

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 4, 2014

F. Ben Jemaa  
G. Pujolle  
Computer Science Laboratory  
of Paris 6  
M. Pariente  
Meteor Network  
July 4, 2014

Labels for common venue-based services  
draft-benjemaa-vbs-urn-00

Abstract

This document describes a service-identifying label that allows venue-based services that are locally offered by the venue owner to be identified. In particular, we define labels for the most known and common venue services (e.g. mapping and printer). These labels can be used by IEEE 802.11 and service discovery protocols to discover the available services specific to the venue and accessible via the Wireless LAN.

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 <http://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."

Copyright Notice

This Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|  |   |
|--|---|
| 1. Introduction . . . . .                      | 2 |
| 2. Conventions used in this document . . . . . | 3 |
| 3. Venue-based services. . . . .               | 3 |
| 4. Security Considerations . . . . .           | 4 |
| 5. IANA Considerations . . . . .               | 4 |
| 6. References . . . . .                        | 4 |

## 1. Introduction

Increasingly, local value-added services are deployed in airports, shopping malls, and other public spaces to improve customer experience. Although these services depend on the venue type and the specific user needs in this venue, there is a set of well-known and common services often required by customers in such indoor area (e.g. indoor mapping).

To enable global and unique identification of such services, it is useful to define a common set of terms, so that the same service is labeled with the same identifier regardless of the actual service deployment which may depends on the venue type.

Availability of such venue service identifiers facilitates service advertisement and discovery. Indeed, it allows network entities to convey information about the available services to user devices while ensuring consistency and compatibility between devices and service providers. Thus, it allows a user device to recognize the desired services among the received information according to the defined user preferences. This allows for more automatism and transparency relative to end-users.

In addition, these URN labels identify services independent of the particular protocol using these identifiers. In particular, it may appear in IEEE 802.11, service discovery, and mapping protocols. For example, for mapping service URNs to URLs, Location to Service Translation Protocol (LoST) [RFC5222] could be used as a resolution system based on geographic location.

Finally, as URN identifiers are extensible, these venue service identifiers may contain a hierarchy of sub-services that further describe the service.

## 2. 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].

## 3. Venue-based services

In this section, we define new URN service labels to identify venue-based services using the "service" URN namespace defined in [RFC5031]. We propose to add a new top-level service label "vbs" (referring to venue-based services).

urn:service:vbs The generic "vbs" service type encompasses all of the services offered by the venue.

We define also additional sub-services corresponding to the most well-known and common indoor venue services which are of general public interest.

urn:service:vbs:mapping The "mapping" service refers to indoor localization and way finding using the venue map.

urn:service:vbs:discount The "discount" service refers to discounts and special deals proposed by the venue (e.g. discount offered by a restaurant , promotions offered by a shopping mall, special price reduction offered by an hotel).

urn:service:vbs:printer The "printer" service refers to printing services that can be offered by the venue such as hotel, library, etc.

urn:service:vbs:info The "info" service gives information related or about the visited venue. For example, in a shopping mall, it gives the list of available shops, brands, restaurants in this mall. Or in the airport, travelers could use this service to access information about their fly.

urn:service:vbs:video The "video" service refers to video streaming or download service offered by a certain venues. For example, in a stadium, it gives exclusive in-venue content such as replays and live video streaming.

#### 4. Security Considerations

This document defines service-identifying labels for venue-based services and this does not raise security issues.

#### 5. IANA Considerations

Services and sub-services are identified by labels managed by IANA, according to the processes outlined in [RFC5226] in a registry called "Service URN Labels". Thus, creating a new service requires IANA action.

This document defines new service URNs in the Service URN Labels registry according to [RFC5031] for venue-based services as listed in Section 3.

[TO BE REMOVED: This registration should take place at the following location: <http://www.iana.org/assignments/urn-serviceid-labels/urn-serviceid-labels.xhtml>]

The following table contains the initial IANA registrations for venue-based services.

| Service      | Reference | Description                               |
|--------------|-----------|---|
| -----        |           |   |
| vbs          | RFC xxxx  | Venue-based services                      |
| vbs.mapping  | RFC xxxx  | Indoor mapping service                    |
| vbs.discount | RFC xxxx  | Discount offers                           |
| vbs.printer  | RFC xxxx  | Printer service                           |
| vbs.info     | RFC xxxx  | Venue-related information service         |
| vbs.video    | RFC xxxx  | Local video download or streaming service |

#### 6. References

##### 6.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5031] Schulzrinne, H., "A Uniform Resource Name (URN) for Emergency and Other Well-Known Services", RFC 5031, January 2008.
- [RFC5226] Narten, T. and H.Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

## 6.2. Informative References

- [RFC5222] Hardie, T., Newton, A., Schulzrinne, H., and H. Tschofenig, "LoST: A Location-to-Service Translation Protocol", RFC 5222, August 2008.

## Authors' Addresses

Fatma Ben Jemaa  
University of Pierre & Marie Curie  
Computer Science Laboratory of Paris 6  
4, Place Jussieu  
Paris 75005  
France  
Email: fatma.ben-jemaa@lip6.fr

Guy Pujolle  
University of Pierre & Marie Curie  
Computer Science Laboratory of Paris 6  
4, Place Jussieu  
Paris 75005  
France  
Email: guy.pujolle@lip6.fr

Michel Pariente  
Meteor Network  
40, rue du general Malleret-Joinville  
Vitry-sur-Seine 94420  
France  
Email: mpariente@meteornetworks.com

Network Working Group  
Internet-Draft  
Obsoletes: 3798 (if approved)  
Intended status: Standards Track  
Expires: January 4, 2015

T. Hansen, Ed.  
AT&T Laboratories  
A. Melnikov, Ed.  
Isode Ltd  
July 3, 2014

Message Disposition Notification  
draft-hansen-mdn-3798bis-02.txt

Abstract

This memo defines a MIME content-type that may be used by a mail user agent (MUA) or electronic mail gateway to report the disposition of a message after it has been successfully delivered to a recipient. This content-type is intended to be machine-processable. Additional message header fields are also defined to permit Message Disposition Notifications (MDNs) to be requested by the sender of a message. The purpose is to extend Internet Mail to support functionality often found in other messaging systems, such as X.400 and the proprietary "LAN-based" systems, and often referred to as "read receipts," "acknowledgements", or "receipt notifications." The intention is to do this while respecting privacy concerns, which have often been expressed when such functions have been discussed in the past.

Because many messages are sent between the Internet and other messaging systems (such as X.400 or the proprietary "LAN-based" systems), the MDN protocol is designed to be useful in a multi-protocol messaging environment. To this end, the protocol described in this memo provides for the carriage of "foreign" addresses, in addition to those normally used in Internet Mail. Additional attributes may also be defined to support "tunneling" of foreign notifications through Internet Mail.

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 <http://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 January 4, 2015.

#### Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

#### Table of Contents

|  |    |
|--|----|
| 1. Introduction . . . . .  | 3  |
| 1.1. Purposes . . . . .  | 3  |
| 1.2. Requirements . . . . .                                      | 4  |
| 1.3. Terminology . . . . .                                       | 5  |
| 2. Requesting Message Disposition Notifications . . . . .        | 5  |
| 2.1. The Disposition-Notification-To Header . . . . .            | 5  |
| 2.2. The Disposition-Notification-Options Header . . . . .       | 7  |
| 2.3. The Original-Recipient Header . . . . .                     | 8  |
| 2.4. Use with the Message/Partial Content Type . . . . .         | 9  |
| 3. Format of a Message Disposition Notification . . . . .        | 9  |
| 3.1. The message/disposition-notification content-type . . . . . | 11 |
| 3.2. Message/disposition-notification Fields . . . . .           | 13 |
| 3.3. Extension-fields . . . . .                                  | 18 |
| 4. Timeline of events . . . . .                                  | 19 |
| 5. Conformance and Usage Requirements . . . . .                  | 20 |
| 6. Security Considerations . . . . .                             | 21 |
| 6.1. Forgery . . . . .   | 21 |

|                    |   |    |
|--------------------|---|----|
| 6.2.               | Privacy . . . . .   | 21 |
| 6.3.               | Non-Repudiation . . . . .   | 22 |
| 6.4.               | Mail Bombing . . . . .  | 22 |
| 7.                 | Collected Grammar . . . . .   | 22 |
| 8.                 | Guidelines for Gatewaying MDNs . . . . .  | 24 |
| 8.1.               | Gatewaying from other mail systems to MDNs . . . . .  | 24 |
| 8.2.               | Gatewaying from MDNs to other mail systems . . . . .  | 25 |
| 8.3.               | Gatewaying of MDN-requests to other mail systems . . . . .  | 25 |
| 9.                 | Example . . . . .   | 26 |
| 10.                | IANA Considerations . . . . .   | 27 |
| 10.1.              | Disposition-Notification-Options header field<br>disposition-notification-parameter names . . . . . | 27 |
| 10.2.              | Disposition modifier names . . . . .  | 28 |
| 10.3.              | MDN extension field names . . . . .   | 28 |
| 11.                | Acknowledgements . . . . .  | 28 |
| 12.                | References . . . . .  | 29 |
| 12.1.              | Normative References . . . . .  | 29 |
| 12.2.              | Informative References . . . . .  | 29 |
| Appendix A.        | Changes from RFC 3798 . . . . .   | 29 |
| Authors' Addresses | . . . . .   | 32 |

## 1. Introduction

This memo defines a RFC-MIME-MEDIA [4] content-type for message disposition notifications (MDNs). An MDN can be used to notify the sender of a message of any of several conditions that may occur after successful delivery, such as display of the message contents, printing of the message, deletion (without display) of the message, or the recipient's refusal to provide MDNs. The "message/disposition-notification" content-type defined herein is intended for use within the framework of the "multipart/report" content type defined in RFC-REPORT [6].

This memo defines the format of the notifications and the RFC-MSGFMT [2] header fields used to request them.

This memo is an update to RFC 3798 and is intended to be published at Internet Standard Level.

This memo is currently marked with the 'pre5378Trust200902' IPR statements until a release has been obtained from all previous authors and editors of this text.

### 1.1. Purposes

The MDNs defined in this memo are expected to serve several purposes:

- a. Inform human beings of the disposition of messages after successful delivery, in a manner that is largely independent of human language;
- b. Allow mail user agents to keep track of the disposition of messages sent, by associating returned MDNs with earlier message transmissions;
- c. Convey disposition notification requests and disposition notifications between Internet Mail and "foreign" mail systems via a gateway;
- d. Allow "foreign" notifications to be tunneled through a MIME-capable message system and back into the original messaging system that issued the original notification, or even to a third messaging system;
- e. Allow language-independent, yet reasonably precise, indications of the disposition of a message to be delivered.

## 1.2. Requirements

These purposes place the following constraints on the notification protocol:

- a. It must be readable by humans, and must be machine-parsable.
- b. It must provide enough information to allow message senders (or their user agents) to unambiguously associate an MDN with the message that was sent and the original recipient address for which the MDN was issued (if such information is available), even if the message was forwarded to another recipient address.
- c. It must also be able to describe the disposition of a message independent of any particular human language or of the terminology of any particular mail system.
- d. The specification must be extensible in order to accommodate future requirements.

### 1.3. Terminology

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 RFC-KEYWORDS [9].

All syntax descriptions use the ABNF specified by RFC-MSGFMT [2], in which the lexical tokens (used below) are defined: "atom", "CRLF", "FWS", "CFWS", "field-name", "mailbox", "msg-id", and "text". The following lexical tokens are defined in the definition of the Content-Type header field in RFC-MIME-BODY [3]: "attribute" and "value".

## 2. Requesting Message Disposition Notifications

Message disposition notifications are requested by including a Disposition-Notification-To header field in the message. Further information to be used by the recipient's MUA in generating the MDN may be provided by also including Original-Recipient and/or Disposition-Notification-Options header fields in the message.

### 2.1. The Disposition-Notification-To Header

A request for the receiving user agent to issue message disposition notifications is made by placing a Disposition-Notification-To header field into the message. The syntax of the header field is

```
mdn-request-header = "Disposition-Notification-To" ":" [FWS]  
                    mailbox *("," [FWS] mailbox)
```

The presence of a Disposition-Notification-To header field in a message is merely a request for an MDN. The recipients' user agents are always free to silently ignore such a request.

An MDN MUST NOT itself have a Disposition-Notification-To header field. An MDN MUST NOT be generated in response to an MDN.

A user agent MUST NOT issue more than one MDN on behalf of each particular recipient. That is, once an MDN has been issued on behalf of a recipient, no further MDNs may be issued on behalf of that recipient, even if another disposition is performed on the message. However, if a message is forwarded, an MDN may have been issued for the recipient doing the forwarding and the recipient of the forwarded message may also cause an MDN to be generated.

While Internet standards normally do not specify the behavior of user interfaces, it is strongly recommended that the user agent obtain the user's consent before sending an MDN. This consent could be obtained

for each message through some sort of prompt or dialog box, or globally through the user's setting of a preference.

MDNs SHOULD NOT be sent automatically if the address in the Disposition-Notification-To header field differs from the address in the Return-Path header field (see RFC-MSGFMT [2]). In this case, confirmation from the user SHOULD be obtained, if possible. If obtaining consent is not possible (e.g., because the user is not online at the time), then an MDN SHOULD NOT be sent.

Confirmation from the user SHOULD be obtained (or no MDN sent) if there is no Return-Path header field in the message, or if there is more than one distinct address in the Disposition-Notification-To header field.

The comparison of the addresses should be done using only the addr-spec (local-part "@" domain) portion, excluding any angle brackets, phrase and route. The comparison MUST be case-sensitive for the local-part and case-insensitive for the domain part. [[ more work needed here ]]

[[CREF1: (From Bruce) the domains might differ, yet refer to the same place (equivalent MX mail exchangers, A vs. CNAME DNS records, DNS names vs. domain literals, etc.) These are not addressed in 3798. ]]

[[CREF2: (From Bruce) local-parts and domains might differ as literal text, but be equivalent when put in canonical form. The issues are discussed in RFC 3696 -- but beware -- 3696 has a number of errors; refer to RFC 5322 for the actual quoting and escaping rules. ]]

[[CREF3: (From Bruce) internationalization issues might further compound comparison issues between local-parts and domains (specifying that the on-the-wire forms must be compared might suffice) ]]

[[CREF4: (From Bruce) there exist some conventions (not standardized as far as I know) regarding subaddressing applied to local parts, e.g. as in tony+rfc3798@maillennium.att.com (that example also illustrates an issue regarding subdomains) ]]

[[CREF5: (From Bruce) Of those, the angle bracket issue ought to be understood, but clarification could benefit implementors, especially as RFC 5322 defined the Return-Path syntax somewhat peculiarly. Canonicalization of local-parts and domains should probably be required prior to comparison, and use of on-the-wire forms should probably also be specified. DNS equivalence issues might be tricky for some implementations (e.g. offline reading); perhaps the

specification could use RFC 2119 "MAY" to give implementations leeway to consider A vs. CNAME and DNS vs domain literal equivalence for situations where DNS is available to the implementation (I'm not sure about MX). About the only thing that can be said w.r.t. subaddressing and subdomains is a caution to sending MUA and address-rewriting MTA authors that a mismatch might result in no MDN being produced. ]]

If the message contains more than one Return-Path header field, the implementation may pick one to use for the comparison, or treat the situation as a failure of the comparison.

The reason for not automatically sending an MDN if the comparison fails or more than one address is specified is to reduce the possibility of mail loops and of MDNs being used for mail bombing.

A message that contains a Disposition-Notification-To header field SHOULD also contain a Message-ID header field as specified in RFC-MSGFMT [2]. This will permit automatic correlation of MDNs with their original messages by user agents.

If the request for message disposition notifications for some recipients and not others is desired, two copies of the message should be sent, one with a Disposition-Notification-To header field and one without. Many of the other header fields of the message (e.g., To, Cc) will be the same in both copies. The recipients in the respective message envelopes determine for whom message disposition notifications are requested and for whom they are not. If desired, the Message-ID header field may be the same in both copies of the message. Note that there are other situations (e.g., Bcc) in which it is necessary to send multiple copies of a message with slightly different header fields. The combination of such situations and the need to request MDNs for a subset of all recipients may result in more than two copies of a message being sent, some with a Disposition-Notification-To header field and some without.

Messages posted to newsgroups SHOULD NOT have a Disposition-Notification-To header field.

