RFC 7482 : Registration Data Access Protocol Query Format

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 26 mars 2015

Date de publication du RFC : Mars 2015

https://www.bortzmeyer.org/7482.html

Le nouveau protocole d'information RDAP <https://www.bortzmeyer.org/weirds-rdap.html>, qui vise à remplacer whois, est décrit dans un ensemble de RFC. Celui présenté ici normalise la façon de former les requêtes RDAP. Celles-ci ont la forme d'une URL, puisque RDAP repose sur l'architecture REST. (Depuis, ce RFC a été légèrement mis à jour par son remplaçant, le RFC 9082 [1].)

Vous pouvez voir les autres RFC sur RDAP dans mon autre article <https://www.bortzmeyer.org/weirds-rdap.html>. Je rappelle juste que le seul transport actuellement normalisé pour RDAP (dans le RFC 7480) est HTTP. RDAP peut être utilisé pour beaucoup de sortes d'entités différentes mais ce RFC ne couvre que ce qui correspond aux usages actuels de whois, les préfixes d'adresses IP, les AS, les noms de domaine, etc. Bien sûr, un serveur RDAP donné ne gère pas forcément tous ces types d'entités, et il renvoie le code HTTP 501 ("*Not implemented*") s'il ne sait pas gérer une demande donnée. Ce RFC ne spécifie que l'URL de la requête, le format de la réponse est variable (JSON, XML...) et le seul actuellement normalisé, au-dessus de JSON, est décrit dans le RFC 7483. D'autre part, ces RFC RDAP ne décrivent que le protocole entre le client RDAP et le serveur, pas « l'arrière-cuisine », c'est-à-dire l'avitaillement (création, modification et suppression) des entités enregistrées. RDAP est en lecture seule et ne modifie pas le contenu des bases de données qu'il interroge.

Passons aux choses concrètes. Une requête RDAP est un URL RFC 3986. Celui-ci est obtenu en ajoutant un chemin spécifique à une base. La base (par exemple https://rdap.example.net/) va être obtenue par des mécanismes divers, comme celui du RFC 7484. On met ensuite un chemin qui dépend du type d'entité sur laquelle on veut se renseigner, et qui indique l'identificateur de l'entité. Par exemple, avec la base ci-dessus, et une recherche du nom de domaine internautique.fr, on construirait un URL complet https://rdap.example.net/domain/internautique.fr. Il y a cinq types d'entités possibles :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc9082.txt

— `ip` : les préfixes IP (notez qu'on peut chercher un préfixe en donnant juste une des adresses IP couvertes par ce préfixe),
— `autnum` : les numéros de systèmes autonomes,
— `domain` : un nom de domaine (notez que cela peut être un domaine dans `in-addr.arpa` ou `ipv6.arpa`),
— `nameserver` : un serveur de noms,
— `entity` : une entité quelconque, comme un bureau d'enregistrement, ou un contact identifié par un "*handle*".

Pour `ip`, le chemin dans l'URL est `/ip/XXX` où `XXX` peut être une adresse IPv4 ou IPv6 sous forme texte. Il peut aussi y avoir une longueur de préfixe à la fin donc `/ip/2001:db8:1:a::/64` est un chemin valable. Ainsi, sur le service RDAP expérimental du RIPE-NCC, `http://rdap.db.ripe.net/ip/2001:4b98` est un URL possible. Testons-le :

```
% curl  http://rdap.db.ripe.net/ip/2001:4b98:dc0:41::
{
  "handle" : "2001:4b98:dc0::/48",
  "startAddress" : "2001:4b98:dc0::/128",
  "endAddress" : "2001:4b98:dc0:ffff:ffff:ffff:ffff:ffff/128",
  "ipVersion" : "v6",
  "name" : "GANDI-HOSTING-DC0",
  "type" : "ASSIGNED",
  "country" : "FR",
  "rdapConformance" : [ "rdap_level_0" ],
  "entities" : [ {
    "handle" : "GAD42-RIPE",
    "vcardArray" : [ "vcard", [ [ "version", { }, "text", "4.0" ], [ "fn", { }, "text", "Gandi Abuse Departm
      "label" : "63-65 Boulevard Massena\n75013 Paris\nFrance"
...
```

