Internet Engineering Task Force (IETF)
Request for Comments: 7071
Category: Standards Track
ISSN: 2070-1721

N. Borenstein
Mimecast
M. Kucherawy

November 2013

A Media Type for Reputation Interchange

Abstract

This document defines the format of reputation response data ("reputons"), the media type for packaging it, and definition of a registry for the names of reputation applications and response sets.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7071.

Table of Contents

1. Introduction

This document defines a data object for use when answering a reputation query. It also defines a media type to carry the response set data when using a transport method that follows the media type framework, such as the query method based on the HyperText Transfer Protocol (HTTP) defined in [RFC7072]. Any future query methods that might be developed are expected to use the same data object.

Also included is the specification for an IANA registry to contain definitions and symbolic names for known reputation applications and corresponding response sets.

2. Terminology and Definitions

This section defines terms used in the rest of the document.

2.1. Reputon

A "reputon" is a single independent object containing reputation information. A particular query about a subject of interest will receive one or more reputons in response, depending on the nature of the data collected and reported by the server.

2.2. Key Words

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 [KEYWORDS].

2.3. Other Definitions

Other terms of importance in this document are defined in [RFC7070], the base document in this document series.

3. Description

The meta-format selected for the representation of a reputon is JavaScript Object Notation (JSON), defined in [JSON]. Accordingly, a new media type, "application/reputon+json", is defined for the JSON representation of reputational data, typically in response to a client making a request for such data about some subject. This media type takes no parameters.

The body of the media type consists of a JSON document that contains the reputation information requested. A detailed description of the expected structure of the reply is provided below.

The media type comprises a single member indicating the name of the application context (see Section 5.1 of [RFC7070]) in which the reputational data are being returned. The application name refers to a registration as described in Section 7.2, which defines the valid assertions and any extensions that might also be valid (i.e., the response set) for that application.

3.1. Reputon Attributes

The key pieces of data found in a reputon for all reputation applications are defined as follows:

rater: The identity of the entity aggregating, computing, and providing the reputation information, typically expressed as a DNS domain name.

assertion: A key word indicating the specific assertion or claim being rated.

rated: The identity of the entity being rated. The nature of this field is application specific; it could be domain names, email addresses, driver's license numbers, or anything that uniquely identifies the entity being rated. Documents that define specific reputation applications are required to define syntax and semantics for this field.

rating: The overall rating score for that entity, expressed as a floating-point number between 0.0 and 1.0 inclusive. See Section 4 for discussion.

The following are OPTIONAL for all applications, to be used in contexts where they are appropriate:

confidence: the level of certainty the reputation provider has that the value presented is appropriate, expressed as a floating-point number between 0.0 and 1.0 inclusive.

normal-rating: An indication of what the reputation provider would normally expect as a rating for the subject. This allows the client to note that the current rating is or is not in line with expectations.

sample-size: The number of data points used to compute the rating, possibly an approximation. Expressed as an unsigned 64-bit integer. Consumers can assume that the count refers to distinct data points rather than a count of aggregations (for example, individual votes rather than aggregated vote counts) unless it is specified out-of-band that some other interpretation is more appropriate. The units are deliberately not normatively specified, since not all reputation service providers will collect data the same way.

generated: A timestamp indicating when this value was generated. Expressed as the number of seconds since January 1, 1970 00:00 UTC.

expires: A timestamp indicating a time beyond which the score reported is likely not to be valid. Expressed as the number of seconds since January 1, 1970 00:00 UTC. See Section 5 for discussion.

A particular application that registers itself with IANA (per Section 7.2, below) can define additional application-specific attribute/value pairs beyond these standard ones.

An application service provider might operate with an enhanced form of common services, which might in turn prompt development and reporting of specialized reputation information. The details of the enhancements and specialized information are beyond the scope of this document, except that the underlying JSON syntax is extensible for encoding such provider-specific information.

4. Ratings

The score presented as the value in the rating attribute appears as a floating-point value between 0.0 and 1.0 inclusive. The intent is that the definition of an assertion within an application will declare what the anchor values 0.0 and 1.0 specifically mean. Generally speaking, 1.0 implies full agreement with the assertion, while 0.0 indicates no support for the assertion.

The definition will also specify the type of scale in use when generating scores, to which all reputation service providers for that application space must adhere. Further discussion can be found in [RFC7070].

5. Caching

A reputon can contain an "expires" field indicating a timestamp after which the client SHOULD NOT use the rating it contains and SHOULD issue a new query.

