AS) Numbers", <http://www.iana.org/assignments/as-numbers/as-numbers.xml>.

[domainreg] Internet Assigned Numbers Authority(IANA), , "Root Zone Database", <http://www.iana.org/domains/root/db>.

[ipv4reg] Internet Assigned Numbers Authority(IANA), , "IPv4 Address Space", <http://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xml>.

[ipv6reg] Internet Assigned Numbers Authority(IANA), , "IPv6 Global Unicast Address Assignments", <http://www.iana.org/assignments/ipv6-unicast-address-assignments/ipv6-unicast-address-assignments.xml>.

[ipv6regparent] Internet Assigned Numbers Authority(IANA), , "Internet Protocol Version 6 Address Space", <http://www.iana.org/assignments/ipv6-address-space/ipv6-address-space.xml>.

Authors' Addresses

Marc Blanchet
Viagenie
246 Aberdeen
Quebec, QC G1R 2E1
Canada

Email: Marc.Blanchet@viagenie.ca
URI: http://viagenie.ca

Guillaume Leclanche
Viagenie
246 Aberdeen
Quebec, QC G1R 2E1
Canada

Email: Guillaume.Leclanche@viagenie.ca
URI: http://viagenie.ca