## 2.2. The Disposition-Notification-Options Header

Future extensions to this specification may require that information be supplied to the recipient's MUA for additional control over how and what MDNs are generated. The Disposition-Notification-Options header field provides an extensible mechanism for such information. The syntax of this header field is as follows:

```

Disposition-Notification-Options =
    "Disposition-Notification-Options" ":" [FWS]
        disposition-notification-parameter-list
disposition-notification-parameter-list =
    disposition-notification-parameter
    *(";" [FWS] disposition-notification-parameter)
disposition-notification-parameter = attribute [FWS] "="
    [FWS] importance "," [FWS] value *("," [FWS] value)
importance = "required" / "optional"

```

An importance of "required" indicates that interpretation of the disposition-notification-parameter is necessary for proper generation of an MDN in response to this request. An importance of "optional" indicates that an MUA that does not understand the meaning of this disposition-notification-parameter MAY generate an MDN in response anyway, ignoring the value of the disposition-notification-parameter.

No disposition-notification-parameter attribute names are defined in this specification. Attribute names may be defined in the future by later revisions or extensions to this specification. Disposition-notification-parameter attribute names beginning with "X-" will never be defined as standard names; such names are reserved for experimental use. disposition-notification-parameter attribute names not beginning with "X-" MUST be registered with the Internet Assigned Numbers Authority (IANA) and described in a standards-track RFC or an experimental RFC approved by the IESG. [[ more work needed here ]] (See Section 10 for a registration form.)

### 2.3. The Original-Recipient Header

Since electronic mail addresses may be rewritten while the message is in transit, it is useful for the original recipient address to be made available by the delivering MTA. The delivering MTA may be able to obtain this information from the ORCPT parameter of the SMTP RCPT TO command, as defined in RFC-SMTP [1] and RFC-DSN-SMTP [7].

RFC-DSN-SMTP [7] is amended as follows: If the ORCPT information is available, the delivering MTA SHOULD insert an Original-Recipient header field at the beginning of the message (along with the Return-Path header field). The delivering MTA MAY delete any other Original-Recipient header fields that occur in the message. The syntax of this header field is as follows:

```

original-recipient-header =
    "Original-Recipient" ":" address-type ";" generic-address

```

The address-type and generic-address token are as specified in the description of the Original-Recipient field in Section 3.2.3.

The purpose of carrying the original recipient information and returning it in the MDN is to permit automatic correlation of MDNs with the original message on a per-recipient basis.

#### 2.4. Use with the Message/Partial Content Type

The use of the header fields `Disposition-Notification-To`, `Disposition-Notification-Options`, and `Original-Recipient` with the MIME message/partial content type (RFC-MIME-MEDIA [4]) requires further definition.

When a message is segmented into two or more message/partial fragments, the three header fields mentioned in the above paragraph SHOULD be placed in the "inner" or "enclosed" message (using the terms of RFC-MIME-MEDIA [4]). These header fields SHOULD NOT be used in the header fields of any of the fragments themselves.

When the multiple message/partial fragments are reassembled, the following applies. If these header fields occur along with the other header fields of a message/partial fragment message, they pertain to an MDN that will be generated for the fragment. If these header fields occur in the header fields of the "inner" or "enclosed" message (using the terms of RFC-MIME-MEDIA [4]), they pertain to an MDN that will be generated for the reassembled message. Section 5.2.2.1 of RFC-MIME-MEDIA [4] is amended to specify that, in addition to the header fields specified there, the three header fields described in this specification are to be appended, in order, to the header fields of the reassembled message. Any occurrences of the three header fields defined here in the header fields of the initial enclosing message must not be copied to the reassembled message.

#### 3. Format of a Message Disposition Notification

A message disposition notification is a MIME message with a top-level content-type of `multipart/report` (defined in RFC-REPORT [6]). When `multipart/report` content is used to transmit an MDN:

- a. The `report-type` parameter of the `multipart/report` content is `"disposition-notification"`.
- b. The first component of the `multipart/report` contains a human-readable explanation of the MDN, as described in RFC-REPORT [6].

- c. The second component of the multipart/report is of content-type message/disposition-notification, described in Section 3.1 of this document.
- d. If the original message or a portion of the message is to be returned to the sender, it appears as the third component of the multipart/report. The decision of whether or not to return the message or part of the message is up to the MUA generating the MDN. However, in the case of encrypted messages requesting MDNs, encrypted message text **MUST** be returned, if it is returned at all, only in its original encrypted form.

NOTE: For message disposition notifications gatewayed from foreign systems, the header fields of the original message may not be available. In this case, the third component of the MDN may be omitted, or it may contain "simulated" RFC-MSGFMT [2] header fields that contain equivalent information. In particular, it is very desirable to preserve the subject and date fields from the original message.

The MDN **MUST** be addressed (in both the message header field and the transport envelope) to the address(es) from the Disposition-Notification-To header field from the original message for which the MDN is being generated.

The From field of the message header field of the MDN **MUST** contain the address of the person for whom the message disposition notification is being issued.

The envelope sender address (i.e., SMTP MAIL FROM) of the MDN **MUST** be null (<>), specifying that no Delivery Status Notification messages or other messages indicating successful or unsuccessful delivery are to be sent in response to an MDN.

A message disposition notification **MUST NOT** itself request an MDN. That is, it **MUST NOT** contain a Disposition-Notification-To header field.

The Message-ID header field (if present) for an MDN **MUST** be different from the Message-ID of the message for which the MDN is being issued.

A particular MDN describes the disposition of exactly one message for exactly one recipient. Multiple MDNs may be generated as a result of one message submission, one per recipient. However, due to the circumstances described in Section 2.1, MDNs may not be generated for some recipients for which MDNs were requested.

### 3.1. The message/disposition-notification content-type

The message/disposition-notification content-type is defined as follows:

MIME type name: message

MIME subtype name: disposition-notification

Optional parameters: none

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

Security considerations: discussed in Section 6 of this memo.

(While the 7bit restriction applies to the message/disposition-notification portion of the multipart/report content, it does not apply to the optional third portion of the multipart/report content.)

The message/disposition-notification report type for use in the multipart/report is "disposition-notification".

The body of a message/disposition-notification consists of one or more "fields" formatted according to the ABNF of RFC-MSGFMT [2] header "fields". The syntax of the message/disposition-notification content is as follows:

```
disposition-notification-content = [ reporting-ua-field CRLF ]
                                   [ mdn-gateway-field CRLF ]
                                   [ original-recipient-field CRLF ]
                                   final-recipient-field CRLF
                                   [ original-message-id-field CRLF ]
                                   disposition-field CRLF
                                   *( failure-field CRLF )
                                   *( error-field CRLF )
                                   *( extension-field CRLF )
extension-field = extension-field-name ":" *(CFWS / text)
extension-field-name = field-name

[[ more work needed here ]]
```

[[CREF6: Is this wording okay ? ]] Note that the order of the above fields is fixed.

### 3.1.1. General conventions for fields

Since these fields are defined according to the rules of RFC-MSGFMT [2], the same conventions for continuation lines and comments apply. Notification fields may be continued onto multiple lines by beginning each additional line with a SPACE or HTAB. Text that appears in parentheses is considered a comment and not part of the contents of that notification field. Field names are case-insensitive, so the names of notification fields may be spelled in any combination of upper and lower case letters. Comments in notification fields may use the "encoded-word" construct defined in RFC-MIME-HEADER [5].

### 3.1.2. "-type" subfields

Several fields consist of a "-type" subfield, followed by a semi-colon, followed by "\*text". [[ more work needed here ]]  
[[CREF7: ( Shouldn't this allow FWS somehow? Alexey: yes! ) ]]  
[[CREF8: ( I see that address-type and mta-name-type uses atom instead of \*text, which not only permits FWS, but goes further to allow CFWS. ) ]]  
For these fields, the keyword used in the address-type or MTA-type subfield indicates the expected format of the address or MTA-name that follows.

The "-type" subfields are defined as follows:

- a. An "address-type" specifies the format of a mailbox address. For example, Internet Mail addresses use the "rfc822" address-type.

address-type = atom

[[ more work needed here ]]  
[[CREF9: This is not \*text ]]

- b. An "MTA-name-type" specifies the format of a mail transfer agent name. For example, for an SMTP server on an Internet host, the MTA name is the domain name of that host, and the "dns" MTA-name-type is used.

mta-name-type = atom

[[ more work needed here ]]  
[[CREF10: This is not \*text ]]

Values for address-type and mta-name-type are case-insensitive. Thus, address-type values of "RFC822" and "rfc822" are equivalent.

The Internet Assigned Numbers Authority (IANA) maintains a registry of address-type and mta-name-type values, along with descriptions of the meanings of each, or a reference to one or more specifications that provide such descriptions. (The "rfc822" address-type is defined in RFC-DSN-SMTP [7].) Registration forms for address-type and mta-name-type appear in RFC-DSN-FORMAT [8].

### 3.2. Message/disposition-notification Fields

#### 3.2.1. The Reporting-UA field

```
reporting-ua-field = "Reporting-UA" ":" ua-name [ ";" ua-product ]
ua-name = *text-no-semi
ua-product = *text-no-semi
text-no-semi = %d1-9 /                ; text characters excluding NUL, CR,
              %d11 / %d12 / %d14-58 / %d60-127 ; LF, or semi-colon
```

The Reporting-UA field is defined as follows:

An MDN describes the disposition of a message after it has been delivered to a recipient. In all cases, the Reporting-UA is the MUA that performed the disposition described in the MDN. This field is optional, but recommended. For Internet Mail user agents, it is recommended that this field contain both: the DNS name of the particular instance of the MUA that generated the MDN, and the name of the product. For example,

```
Reporting-UA: pc.example.com; Foomail 97.1
```

If the reporting MUA consists of more than one component (e.g., a base program and plug-ins), this may be indicated by including a list of product names.

#### 3.2.2. The MDN-Gateway field

The MDN-Gateway field indicates the name of the gateway or MTA that translated a foreign (non-Internet) message disposition notification into this MDN. This field **MUST** appear in any MDN that was translated by a gateway from a foreign system into MDN format, and **MUST NOT** appear otherwise.

```
mdn-gateway-field = "MDN-Gateway" ":" mta-name-type ";" mta-name
mta-name = *text
```

For gateways into Internet Mail, the MTA-name-type will normally be "smtp", and the mta-name will be the Internet domain name of the gateway.

### 3.2.3. Original-Recipient field

The Original-Recipient field indicates the original recipient address as specified by the sender of the message for which the MDN is being issued. For Internet Mail messages, the value of the Original-Recipient field is obtained from the Original-Recipient header field from the message for which the MDN is being generated. If there is no Original-Recipient header field in the message, then the Original-Recipient field MUST be omitted, unless the same information is reliably available some other way. If there is an Original-Recipient header field in the original message (or original recipient information is reliably available some other way), then the Original-Recipient field must be supplied. If there is more than one Original-Recipient header field in the message, the MUA may choose the one to use, or act as if no Original-Recipient header field is present.

```
original-recipient-field =  
    "Original-Recipient" ":" address-type ";" generic-address  
generic-address = *text
```

The address-type field indicates the type of the original recipient address. If the message originated within the Internet, the address-type field will normally be "rfc822", and the address will be according to the syntax specified in RFC-MSGFMT [2]. The value "unknown" should be used if the Reporting MUA cannot determine the type of the original recipient address from the message envelope. This address is the same as that provided by the sender and can be used to automatically correlate MDN reports with original messages on a per recipient basis.

### 3.2.4. Final-Recipient field

The Final-Recipient field indicates the recipient for which the MDN is being issued. This field MUST be present.

The syntax of the field is as follows:

```
final-recipient-field =  
    "Final-Recipient" ":" address-type ";" generic-address
```

The generic-address subfield of the Final-Recipient field MUST contain the mailbox address of the recipient (from the From header field of the MDN) as it was when the MDN was generated by the MUA.

The Final-Recipient address may differ from the address originally provided by the sender, because it may have been transformed during forwarding and gatewaying into a totally unrecognizable mess. However, in the absence of the optional Original-Recipient field, the Final-Recipient field and any returned content may be the only information available with which to correlate the MDN with a particular message recipient.

The address-type subfield indicates the type of address expected by the reporting MTA in that context. Recipient addresses obtained via SMTP will normally be of address-type "rfc822".

Since mailbox addresses (including those used in the Internet) may be case sensitive, the case of alphabetic characters in the address MUST be preserved.

### 3.2.5. Original-Message-ID field

The Original-Message-ID field indicates the message-ID of the message for which the MDN is being issued. It is obtained from the Message-ID header field of the message for which the MDN is issued. This field MUST be present if the original message contained a Message-ID header field. The syntax of the field is as follows:

```
original-message-id-field =  
    "Original-Message-ID" ":" msg-id
```

The msg-id token is as specified in RFC-MSGFMT [2].

### 3.2.6. Disposition field

The Disposition field indicates the action performed by the Reporting-MUA on behalf of the user. This field MUST be present.

The syntax for the Disposition field is:

```
disposition-field =
    "Disposition" ":" [FWS] disposition-mode ";"
    [FWS] disposition-type
    [ "/" disposition-modifier
    *( "," disposition-modifier ) ]
disposition-mode = action-mode "/" [FWS] sending-mode
action-mode = "manual-action" / "automatic-action"
sending-mode = "MDN-sent-manually" / "MDN-sent-automatically"
disposition-type = "displayed" / "deleted" / "dispatched" /
    "processed"
disposition-modifier = [FWS]
    ("error" / disposition-modifier-extension)
disposition-modifier-extension = atom
```

The disposition-mode, disposition-type, and disposition-modifier may be spelled in any combination of upper and lower case characters.

#### 3.2.6.1. Disposition modes

The following disposition modes are defined:

"manual-action"      The disposition described by the disposition type was a result of an explicit instruction by the user rather than some sort of automatically performed action.

"automatic-action"   The disposition described by the disposition type was a result of an automatic action, rather than an explicit instruction by the user for this message.

"Manual-action" and "automatic-action" are mutually exclusive. One or the other MUST be specified.

"MDN-sent-manually"   The user explicitly gave permission for this particular MDN to be sent.

"MDN-sent-automatically"   The MDN was sent because the MUA had previously been configured to do so automatically.

"MDN-sent-manually" and "MDN-sent-automatically" are mutually exclusive. One or the other MUST be specified.

### 3.2.6.2. Disposition types

The following disposition-types are defined:

|              |  |
|--------------|--|
| "displayed"  | The message has been displayed by the MUA to someone reading the recipient's mailbox. There is no guarantee that the content has been read or understood.  |
| "dispatched" | The message has been sent somewhere in some manner (e.g., printed, faxed, forwarded) without necessarily having been previously displayed to the user. The user may or may not see the message later.  |
| "processed"  | The message has been processed in some manner (i.e., by some sort of rules or server) without being displayed to the user. The user may or may not see the message later, or there may not even be a human user associated with the mailbox. |
| "deleted"    | The message has been deleted. The recipient may or may not have seen the message. The recipient might "undelete" the message at a later time and read the message.   |

### 3.2.6.3. Disposition modifiers

Only the extension disposition modifiers is defined:

#### disposition-modifier-extension

Disposition modifiers may be defined in the future by later revisions or extensions to this specification. Disposition value names beginning with "X-" will never be defined as standard values; such names are reserved for experimental use. MDN disposition value names NOT beginning with "X-" MUST be registered with the Internet Assigned Numbers Authority (IANA) and described in a standards-track RFC or an experimental RFC approved by the IESG. (See Section 10 for a registration form.) MDNs with disposition modifier names not understood by the receiving

MUA MAY be silently ignored or placed in the user's mailbox without special interpretation. They MUST not cause any error message to be sent to the sender of the MDN.

If an MUA developer does not wish to register the meanings of such disposition modifier extensions, "X-" modifiers may be used for this purpose. To avoid name collisions, the name of the MUA implementation should follow the "X-", (e.g., "X-Foomail-").

It is not required that an MUA be able to generate all of the possible values of the Disposition field.

A user agent MUST NOT issue more than one MDN on behalf of each particular recipient. That is, once an MDN has been issued on behalf of a recipient, no further MDNs may be issued on behalf of that recipient, even if another disposition is performed on the message. However, if a message is forwarded, a "dispatched" MDN MAY be issued for the recipient doing the forwarding and the recipient of the forwarded message may also cause an MDN to be generated.