This specification does not mandate any caching of ratings on the part of the client, but there are obvious operational benefits to doing so. In the context of reputation, a cached (and hence, stale) rating can cause desirable traffic to be identified as undesirable, or vice versa.

Reputation data is typically most volatile when the subject of the reputation is young. Accordingly, if a service chooses to include expiration timestamps as part a reply, these values SHOULD be lower for subjects about which little data has been collected.

6. Reputons

6.1. Syntax

A reputon expressed in JSON is a set of key-value pairs, where the keys are the names of particular attributes that comprise a reputon (as listed above, or as provided with specific applications), and values are the content associated with those keys. The set of keys that make up a reputon within a given application are known as that application's "response set".

A reputon object typically contains a reply corresponding to the assertion for which a client made a specific request. For example, a client asking for assertion "sends-spam" about domain "example.com" would expect a reply consisting of a reputon making a "sends-spam" assertion about "example.com" and nothing more. If a client makes a request about a subject but does not specify an assertion of interest, the server can return reputons about any assertion for which it has data; in effect, the client has asked for any available information about the subject. A client that receives an irrelevant reputon simply ignores it.

An empty reputon is an acknowledgment by the server that the request has been received, and serves as a positive indication that the server does not have the information requested. This is semantically equivalent to returning a reputon with a "sample-size" of zero.

6.2. Formal Definition

[JSON] defines the structure of JSON objects and arrays using a set of primitive elements. Those elements will be used to describe the JSON structure of a reputation object.

6.2.1. Imported JSON Terms

OBJECT: a JSON object, defined in Section 2.2 of [JSON]

MEMBER: a member of a JSON object, defined in Section 2.2 of [JSON]

MEMBER-NAME: the name of a MEMBER, defined as a "string" in Section 2.2 of [JSON]

MEMBER-VALUE: the value of a MEMBER, defined as a "value" in Section 2.2 of [JSON]

ARRAY: an array, defined in Section 2.3 of [JSON]

ARRAY-VALUE: an element of an ARRAY, defined in Section 2.3 of [JSON]

NUMBER: a "number" as defined in Section 2.4 of [JSON]

INTEGER: an "integer" as defined in Section 2.4 of [JSON]

STRING: a "string" as defined in Section 2.5 of [JSON]

6.2.2. Reputon Structure

Using the above terms for the JSON structures, the syntax of a reputation object is defined as follows:

reputation-object: an OBJECT containing a MEMBER reputation-context and a MEMBER reputon-list

reputation-context: a MEMBER with MEMBER-NAME "application" and MEMBER-VALUE a STRING (see Section 3)

reputon-list: a MEMBER with MEMBER-NAME "reputons" and MEMBER-VALUE a reputon-array

reputon-array: an ARRAY, where each ARRAY-VALUE is a reputon

reputon: an OBJECT, where each MEMBER is a reputon-element

reputon-element: one of the following, defined below: rater-value, assertion-value, rated-value, rating-value, conf-value, normal-value, sample-value, gen-value, expire-value, ext-value; note the following:

* The order of reputon-element members is not significant.

* A specific reputon-element MUST NOT appear more than once.

* rater-value, assertion-value, rated-value, and rating-value are REQUIRED.

rater-value: a MEMBER with MEMBER-NAME "rater" and MEMBER-VALUE a STRING (see "rater" in Section 3.1)

assertion-value: a MEMBER with MEMBER-NAME "assertion" and MEMBER-VALUE a STRING (see "assertion" in Section 3.1)

rated-value: a MEMBER with MEMBER-NAME "rated" and MEMBER-VALUE a STRING (see "rated" in Section 3.1)

rating-value: a MEMBER with MEMBER-NAME "rating" and MEMBER-VALUE a NUMBER between 0.0 and 1.0 inclusive (see "rating" in Section 3.1); the number SHOULD NOT not have more than three decimal places of precision

conf-value: a MEMBER with MEMBER-NAME "confidence" and MEMBER-VALUE a NUMBER between 0.0 and 1.0 inclusive (see "confidence" in Section 3.1); the number SHOULD NOT not have more than three decimal places of precision

normal-value: a MEMBER with MEMBER-NAME "normal-rating" and MEMBER-VALUE a NUMBER between 0.0 and 1.0 inclusive (see "normal" in Section 3.1); the number SHOULD NOT not have more than three decimal places of precision

sample-value: a MEMBER with MEMBER-NAME "sample-size" and MEMBER-VALUE a non-negative INTEGER (see "sample-size" in "normal" in Section 3.1)

gen-value: a MEMBER with MEMBER-NAME "generated" and MEMBER-VALUE a non-negative INTEGER (see "generated" in Section 3.1)

expire-value: a MEMBER with MEMBER-NAME "expires" and MEMBER-VALUE a non-negative INTEGER (see "expires" in Section 3.1)

ext-value: a MEMBER, for extension purposes; MEMBER-NAME and MEMBER-VALUE will be defined in separate application registrations