Pour `autnum`, on met le numéro de l'AS après `/autnum/`. Toujours dans l'exemple RIPE-NCC, `http://rdap.db.ripe.net/autnum/29169` permet de chercher de l'information sur l'AS 29169 :

```
% curl   http://rdap.db.ripe.net/autnum/29169
{
  "handle" : "AS29169",
  "name" : "GANDI-AS",
  "type" : "DIRECT ALLOCATION",
  "rdapConformance" : [ "rdap_level_0" ],
  "entities" : [ {
    "handle" : "GANDI-NOC",
    "roles" : [ "registrant" ]
  }, {
...
```

Pour les noms de domaines, on met le nom après `/domain/`. Ainsi, sur le serveur RDAP expérimental d'Afilias, `http://rdg.afilias.info/rdap/domain/rmll.info` nous donnera de l'information sur le domaine `rmll.info`. On peut mettre un nom en Unicode donc `https://rdap.example.net/domain/pot` est valable, mais il devra être encodé comme l'explique la section 6.1, plus loin. Si on ne veut pas lire cette information sur l'encodage, on peut aussi utiliser la forme Punycode, donc chercher avec `https://rdap.example.net/domain/xn--potamochre-66a.fr`.

On peut aussi se servir de RDAP pour les noms de domaines qui servent à traduire une adresse IP en nom :

```
% curl http://rdap.db.ripe.net/domain/1.8.a.4.1.0.0.0.0.d.1.4.1.0.0.2.ip6.arpa
{
  "handle" : "0.d.1.4.1.0.0.2.ip6.arpa",
  "ldhName" : "0.d.1.4.1.0.0.2.ip6.arpa",
  "nameServers" : [ {
    "ldhName" : "dns15.ovh.net"
  }, {
    "ldhName" : "ns15.ovh.net"
  } ],
  "rdapConformance" : [ "rdap_level_0" ],
  "entities" : [ {
    "handle" : "OK217-RIPE",
    "roles" : [ "administrative" ]
  }, {
    "handle" : "OTC2-RIPE",
    "roles" : [ "zone", "technical" ]
  }, {
    "handle" : "OVH-MNT",
    "roles" : [ "registrant" ]
  } ],
  "remarks" : [ {
    "description" : [ "OVH IPv6 reverse delegation" ]
  } ],
...
```

Pour un serveur de noms, on met son nom après `/nameserver` donc, chez Afilias :

```
% curl  http://rdg.afilias.info/rdap/nameserver/rmll1.rmll.info
{
...
   "ipAddresses": {
      "v4": [
         "80.67.169.65"
      ]
   },
   "lang": "en",
   "ldhName": "rmll1.rmll.info",
...
```

Pour `entity`, on indique juste un identificateur. Voici un exemple :

```
% curl  http://rdg.afilias.info/rdap/entity/R-R191-LRMS
{
   "handle": "R-R191-LRMS",
   "lang": "en",
...
   "roles": [
      "registrar"
   ],
   "vcardArray": [
      "vcard",
      [
         [
            "version",
            {},
            "text",
            "4.0"
         ],
         [
            "fn",
            {},
            "text",
```

```
            "Gandi SAS"
         ],
         [
            "adr",
            {},
            "text",
            [
               "",
               "",
               "63-65 boulevard Massena",
               "Paris",
               "",
               "F-75013",
               "FR"
            ]
...
```

Dernière possibilité, un chemin spécial indique qu'on veut récupérer de l'aide sur ce serveur RDAP particulier. En envoyant `help` (par exemple `https://rdap.example.net/help`, on obtient un document décrivant les capacités de ce serveur, ses conditions d'utilisation, sa politique vis-à-vis de la vie privée, ses possibilités d'authentification, l'adresse où contacter les responsables, etc. C'est l'équivalent de la fonction d'aide qu'offrent certains serveurs whois, ici celui de l'AFNIC :

```
% whois -h whois.nic.fr  -- -h
...
%% Option   Function
%% -------  ------------------------------------
%% -r       turn off recursive lookups
%% -n       AFNIC output format
%% -o       old fashioned output format (Default)
%% -7       force 7bits ASCII output format
%% -v       verbose mode for templates and help options
%%          (may be use for reverse query)
%% -T type  return only objects of specified type
%% -P       don't return individual objects in case of contact search
%% -h       informations about server features
%% -l lang  choice of a language for informations (you can specify US|EN|UK for
%%          english or FR for french)
%%
...
```

Pour RDAP, voyez par exemple <http://rdg.afilias.info/rdap/help> ou, dans un genre très différent, <http://rdap.apnic.net/help>.