#### 3.2.7. Failure and Error Fields

The Failure and Error fields are used to supply additional information in the form of text messages when the "failure" disposition type or "error" disposition modifier appear. The syntax is as follows:

```
failure-field = "Failure" ":" *text
error-field  = "Error"   ":" *text
```

#### 3.3. Extension-fields

Additional MDN fields may be defined in the future by later revisions or extensions to this specification. Extension-field names beginning with "X-" will never be defined as standard fields; such names are reserved for experimental use. MDN field names NOT beginning with "X-" MUST be registered with the Internet Assigned Numbers Authority (IANA) and described in a standards-track RFC or an experimental RFC approved by the IESG. (See Section 10 for a registration form.) MDN Extension-fields may be defined for the following reasons:

- a. To allow additional information from foreign disposition reports to be tunneled through Internet MDNs. The names of such MDN fields should begin with an indication of the foreign environment name (e.g., X400-Physical-Forwarding-Address).

- b. To allow transmission of diagnostic information that is specific to a particular mail user agent (MUA). The names of such MDN fields should begin with an indication of the MUA implementation that produced the MDN (e.g., Foomail-information).

If an application developer does not wish to register the meanings of such extension fields, "X-" fields may be used for this purpose. To avoid name collisions, the name of the application implementation should follow the "X-", (e.g., "X-Foomail-Log-ID" or "X-Foomail-EDI-info").

#### 4. Timeline of events

The following timeline shows when various events in the processing of a message and generation of MDNs take place:

- User composes message
- User tells MUA to send message
- MUA passes message to MTA (original recipient information passed along)
- MTA sends message to next MTA
- Final MTA receives message
- Final MTA delivers message to MUA (possibly generating a DSN)
- MUA performs automatic processing and generates corresponding MDNs ("dispatched", "processed" or "deleted" disposition type with "automatic-action" and "MDN-sent-automatically" disposition modes)
- MUA displays list of messages to user
- User selects a message and requests that some action be performed on it.

- MUA performs requested action and, with user's permission, sends an appropriate MDN ("displayed", "dispatched", "processed", or "deleted" disposition type, with "manual-action" and "MDN-sent-manually" or "MDN-sent-automatically" disposition mode).
- User possibly performs other actions on message, but no further MDNs are generated.

## 5. Conformance and Usage Requirements

An MUA or gateway conforms to this specification if it generates MDNs according to the protocol defined in this memo. It is not necessary to be able to generate all of the possible values of the Disposition field.

MUAs and gateways **MUST NOT** generate the Original-Recipient field of an MDN unless the mail protocols provide the address originally specified by the sender at the time of submission. Ordinary SMTP does not make that guarantee, but the SMTP extension defined in RFC-DSN-SMTP [7] permits such information to be carried in the envelope if it is available. The Original-Recipient header field defined in this document provides a way for the MTA to pass the original recipient address to the MUA.

Each sender-specified recipient address may result in more than one MDN. If an MDN is requested for a recipient that is forwarded to multiple recipients of an "alias" (as defined in RFC-DSN-SMTP [7], section 6.2.7.3), each of the recipients may issue an MDN.

Successful distribution of a message to a mailing list exploder **SHOULD** be considered the final disposition of the message. A mailing list exploder **MAY** issue an MDN with a disposition type of "processed" and disposition modes of "automatic-action" and "MDN-sent-automatically" indicating that the message has been forwarded to the list. In this case, the request for MDNs is not propagated to the members of the list.

Alternatively, the mailing list exploder **MAY** issue no MDN and propagate the request for MDNs to all members of the list. The latter behavior is not recommended for any but small, closely knit lists, as it might cause large numbers of MDNs to be generated and may cause confidential subscribers to the list to be revealed. The mailing list exploder **MAY** also direct MDNs to itself, correlate them, and produce a report to the original sender of the message.

This specification places no restrictions on the processing of MDNs received by user agents or mailing lists.

## 6. Security Considerations

The following security considerations apply when using MDNs:

### 6.1. Forgery

MDNs may be forged as easily as ordinary Internet electronic mail. User agents and automatic mail handling facilities (such as mail distribution list exploders) that wish to make automatic use of MDNs should take appropriate precautions to minimize the potential damage from denial-of-service attacks.

Security threats related to forged MDNs include the sending of:

- a. A falsified disposition notification when the indicated disposition of the message has not actually occurred,
- b. Unsolicited MDNs

### 6.2. Privacy

Another dimension of security is privacy. There may be cases in which a message recipient does not wish the disposition of messages addressed to him to be known, or is concerned that the sending of MDNs may reveal other sensitive information (e.g., when the message was read). In this situation, it is acceptable for the MUA to silently ignore requests for MDNs.

If the Disposition-Notification-To header field is passed on unmodified when a message is distributed to the subscribers of a mailing list, the subscribers to the list may be revealed to the sender of the original message by the generation of MDNs.

Headers of the original message returned in part 3 of the multipart/report could reveal confidential information about host names and/or network topology inside a firewall.

An unencrypted MDN could reveal confidential information about an encrypted message, especially if all or part of the original message is returned in part 3 of the multipart/report. Encrypted MDNs are not defined in this specification.

In general, any optional MDN field may be omitted if the Reporting MUA site or user determines that inclusion of the field would impose too great a compromise of site confidentiality. The need for such

confidentiality must be balanced against the utility of the omitted information in MDNs.

In some cases, someone with access to the message stream may use the MDN request mechanism to monitor the mail reading habits of a target. If the target is known to generate MDN reports, they could add a disposition-notification-to field containing the envelope from address along with a source route. The source route is ignored in the comparison so the addresses will always match. But if the source route is honored when the notification is sent, it could direct the message to some other destination. This risk can be minimized by not sending MDN's automatically.

### 6.3. Non-Repudiation

MDNs do not provide non-repudiation with proof of delivery. Within the framework of today's Internet Mail, the MDNs defined in this document provide valuable information to the mail user; however, MDNs cannot be relied upon as a guarantee that a message was or was not seen by the recipient. Even if MDNs are not actively forged, they may be lost in transit. The recipient may bypass the MDN issuing mechanism in some manner.

One possible solution for this purpose can be found in RFC-SEC-SERVICES [10].

### 6.4. Mail Bombing

The MDN request mechanism introduces an additional way of mailbombing a mailbox. The MDN request notification provides an address to which MDN's should be sent. It is possible for an attacking agent to send a potentially large set of messages to otherwise unsuspecting third party recipients with a false "disposition-notification-to:" address. Automatic, or simplistic processing of such requests would result in a flood of MDN notifications to the target of the attack. Such an attack could overrun the capacity of the targeted mailbox and deny service.

For that reason, MDN's SHOULD NOT be sent automatically where the "disposition-notification-to:" address is different from the envelope MAIL FROM address. See Section 2.1 for further discussion.

## 7. Collected Grammar

NOTE: The following lexical tokens are defined in RFC-MSGFMT [2]: atom, CRLF, FWS, CFWS, field-name, mailbox, msg-id, text. The definitions of attribute and value are as in the definition of the Content-Type header field in RFC-MIME-BODY [3].

## Message header fields:

```

mdn-request-header =
    "Disposition-Notification-To" ":" [FWS]
        mailbox *("," [FWS] mailbox)
Disposition-Notification-Options =
    "Disposition-Notification-Options" ":" [FWS]
        disposition-notification-parameter-list
disposition-notification-parameter-list =
    disposition-notification-parameter
        *(";" [FWS] disposition-notification-parameter)
disposition-notification-parameter = attribute [FWS] "=" [FWS]
    importance "," [FWS] value *("," [FWS] value)
importance = "required" / "optional"
original-recipient-header =
    "Original-Recipient" ":" address-type ";" generic-address

```

## Report content:

```

disposition-notification-content =
    [ reporting-ua-field CRLF ]
    [ mdn-gateway-field CRLF ]
    [ original-recipient-field CRLF ]
    final-recipient-field CRLF
    [ original-message-id-field CRLF ]
    disposition-field CRLF
    *( failure-field CRLF )
    *( error-field CRLF )
    *( extension-field CRLF )
address-type = atom
mta-name-type = atom
reporting-ua-field = "Reporting-UA" ":" ua-name [ ";" ua-product ]
ua-name = *text-no-semi
ua-product = *text-no-semi
text-no-semi = %d1-9 /           ; text characters excluding NUL, CR,
    %d11 / %d12 / %d14-58 / %d60-127 ; LF, or semi-colon
mdn-gateway-field = "MDN-Gateway" ":" mta-name-type ";" mta-name
mta-name = *text
original-recipient-field =
    "Original-Recipient" ":" address-type ";" generic-address
generic-address = *text
final-recipient-field =
    "Final-Recipient" ":" address-type ";" generic-address
original-message-id-field = "Original-Message-ID" ":" msg-id
disposition-field =
    "Disposition" ":" [FWS] disposition-mode ";"
    [FWS] disposition-type
    [ "/" disposition-modifier
    *( "," disposition-modifier ) ]
disposition-mode = action-mode "/" [FWS] sending-mode
action-mode = "manual-action" / "automatic-action"

```

```
sending-mode = "MDN-sent-manually" / "MDN-sent-automatically"
disposition-type = "displayed" / "deleted" / "dispatched" /
    "processed"
disposition-modifier = [FWS]
    ("error" / disposition-modifier-extension)
disposition-modifier-extension = atom
failure-field = "Failure" ":" *text
error-field = "Error" ":" *text
extension-field = extension-field-name ":" *text
extension-field-name = field-name
```

## 8. Guidelines for Gatewaying MDNs

NOTE: This section provides non-binding recommendations for the construction of mail gateways that wish to provide semi-transparent disposition notifications between the Internet and another electronic mail system. Specific MDN gateway requirements for a particular pair of mail systems may be defined by other documents.

### 8.1. Gatewaying from other mail systems to MDNs

A mail gateway may issue an MDN to convey the contents of a "foreign" disposition notification over Internet Mail. When there are appropriate mappings from the foreign notification elements to MDN fields, the information may be transmitted in those MDN fields. Additional information (such as might be needed to tunnel the foreign notification through the Internet) may be defined in extension MDN fields. (Such fields should be given names that identify the foreign mail protocol, e.g., X400-\* for X.400 protocol elements).

The gateway must attempt to supply reasonable values for the Reporting-UA, Final-Recipient, and Disposition fields. These will normally be obtained by translating the values from the foreign notification into their Internet-style equivalents. However, some loss of information is to be expected.

The sender-specified recipient address and the original message-id, if present in the foreign notification, should be preserved in the Original-Recipient and Original-Message-ID fields.

The gateway should also attempt to preserve the "final" recipient address from the foreign system. Whenever possible, foreign protocol elements should be encoded as meaningful printable ASCII strings.

For MDNs produced from foreign disposition notifications, the name of the gateway MUST appear in the MDN-Gateway field of the MDN.

## 8.2. Gatewaying from MDNs to other mail systems

It may be possible to gateway MDNs from the Internet into a foreign mail system. The primary purpose of such gatewaying is to convey disposition information in a form that is usable by the destination system. A secondary purpose is to allow "tunneling" of MDNs through foreign mail systems in case the MDN may be gatewayed back into the Internet.

In general, the recipient of the MDN (i.e., the sender of the original message) will want to know, for each recipient: the closest available approximation to the original recipient address, and the disposition (displayed, printed, etc.).

If possible, the gateway should attempt to preserve the Original-Recipient address and Original-Message-ID (if present) in the resulting foreign disposition report.

If it is possible to tunnel an MDN through the destination environment, the gateway specification may define a means of preserving the MDN information in the disposition reports used by that environment.

## 8.3. Gatewaying of MDN-requests to other mail systems

By use of the separate disposition-notification-to request header field, this specification offers a richer functionality than most, if not all, other email systems. In most other email systems, the notification recipient is identical to the message sender as indicated in the "from" address. There are two interesting cases when gatewaying into such systems:

1. If the address in the disposition-notification-to header field is identical to the address in the SMTP "MAIL FROM", the expected behavior will result, even if the disposition-notification-to information is lost. Systems should propagate the MDN request.
2. If the address in the disposition-notification-to header field is different from the address in the SMTP "MAIL FROM", gatewaying into a foreign system without a separate notification address will result in unintended behavior. This is especially important when the message arrives via a mailing list expansion software that may specifically replace the SMTP "MAIL FROM" address with an alternate address. In such cases, the MDN request should not be gatewayed and should be silently dropped. This is consistent with other forms of non-support for MDN.

## 9. Example

NOTE: This example is provided as illustration only, and is not considered part of the MDN protocol specification. If the example conflicts with the protocol definition above, the example is wrong.

Likewise, the use of \*-type subfield names or extension fields in this example is not to be construed as a definition for those type names or extension fields.

This is an MDN issued after a message has been displayed to the user of an Internet Mail user agent.

```
Date: Wed, 20 Sep 1995 00:19:00 (EDT) -0400
From: Joe Recipient <Joe_Recipient@example.com>
Message-Id: <199509200019.12345@example.com>
Subject: Disposition notification
To: Jane Sender <Jane_Sender@example.org>
MIME-Version: 1.0
Content-Type: multipart/report; report-type=disposition-notification;
    boundary="RAA14128.773615765/example.com"
```

--RAA14128.773615765/example.com

The message sent on 1995 Sep 19 at 13:30:00 (EDT) -0400 to Joe Recipient <Joe\_Recipient@example.com> with subject "First draft of report" has been displayed.  
This is no guarantee that the message has been read or understood.

--RAA14128.773615765/example.com  
content-type: message/disposition-notification

```
Reporting-UA: joes-pc.cs.example.com; Foomail 97.1
Original-Recipient: rfc822;Joe_Recipient@example.com
Final-Recipient: rfc822;Joe_Recipient@example.com
Original-Message-ID: <199509192301.23456@example.org>
Disposition: manual-action/MDN-sent-manually; displayed
```

--RAA14128.773615765/example.com  
content-type: message/rfc822

[original message optionally goes here]

--RAA14128.773615765/example.com--

## 10. IANA Considerations

This document specifies three types of parameters that must be registered with the Internet Assigned Numbers Authority (IANA).

The forms below are for use when registering a new disposition-notification-parameter name for the Disposition-Notification-Options header field, a new disposition modifier name, or a new MDN extension field. Each piece of information required by a registration form may be satisfied either by providing the information on the form itself, or by including a reference to a published, publicly available specification that includes the necessary information. IANA MAY reject registrations because of incomplete registration forms or incomplete specifications.

To register, complete the following applicable form and send it via electronic mail to <IANA@IANA.ORG>.

### 10.1. Disposition-Notification-Options header field disposition-notification-parameter names

A registration for a Disposition-Notification-Options header field disposition-notification-parameter name MUST include the following information:

- a. The proposed disposition-notification-parameter name.
- b. The syntax for disposition-notification-parameter values, specified using BNF, ABNF, regular expressions, or other non-ambiguous language.
- c. If disposition-notification-parameter values are not composed entirely of graphic characters from the US-ASCII repertoire, a specification for how they are to be encoded as graphic US-ASCII characters in a Disposition-Notification-Options header field.
- d. A reference to a standards track RFC or experimental RFC approved by the IESG that describes the semantics of the disposition-notification-parameter values.

## 10.2. Disposition modifier names

A registration for a disposition-modifier name (used in the Disposition field of a message/disposition-notification) MUST include the following information:

- a. The proposed disposition-modifier name.
- b. A reference to a standards track RFC or experimental RFC approved by the IESG that describes the semantics of the disposition modifier.

## 10.3. MDN extension field names

A registration for an MDN extension-field name MUST include the following information:

- a. The proposed extension field name.
- b. The syntax for extension values, specified using BNF, ABNF, regular expressions, or other non-ambiguous language.
- c. If extension-field values are not composed entirely of graphic characters from the US-ASCII repertoire, a specification for how they are to be encoded as graphic US-ASCII characters in a Disposition-Notification-Options header field.
- d. A reference to a standards track RFC or experimental RFC approved by the IESG that describes the semantics of the extension field.

## 11. Acknowledgements

The contributions of Bruce Lilly and Alfred Hoenes are gratefully acknowledged for this revision.

The contributions of Roger Fajman and Greg Vaudreuil to earlier versions of this document are also gratefully acknowledged.

## 12. References

### 12.1. Normative References

- [1] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001.
- [2] Resnick, P., Ed., "Internet Message Format", RFC 5322, October 2008.
- [3] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.
- [4] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.
- [5] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.
- [6] Vaudreuil, G., "The Multipart/Report Content Type for the Reporting of Mail System Administrative Messages", RFC 3462, January 2003.
- [7] Moore, K., "Simple Mail Transfer Protocol (SMTP) Service Extension for Delivery Status Notifications (DSNs)", RFC 3461, January 2003.
- [8] Moore, K. and G. Vaudreuil, "An Extensible Message Format for Delivery Status Notifications", RFC 3464, January 2003.
- [9] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

### 12.2. Informative References

- [10] Hoffman, P., "Enhanced Security Services for S/MIME", RFC 2634, June 1999.