6.3. Examples

The following simple example:

```
Content-Type: application/reputon+json

{
  "application": "baseball",
  "reputons": [
    {
      "rater": "RatingsRUs.example.com",
      "assertion": "is-good",
      "rated": "Alex Rodriguez",
      "rating": 0.99,
      "sample-size": 50000
    }
  ]
}
```

...indicates to the client that "RatingsRUs.example.com" consolidated 50000 data points (perhaps from everyone in Yankee Stadium) and concluded that Alex Rodriguez is very, very good (0.99) at something. It doesn't tell us what he's good at, and while it might be playing baseball, it could just as well be paying his taxes on time.

A more sophisticated usage would define a baseball application with a response set of specific assertions, so that this example:

```
Content-Type: application/reputon+json

{
  "application": "baseball",
  "reputons:" [
    {
      "rater": "baseball-reference.example.com",
      "assertion": "hits-for-power",
      "rated": "Alex Rodriguez",
      "rating": 0.99,
      "sample-size": 50000
    }
  ]
}
```

...would indicate that 50000 fans polled by the entity baseball-reference.example.com rate Alex Rodriguez very highly in hitting for power, whereas this example:

```
Content-Type: application/reputon+json

{
  "application": "baseball",
  "reputons": [
    {
      "rater": "baseball-reference.example.com",
      "assertion": "strong-hitter",
      "rated": "Alex Rodriguez",
      "rating": 0.4,
      "confidence": 0.2,
      "sample-size": 50000
    }
  ]
}
```

...would indicate that a similar poll indicated a somewhat weak consensus that Alex Rodriguez tends to fail in critical batting situations during baseball games.

The following is an example reputon generated using this schema, including the media type definition line that identifies a specific reputation application context. Here, reputation agent "rep.example.net" is asserting within the context of the "email-id" application (see [RFC7073]) that "example.com" appears to be associated with spam 1.2% of the time, based on just short of 17 million messages analyzed or reported to date. The "email-id" application has declared the extension key "email-id-identity" to indicate how the subject identifier was used in the observed data, establishing some more-specific semantics for the "rating" value. In this case, the extension is used to show the identity "example.com", the subject of the query, is extracted from the analyzed messages using the DomainKeys Identified Mail [DKIM] "d=" parameter for messages where signatures validate. The reputation agent is 95% confident of this result. A second reputon is also present indicating similar information for the same domain as it is used in the context of Sender Policy Framework [SPF] evaluations. (See [RFC7073] for details about the registered email identifiers application.)

```
Content-Type: application/reputon+json

{
  "application": "email-id",
  "reputons": [
    {
      "rater": "rep.example.net",
      "assertion": "spam",
      "identity": "dkim",
      "rated": "example.com",
      "confidence": 0.95,
      "rating": 0.012,
      "sample-size": 16938213,
      "updated": 1317795852
    },
    {
      "rater": "rep.example.net",
      "assertion": "spam",
      "identity": "spf",
      "rated": "example.com",
      "confidence": 0.98,
      "rating": 0.023,
      "sample-size": 16938213,
      "updated": 1317795852
    }
  ]
}
```

7. IANA Considerations

This document presents two actions for IANA -- namely, the creation of the new media type "application/reputon+json" and the creation of a registry for reputation application types. Another document in this series creates an initial registry entry for the latter.

7.1. application/reputon+json Media Type Registration

This section provides the media type registration application from [MIME-REG] for processing by IANA.

To: media-types@iana.org

Subject: Registration of media type application/reputon+json

Type name: application

Subtype name: reputon+json

Required parameters: none

Optional parameters: none

Encoding considerations: "7bit" encoding is sufficient and is used to maintain readability when viewed by non-MIME mail readers.

Security considerations: See Section 8 of [RFC7071].

Interoperability considerations: Implementers may encounter "app" values, attribute/value pairs, or response set items that they do not support, which are to be ignored.

Published specification: [RFC7071]

Applications that use this media type: Any application that wishes to query a service that provides reputation data using the form defined in [RFC7072]. The example application is one that provides reputation data about DNS domain names and other identifiers found in email messages.

