RFC 9537 : Redacted Fields in the Registration Data Access Protocol (RDAP) Response

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 14 juillet 2024

Date de publication du RFC : Mars 2024

https://www.bortzmeyer.org/9537.html

RDAP est le protocole recommandé pour accéder aux données sociales sur un nom de domaine, comme le nom du titulaire ou son adresse postale. Pour d'évidentes raisons de vie privée, certains registres ne renvoient pas la totalité de l'information dont ils disposent. Que doit-on mettre dans RDAP dans ce cas ? La question n'était pas tranchée et chaque registre faisait différemment. Désormais, il existe une solution normalisée.

Au passage, oui, d'accord, il n'y a pas que RDAP pour obtenir les données sociales (cet article <https://www.afnic.fr/observatoire-ressources/papier-expert/trouver-les-informations-soci> vous en indiquera d'autres). Mais c'est le service le plus moderne et le plus adapté aux programmeur·ses <https://www.afnic.fr/observatoire-ressources/papier-expert/rdap-obtenir-des-inform>. Contacté en HTTPS, le serveur RDAP va renvoyer du JSON que le client n'aura plus qu'à filtrer et formatter. Voici par exemple une partie de la réponse RDAP obtenue en se renseignant sur nouveaufrontpopulaire.fr :

```
% curl -s https://rdap.nic.fr/domain/nouveaufrontpopulaire.fr | jq
...
          [
            "fn",
            {},
            "text",
            "Parti Socialiste"
          ],
          [
            "org",
            {},
            "text",
            "Parti Socialiste"
          ],
          [
            "adr",
            {},
            "text",
```

```
            [
              "",
              "",
              "99 Rue Moliere",
              "Ivry-sur-Seine",
              "",
              "94200",
              "FR"
            ]
          ],
...
```

Ici, il s'agit d'une personne morale donc les données sont toutes envoyées. Et s'il s'agissait d'une personne physique, pour laquelle la loi Informatique & et Libertés s'applique, depuis 1978 ? La solution évidente est de ne pas envoyer les données qu'on ne veut pas diffuser mais attention, il y a un piège, il ne faut pas casser la syntaxe JSON. Par exemple, RDAP utilise (c'est en cours de changement, cf. RFC 9553 [1]) jCard pour formater les adresses (RFC 7095) et les champs dans jCard ne sont pas étiquetés, c'est leur position dans le tableau qui indique leur rôle (c'est un des nombreux inconvénients de jCard). On ne peut donc pas supprimer, par exemple, la rue, en indiquant :

```
[
            "adr",
            {},
            "text",
            [
              "",
              "",
              "Ivry-sur-Seine",
              "",
              "94200",
              "FR"
            ]
],
          [
            "email",
            {},
            "text",
            "e5d92838d5f0268143ac47d86880b5f7-48916400@contact.gandi.net"
          ],
```

Car, alors, on ne saurait plus si "Ivry-sur-Seine" est la rue ou bien la ville.

Le principe de notre RFC est donc : si on peut, retirer le membre JSON. Si on ne peut pas (cas du tableau de taille fixe), mettre une valeur vide ou `null`.

Petit point de terminologie : comment traduire le "*redacted*" du titre ? « Censuré » est inadapté (l'intervention ne vient pas d'un tiers mais d'une des deux parties). Je vais dire « élidé <https://fr.wiktionary.org/wiki/%C3%A9lid%C3%A9> », mais « caviardé », « biffé » et « expurgé » sont également de bonnes solutions. Le nom correspondant est « élision <https://fr.wiktionary.org/wiki/%C3%A9lision> ». Évidemment, il ne faut surtout pas dire « anonymisé », il n'y a rien d'anonyme ici, puisque le registre connait toute l'information, il refuse simplement de la diffuser.

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc9553.txt`

La section 1 du RFC expose les grands principes de l'élision. Elle explique notamment qu'en cas d'élision, il faut ajouter un membre (nommé `redacted`) à la réponse JSON, expliquant les raisons et utilisant le langage JSONPath (RFC 9535) pour désigner de manière formelle la partie élidée.