## Appendix A. Changes from RFC 3798

The values of "dispatched" and "processed" were lost from the ABNF for "disposition-type".

Because the warning disposition modifier was previously removed, warning-field has also been removed.

The ABNF for ua-name and ua-product included semi-colon, which could not be distinguished from \*text in the production. The ua-name and ua-product definitions were restricted to not include semi-colon.

The ABNF did not indicate all places that whitespace was allowable, in particular folding whitespace, although all implementations allow whitespace and folding in the header fields just like any other RFC5322 [2]-formatted header field. There were also a number of places in the ABNF that inconsistently permitted comments and whitespace in one leg of the production and not another. The ABNF now specifies FWS and CFWS in several places that should have already been specified by the grammar.

Extension-field was defined in the collected grammar but not in the main text.

[[CREF11: Shouldn't the places we use \*text and \*text-no-semi allow FWS? ]]

The comparison of mailboxes in Disposition-Notification-To to the Return-Path addr-spec was clarified.

The use of the grammar production "parameter" was confusing with the RFC2045 [3] production of the same name, as well as other uses of the same term. These have been clarified.

[[CREF12: Not sure what to do with this one: (From Bruce) In the case of the message header fields, RFC 2822 also specifies minimum and maximum counts for each header field, and similar guidance would clarify 3798 (e.g. are multiple Disposition-Notification-Options fields permitted in a single message header, and if so, what semantics apply?). ]]

[[CREF13: Not sure what to do with this one: (From Bruce) Note also that RFC 2045 is itself based on RFC 822 rather than 2822, so the issue of where CFWS is permitted or prohibited should probably be clearly specified where "attribute" and "value" are used. Note further that the RFC 2045 definitions are clarified by errata and modified by RFC 2231, and by RFC 2231 errata. Finally, note that RFC 2231 has provisions for continuation of long parameter values (where there would otherwise be problems with the maximum line length specifications of RFCs 822 and 2822), specification of language and charset, and provision for compatible handling of non-ASCII text, none of which are provided for in the RFC 3798 disposition-notification parameters. It might be a good idea to think about that now, as a future change would almost certainly reset the document status to "Proposed". ]]

A clarification was added on the extent of the 7bit nature of MDNs.

Uses of the terms "may" and "might" were clarified.

A clarification was added on the order of the fields in the message/disposition-notification content.

[[CREF14: Not sure what to do with this one: (From Bruce) 3.1.1 explicitly mentions use of RFC 2047 encoded-words in comments (however, as noted above there is no explicit provision for comments), but fails to mention the other contexts in which encoded-words may be used, viz. in an RFC [2]822 "phrase" (e.g. in the display name of a name-addr mailbox in Disposition-Notification-To (therefore, the discussion of encoded-words should probably be moved earlier in the document, prior to the specification of Disposition-Notification-To)), and in unstructured text (i.e. every instance of \*text in the ABNF). In particular, use of encoded-words might be highly desirable in the following places: \*) the ua-product portion of the Reporting-UA field; \*) the generic-address part of the Original-Recipient and Final-Recipient fields; \*) the (unstructured) field bodies of Error, Failure, and Warning fields; in structured extension fields where the context (per RFC 2047) is appropriate in unstructured extension fields; \*) in X- extension fields (see RFC 2047 for related X- message header fields). In cases where the field syntax is shared with DSN fields, some coordination with the RFC 346x authors might be desirable. ]]

[[CREF15: I think a couple of clarifications are in order: 1) This restriction is within a given mail user agent. If the user uses multiple MUAs, it is possible that multiple MDNs MAY be generated. 2) A mail user agent SHOULD use underlying protocol support when possible to prevent multiple MDNs from being generated. If underlying protocol support is not available, the mail user agent MUST use local knowledge to prevent multiple MDNs. I don't think we need to worry about the case of an MUA error; accidents and bad implementations DO happen. (From Bruce) 3.2.6.3 prohibition against multiple MDNs being issued on behalf of each recipient poses some implementation difficulties: \*) While IMAP servers maintain state that could possibly be used to prevent issuance of multiple MDNs, the POP protocol has no such provision. Even in the case of IMAP, there is some ambiguity in the case of shared mailboxes. \*) Some MUAs are known to have extreme difficulty keeping track of which messages have been seen, let alone responded to. Software version updates, minor configuration changes (e.g. domain name or IP address change of POP or IMAP server) are known to "confuse" some MUAs. \*) there is no standardized mechanism for communicating status between multiple MUAs accessing the same mailbox (except in the case of IMAP, as noted above). Therefore, if an MDN is sent when a message is viewed (etc.) using one MUA, a different MUA subsequently being used to view the same message in the same user's mailbox (either via POP, or from a

flat file mailbox) might have no way to determine that an MDN had already been sent. This is a fundamental difficulty with the specified protocol (relaxing "MUST NOT" to "SHOULD NOT" is one possible way around that difficulty -- otherwise the document contains a "known technical omission" viz. no defined means of establishing whether or not an MDN has already been sent for a particular message. I believe that "known technical omissions" are a barrier to further Standards Track progress). \*) Due to aliases, forwarding, etc. an original message sent to multiple addresses might end up as multiple copies in a single recipient's mailbox. It is unclear whether or not multiple MDNs are permitted in that case (the Message-ID, if present in the original, will be the same in the copies, and the "particular recipient" could be interpreted as being the same, even though the addresses specified in the original message transport envelope might have appeared to have been distinct to the originator who requested MDNs. ]]

#### Authors' Addresses

Tony Hansen (editor)  
AT&T Laboratories  
200 Laurel Ave. South  
Middletown, NJ 07748  
USA

Email: [tony+rfc3798@maillennium.att.com](mailto:tony+rfc3798@maillennium.att.com)

Alexey Melnikov (editor)  
Isode Ltd  
14 Castle Mews  
Hampton, Middlesex TW12 2NP  
UK

Email: [Alexey.Melnikov@isode.com](mailto:Alexey.Melnikov@isode.com)

Network Working Group  
Internet-Draft  
Updates: 7208 (if approved)  
Intended status: Standards Track  
Expires: February 9, 2015

M. Kucherawy  
August 8, 2014

Email Authentication Status Codes  
draft-ietf-appsawg-email-auth-codes-07

Abstract

This document registers code points to allow status codes to be returned to an email client to indicate that a message is being rejected or deferred specifically because of email authentication failures.

This document updates [RFC7208] since some of the code points registered replace the ones recommended for use in that document.

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 <http://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 February 9, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|  |   |
|--|---|
| 1. Introduction . . . . .                            | 3 |
| 2. Key Words . . . . .                               | 3 |
| 3. New Enhanced Status Codes . . . . .               | 3 |
| 3.1. DKIM Failure Codes . . . . .                    | 3 |
| 3.2. SPF Failure Codes . . . . .                     | 4 |
| 3.3. Reverse DNS Failure Code . . . . .              | 5 |
| 3.4. Multiple Authentication Failures Code . . . . . | 5 |
| 4. General Considerations . . . . .                  | 6 |
| 5. Security Considerations . . . . .                 | 7 |
| 6. IANA Considerations . . . . .                     | 7 |
| 7. Normative References . . . . .                    | 7 |
| Appendix A. Acknowledgments . . . . .                | 7 |

## 1. Introduction

[RFC3463] introduced Enhanced Mail System Status Codes, and [RFC5248] created an IANA registry for these.

[RFC6376] and [RFC7208] introduced, respectively, DomainKeys Identified Mail (DKIM) and Sender Policy Framework (SPF), two protocols for conducting message authentication. Another common email acceptance test is the reverse Domain Name System (DNS) check on an email client's IP address, as described in Section 3 of [RFC7001].

The current set of enhanced status codes does not include any code for indicating that a message is being rejected or deferred due to local policy reasons related to any of these mechanisms. This is potentially useful information to agents that need more than rudimentary handling information about the reason a message was rejected on receipt. This document introduces enhanced status codes for reporting those cases to clients.

Section 3.2 updates [RFC7208], as new enhanced status codes relevant to that specification are being registered and recommended for use.

## 2. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

## 3. New Enhanced Status Codes

The following new enhanced status codes are defined:

### 3.1. DKIM Failure Codes

In the code point definitions below, the following definitions are used:

passing: A signature is "passing" if the basic DKIM verification algorithm as defined in [RFC6376] succeeds.

acceptable: A signature is "acceptable" if it satisfies all locally defined requirements (if any) in addition to passing the basic DKIM verification algorithm (e.g., certain header fields are included in the signed content; no partial signatures; etc.).

Code: X.7.20  
Sample Text: No passing DKIM signature found  
Associated basic status code: 550  
Description: This status code is returned when a message did not contain any passing DKIM signatures. (This violates the advice of Section 6.1 of RFC6376.)  
Reference: [this document]; RFC6376  
Submitter: M. Kucherawy  
Change controller: IESG

Code: X.7.21  
Sample Text: No acceptable DKIM signature found  
Associated basic status code: 550  
Description: This status code is returned when a message contains one or more passing DKIM signatures, but none are acceptable. (This violates the advice of Section 6.1 of RFC6376.)  
Reference: [this document]; RFC6376  
Submitter: M. Kucherawy  
Change controller: IESG

Code: X.7.22  
Sample Text: No valid author-matched DKIM signature found  
Associated basic status code: 550  
Description: This status code is returned when a message contains one or more passing DKIM signatures, but none are acceptable because none have an identifier(s) that matches the author address(es) found in the From header field. This is a special case of X.7.21. (This violates the advice of Section 6.1 of RFC6376.)  
Reference: [this document]; RFC6376  
Submitter: M. Kucherawy  
Change controller: IESG

### 3.2. SPF Failure Codes

Code: X.7.23  
Sample Text: SPF validation failed  
Associated basic status code: 550  
Description: This status code is returned when a message completed an SPF check that produced a "fail" result, contrary to local policy requirements. Used in place of 5.7.1 as described in Section 8.4 of RFC7208.  
Reference: [this document]; RFC7208  
Submitter: M. Kucherawy  
Change controller: IESG

Code: X.7.24  
Sample Text: SPF validation error  
Associated basic status code: 451/550  
Description: This status code is returned when evaluation of SPF relative to an arriving message resulted in an error. Used in place of 4.4.3 or 5.5.2 as described in Sections 8.6 and 8.7 of RFC7208.  
Reference: [this document]; RFC7208  
Submitter: M. Kucherawy  
Change controller: IESG

### 3.3. Reverse DNS Failure Code

Code: X.7.25  
Sample Text: Reverse DNS validation failed  
Associated basic status code: 550  
Description: This status code is returned when an SMTP client's IP address failed a reverse DNS validation check, contrary to local policy requirements.  
Reference: [this document]; Section 3 of RFC7001  
Submitter: M. Kucherawy  
Change controller: IESG

### 3.4. Multiple Authentication Failures Code

Code: X.7.26  
Sample Text: Multiple authentication checks failed  
Associated basic status code: 550  
Description: This status code is returned when a message failed more than one message authentication check, contrary to local policy requirements. The specific mechanisms that failed are not specified.  
Reference: [this document]  
Submitter: M. Kucherawy  
Change controller: IESG

#### 4. General Considerations

By the nature of the Simple Mail Transfer Protocol (SMTP), only one enhanced status code can be returned for a given exchange between client and server. However, an operator might decide to defer or reject a message for a plurality of reasons. Clients receiving these codes need to consider that the failure reflected by one of these status codes might not reflect the only reason, or the most important reason, for non-acceptance of the message or command.

It is important to note that Section 6.1 of [RFC6376] discourages special treatment of messages bearing no valid DKIM signature. There are some operators that disregard this advice, a few of which go so far as to require a valid Author Domain signature (that is, one matching the domain(s) in the From header field) in order to accept the message. Moreover, some nascent technologies built atop SPF and DKIM depend on such authentications. This work does not endorse configurations that violate DKIM's recommendations, but rather acknowledges that they do exist and merely seeks to provide for improved interoperability with such operators.

A specific use case for these codes is mailing list software, which processes rejections in order to remove from the subscriber set those addresses that are no longer valid. There is a need in that case to distinguish authentication failures versus indications that the recipient address is no longer valid.

If a receiving server performs multiple authentication checks, and more than one of them fails thus warranting rejection of the message, the SMTP server SHOULD use the code that indicates multiple methods failed rather than only reporting the first one that failed. It may be the case that one method is always expected to fail, and thus returning that method's specific code is not information useful to the sending agent.

The reverse IP DNS check is defined in Section 2.6.3 of [RFC7001].

Any message authentication or policy enforcement technologies developed in the future should also include registration of their own enhanced status codes so that this kind of specific reporting is available to operators that wish to use them.

## 5. Security Considerations

Use of these codes reveals local policy with respect to email authentication, which can be useful information to actors attempting to deliver undesired mail. It should be noted that there is no specific obligation to use these codes; if an operator wishes not to reveal this aspect of local policy, it can continue using a generic result code such as 5.7.7, 5.7.1, or even 5.7.0.

## 6. IANA Considerations

Registration of new enhanced status codes, for addition to the Enumerated Status Codes sub-registry of the SMTP Enhanced Status Codes Registry, can be found in Section 3.

## 7. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3463] Vaudreuil, G., "Enhanced Mail System Status Codes", RFC 3463, January 2003.
- [RFC5248] Hansen, T. and J. Klensin, "A Registry for SMTP Enhanced Mail System Status Codes", BCP 138, RFC 5248, June 2008.
- [RFC6376] Crocker, D., Hansen, T., and M. Kucherawy, "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, September 2011.
- [RFC7001] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7001, September 2013.
- [RFC7208] Kitterman, S., "Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1", RFC 7208, April 2014.

## Appendix A. Acknowledgments

Claudio Allocchio, Dave Crocker, Ned Freed, Arnt Gulbrandsen, Scott Kitterman, Barry Leiba, Alexey Melnikov, S. Moonesamy, Hector Santos, and Stephen Turnbull contributed to this work.

Author's Address

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

EMail: [superuser@gmail.com](mailto:superuser@gmail.com)



Applications Area Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: February 22, 2015

P. Hoffman  
VPN Consortium  
J. Snell

August 21, 2014

JSON Merge Patch  
draft-ietf-appsawg-json-merge-patch-07

Abstract

This specification defines the JSON merge patch format and processing rules. The merge patch format is primarily intended for use with the HTTP PATCH method as a means of describing a set of modifications to a target resource's content.

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 <http://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 February 22, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |   |
|---|---|
| 1. Introduction . . . . .                     | 2 |
| 2. Processing Merge Patch Documents . . . . . | 3 |
| 3. Example . . . . .                          | 4 |
| 4. IANA Considerations . . . . .              | 5 |
| 5. Security Considerations . . . . .          | 6 |
| 6. Acknowledgements . . . . .                 | 6 |
| 7. References . . . . .                       | 6 |
| 7.1. Normative References . . . . .           | 6 |
| 7.2. Informative References . . . . .         | 7 |
| Appendix A. Example Test Cases . . . . .      | 7 |
| Authors' Addresses . . . . .                  | 9 |

## 1. Introduction

This specification defines the JSON merge patch document format, processing rules, and associated MIME media type identifier. The merge patch format is primarily intended for use with the HTTP PATCH method [RFC5789] as a means of describing a set of modifications to a target resource's content.

A JSON merge patch document describes changes to be made to a target JSON document using a syntax that closely mimics the document being modified. Recipients of a merge patch document determine the exact set of changes being requested by comparing the content of the provided patch against the current content of the target document. If the provided merge patch contains members that do not appear within the target, those members are added. If the target does contain the member, the value is replaced. Null values in the merge patch are given special meaning to indicate the removal of existing values in the target.

For example, given the following original JSON document:

```
{
  "a": "b",
  "c": {
    "d": "e",
    "f": "g"
  }
}
```

Changing the value of "a" and removing "f" can be achieved by sending:

```
PATCH /target HTTP/1.1
Host: example.org
Content-Type: application/merge-patch+json
```

```
{
  "a": "z",
  "c": {
    "f": null
  }
}
```

When applied to the target resource, the value of the "a" member is replaced with "z" and "f" is removed, leaving the remaining content untouched.

