draft-...-considerations-on-rfc9537

Andy Newton
IETF 120 - 24 July 2024

RFC 9537 - Minted March 2024

Abstract

This document describes a Registration Data Access Protocol (RDAP) extension for specifying methods of redaction of RDAP responses and explicitly identifying redacted RDAP response fields, using JSONPath as the default expression language.

The Basics

Extension defines a “redacted” array of objects:

```
{
  "name": {
    "description": "Registrar Email"
  },
  "prePath": "$.entities[?(@.roles[0]=='registrar')].vcardArray[1][?(@[0]=='email')]",
  "method": "removal",
  "reason": {
    "description": "Server policy"
  }
}
```

Redaction methods:

redaction by removal, empty value, partial value, replacement value.

Depending on method, JSONPath expression found in some combination of:

prePath (optional), postPath, replacementPath.

Implementations at the Time of Publication

Servers:

1. IIT-CNR/Registro.it
    - Public test environment of .it ccTLD.
    - Level of Maturity: This is an "alpha" test implementation.
    - Coverage: This implementation includes all of the features described in this specification.
    - Source Code: Not known to be available.

Clients:

None known to be implemented.

Work Informing these Considerations

1. Two clients, each with a separate development team:
   a. Web client at https://lookup.icann.org/en .
      i. Extension added to existing client.
      ii. Development done with internal ICANN development team.
   b. Command-Line Interface (CLI) client:
      i. Extension added to existing client.
      ii. Development done by Cobenian, contractor with experience doing protocol development.
      iii. Open source: https://github.com/icann/icann-rdap .
2. Test server
   a. Extension added to existing server.
   b. Development done by Cobenian.
   c. Open source: https://github.com/icann/icann-rdap .
3. Test cases:
   a. Corpus of 21 test cases created: https://github.com/anewton1998/redacted_examples .
   b. Includes Dockerfile to help any client implementor to stand up test server with all examples.

Limitations of Redaction by Removal

- prePath is optional.
- Even if it was present, it should always evaluate to an empty set.
- There is no explicit means to tell a client what has been removed.

Non-normative language in Section 5.1:

“The client can key off the "name" member for display logic related to the redaction.”

- This guidance is ambiguous because “name” may contain either a registered value or an unregistered value.
- Reduces redaction by removal to “redaction by notice” (more later).

Limitations of Redaction by Empty Value

The Redaction by Empty Value Method is when a redacted field is not removed, but its value is set to an empty value, such as "" for a jCard [RFC7095] Text ("text") property or null for a non-Text property.

- Ambiguous:
  - No definition of “empty value” for numbers or booleans.
  - Empty JSON arrays are “[ ]” and empty JSON objects are “{}”.
  - Is this only for jCard?
- List discussion:
  - RFC 9083 Section 3.1 mentions null, therefore null in place of any value is legal.
  - This is a specious argument. 3.1 is merely listing the types of JSON primitive values. No where in the RFC does it say null may be given in place of any other value.
  - Same is true for RFC 7095.
  - If true, can “1234” be represented as 1234?
- For all practical purposes, redaction by empty value only applies to strings.

Limitations of Redaction by Partial Value

- *“The Redaction by Partial Value Method SHOULD be used only when redacting JSON response fields that use a formatted value, where a portion of the value is removed.”*
  - What is a formatted value for anything but a string?
  - What is the partial value of a number, boolean, or null?
  - Ambiguous for things like “tel” which can be either formatted or unformatted strings.
  - Cannot identify which parts of a string, array, or object have been removed.
- Identifying the parts of a string that are redacted is important:
  - Unstructured postal addresses in jCard:
    - `[“adr”,{“label”:”123 Maple\nVancouver\nBC\nCA\n294-QL”]`
  - Personal data in RDAP remarks:
    - `“description”: [ “send routing requests to bob@bobsisp.com”]`
- List discussion:
  - This can be fixed with an extension to this extension.

Redaction by Replacement Value

- When used with “prePath”, this has the same limitations as redaction by removal:
  - The client cannot explicitly identify the value being replaced.

JSONPath is Inappropriate for this Use Case

- JSONPath implementations vary too much in conformance to RFC 9395 to be usable in a protocol that exchanges JSONPath expressions.
- https://cburgmer.github.io/json-path-comparison/
- Our experience, there is no JSONPath library that we could find which was useable for this purpose.
  - Our developers were forced to write their own JSONPath parsers to either fixup JSONPath expressions from the server or convert them to arrays of JSONPointer expressions.
- A simplified profile of JSONPath must be specified for it to be useable.
  - Either to allow RDAP implementors to find JSONPath implementations compliant with the profile; or
  - Write their own JSONPath implementation (still not a good choice).

Redaction Using JSONPath is Complex

```
{
  "name": {
    "description": "Developer Quotes About RFC 9537"
  },
  "method": "removal",
  "reason": {
    "description": "Cannot be repeated in professional venues."
  }
}
```

- Requires clients to alter their internal data model for every aspect of an RDAP response that may be redacted.
- May require a two-pass parsing strategy.
- Either/both are significant structural changes to a client.