Toutes les recherches jusque-là ont été des recherches exactes (pas complètement pour les adresses IP, où on pouvait chercher un réseau par une seule des adresses contenues dans le réseau). Mais on peut aussi faire des recherches plus ouvertes, sur une partie de l'identificateur. Cela se fait en ajoutant une requête (la partie après le point d'interrogation) dans l'URL et en ajoutant un astérisque (cf. section 4.1). Ainsi, `https://rdap.example.net/domains?name=foo*` cherchera tous les domaines dont le nom comporte la chaîne de caractères `foo`. (Vous avez noté que c'est `/domains`, au pluriel, et non plus `/domain`?) Voici un exemple d'utilisation :

```
% curl  http://rdg.afilias.info/rdap/domains\?name=rm\*|more
   "domainSearchResults": [
      {
         "handle": "D10775367-LRMS",
         "ldhName": "RMLL.INFO",
...
         "remarks": [
```

```
            {
               "description": [
                  "Summary data only. For complete data, send a specific query for the object."
               ],
               "title": "Incomplete Data",
               "type": "object truncated due to unexplainable reasons"
            }
...
         "ldhName": "RMLL2015.INFO",
...
         "ldhName": "RMLLOCKSMITHS.INFO",
...
      {
         "description": [
            "Search results are limited to 50 per query."
         ],
         "title": "Search Policy",
         "type": "result set truncated due to unexplainable reasons"
      }
```

On peut aussi chercher un domaine d'après ses serveurs de noms, par exemple `https://rdap.example.net/domai` chercherait tous les domaines délégués au serveur DNS `ns1.example.com`. Une telle fonction peut être jugée très indiscrète et le serveur RDAP est toujours libre de répondre ou pas :

```
% curl http://rdg.afilias.info/rdap/domains\?nsLdhName=ns0.abul.org
...
{
   "description": [
      "Domain name search by nameserver is not supported."
   ],
   "errorCode": 501,
...
```

Deux autres types permettent ces recherches ouvertes, `/nameservers` (comme dans `https://rdap.example.net/` mais notez qu'on peut aussi chercher un serveur par son nom) et `/entities` (comme dans `https://rdap.example.net`

```
% curl  http://rdg.afilias.info/rdap/entities\?fn=go\*
{
   "entitySearchResults": [
      {
         "fn": "Go China Domains, Inc.",
...
         "fn": "Gotnames.ca Inc.",
...
```

Notez que ce type de recherche peut représenter un sérieux danger pour la vie privée (comme noté dans le RFC, par exemple en section 4.2) puisqu'elle permettrait, par exemple de trouver tous les titulaires prénommés Jean. Il faut donc espérer qu'elle ne sera accessible qu'à des clients authentifiés, et de confiance.

La section 4 détaille le traitement des requêtes. N'oubliez pas qu'on travaille ici sur HTTP et que, par défaut, les codes de retour RDAP suivent la sémantique HTTP (404 pour un objet non trouvé, par exemple). Il y a aussi quelques cas où le code à retourner est moins évident. Ainsi, si un serveur ne veut pas faire une recherche ouverte, il va répondre 422 ("*Unprocessable Entity*").

Vous avez noté plus haut, mais la section 6 le rappelle aux distraits, que le nom de domaine peut être exprimé en Unicode ou en ASCII. Donc, `https://rdap.example.net/domain/potamochère.fr` et `https://rdap.example.net/domain/xn--potamochre-66a` sont deux requêtes acceptables.

Enfin, la section 8 rappelle quelques règles de sécurité comme :

— Les requêtes ouvertes peuvent mener à une forte consommation de ressources sur le serveur. Le serveur qui ne vaut pas se faire DoSer doit donc faire attention avant de les accepter.
— Les requêtes RDAP, et surtout les requêtes ouvertes, peuvent soulever des questions liées à la vie privée. Les serveurs RDAP doivent donc réfléchir avant de renvoyer de l'information. Rappelez-vous que RDAP, contrairement à whois, peut avoir un mécanisme d'authentification, donc peut envoyer des réponses différentes selon le client.
— Et, corollaire du précédent point, les gérants de serveurs RDAP doivent définir une politique d'autorisation : qu'est-ce que je renvoie, et à qui ?