This design means that merge patch documents are suitable for describing modifications to JSON documents that primarily use objects for their structure and do not make use of explicit null values. The merge patch format is not appropriate for all JSON syntaxes.

## 2. Processing Merge Patch Documents

JSON merge patch documents describe, by example, a set of changes that are to be made to a target resource. Recipients of merge patch documents are responsible for comparing the merge patch with the current content of the target resource to determine the specific set of change operations to be applied to the target.

To apply the merge patch document to a target resource, the system realizes the effect of the following function, described in pseudocode. For this description, the function is called `MergePatch`, and it takes two arguments: the target resource document and the merge patch document. The Target argument can be any JSON value, or undefined. The Patch argument can be any JSON value.

```
define MergePatch(Target, Patch):
  if Patch is an Object:
    if Target is not an Object:
      Target = {} # Ignore the contents and set it to an empty Object
    for each Name/Value pair in Patch:
      if Value is null:
        if Name exists in Target:
          remove the Name/Value pair from Target
      else:
        Target[Name] = MergePatch(Target[Name], Value)
    return Target
  else:
    return Patch
```

There are a few things to note about the function. If the patch is anything other than an object, the result will always be to replace the entire target with the entire patch. Also, it is not possible to patch part of a target that is not an object, such as to replace just some if the values in an array.

The MergePatch operation is defined to operate at the level of data items, not at the level of textual representation. There is no expectation that the MergePatch operation will preserve textual representation-level features such as white space, member ordering, number precision beyond what is available in the target's implementation, and so forth. In addition, even if the target implementation allows multiple name/value pairs with the same name, the result of the patch operation on such objects is not defined.

### 3. Example

For example, given the following example JSON document:

```
{
  "title": "Goodbye!",
  "author" : {
    "givenName" : "John",
    "familyName" : "Doe"
  },
  "tags":[ "example", "sample" ],
  "content": "This will be unchanged"
}
```

A user-agent wishing to change the value of the "title" member from "Goodbye!" to the value "Hello!", add a new "phoneNumber" member, remove the "familyName" member from the "author" object, and replace the "tags" Array so that it doesn't include the word "sample", would send the following request:

```
PATCH /my/resource HTTP/1.1
Host: example.org
Content-Type: application/merge-patch+json
```

```
{
  "title": "Hello!",
  "phoneNumber": "+01-123-456-7890",
  "author": {
    "familyName": null
  },
  "tags": [ "example" ]
}
```

The resulting JSON document would be:

```
{
  "title": "Hello!",
  "author" : {
    "givenName" : "John"
  },
  "tags": [ "example" ],
  "content": "This will be unchanged",
  "phoneNumber": "+01-123-456-7890"
}
```

#### 4. IANA Considerations

This specification registers the following additional MIME Media Types:

Type name: application

Subtype name: merge-patch+json

Required parameters: None

Optional parameters: None

Encoding considerations: Resources that use the "application/merge-patch+json" media type are required to conform to the "application/json" Media Type and are therefore subject to the same encoding considerations specified in Section 8 of [RFC7159].

Security considerations: As defined in this specification

Published specification: This specification.

Applications that use this media type: None currently known.

## Additional information:

Magic number(s): N/A

File extension(s): N/A

Macintosh file type code(s): TEXT

Person &amp; email address to contact for further information: IESG

Intended usage: COMMON

Restrictions on usage: None.

Author: James M Snell &lt;jasnell@gmail.com&gt;

Change controller: IESG

## 5. Security Considerations

The "application/merge-patch+json" Media Type allows user agents to indicate their intention that the server determine the specific set of change operations to be applied to a target resource. As such, it is the server's responsibility to determine the appropriateness of any given change as well as the user agent's authorization to request such changes. How such determinations are made is considered out of the scope of this specification.

All of the the security considerations discussed in Section 5 of [RFC5789] apply to all uses of the HTTP PATCH method with the "application/merge-patch+json" Media Type.

## 6. Acknowledgements

Many people contributed significant ideas to this document. These people include, but are not limited to, James Manger, Matt Miller, Carsten Bormann, and Bjoern Hoehrmann, Pete Resnick, and Richard Barnes.

## 7. References

## 7.1. Normative References

[RFC7159] Bray, T., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, March 2014.

## 7.2. Informative References

[RFC5789] Dusseault, L. and J. Snell, "PATCH Method for HTTP", RFC 5789, March 2010.

## Appendix A. Example Test Cases

| ORIGINAL  | PATCH   | RESULT                                       |
|---|---|--|
| <code>{"a": "b"}</code>                           | <code>{"a": "c"}</code>   | <code>{"a": "c"}</code>                      |
| <code>{"a": "b"}</code>                           | <code>{"b": "c"}</code>   | <code>{"a": "b",<br/>"b": "c"}</code>        |
| <code>{"a": "b"}</code>                           | <code>{"a": null}</code>  | <code>{}</code>                              |
| <code>{"a": "b",<br/>"b": "c"}</code>             | <code>{"a": null}</code>  | <code>{"b": "c"}</code>                      |
| <code>{"a": ["b"]}</code>                         | <code>{"a": "c"}</code>   | <code>{"a": "c"}</code>                      |
| <code>{"a": "c"}</code>                           | <code>{"a": ["b"]}</code>   | <code>{"a": ["b"]}</code>                    |
| <code>{"a": {<br/>  "b": "c"<br/>}}</code>        | <code>{"a": {<br/>  "b": "d",<br/>  "c": null<br/>}}</code>       | <code>{"a": {<br/>  "b": "d"<br/>}}</code>   |
| <code>{"a": [<br/>  {"b": "c"<br/>}]<br/>}</code> | <code>{"a": [1]}</code>   | <code>{"a": [1]}</code>                      |
| <code>["a", "b"]</code>                           | <code>["c", "d"]</code>   | <code>["c", "d"]</code>                      |
| <code>{"a": "b"}</code>                           | <code>["c"]</code>  | <code>["c"]</code>                           |
| <code>{"a": "foo"}</code>                         | <code>null</code>   | <code>null</code>                            |
| <code>{"a": "foo"}</code>                         | <code>"bar"</code>  | <code>"bar"</code>                           |
| <code>{"e": null}</code>                          | <code>{"a": 1}</code>   | <code>{"e": null,<br/>"a": 1}</code>         |
| <code>[1, 2]</code>                               | <code>{"a": "b",<br/>"c": null}</code>                            | <code>{"a": "b"}</code>                      |
| <code>{}</code>                                   | <code>{"a":<br/>  {"bb":<br/>    {"ccc":<br/>      null}}}</code> | <code>{"a":<br/>  {"bb":<br/>    {}}}</code> |

Authors' Addresses

Paul Hoffman  
VPN Consortium

Email: paul.hoffman@vpnc.org

James M Snell

Email: jasnell@gmail.com

Applications Area Working Group  
Internet-Draft  
Updates: 4466 (if approved)  
Obsoletes: 6237 (if approved)  
Intended status: Standards Track  
Expires: February 06, 2015

B. Leiba  
Huawei Technologies  
A. Melnikov  
Isode Limited  
August 07, 2014

IMAP4 Multimailbox SEARCH Extension  
draft-ietf-appsawg-multimailbox-search-04

Abstract

The IMAP4 specification allows the searching of only the selected mailbox. A user often wants to search multiple mailboxes, and a client that wishes to support this must issue a series of SELECT and SEARCH commands, waiting for each to complete before moving on to the next. This extension allows a client to search multiple mailboxes with one command, limiting the delays caused by many round trips, and not requiring disruption of the currently selected mailbox. This extension also uses MAILBOX, UIDVALIDITY, and TAG fields in ESEARCH responses, allowing a client to pipeline these searches if it chooses. This document updates RFC 4466 and obsoletes RFC 6237.

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 <http://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 February 06, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |    |
|---|----|
| 1. Introduction . . . . .                                     | 2  |
| 1.1. Conventions Used in This Document . . . . .              | 3  |
| 2. New ESEARCH Command . . . . .                              | 3  |
| 2.1. The ESEARCH Response . . . . .                           | 4  |
| 2.2. Source Options: Specifying Mailboxes to Search . . . . . | 5  |
| 2.3. Strictness in Search Matches . . . . .                   | 6  |
| 2.4. Server-Side Limitations on Search Volume . . . . .       | 6  |
| 3. Examples . . . . .   | 7  |
| 4. Formal Syntax . . . . .                                    | 7  |
| 5. Security Considerations . . . . .                          | 8  |
| 6. IANA Considerations . . . . .                              | 9  |
| 7. Implementation Status . . . . .                            | 9  |
| 8. Changes Since RFC 6237 . . . . .                           | 10 |
| 9. Acknowledgments . . . . .                                  | 10 |
| 10. References . . . . .                                      | 10 |
| 10.1. Normative References . . . . .                          | 10 |
| 10.2. Informative References . . . . .                        | 11 |
| Authors' Addresses . . . . .                                  | 11 |

## 1. Introduction

The IMAP4 specification allows the searching of only the selected mailbox. A user often wants to search multiple mailboxes, and a client that wishes to support this must issue a series of SELECT and SEARCH commands, waiting for each to complete before moving on to the next. The commands can't be pipelined, because the server might run them in parallel, and the untagged SEARCH responses could not then be distinguished from each other.

This extension allows a client to search multiple mailboxes with one command, and includes MAILBOX and TAG fields in the ESEARCH response, yielding the following advantages:

- o A single command limits the number of round trips needed to search a set of mailboxes.
- o A single command eliminates the need to wait for one search to complete before starting the next.
- o A single command allows the server to optimize the search, if it can.

- o A command that is not dependent upon the selected mailbox eliminates the need to disrupt the selection state or to open another IMAP connection.
- o The MAILBOX, UIDVALIDITY, and TAG fields in the responses allow a client to distinguish which responses go with which search (and which mailbox). A client can safely pipeline these search commands without danger of confusion. The addition of the MAILBOX and UIDVALIDITY fields updates the search-correlator item defined in [RFC4466].

This extension was previously published as experimental. There is now implementation experience, giving confidence in the protocol, so this document puts the extension on the Standards Track, with some minor updates that were informed by the implementation experience. A brief summary of changes is in Section 8.

### 1.1. Conventions Used in This Document

In examples, "C:" indicates lines sent by a client that is connected to a server. "S:" indicates lines sent by the server to the client.

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 RFC 2119 [RFC2119].

## 2. New ESEARCH Command

Arguments: OPTIONAL source options  
OPTIONAL result options  
OPTIONAL charset specification (see [RFC2978])  
searching criteria (one or more)

Responses: REQUIRED untagged response: ESEARCH

Result: OK -- search completed  
NO -- error: cannot search that charset or criteria  
BAD -- command unknown or arguments invalid

This section defines a new ESEARCH command, which works similarly to the UID SEARCH command described in Section 2.6.1 of [RFC4466] (initially described in Section 6.4.4 of [RFC3501] and extended by [RFC4731]).

The ESEARCH command further extends searching by allowing for optional source and result options. This document does not define any new result options (see Section 3.1 of [RFC4731]). A server that supports this extension includes "MULTISEARCH" in its IMAP capability string.

Because there has been confusion about this, it is worth pointing out that with ESEARCH, as with any SEARCH or UID SEARCH command, it MUST NOT be considered an error if the search terms include a range of message numbers that extends (or, in fact, starts) beyond the end of the mailbox. For example, a client might want to establish a rolling window through the search results this way:

```
C: tag1 UID ESEARCH FROM "frobozz" 1:100
```

...followed later by this:

```
C: tag1 UID ESEARCH FROM "frobozz" 101:200
```

...and so on. This tells the server to match only the first hundred messages in the mailbox the first time, the second hundred the second time, etc. In fact, it might likely allow the server to optimize the search significantly. In the above example, whether the mailbox contains 50 or 150 or 250 messages, neither of the search commands shown will result in an error. It is up to the client to know when to stop moving its search window.

## 2.1. The ESEARCH Response

In response to an ESEARCH command, the server MUST return ESEARCH responses [RFC4731] (that is, not SEARCH responses). Because message numbers are not useful for mailboxes that are not selected, the responses MUST contain information about UIDs, not message numbers. This is true even if the source options specify that only the selected mailbox be searched.

Presence of a source option in the absence of a result option implies the "ALL" result option (see Section 3.1 of [RFC4731]). Note that this is not the same as the result from the SEARCH command described in the IMAP base protocol [RFC3501].

Source options describe which mailboxes must be searched for messages. An ESEARCH command with source options does not affect which mailbox, if any, is currently selected, regardless of which mailboxes are searched.

For each mailbox satisfying the source options, a single ESEARCH response MUST be returned if any messages in that mailbox match the search criteria. An ESEARCH response MUST NOT be returned for mailboxes that contain no matching messages. This is true even when result options such as MIN, MAX, and COUNT are specified (see Section 3.1 of [RFC4731]), and the values returned (lowest UID matched, highest UID matched, and number of messages matched, respectively) apply to the mailbox reported in that ESEARCH response.

Note that it is possible for an ESEARCH command to return no untagged responses (no ESEARCH responses at all), in the case that there are no matches to the search in any of the mailboxes that satisfy the source options. Clients can detect this situation by finding the tagged OK response without having received any matching untagged ESEARCH responses.

Each ESEARCH response MUST contain the MAILBOX, TAG, and UIDVALIDITY correlators. Correlators allow clients to issue several ESEARCH commands at once (pipelined). If the SEARCHRES [RFC5182] extension is used in an ESEARCH command, that ESEARCH command MUST be executed by the server after all previous SEARCH/ESEARCH commands have completed and before any subsequent SEARCH/ESEARCH commands are executed. The server MAY perform consecutive ESEARCH commands in parallel as long as none of them use the SEARCHRES extension.

## 2.2. Source Options: Specifying Mailboxes to Search

The source options, if present, MUST contain a mailbox specifier as defined in the IMAP NOTIFY extension [RFC5465], Section 6 (using the "filter-mailboxes" ABNF item), with the following differences:

1. The "selected-delayed" specifier is not valid here.
2. A "subtree-one" specifier is added. The "subtree" specifier results in a search of the specified mailbox and all selectable mailboxes that are subordinate to it, through an indefinitely deep hierarchy. The "subtree-one" specifier results in a search of the specified mailbox and all selectable child mailboxes, one hierarchy level down.

If "subtree" is specified, the server MUST defend against loops in the hierarchy (for example, those caused by recursive file-system links within the message store). The server SHOULD do this by keeping track of the mailboxes that have been searched, and terminating the hierarchy traversal when a repeat is found. If it cannot do that, it MAY do it by limiting the hierarchy depth.

If the source options are not present, the value "selected" is assumed -- that is, only the currently selected mailbox is searched.

The "personal" source option is a particularly convenient way to search all of the current user's mailboxes. Note that there is no way to use wildcard characters to search all mailboxes; the "mailboxes" source option does not do wildcard expansion.

If the source options include (or default to) "selected", the IMAP session MUST be in "selected" state. If the source options specify other mailboxes and NOT "selected", then the IMAP session MUST be in either "selected" or "authenticated" state. If the session is not in a correct state, the ESEARCH command MUST return a "BAD" result.

The client SHOULD NOT provide source options that resolve to including the same mailbox more than once. A server can, of course, remove the duplicates before processing, but the server MAY return "BAD" to an ESEARCH command with duplicate source mailboxes.

If the server supports the SEARCHRES [RFC5182] extension, then the "SAVE" result option is valid only if "selected" is specified or defaulted as the sole mailbox to be searched. If any source option other than "selected" is specified, the ESEARCH command MUST return a "BAD" result.

If the server supports the CONTEXT=SEARCH and/or CONTEXT=SORT extension [RFC5267], then the following additional rules apply:

- o The CONTEXT return option (Section 4.2 of [RFC5267]) can be used with an ESEARCH command.
- o If the UPDATE return option is used (Section 4.3 of [RFC5267]), it MUST apply only to the currently selected mailbox. If UPDATE is used and there is no mailbox currently selected, the ESEARCH command MUST return a "BAD" result.
- o The PARTIAL search return option (Section 4.4 of [RFC5267]) can be used and applies to each mailbox searched by the ESEARCH command.

If the server supports the Access Control List (ACL) [RFC4314] extension, then the logged-in user is required to have the "r" right for each mailbox she wants to search. In addition, any mailboxes that are not explicitly named (accessed through "personal" or "subtree", for example) are required to have the "l" right. Mailboxes matching the source options for which the logged-in user lacks sufficient rights MUST be ignored by the ESEARCH command processing. In particular, ESEARCH responses MUST NOT be returned for those mailboxes.

