Abstract

The Registration Data Access Protocol (RDAP) does not include query capabilities to find the list of domains related to a set of entities matching a given search pattern. Even if such capabilities, commonly referred to as reverse search, respond to some needs not yet readily fulfilled by the current Whois protocol, they have raised concern from two perspectives: server processing impact and data privacy. Anyway, the impact of the reverse queries on RDAP servers processing is the same of the standard searches and it can be reduced by implementing policies to deal with big result sets, while data privacy risks can be mitigated by RDAP access control functionalities. This document describes RDAP query extensions that allows clients to request a reverse search based on the domains-entities relationship.

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 April 19, 2019.

Copyright Notice

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

Reverse Whois is a service provided by many web applications that allows users to find domain names owned by an individual person or company starting from the owner details (like name, email). Even if the availability of this service might raise some objections due to potential privacy risks, ICANN itself, in its report about Next-Gen Registration Directory Service (RDS) [ICANN-RDS], states that it is allowed when driven by some permissible purposes (e.g. legal actions, criminal investigations) and if it provides adequate policies to enforce the requestor accreditation, authentication, authorization, and terms and conditions of data use.

It is well known that such policies are not implemented in Whois [RFC3912], while they are in RDAP. In fact, RDAP relies on security features, available in the HTTP protocol, to support access control based on local policy [RFC7481].

Another objection to the implementation of Reverse Whois is connected with its impact on server processing. Since RDAP supports search
queries, the impact of both standard and reverse searches can be mitigated by servers adopting ad hoc strategies. Furthermore, reverse search is almost always performed by specifying an entity role (e.g. registrant, technical contact) and this can contribute to restrict the result set.

Reverse searches, such as finding the list of domain names associated to contacts, nameservers or DNSSEC keys, may be useful to registrars as well. Usually, registries adopt out-of-band mechanisms to provide results to registrars asking for reverse searches on their domains. Possible reasons of such requests are:

- the loss of synchronization between the registrar database and the registry database;
- the need of such data to perform massive EPP updates (i.e. changing the contacts of a set of domains, etc.).

Currently, RDAP does not provide any way for a client to search for the collection of domains associated to an entity [RFC7482]. A query (lookup or search) on domains can return the array of entities related to a domain with different roles (registrant, registrar, administrative, technical, reseller, etc.), but the reverse operation is not allowed. Only reverse searches to find the collection of domains related to a nameserver (idhName or ip) can be requested. Since entities can be in relationship with all RDAP objects [RFC7483], the availability of a reverse search can be common to all RDAP query paths.

The protocol described in this specification aims to extend the RDAP query capabilities to enable reverse search based on the domains-entities relationship (the classic Reverse Whois scenario). The extension is implemented by adding new path segments (i.e. search paths) and using a RESTful web service [REST]. The service is implemented using the Hypertext Transfer Protocol (HTTP) [RFC7230] and the conventions described in RFC 7480 [RFC7480].

1.1. 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. RDAP Path Segment Specification

The new search paths are OPTIONAL extensions of path segments defined in RFC 7482 [RFC7482]. The search paths are:
The reverse search pattern is a JSON [RFC8259] object including two members: "value" and "role". The "value" member represents the search pattern to be applied to the corresponding entity field and can be a JSON type primitive or object. The "role" member is a string whose possible values are those detailed in Section 10.2.4 of RFC 7483 [RFC7483]. The former is REQUIRED while the latter is OPTIONAL to allow RDAP servers to provide reverse search capabilities without specifying any role.

The search patterns corresponding to the "value" in the first two cases are the same as specified in paragraph Section 3.2.3 of RFC 7482 [RFC7482] (Figure 1).

domains?entityHandle={"value":"CID-40*","role":"registrant"}

domains?entityFn={"value":"Bobby*","role":"registrant"}

Figure 1: Examples of RDAP queries to find all domains related to a registrant whose handle matches "CID-40*" and whose formatted name matches "Bobby*"

The last two reverse searches are considered by ICANN very useful to improve searchability capabilities of the Registry Directory Services ([ICANN-RDS], [ICANN-RA]).

Searches for domains by related entity email are specified using this form:

domains?entityEmail={"value":"XXXX","role":"ZZZZ"}

where XXXX is a search pattern representing an email address as defined in RFC 5322 [RFC5322].