Fragment identifier considerations: N/A

Additional information: The value of the "app" parameter is registered with IANA.

Deprecated alias names for this type: N/A

Magic number(s): N/A

File extension(s): N/A

Macintosh file type code(s): N/A

Person and email address to contact for further information:

Murray S. Kucherawy <superuser@gmail.com>

Intended usage: COMMON

Restrictions on usage: N/A

Author:
Nathaniel Borenstein
Murray S. Kucherawy

Change controller: IESG

Provisional registration?: no

7.2. Reputation Applications Registry

IANA has created the "Reputation Applications" registry. This registry contains names of applications used with the application/reputon+json media type (and other media types that carry reputons), as defined by this document.

New registrations or updates are published in accordance with either the "IETF Review" or "Specification Required" guidelines as described in [IANA-CONSIDERATIONS].

New registrations and updates are to contain the following information:

1. Symbolic name of the application being registered or updated. Valid names conform to the ABNF construction "token" as defined in Multipurpose Internet Mail Extensions (MIME) Part One [MIME]

2. Short description of the application (i.e., the class of entity about which it reports reputation data)

3. The document in which the application is defined

4. New or updated status, which is to be one of:

   current: The application is in current use

   deprecated: The application is in current use but its use is discouraged

   historic: The application is no longer in current use

A specification for an application space needs to be specific and clear enough to allow interoperability, and include at least the following details:

- The application’s symbolic name, as it appears in the registration (see above)

- A description of the subject of a query within this reputation, and a legal syntax for the same

- An optional table of query parameters that are specific to this application; each table entry must include:

  Name: Name of the query parameter

  Status: (as above)

  Description: A short description of the purpose of this parameter

  Syntax: A reference to a description of valid syntax for the parameter's value

  Required: "yes" if the parameter is mandatory; "no" otherwise

- A list of one or more assertions registered within this application; each table entry is to include:

  Name: Name of the assertion

  Description: A short description of the assertion, with specific meanings for values of 0.0 and 1.0

  Scale: A short description of the scale used in computing the value (see Section 4 of this document)

- An optional list of one or more response set extension keys for use within this application; each table entry is to include:

  Name: Name of the extension key

  Description: A short description of the key's intended meaning

  Syntax: A description of valid values that can appear associated with the key

The names of attributes registered should be prefixed by the name of the application itself (e.g., the "foo" application registering a "bar" attribute should call it "foo-bar") to avoid namespace collisions.

For registrations qualifying under "Specification Required" rules, the Designated Expert [IANA-CONSIDERATIONS] should confirm the document meets the minima described above and otherwise looks generally acceptable, and then approve the registration.

8. Security Considerations

This document is primarily an IANA action registering a media type. It does not describe a new protocol that might introduce security considerations.

Discussion of the security and operational impacts of using reputation services in general can be found throughout [CONSIDERATIONS].

9. References

9.1. Normative References

[JSON] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.

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

[RFC7070] Borenstein, N., Kucherawy, M., and A. Sullivan, "An Architecture for Reputation Reporting", RFC 7070, November 2013.

[RFC7072] Borenstein, N. and M. Kucherawy, "A Reputation Query Protocol", RFC 7072, November 2013.

9.2. Informative References

[CONSIDERATIONS] Kucherawy, M., "Operational Considerations Regarding Reputation Services", Work in Progress, May 2013.

[DKIM] Crocker, D., Ed., Hansen, T., Ed., and M. Kucherawy, Ed., "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, September 2011.

[IANA-CONSIDERATIONS] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[MIME-REG] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, January 2013.

[MIME] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC7073] Borenstein, N. and M. Kucherawy, "A Reputation Response Set for Email Identifiers", RFC 7073, November 2013.

[SPF] Wong, M. and W. Schlitt, "Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1", RFC 4408, April 2006.

Appendix A. Acknowledgments

The authors wish to acknowledge the contributions of the following to this specification: Frank Ellermann, Tony Hansen, Jeff Hodges, Simon Hunt, John Levine, David F. Skoll, and Mykyta Yevstifeyev.

Authors' Addresses

Nathaniel Borenstein
Mimecast
203 Crescent St., Suite 303
Waltham, MA 02453
USA

Phone: +1 781 996 5340
EMail: nsb@guppylake.com

Murray S. Kucherawy
270 Upland Drive
San Francisco, CA 94127
USA

EMail: superuser@gmail.com