### 2.3. Strictness in Search Matches

The base IMAP SEARCH command (Section 6.4.4. of [RFC3501]) requires strict substring matching in text searches. Many servers, however, use search engines that match strings in different ways, matching, for example, "swim" to "swam" and "swum" as well, or only doing full word matching (where "swim" will not match "swimming"). This is covered by the "Fuzzy Search" extension to IMAP [RFC6203], and that extension is compatible with this one and can be combined with it.

Whether or not Fuzzy Search is implemented or used, this extension explicitly allows flexible searching with respect to TEXT and BODY searches. Servers MAY use fuzzy text matching in multimapbox searches.

### 2.4. Server-Side Limitations on Search Volume

To avoid having a search use more than a reasonable share of server resources, servers MAY apply limits that go beyond loop protection, such as limits on the number of mailboxes that may be searched at once, and/or limits on the number or total size of messages searched. A server can apply those limits up front, responding with "NO [LIMIT]" if a limit is exceeded (see [RFC5530] for information about response codes). Alternatively, a server can process the search and terminate it when a limit is exceeded, responding with "OK [LIMIT]" and returning partial results. Note that searches that return partial results can cause complexity for client implementations and confusion to users.

### 3. Examples

In the following example, note that two ESEARCH commands are pipelined, and that the server is running them in parallel, interleaving a response to the second search amid the responses to the first (watch the tags).

```
C: tag1 ESEARCH IN (mailboxes "folder1" subtree "folder2") unseen
C: tag2 ESEARCH IN (mailboxes "folder1" subtree-one "folder2")
subject "chad"
S: * ESEARCH (TAG "tag1" MAILBOX "folder1" UIDVALIDITY 1) UID ALL
4001,4003,4005,4007,4009
S: * ESEARCH (TAG "tag2" MAILBOX "folder1" UIDVALIDITY 1) UID ALL
3001:3004,3788
S: * ESEARCH (TAG "tag1" MAILBOX "folder2/banana" UIDVALIDITY 503)
UID ALL 3002,4004
S: * ESEARCH (TAG "tag1" MAILBOX "folder2/peach" UIDVALIDITY 3) UID
ALL 921691
S: tag1 OK done
S: * ESEARCH (TAG "tag2" MAILBOX "folder2/salmon" UIDVALIDITY
1111111) UID ALL 50003,50006,50009,50012
S: tag2 OK done
```

### 4. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form (ABNF) as described in [RFC5234]. Terms not defined here are taken from [RFC3501], [RFC5465], or [RFC4466].

```
command-auth =/ esearch
               ; Update definition from IMAP base [RFC3501].
               ; Add new "esearch" command.

command-select =/ esearch
                ; Update definition from IMAP base [RFC3501].
                ; Add new "esearch" command.
```

```
filter-mailboxes-other =/ ("subtree-one" SP one-or-more-mailbox)
    ; Update definition from IMAP Notify [RFC5465].
    ; Add new "subtree-one" selector.

filter-mailboxes-selected = "selected"
    ; Update definition from IMAP Notify [RFC5465].
    ; We forbid the use of "selected-delayed".

one-correlator = ("TAG" SP tag-string) / ("MAILBOX" SP astring) /
    ("UIDVALIDITY" SP nz-number)
    ; Each correlator MUST appear exactly once.

scope-option = scope-option-name [SP scope-option-value]
    ; No options defined here.  Syntax for future extensions.

scope-option-name = tagged-ext-label
    ; No options defined here.  Syntax for future extensions.

scope-option-value = tagged-ext-val
    ; No options defined here.  Syntax for future extensions.

scope-options = scope-option *(SP scope-option)
    ; A given option may only appear once.
    ; No options defined here.  Syntax for future extensions.

esearch = "ESEARCH" [SP esearch-source-opts]
    [SP search-return-opts] SP search-program

search-correlator = SP "(" one-correlator *(SP one-correlator) ")"
    ; Updates definition in IMAP4 ABNF [RFC4466].

esearch-source-opts = "IN" SP "(" source-mbox [SP
    "(" scope-options ")" ] ")"

source-mbox = filter-mailboxes *(SP filter-mailboxes)
    ; "filter-mailboxes" is defined in IMAP Notify [RFC5465].
    ; See updated definition of filter-mailboxes-other, above.
    ; See updated definition of filter-mailboxes-selected, above.
```

## 5. Security Considerations

This new IMAP ESEARCH command allows a single command to search many mailboxes at once. On the one hand, a client could do that by sending many IMAP SEARCH commands. On the other hand, this makes it easier for a client to overwork a server, by sending a single command that results in an expensive search of tens of thousands of mailboxes. Server implementations need to be aware of that, and provide mechanisms that prevent a client from adversely affecting other users. Limitations on the number of mailboxes that may be searched in one command, and/or on the server resources that will be devoted to responding to a single client, are reasonable limitations for an implementation to impose (see also Section 2.4).

Implementations MUST, of course, apply access controls appropriately, limiting a user's access to ESEARCH in the same way its access is limited for any other IMAP commands. This extension has no data-access risks beyond what may be there in the unextended IMAP implementation.

Mailboxes matching the source options for which the logged-in user lacks sufficient rights MUST be ignored by the ESEARCH command processing (see the paragraph about this in Section 2.2). In particular, any attempt to distinguish insufficient access from non-existent mailboxes may expose information about the mailbox hierarchy that isn't otherwise available to the client.

If "subtree" is specified, the server MUST defend against loops in the hierarchy (see the paragraph about this in Section 2.2).

## 6. IANA Considerations

The "IMAP Capabilities" registry is currently located at <http://www.iana.org/assignments/imap-capabilities>.

IANA is asked to change the reference for the IMAP capability "MULTISEARCH" to point to this document.

## 7. Implementation Status

[[RFC Editor: Please remove this section at publication.]]

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 6982. 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 6982, "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".

The following implementations are known to exist:

- o Oracle has a server implementation that is not currently in a product.

- o There is a client implementation that has been tested with the Oracle server. No further information is available.

Interest has been expressed in creating the following implementations:

- o Isode Limited

## 8. Changes Since RFC 6237

- o Change to Standards Track.
- o Added paragraph about duplicate mailboxes.
- o Added Section 2.3 about fuzzy search.
- o Added Section 7, "Implementation Status".  
[[RFC Editor: Please remove this bullet at publication.]]

## 9. Acknowledgments

The authors gratefully acknowledge feedback provided by Timo Sirainen, Peter Coates, Arnt Gulbrandsen, and Chris Newman.

## 10. References

### 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2978] Freed, N. and J. Postel, "IANA Charset Registration Procedures", BCP 19, RFC 2978, October 2000.
- [RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.
- [RFC4314] Melnikov, A., "IMAP4 Access Control List (ACL) Extension", RFC 4314, December 2005.
- [RFC4466] Melnikov, A. and C. Daboo, "Collected Extensions to IMAP4 ABNF", RFC 4466, April 2006.
- [RFC4731] Melnikov, A. and D. Cridland, "IMAP4 Extension to SEARCH Command for Controlling What Kind of Information Is Returned", RFC 4731, November 2006.
- [RFC5182] Melnikov, A., "IMAP Extension for Referencing the Last SEARCH Result", RFC 5182, March 2008.
- [RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

- [RFC5267] Cridland, D. and C. King, "Contexts for IMAP4", RFC 5267, July 2008.
- [RFC5465] Gulbrandsen, A., King, C. and A. Melnikov, "The IMAP NOTIFY Extension", RFC 5465, February 2009.
- [RFC5530] Gulbrandsen, A., "IMAP Response Codes", RFC 5530, May 2009.

## 10.2. Informative References

- [RFC6203] Sirainen, T., "IMAP4 Extension for Fuzzy Search", RFC 6203, March 2011.

## Authors' Addresses

Barry Leiba  
Huawei Technologies

Phone: +1 646 827 0648  
Email: [barryleiba@computer.org](mailto:barryleiba@computer.org)  
URI: <http://internetmessagingtechnology.org/>

Alexey Melnikov  
Isode Limited  
5 Castle Business Village  
36 Station Road  
Hampton, Middlesex TW12 2BX  
UK

Email: [Alexey.Melnikov@isode.com](mailto:Alexey.Melnikov@isode.com)  
URI: <http://www.melnikov.ca/>

APPSAWG  
Internet-Draft  
Obsoletes: 2388 (if approved)  
Intended status: Standards Track  
Expires: October 12, 2015

L. Masinter  
Adobe  
April 10, 2015

Returning Values from Forms: multipart/form-data  
draft-ietf-appsawg-multipart-form-data-11

Abstract

This specification defines the multipart/form-data Internet Media Type, which can be used by a wide variety of applications and transported by a wide variety of protocols as a way of returning a set of values as the result of a user filling out a form. It obsoletes RFC 2388.

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 <http://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 October 12, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |    |
|---|----|
| 1. Introduction . . . . .   | 2  |
| 2. percent-encoding option . . . . .                                  | 3  |
| 3. Advice for Forms and Form Processing . . . . .                     | 3  |
| 4. Definition of multipart/form-data . . . . .                        | 4  |
| 4.1. Boundary parameter of multipart/form-data . . . . .              | 4  |
| 4.2. Content-Disposition header for each part . . . . .               | 4  |
| 4.3. filename attribute of content-distribution part header . . . . . | 4  |
| 4.4. Multiple files for one form field . . . . .                      | 5  |
| 4.5. Content-Type header for each part . . . . .                      | 5  |
| 4.6. The charset parameter for text/plain form data . . . . .         | 5  |
| 4.7. The _charset_ field for default charset . . . . .                | 6  |
| 4.8. Content-Transfer-Encoding deprecated . . . . .                   | 6  |
| 4.9. Other Content- headers . . . . .                                 | 7  |
| 5. Operability considerations . . . . .                               | 7  |
| 5.1. Non-ASCII field names and values . . . . .                       | 7  |
| 5.1.1. Avoid non-ASCII field names . . . . .                          | 7  |
| 5.1.2. Interpreting forms and creating form-data . . . . .            | 7  |
| 5.1.3. Parsing and interpreting form data . . . . .                   | 8  |
| 5.2. Ordered fields and duplicated field names . . . . .              | 8  |
| 5.3. Interoperability with web applications . . . . .                 | 8  |
| 5.4. Correlating form data with the original form . . . . .           | 9  |
| 6. IANA Considerations . . . . .                                      | 9  |
| 7. Security Considerations . . . . .                                  | 9  |
| 8. Media type registration for multipart/form-data . . . . .          | 10 |
| 9. References . . . . .   | 11 |
| 9.1. Normative References . . . . .                                   | 11 |
| 9.2. Informative References . . . . .                                 | 12 |
| Appendix A. Changes from RFC 2388 . . . . .                           | 12 |
| Appendix B. Alternatives . . . . .                                    | 13 |
| Author's Address . . . . .  | 13 |

## 1. Introduction

In many applications, it is possible for a user to be presented with a form. The user will fill out the form, including information that is typed, generated by user input, or included from files that the user has selected. When the form is filled out, the data from the form is sent from the user to the receiving application.

The definition of "multipart/form-data" is derived from one of those applications, originally set out in [RFC1867] and subsequently incorporated into HTML 3.2 [W3C.REC-html32-19970114], where forms are expressed in HTML, and in which the form data is sent via HTTP or

electronic mail. This representation is widely implemented in numerous web browsers and web servers.

However, "multipart/form-data" is also used for forms that are presented using representations other than HTML (spreadsheets, PDF, etc.), and for transport using means other than electronic mail or HTTP; it is used in distributed applications which do not involve forms at all, or do not have users filling out the form. For this reason, this document defines a general syntax and semantics independent of the application for which it is used, with specific rules for web applications noted in context.

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 BCP 14, RFC 2119 [RFC2119].

## 2. percent-encoding option

Within this specification, "percent-encoding" (as defined in [RFC3986]) is offered as a possible way of encoding characters in file names that are otherwise disallowed, including non-ASCII characters, spaces, control characters and so forth. The encoding is created replacing each non-ASCII or disallowed character with a sequence, where each byte of the UTF-8 encoding of the character is represented by a percent-sign (%) followed by the (case-insensitive) hexadecimal of that byte.

## 3. Advice for Forms and Form Processing

The representation and interpretation of forms and the nature of form processing is not specified by this document. However, for forms and form-processing that result in generation of multipart/form-data, some suggestions are included.

In a form, there is generally a sequence of fields, where each field is expected to be supplied with a value, e.g. by a user who fills out the form. Each field has a name. After a form has been filled out, and the form's data is "submitted": the form processing results in a set of values for each field-- the "form data".

In forms that work with multipart/form-data, field names could be arbitrary Unicode strings; however, restricting field names to ASCII will help avoid some interoperability issues (see Section 5.1).

Within a given form, ensuring field names are unique is also helpful. Some fields may have default values or presupplied values in the form itself. Fields with presupplied values might be hidden or invisible;

this allows using generic processing for form data from a variety of actual forms.

#### 4. Definition of multipart/form-data

The media-type "multipart/form-data" follows the model of multipart MIME data streams as specified in [RFC2046] Section 5.1; changes are noted in this document.

A "multipart/form-data" body contains a series of parts, separated by a boundary.

##### 4.1. Boundary parameter of multipart/form-data

As with other multipart types, the parts are delimited with a boundary delimiter, constructed using CRLF, "--", the value of the boundary parameter. The boundary is supplied as a "boundary" parameter to the "multipart/form-data" type. As noted in [RFC2046] Section 5.1, the boundary delimiter MUST NOT appear inside any of the encapsulated parts, and it is often necessary to enclose the boundary parameter values in quotes on the Content-type line.

##### 4.2. Content-Disposition header for each part

Each part MUST contain a "content-disposition" header [RFC2183] and where the disposition type is "form-data". The "content-disposition" header MUST also contain an additional parameter of "name"; the value of the "name" parameter is the original field name from the form (possibly encoded; see Section 5.1). For example, a part might contain a header:

```
Content-Disposition: form-data; name="user"
```

with the body of the part containing the form data of the "user" field.

##### 4.3. filename attribute of content-distribution part header

For form data that represents the content of a file, a name for the file SHOULD be supplied as well, by using a "filename" parameter of the "content-disposition" header. The file name isn't mandatory for cases where the file name isn't available or is meaningless or private; this might result, for example, from selection or drag-and-drop or where the form data content is streamed directly from a device.

If a filename parameter is supplied, the requirements of [RFC2183] Section 2.3 for "receiving MUA" apply to receivers of "multipart/

form-data" as well: Do not use the file name blindly, check and possibly change to match local filesystem conventions if applicable, do not use directory path information that may be present.

In most multipart types, the MIME headers in each part are restricted to US-ASCII; for compatibility with those systems, file names normally visible to users MAY be encoded using the percent-encoding method in Section 2, following how a "file:" URI [I-D.ietf-appsawg-file-scheme] might be encoded.

NOTE: The encoding method described in [RFC5987], which would add a "filename\*" paramter to the "Content-Disposition" header, MUST NOT be used.

Some commonly deployed systems use multipart/form-data with file names directly encoded including octets outside the US-ASCII range. The encoding used for the file names is typically UTF-8, although HTML forms will use the charset associated with the form.

#### 4.4. Multiple files for one form field

The form data for a form field might include multiple files.

[RFC2388] suggested that multiple files for a single form field be transmitted using a nested multipart/mixed part. This usage is deprecated.

To match widely deployed implementations, multiple files MUST be sent by supplying each file in a separate part, but all with the same "name" parameter.

Receiving applications intended for wide applicability (e.g. multipart/form-data parsing libraries) SHOULD also support the older method of supplying multiple files.

#### 4.5. Content-Type header for each part

Each part MAY have an (optional) "content-type", which defaults to "text/plain". If the contents of a file are to be sent, the file data SHOULD be labeled with an appropriate media type, if known, or "application/octet-stream".

#### 4.6. The charset parameter for text/plain form data

In the case where the form data is text, the charset parameter for the "text/plain" Content-Type MAY be used to indicate the character encoding used in that part. For example, a form with a text field in

which a user typed "Joe owes <eu>100" where <eu> is the Euro symbol might have form data returned as:

```
--AaB03x
content-disposition: form-data; name="field1"
content-type: text/plain;charset=UTF-8
content-transfer-encoding: quoted-printable

Joe owes =E2=82=AC100.
--AaB03x
```

In practice, many widely deployed implementations do not supply a charset parameter in each part, but, rather, they rely on the notion of a "default charset" for a multipart/form-data instance. Subsequent sections will explain how the default charset is established.