Searches for domains by related entity postal address are specified using this form:

domains?entityAddr={"value":YYYY,"role":"ZZZZ"}
where YYYY is a JSON object containing the information described in Section 2.4 of RFC 5733 [RFC5733], respectively: "street", "city", "sp", "pc" and "cc" (Figure 2). All the members of the postal address object are OPTIONAL but at least one is REQUIRED. The constraints on the members are implicitly joined by AND.

domains?entityAddr={"value":{"cc":"CA","city":"Sydney"},"role":"registrant"}

Figure 2: Example of a RDAP query to find all domains related to a registrant whose postal address contains the country code equals to "CA" and the city equals to "Sydney"

3. Implementation Considerations

The implementation of the proposed extension is technically feasible. The search paths "handle" and "fn" are used as standard paths to search for entities. With regards to the last two reverse searches, both email and postal address information are usually required by the registries but, while the former is usually mapped on a DBMS indexed field, the latter is mapped on a combination of non-indexed fields. As a consequence while the former should not significantly decrease the performance, the latter might have an impact on server processing. Anyway, this impact is evaluated to be the same as other query capabilities already presented in RDAP (e.g. wildcard prefixed search pattern) so the risks to generate huge result sets are the same existing for the other standard searches and can be mitigated by adopting the same policies (e.g. restricting search functionalities, limiting the rate of search requests according to the user profile, truncating and paging the results, returning partial responses).

3.1. JSON in URLs

Many web services, including RDAP, rely on the HTTP GET method to take advantage from some of its features:

- GET requests can be cached;
- GET requests remain in the browser history;
- GET requests can be bookmarked.

Sometimes it happens that such advantages should be combined with the requirement to pass objects and arrays in the query string. JSON is the best candidate as data interchange format but it contains some characters that are forbidden from appearing in a URL. Anyway, escaping the invalid characters is not an issue because, on client side, modern browsers automatically encode URLs and, on server side, a lot of URL encoding/decoding libraries for all web development programming languages are available. The downside of URL encoding is
that it can make a URL rather long which, depending on the initial length and the number of invalid characters, might exceed the practical limit of web browsers (i.e. 2,000 characters).

Other solutions to pass a JSON expression in a URL could be:

- converting JSON to Base64 ([RFC4648]) but binary data are unreadable;
- using a JSON variation that complies with URL specifications and maintains readability like Rison ([RISON]), URLON ([URLON]) or JSURL ([JSURL]).

The extensions proposed in this document rely on URL encoding because it is widely supported and the risk to exceed the maximum URL length is considered to be very unlikely in RDAP.

4. 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."

4.1. IIT-CNR/Registro.it

Responsible Organization: Institute of Informatics and Telematics of National Research Council (IIT-CNR)/Registro.it
Location: https://rdap.pubtest.nic.it/
Description: This implementation includes support for RDAP queries using data from the public test environment of .it ccTLD. The RDAP server does not implement any security policy because data returned by this server are only for experimental testing purposes.
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: Mario Loffredo, mario.loffredo@iit.cnr.it

5. Security Considerations

Security services for the operations specified in this document are described in RFC 7481 [RFC7481]. It is quite easy to imagine that, in order to be compliant with ICANN recommendations about its use, RDAP servers will provide reverse search, like other query capabilities, only to restricted communities. One realistic scenario for servers is to provide reverse search only for registrars searching for their own domains. Another one is to prevent users to start a reverse search from a registrant detail, by removing "registrant" from the possible "role" values.

6. IANA Considerations

This document has no actions for IANA.

7. Acknowledgements

The authors would like to acknowledge Scott Hollenbeck, Francisco Arias, Gustavo Lozano and Eduardo Alvarez for their contribution to this document.

8. References

8.1. Normative References


8.2. Informative References

[ICANN-RA]
Internet Corporation For Assigned Names and Numbers, "Registry Agreement", July 2017,
Appendix A.  Change Log

00:  Initial version.
01:  Revised some sentences and references.
02:  Added "entityEmail" and "entityAddr" path segments. Removed
    "entityRole" path segment. Revised "Acknowledgements" section
03:  Added "JSON in URLs" section

Authors’ Addresses

Mario Loffredo
IIT-CNR/Registro.it
Via Moruzzi,1
Pisa 56124
IT

Email: mario.loffredo@iit.cnr.it
URI:    http://www.iit.cnr.it