Compte-tenu des contraintes sur la syntaxe de la réponse JSON (RFC 9083), le RFC normalise quatre façons d'élider (section 3) :

— Suppression d'un champ, si possible (c'est la méthode préférée). Cela marche dans la plupart des cas. Par exemple, si on ne veut pas publier le titulaire d'un domaine, on ne l'inclut pas dans la réponse, point.
— Mettre une valeur vide. Quand les contraintes de syntaxe empêchent de supprimer complètement un champ, on lui met une valeur vide. C'est notamment nécessaire pour les tableaux que jCard utilise abondamment : le retrait d'un champ casserait le tableau, qui est censé avoir un nombre d'élements fixe.
— Une valeur trop détaillée peut être remplacée par une valeur partielle. L'adresse `"label":"123 Maple Ave\nSuite 901\nVancouver BC\nCanada"` (un exemple du RFC 7095) peut être remplacée par une valeur moins précise comme `"label":"Vancouver\nBC\nCanada\n"`.
— Enfin, la dernière méthode est de remplacer une valeur par une autre, par exemple une adresse de courrier par une adresse qui masque la vraie, comme `e5d92838d5f0268143ac47d86880b5f7-48916400@cont` dans l'exemple plus haut.

Et le RFC insiste qu'il ne faut **pas** utiliser de texte bidon (« XXX », « "*lorem ipsum dolor*" » ou « Ano Nymous ») car ce texte ne correspond pas forcément aux règles de syntaxe du champ (et, j'ajoute, peut être difficile à identifier pour le lecteur, qui peut ne pas avoir la référence).

Pour la première méthode, la suppression d'un champ, si on supprime le titulaire, on aura un membre nommé `redacted` (élidé) ajouté ainsi :

```
 "redacted": [
  {
    "name": {
      "description": "Remove registrant"
    },
    "prePath": "$.entities[?(@.roles[0]=='registrant')]",
    "method": "removal"
  }
  ]
```

Notez le (difficile à lire) code JSONPath `$.entities[?(@.roles[0]=='registrant')]`.

Le deuxième cas, celui d'une valeur vide, donnerait, pour le cas où on supprime juste le nom du titulaire (qui est en position 1 dans le jCard, et son nom en position 3 - sachant qu'on part de 0) :

```
[
  "fn",
  {},
  "text",
  ""
]
...
   "redacted": [
  {
    "name": {
      "description": "Registrant Name"
    },
    "postPath": "$.entities[?(@.roles[0]=='registrant')].
```

```
      vcardArray[1][?(@[0]=='fn')][3]",
    "pathLang": "jsonpath",
    "method": "emptyValue",
    "reason": {
      "description": "Server policy"
    }
  }
  ]
```

Troisième technique d'élision, réduire une valeur. Le `redacted` devient :

```
"redacted": [
  {
    "name": {
      "description": "Home Address Label"
    },
    "postPath": "$.vcardArray[1][?(@[0]=='adr')][1].label",
    "pathLang": "jsonpath",
    "method": "partialValue",
    "reason": {
      "description": "Server policy"
    }
  }
]
```

Et pour finir, la quatrième et dernière méthode, le remplacement :

```
"redacted": [
  {
    "name": {
      "description": "Registrant Email"
    },
    "postPath": "$.entities[?(@.roles[0]=='registrant')].
                vcardArray[1][?(@[0]=='email')][3]",
    "pathLang": "jsonpath",
    "method": "replacementValue",
  }
  ]
```

Ce membre appelé `redacted` est spécifié en détail dans la section 4 du RFC. (Et il est enregistré à l'IANA <`https://www.iana.org/assignments/rdap-extensions/rdap-extensions.xml#rdap-extensions-1`> parmi les extensions RDAP.) Pour signaler qu'il peut apparaitre, le membre `rdapConformance` de la réponse JSON va l'indiquer :

```
{
  "rdapConformance": [
    "itNic",
    "redacted",
    "rdap_level_0"
  ],
...
```

Dès qu'il y a élision, `redacted` doit être ajouté. Il contient un tableau JSON d'objets, dont les membres peuvent être (seul le premier est obligatoire) :

- `name` : un terme qui décrit le champ élidé,
- `prePath` et `postPath` : des expressions JSONPath (RFC 9535) qui dénotent le membre retiré ou modifié,
- `method` : la technique d'élision utilisée (suppression, nettoyage, remplacement, etc),
- `reason` : texte libre décrivant la raison de l'élision.