#### 4.7. The `_charset_` field for default charset

Some form processing applications (including HTML) have the convention that the value of a form entry with entry name "`_charset_`" and type "hidden" is automatically set when the form is opened; the value is used as the default charset of text field values (see form-charset in Section 5.1.2). In such cases, the value of the default charset for each text/plain part without a charset parameter is the supplied value. For example:

```
--AaB03x
content-disposition: form-data; name="_charset_"

iso-8859-1
--AaB03x--
content-disposition: form-data; name="field1"

...text encoded in iso-8859-1 ...
AaB03x--
```

#### 4.8. Content-Transfer-Encoding deprecated

Previously, it was recommended that senders use a "Content-Transfer-Encoding" encoding (such as "quoted-printable") for each non-ASCII part of a multipart/form-data body, because that would allow use in transports that only support a "7BIT" encoding. This use is deprecated for use in contexts that support binary data such as HTTP. Senders SHOULD NOT generate any parts with a "Content-Transfer-Encoding" header.

Currently, no deployed implementations that send such bodies have been discovered.

#### 4.9. Other Content- headers

The "multipart/form-data" media type does not support any MIME headers in the parts other than Content-Type, Content-Disposition, and (in limited circumstances) Content-Transfer-Encoding. Other headers MUST NOT be included and MUST be ignored.

### 5. Operability considerations

#### 5.1. Non-ASCII field names and values

Normally, MIME headers in multipart bodies are required to consist only of 7-bit data in the US-ASCII character set. While [RFC2388] suggested that non-ASCII field names be encoded according to the method in [RFC2047], this practice doesn't seem to have been followed widely.

This specification makes three sets of recommendations for three different states of workflow.

##### 5.1.1. Avoid non-ASCII field names

For broadest interoperability with existing deployed software, those creating forms SHOULD avoid non-ASCII field names. This should not be a burden, because in general the field names are not visible to users. The field names in the underlying need not match what the user sees on the screen.

If non-ASCII field names are unavoidable, form or application creators SHOULD use UTF-8 uniformly. This will minimize interoperability problems.

##### 5.1.2. Interpreting forms and creating form-data

Some applications of this specification will supply a character encoding to be used for interpretation of the multipart/form-data body. In particular, HTML 5 [W3C.REC-html5-20141028] uses:

- o The content of a '\_charset\_' field, if there is one.
- o the value of an accept-charset attribute of the <form> element, if there is one,
- o the character encoding of the document containing the form, if it is US-ASCII compatible,

- o otherwise UTF-8.

Call this value the form-charset. Any text, whether field name, field value, or (text/plain) form data which uses characters outside the ASCII range MAY be represented directly encoded in the form-charset.

### 5.1.3. Parsing and interpreting form data

While this specification provides guidance for creation of multipart/form-data, parsers and interpreters should be aware of the variety of implementations. File systems differ as to whether and how they normalize Unicode names, for example. The matching of form elements to form-data parts may rely on a fuzzier match. In particular, some multipart/form-data generators might have followed the previous advice of [RFC2388] and used the [RFC2047] "encoded-word" method of encoding non-ASCII values:

```
encoded-word = "=?" charset "?" encoding "?" encoded-text "=?"
```

Others have been known to follow [RFC2231], to send unencoded UTF-8, or even strings encoded in the form-charset.

For this reason, interpreting "multipart/form-data" (even from conforming generators) may require knowing the charset used in form encoding, in cases where the `_charset_` field value or a charset parameter of a text/plain Content-Type header is not supplied.

### 5.2. Ordered fields and duplicated field names

Form processors given forms with a well-defined ordering SHOULD send back results in order (note that there are some forms which do not define a natural order.) Intermediaries MUST NOT reorder the results. Form parts with identical field names MUST NOT be coalesced.

### 5.3. Interoperability with web applications

Many web applications use the "application/x-url-encoded" method for returning data from forms. This format is quite compact, e.g.:

```
name=Xavier+Xantico&verdict=Yes&colour=Blue&happy=sad&Utf%F6r=Send
```

However, there is no opportunity to label the enclosed data with content type, apply a charset, or use other encoding mechanisms.

Many form-interpreting programs (primarily web browsers) now implement and generate multipart/form-data, but an existing

application might need to optionally support both the application/x-url-encoded format as well.

#### 5.4. Correlating form data with the original form

This specification provides no specific mechanism by which multipart/form-data can be associated with the form that caused it to be transmitted. This separation is intentional; many different forms might be used for transmitting the same data. In practice, applications may supply a specific form processing resource (in HTML, the ACTION attribute in a FORM tag) for each different form. Alternatively, data about the form might be encoded in a "hidden field" (a field which is part of the form but which has a fixed value to be transmitted back to the form-data processor.)

#### 6. IANA Considerations

Please update the Internet Media Type registration of multipart/form-data to point to this document, using the template in Section 8. In addition, please update the registrations of the "name" parameter and the "form-data" value in the "Content Disposition Values and Parameters" registry to both point to this document.

#### 7. Security Considerations

All form processing software should treat user supplied form-data with sensitivity, as it often contains confidential or personally identifying information. There is widespread use of form "auto-fill" features in web browsers; these might be used to trick users to unknowingly send confidential information when completing otherwise innocuous tasks. Multipart/form-data does not supply any features for checking integrity, ensuring confidentiality, avoiding user confusion, or other security features; those concerns must be addressed by the form-filling and form-data-interpreting applications.

Applications which receive forms and process them must be careful not to supply data back to the requesting form processing site that was not intended to be sent.

It is important when interpreting the filename of the Content-Disposition header to not overwrite files in the recipient's file space inadvertently.

User applications that request form information from users must be careful not to cause a user to send information to the requestor or a third party unwillingly or unwittingly. For example, a form might request 'spam' information to be sent to an unintended third party,

or private information to be sent to someone that the user might not actually intend. While this is primarily an issue for the representation and interpretation of forms themselves (rather than the data representation of the form data), the transportation of private information must be done in a way that does not expose it to unwanted prying.

With the introduction of form-data that can reasonably send back the content of files from a user's file space, the possibility arises that a user might be sent an automated script that fills out a form and then sends one of the user's local files to another address. Thus, additional caution is required when executing automated scripting where form-data might include a user's files.

Files sent via multipart/form-data may contain arbitrary executable content, and precautions against malicious content are necessary.

The considerations of [RFC2183] Sections 2.3 and 5 with respect to the filename parameter of the Content-Disposition header also apply to its usage here.

#### 8. Media type registration for multipart/form-data

This section is the [RFC6838] media type registration.

Type name: multipart

Subtype name: form-data

Required parameters: boundary

Optional parameters: none

Encoding considerations: Common use is BINARY.

In limited use (or transports that restrict the encoding to 7BIT or 8BIT each part is encoded separately using Content-Transfer-Encoding Section 4.8).

Security considerations: See Section 7 of this document.

Interoperability considerations: This document makes several recommendations for interoperability with deployed implementations, including Section 4.8.

Published specification: This document.

Applications that use this media type: Numerous web browsers, servers, and web applications.

Fragment identifier considerations: None: Fragment identifiers are not defined for this type.

Additional information: None: no deprecated alias names, magic numbers, file extensions or Macintosh sssfile type codes.

Person & email address to contact                      for further information  
Author of this document.

Intended Usage: COMMON

Restrictions on usage: none

Author: Author of this document.

Change controller: IETF

Provisional registration: N/A

## 9. References

### 9.1. Normative References

- [RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.
- [RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2183] Troost, R., Dorner, S., and K. Moore, "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, August 1997.
- [RFC2231] Freed, N. and K. Moore, "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations", RFC 2231, November 1997.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

## 9.2. Informative References

- [I-D.ietf-appsawg-file-scheme]  
Kerwin, M., "The file URI Scheme", draft-ietf-appsawg-file-scheme-00 (work in progress), January 2015.
- [RFC1867] Nebel, E. and L. Masinter, "Form-based File Upload in HTML", RFC 1867, November 1995.
- [RFC2388] Masinter, L., "Returning Values from Forms: multipart/form-data", RFC 2388, August 1998.
- [RFC5987] Reschke, J., "Character Set and Language Encoding for Hypertext Transfer Protocol (HTTP) Header Field Parameters", RFC 5987, August 2010.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, January 2013.
- [W3C.REC-html32-19970114]  
Raggett, D., "HTML 3.2 Reference Specification", World Wide Web Consortium Recommendation REC-html32-19970114, January 1997, <<http://www.w3.org/TR/REC-html32-19970114>>.
- [W3C.REC-html5-20141028]  
Hickson, I., Berjon, R., Faulkner, S., Leithead, T., Navara, E., O'Connor, E., and S. Pfeiffer, "HTML5", World Wide Web Consortium Recommendation REC-html5-20141028, October 2014, <<http://www.w3.org/TR/2014/REC-html5-20141028>>.

## Appendix A. Changes from RFC 2388

The handling of non-ASCII field names changed-- no longer recommending the RFC 2047 method, instead suggesting senders send UTF-8 field names directly, and file names directly in the form-charset.

The handling of multiple files submitted as the result of a single form field (e.g. HTML's <input type=file multiple> element) results in each file having its own top level part with the same name parameter; the method of using a nested "multipart/mixed" from [RFC2388] is no longer recommended for creators, and not required for receivers as there are no known implementations of senders.

The `_charset_` convention and use of an explicit form-data charset is documented.

'boundary' is a required parameter in Content-Type.

The relationship of the ordering of fields within a form and the ordering of returned values within multipart/form-data was not defined before, nor was the handling of the case where a form has multiple fields with the same name.

Editorial: Removed obsolete discussion of alternatives in appendix. Update references. Move outline of form processing into Introduction.

#### Appendix B. Alternatives

There are numerous alternative ways in which form data can be encoded; many are listed in [RFC2388] section 5.2. The multipart/form-data encoding is verbose, especially if there are many fields with short values. In most use cases, this overhead isn't significant.

More problematic are the differences introduced when implementors opted to not follow [RFC2388] when encoding non-ASCII field names (perhaps because "may" should have been "MUST"). As a result, parsers need to be more complex for matching against the possible outputs of various encoding methods.

#### Author's Address

Larry Masinter  
Adobe

Email: [masinter@adobe.com](mailto:masinter@adobe.com)  
URI: <http://larry.masinter.net>

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: March 17, 2015

J. Levine  
Taughannock Networks  
M. Delany  
Apple Inc.  
September 13, 2014

A "Null MX" No Service Resource Record for Domains that Accept No Mail  
draft-ietf-appsawg-nullmx-10

## Abstract

Internet mail determines the address of a receiving server through the DNS, first by looking for an MX record and then by looking for an A/AAAA record as a fallback. Unfortunately this means that the A/AAAA record is taken to be mail server address even when that address does not accept mail. The no service MX RR, informally called null MX, formalizes the existing mechanism by which a domain announces that it accepts no mail, without having to provide a mail server, which permits significant operational efficiencies.

## 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 <http://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 March 17, 2015.

## Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |   |
|---|---|
| 1. Conventions Used in This Document . . . . .                | 2 |
| 2. Introduction . . . . .                                     | 2 |
| 3. MX Resource Records Specifying Null MX . . . . .           | 3 |
| 4. Effects of Null MX . . . . .                               | 3 |
| 4.1. SMTP Server Benefits . . . . .                           | 3 |
| 4.2. Sending Mail from Domains that Publish Null MX . . . . . | 4 |
| 5. Security Considerations . . . . .                          | 5 |
| 6. IANA Considerations . . . . .                              | 5 |
| 7. Acknowledgements . . . . .                                 | 5 |
| 8. References . . . . .                                       | 5 |
| 8.1. Normative References . . . . .                           | 5 |
| 8.2. Informative References . . . . .                         | 6 |
| Appendix A. Change Log . . . . .                              | 6 |
| A.1. Change to appsawg-nullmx-10 . . . . .                    | 6 |
| A.2. Change to appsawg-nullmx-09 . . . . .                    | 6 |
| A.3. Change to appsawg-nullmx-08 . . . . .                    | 6 |
| A.4. Change to appsawg-nullmx-07 . . . . .                    | 6 |
| A.5. Change to appsawg-nullmx-06 . . . . .                    | 7 |
| A.6. Change to appsawg-nullmx-05 . . . . .                    | 7 |
| A.7. Change to appsawg-nullmx-04 . . . . .                    | 7 |
| A.8. Change to appsawg-nullmx-03 . . . . .                    | 7 |
| A.9. Change to appsawg-nullmx-02 . . . . .                    | 7 |
| A.10. Change to appsawg-nullmx-1 . . . . .                    | 7 |
| A.11. Change to appsawg-nullmx-0 . . . . .                    | 7 |
| Authors' Addresses . . . . .                                  | 7 |

## 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].

The terms RFC5321.MailFrom and RFC5322.From are used as defined in [RFC5598].

## 2. Introduction

This document defines the No Service MX, informally called null MX, as a simple mechanism by which a domain can indicate that it does not accept email.

SMTP clients have a prescribed sequence for identifying a server that accepts email for a domain. Section 5 of [RFC5321] covers this in detail, but in essence the SMTP client first looks up a DNS MX RR and if that is not found it falls back to looking up a DNS A or AAAA RR. Hence this overloads an email service semantic onto a DNS record with a different primary mission.

If a domain has no MX records, senders will attempt to deliver mail to the hosts at the domain's A or AAAA record's addresses. If there is no SMTP listener at the A/AAAA address, message delivery will be attempted repeatedly for a long period, typically a week, before the sending MTA gives up. This will delay notification to the sender in the case of misdirected mail, and will consume resources at the sender.

This document defines a null MX that will cause all mail delivery attempts to a domain to fail immediately, without requiring domains to create SMTP listeners dedicated to preventing delivery attempts.

### 3. MX Resource Records Specifying Null MX

To indicate that a domain does not accept email, it advertises a single MX RR (see [RFC1035], section 3.3.9) with an RDATA section consisting of preference number 0, and a zero length label, written in master files as ".", as the exchange domain, to denote that there exists no mail exchanger for a domain. Since "." is not a valid host name, a null MX record can not be confused with an ordinary MX record. The use of "." as a pseudo-host name meaning no service available is modeled on the SRV RR [RFC2782] where it has a similar meaning.

A domain that advertises a null MX MUST NOT advertise any other MX RR.

### 4. Effects of Null MX

The null MX record has a variety of efficiency and usability benefits.

#### 4.1. SMTP Server Benefits

Mail often has an incorrect address due to user error, where the address was mistranscribed or misunderstood, for example, to `alice@www.example.com` or `alice@example.org` or `alice@example.com` rather than `alice@example.com`. Null MX allows a mail system to report the delivery failure when the user sends the message, rather than hours or days later.

Senders of abusive mail often use forged undeliverable return addresses. Null MX allows Delivery Status Notifications (DSNs) and other attempted responses to such mail to be disposed of efficiently.

The ability to detect domains that do not accept email offers resource savings to an SMTP client. It will discover on the first sending attempt that an address is not deliverable, avoiding queuing and retries.

When a submission or SMTP relay server rejects an envelope recipient due to a domain's null MX record, it SHOULD use a 556 reply code[code521556] (Requested action not taken: domain does not accept mail) and a 5.1.TBD enhanced status code (Permanent failure: Recipient address has null MX).

A receiving SMTP server that chooses to reject email during the SMTP conversation that presents an undeliverable RFC5321.MailFrom or RFC5322.From domain can be more confident that for other messages a subsequent attempt to send a DSN or other response will reach a recipient SMTP server.

SMTP servers that reject mail because a RFC5321.MailFrom or RFC5322.From domain has a null MX record SHOULD use a 550 reply code (Requested action not taken: mailbox unavailable) and a 5.7.TBD enhanced status code (Permanent failure: Sender address has null MX).

#### 4.2. Sending Mail from Domains that Publish Null MX

Null MX is primarily intended for domains that do not send or receive any mail, but have mail sent to them anyway due to mistakes or malice. Many receiving systems reject mail that has an invalid return address. Return addresses are needed to allow the sender to handle message delivery errors. An invalid return address often signals that the message is spam. Hence mail systems SHOULD NOT publish a null MX record for domains that they use in RFC5321.MailFrom or RFC5322.From addresses. If a server nonetheless does so, it risks having its mail rejected.

Operators of domains that do not send mail can publish SPF -all [RFC7208] policies to make an explicit declaration that the domains send no mail.

Null MX is not intended to be a replacement for the null reverse path described in RFC 5321 section 4.5.5 and does not change the meaning or use of a null reverse path.