JSONPath is OPTIONAL

- The RFC authors claim JSONPath in RFC 9537 is OPTIONAL and have filed two errata:
  a. Change the abstract.
  b. Change MUST to MAY for postPath and replacementPath.
- This may have been the intent, but no fair reading of RFC 9537 would lead an implementor to that conclusion:
  a. Abstract: “explicitly identifying .. using JSONPath”
  b. Section 1: “... explicitly identifying redacted RDAP response fields …”
  c. Section 1: “... explicitly specify which RDAP fields are not included …”
  d. 22 examples of redaction in the RFC, and every one of them uses JSONPath.
  e. Section 5 is dedicated to discussing JSONPath issues.
- Unclear if JSONPath is to be optional for clients but not servers.
  a. This presentation focuses on clients, but there is considerable effort to make servers compliant (see section 5.2).

OPTIONAL JSONPath is Redaction by Notice

That is, without JSONPath the redaction array in RFC 9537 is just an RDAP notice already defined in RFC 9083:

```
"notices" : [
  {
    “type” : “registrant name redacted”
    "description" : [
      "The registrant name was redacted."
    ]
  }
]
```

Except less flexible as RFC 9083 notices can be repeated in multiple languages and can contain links (such as to a redaction policy).

Redaction by Notice Can Be Bad UI

- Many RDAP clients visually separate different parts of an RDAP response.
  - It is common for RDAP clients to use UI features to nest (collapse / expand / indent) entities to make the responses easier to read.
- A notice, either as defined by RFC 9083 or by Redaction by Notice in RFC 9537, moves that information away from the data that has been redacted.
  - It would get mixed into terms of use, copyright notices, etc…
- Core RDAP also has a better solution: “remarks” which are co-located with objects.

Retrograde UI compared to Whois

```
$ whois -h whois.nic.nyc oneworld.nyc
Domain Name: oneworld.nyc
Registry Domain ID: D1267769-NYC
Registrar WHOIS Server: whois.godaddy.com
Registrar URL: whois.godaddy.com
Updated Date: 2022-10-14T19:17:28Z
Creation Date: 2014-10-09T17:08:53Z
Registry Expiry Date: 2024-10-08T23:59:59Z
Registrar: GoDaddy.com, LLC
Registrar IANA ID: 146
Registrar Abuse Contact Email: abuse@godaddy.com
Registrar Abuse Contact Phone: +1.4806242505
Domain Status: clientRenewProhibited https://icann.org/epp#clientRenewProhibited
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
Registry Registrant ID: REDACTED FOR PRIVACY
Registrant Name: REDACTED FOR PRIVACY
Registrant Organization: OneWorld Relocation Services
Registrant Street: REDACTED FOR PRIVACY
Registrant Street: REDACTED FOR PRIVACY
Registrant Street: REDACTED FOR PRIVACY
Registrant City: REDACTED FOR PRIVACY
Registrant State/Province: Florida
Registrant Postal Code: REDACTED FOR PRIVACY
Registrant Country: US
Registrant Phone: REDACTED FOR PRIVACY
Registrant Phone Ext: REDACTED FOR PRIVACY
Registrant Fax: REDACTED FOR PRIVACY
Registrant Fax Ext: REDACTED FOR PRIVACY
```

Redaction by Registered Notice

- `name.type` instead of `name.description`
- Display logic in the client is based solely on the implementors interpretation of the IANA registered value.
- Turns IANA registrations into pseudo-specifications.
- There would need to be registration restrictions to avoid confusion:
  - Same registration used for different redaction methods
  - Same registration used across multiple redacted fields.
- Not as flexible or precise as an algorithmic approach.
- Requires client implementors to periodically check for IANA updates.
  - May not be suitable to the needs of all types of registries.

Not Flexible for Nexus Policy / Contact

```
$ whois -h whois.nic.nyc oneworld.nyc
…
nyc ID: C1267762-NYC
nyc Name: REDACTED FOR PRIVACY
nyc Organization: REDACTED FOR PRIVACY
nyc Street: REDACTED FOR PRIVACY
nyc Street: REDACTED FOR PRIVACY
nyc Street: REDACTED FOR PRIVACY
nyc City: REDACTED FOR PRIVACY
nyc State/Province: REDACTED FOR PRIVACY
nyc Postal Code: REDACTED FOR PRIVACY
nyc Country: REDACTED FOR PRIVACY
nyc Phone: REDACTED FOR PRIVACY
nyc Phone Ext: REDACTED FOR PRIVACY
nyc Fax: REDACTED FOR PRIVACY
nyc Fax Ext: REDACTED FOR PRIVACY
nyc Email:
nyc Nexus Category: ORG
URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of WHOIS database: 2024-07-24T14:19:40Z <<<
```

Wait & See

- The RFC authors have also suggested that more time and implementation experience is needed.
    - Wait & See if better JSONPath implementations appear?
    - This does not address the limitations inherent in RFC 9537.
- Wait & See Specifications in the IETF are Experimental.
    - Should RFC 9537 be reclassified as Experimental?

Q&A

Questions and comments?

Let us consider the Tao of the IETF (RFC 4677):

> rough consensus and running code

and

> The protocol itself can be changed later for a number of reasons, the most common of which is that implementors discover a problem as they implement the standard.

It takes time, effort, and courage for implementors to provide feedback. It should not be dismissed.