## 5. Security Considerations

Within the DNS, a null MX RR is an ordinary MX record and presents no new security issues. If desired, it can be secured in the same manner as any other DNS record using DNSSEC.

## 6. IANA Considerations

IANA is requested to add the following entries to the "Enumerated Status Codes" sub-registry of the Simple Mail Transfer Protocol (SMTP) Enhanced Status Codes Registry.

Code: X.1.TBD  
Sample Text: Recipient address has null MX  
Associated basic status code: 556  
Description: This status code is returned when the associated address is marked as invalid using a null MX.  
Reference: [this document]  
Submitter: [authors of this document]  
Change controller: IESG

Code: X.7.TBD  
Sample Text: Sender address has null MX  
Associated basic status code: 550  
Description: This status code is returned when the associated sender address has a null MX, and the SMTP receiver is configured to reject mail from such sender (e.g. because it could not return a DSN).  
Reference: [this document]  
Submitter: [authors of this document]  
Change controller: IESG

## 7. Acknowledgements

We thank Dave Crocker for his diligent and lengthy shepherding of this document, and members of the appsawg working group for their constructive suggestions.

## 8. References

### 8.1. Normative References

- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008.

[code521556]  
Klensin, J., "SMTP 521 and 556 Reply Codes", internet-draft draft-klensin-smtp-521code, .

## 8.2. Informative References

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC5598] Crocker, D., "Internet Mail Architecture", RFC 5598, July 2009.

[RFC7208] Kitterman, S., "Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1", RFC 7208, April 2014.

## Appendix A. Change Log

\*NOTE TO RFC EDITOR: This section may be removed upon publication of this document as an RFC.\*

### A.1. Change to appsawg-nullmx-10

Minor twiddle to clarify reference.

### A.2. Change to appsawg-nullmx-09

Change 521 to 556, change reference.

### A.3. Change to appsawg-nullmx-08

Fix name of IANA registry.

Yea, even yet more editorial cleanup.

### A.4. Change to appsawg-nullmx-07

Add new enhanced status codes and ref for 521 return code.

Even yet more editorial cleanup.

A.5. Change to appsawg-nullmx-06

Even more editorial cleanup.

Mention SRV

you SHOULD NOT put a null MX on domains that send mail

A.6. Change to appsawg-nullmx-05

Fix ID nits, add NULL IANA section. More editorial cleanup.

A.7. Change to appsawg-nullmx-04

Reorganize.

A.8. Change to appsawg-nullmx-03

Editorial nits per Murray.

A.9. Change to appsawg-nullmx-02

Should not publish NULL MX with other MX.

Never say never.

Add 5.1.2 enhanced status code.

Minor editorial changes.

A.10. Change to appsawg-nullmx-1

Editorial improvements per D. Crocker's review.

A.11. Change to appsawg-nullmx-0

Fix typos.

Authors' Addresses

John Levine  
Taughannock Networks  
PO Box 727  
Trumansburg, NY 14886

Phone: +1 831 480 2300  
Email: standards@taugh.com  
URI: <http://jl.ly>

Internet-Draft

Null MX

September 2014

Mark Delany  
Apple Inc.  
1 Infinite Loop  
Cupertino, CA 95014

Email: mx0dot@yahoo.com

APPSAWG  
Internet-Draft  
Intended status: Standards Track  
Expires: December 28, 2014

S. Bosch  
June 26, 2014

Sieve Email Filtering: Detecting Duplicate Deliveries  
draft-ietf-appsawg-sieve-duplicate-09

Abstract

This document defines a new test command "duplicate" for the "Sieve" email filtering language. This test adds the ability to detect duplications. The main application for this new test is handling duplicate deliveries commonly caused by mailing list subscriptions or redirected mail addresses. The detection is normally performed by matching the message ID to an internal list of message IDs from previously delivered messages. For more complex applications, the "duplicate" test can also use the content of a specific header or other parts of the message.

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 <http://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 December 28, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|  |    |
|--|----|
| 1. Introduction . . . . .                              | 3  |
| 2. Conventions Used in This Document . . . . .         | 3  |
| 3. Test "duplicate" . . . . .                          | 3  |
| 3.1. Arguments ":header" and ":uniqueid" . . . . .     | 5  |
| 3.2. Argument ":handle" . . . . .                      | 6  |
| 3.3. Arguments ":seconds" and ":last" . . . . .        | 7  |
| 3.4. Interaction with Other Sieve Extensions . . . . . | 8  |
| 4. Sieve Capability Strings . . . . .                  | 8  |
| 5. Examples . . . . .                                  | 9  |
| 5.1. Example 1 . . . . .                               | 9  |
| 5.2. Example 2 . . . . .                               | 9  |
| 5.3. Example 3 . . . . .                               | 10 |
| 5.4. Example 4 . . . . .                               | 11 |
| 6. Security Considerations . . . . .                   | 11 |
| 7. IANA Considerations . . . . .                       | 12 |
| 8. Acknowledgements . . . . .                          | 13 |
| 9. References . . . . .                                | 13 |
| 9.1. Normative References . . . . .                    | 13 |
| 9.2. Informative References . . . . .                  | 14 |
| Author's Address . . . . .                             | 14 |

## 1. Introduction

This document specifies an extension to the Sieve filtering language defined by RFC 5228 [SIEVE]. It adds a test to track whether or not a text string was seen before by the delivery agent in an earlier execution of the Sieve script. This can be used to detect and handle duplicate message deliveries.

Duplicate deliveries are a common side-effect of being subscribed to a mailing list. For example, if a member of the list decides to reply to both the user and the mailing list itself, the user will often get one copy of the message directly and another through the mailing list. Also, if someone cross-posts over several mailing lists to which the user is subscribed, the user will likely receive a copy from each of those lists. In another scenario, the user has several redirected mail addresses all pointing to his main mail account. If one of the user's contacts sends the message to more than one of those addresses, the user will likely receive more than a single copy. Using the "duplicate" extension, users have the means to detect and handle such duplicates, e.g. by discarding them, marking them as "seen", or putting them in a special folder.

Duplicate messages are normally detected using the Message-ID header field, which is required to be unique for each message. However, the "duplicate" test is flexible enough to use different criteria for defining what makes a message a duplicate, for example using the subject line or parts of the message body. Other applications of this new test command are also possible, as long as the tracked unique value is a string.

## 2. 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 [KEYWORDS].

Conventions for notations are as in [SIEVE] Section 1.1, including use of the "Usage:" label for the definition of action and tagged arguments syntax.

## 3. Test "duplicate"

```
Usage: "duplicate" [":handle" <handle: string>]
           [":header" <header-name: string> /
             ":uniqueid" <value: string>]
           [":seconds" <timeout: number>] [":last"]
```

The "duplicate" test identifies the message by a "unique ID", and, using that unique ID, keeps track of which messages were seen by a "duplicate" test during an earlier Sieve execution. In its basic form, the test gets the unique ID from the content of the message's Message-ID header. The "duplicate" test evaluates to "true" when the message was seen before and it evaluates to "false" when it was not.

As a side-effect, the "duplicate" test adds the unique ID to an internal duplicate tracking list once the Sieve execution finishes successfully. The first time a particular unique ID is seen, the message is not a duplicate, and the unique ID is added to the tracking list. If a future Sieve execution sees a message whose unique ID appears in the tracking list, that test will evaluate to "true", and that message will be considered a duplicate.

Note that this side-effect is performed only when the "duplicate" test is actually evaluated. If the "duplicate" test is nested in a control structure or it is not the first item of an "allof" or "anyof" test list, its evaluation depends on the result of preceding tests, which may produce unexpected results

Implementations MUST only update the internal duplicate tracking list when the Sieve script execution finishes successfully. If failing script executions add the unique ID to the duplicate tracking list, all "duplicate" tests in the Sieve script would erroneously yield "true" for the next delivery attempt of the same message, which can -- depending on the action taken for a duplicate -- easily lead to discarding the message without further notice.

However, deferring the definitive modification of the tracking list to the end of a successful Sieve script execution is not without problems. It can cause a race condition when a duplicate message is delivered in parallel before the tracking list is updated. This way, a duplicate message could be missed by the "duplicate" test. More complex implementations could use a locking mechanism to prevent this problem. But, irrespective of what implementation is chosen, situations in which the "duplicate" test erroneously yields "true" MUST be prevented.

The "duplicate" test MUST only check for duplicates amongst unique ID values encountered in previous executions of the Sieve script; it MUST NOT consider ID values encountered earlier in the current Sieve script execution as potential duplicates. This means that all "duplicate" tests in a Sieve script execution, including those located in scripts included using the "include" [INCLUDE] extension, MUST always yield the same result if the arguments are identical.

The Message-ID header field is assumed to be globally unique as

required in Section 3.6.4 of RFC 5322 [IMAIL]. In practice, this assumption may not always prove to be true. The "duplicate" test does not deal with this situation, which means that false duplicates may be detected in this case. However, the user can address such situations by specifying an alternative means of message identification using the ":header" or the ":uniqueid" argument, as described in the next section.

### 3.1. Arguments ":header" and ":uniqueid"

Duplicate tracking involves determining the unique ID for a given message, and checking whether that unique ID is in the duplicate tracking list. The unique ID for a message is determined as follows:

- o When neither the ":header" argument nor the ":uniqueid" argument is used, the unique ID is the content of the message's Message-ID header field.
- o When the ":header" argument is used, the unique ID is the content of the specified header field in the message. The header field name is not part of the resulting unique ID; it consists only of the field value.
- o When the ":uniqueid" argument is used, the unique ID is the string parameter that is specified with the argument.

The ":header" and ":uniqueid" arguments are mutually exclusive and specifying both for a single "duplicate" test command MUST trigger an error.

The syntax rules for the header name parameter of the ":header" argument are specified in Section 2.4.2.2 of RFC 5228 [SIEVE]. Note that implementations MUST NOT trigger an error for an invalid header name. Instead, the "duplicate" test MUST yield "false" unconditionally in this case. The parameter of the ":uniqueid" argument can be any string.

If the tracked unique ID value is extracted directly from a message header field, i.e., when the ":uniqueid" argument is not used, the following operations MUST be performed before the actual duplicate verification:

- o Unfold the header line as described in RFC 5322 [IMAIL], Section 2.2.3. (see also Section 2.4.2.2 of RFC 5228 [SIEVE]).
- o If possible, convert the header value to Unicode, encoded as UTF-8 (see Section 2.7.2 of RFC 5228 [SIEVE]). If conversion is not possible, the value is left unchanged.

- o Trim leading and trailing whitespace from the header value (see Section 2.2 of RFC 5228 [SIEVE]).

Note that these rules also apply to the Message-ID header field used by the basic "duplicate" test without a ":header" or ":uniqueid" argument. When the ":uniqueid" argument is used, any normalization needs to be done in the Sieve script itself, as the unique ID is created.

If the header field specified using the ":header" argument exists multiple times in the message, extraction of the unique ID MUST use only the first occurrence. This is true whether or not multiple occurrences are allowed by RFC 5322 [IMAIL], Section 3.6. If the specified header field is not present in the message, the "duplicate" test MUST yield "false" unconditionally. In that case the duplicate tracking list is left unmodified by this test, since no unique ID value is available. The same rules apply with respect to the Message-ID header field for the basic "duplicate" test without a ":header" or ":uniqueid" argument, since that header field could also be missing or occur multiple times.

The string parameter of the ":uniqueid" argument can be composed from arbitrary text extracted from the message using the "variables" [VARIABLES] extension. To extract text from the message body, the "foreverypart" and "extracttext" [SIEVE-MIME] extensions need to be used as well. This provides the user with detailed control over how the message's unique ID is created.

The unique ID MUST be matched case-sensitively with the contents of the duplicate tracking list, irrespective of how the unique ID was determined. To achieve case-insensitive behavior when the ":uniqueid" argument is used, the "set" command added by the "variables" [VARIABLES] extension can be used to normalize the unique ID value to upper or lower case.

### 3.2. Argument ":handle"

The "duplicate" test MUST track a unique ID value independent of its source. This means that all values in the duplicate list should be used for duplicate testing, regardless of whether they were obtained from the Message-ID header, from an arbitrary header specified using the ":header" argument or explicitly from the ":uniqueid" argument. The following three examples are equivalent and match the same entry in the duplicate tracking list:

```
require "duplicate";
if duplicate {
    discard;
```

```
}

require "duplicate";
if duplicate :header "message-id" {
    discard;
}

require ["duplicate", "variables"];
if header :matches "message-id" "*" {
    if duplicate :uniqueid "${0}" {
        discard;
    }
}
```

The `:handle` argument can be used to override this default behavior. The `:handle` argument separates a "duplicate" test from other duplicate tests with a different or omitted `:handle` argument. Using the `:handle` argument, unrelated "duplicate" tests can be prevented from interfering with each other: a message is only recognized as a duplicate when the tracked unique ID was seen before in an earlier script execution by a "duplicate" test with the same `:handle` argument.

NOTE: The necessary mechanism to track duplicate messages is very similar to the mechanism that is needed for tracking duplicate responses for the "vacation" [VACATION] action. One way to implement the necessary mechanism for the "duplicate" test is therefore to store a hash of the tracked unique ID and, if provided, the `:handle` argument.

### 3.3. Arguments `:seconds` and `:last`

Implementations SHOULD let entries in the tracking list expire after a short period of time. The user can explicitly control the length of this expiration time by means of the `:seconds` argument, which accepts an integer value specifying the timeout value in seconds. If the `:seconds` argument is omitted, an appropriate default value MUST be used. A default expiration time of around 7 days is usually appropriate. Sites SHOULD impose a maximum limit on the expiration time. If that limit is exceeded by the `:seconds` argument, the maximum value MUST silently be substituted; exceeding the limit MUST NOT produce an error. If the `:seconds` argument is zero, the "duplicate" test MUST yield "false" unconditionally.

When the `:last` argument is omitted, the expiration time for entries in the duplicate tracking list MUST be measured relative to the

moment at which the entry was first created; i.e., at the end of the successful script execution during which "duplicate" test returned "false" for a message with that particular unique ID value. This means that subsequent duplicate messages have no influence on the time at which the entry in the duplicate tracking list finally expires.

In contrast, when the ":last" argument is specified, the expiration time MUST be measured relative to the last script execution during which the "duplicate" test was used to check the entry's unique ID value. This effectively means that the entry in the duplicate tracking list will not expire while duplicate messages with the corresponding unique ID keep being delivered within intervals smaller than the expiration time.

It is possible to write Sieve scripts where during a single execution more than one "duplicate" test is evaluated with the same unique ID value and ":handle" argument but different ":seconds" or ":last" arguments. The resulting behavior is left undefined by this specification, so such constructs should be avoided. Implementations MAY choose to use the ":seconds" and ":last" arguments from the "duplicate" test that was evaluated last.

### 3.4. Interaction with Other Sieve Extensions

The "duplicate" test does not support either the "index" [DATE-INDEX], or "mime" [SIEVE-MIME] extensions directly, meaning that none of the ":index", ":mime" or associated arguments are added to the "duplicate" test when these extensions are active. The ":uniqueid" argument can be used in combination with the "variables" [VARIABLES] extension to achieve the same result indirectly.

Normally, Sieve scripts are executed at final delivery. However, with the "imapsieve" [IMAPSIEVE] extension, Sieve scripts are invoked when the IMAP [IMAP] server performs operations on the message store, e.g. when messages are uploaded, flagged, or moved to another location. The "duplicate" test is devised for use at final delivery and the semantics in the "imapsieve" context are left undefined. Therefore, implementations SHOULD NOT allow the "duplicate" test to be used in the context of "imapsieve".

## 4. Sieve Capability Strings

A Sieve implementation that defines the "duplicate" test command will advertise the capability string "duplicate".

## 5. Examples

### 5.1. Example 1

In this basic example, message duplicates are detected by tracking the Message-ID header. Duplicate deliveries are stored in a special folder contained in the user's Trash folder. If the folder does not exist, it is created automatically using the "mailbox" [MAILBOX] extension. This way, the user has a chance to recover messages when necessary. Messages that are not recognized as duplicates are stored in the user's inbox as normal.

```
require ["duplicate", "fileinto", "mailbox"];

if duplicate {
    fileinto :create "Trash/Duplicate";
}
```

### 5.2. Example 2

This example shows a more complex use of the "duplicate" test. The user gets network alerts from a set of remote automated monitoring systems. Several notifications can be received about the same event from different monitoring systems. The Message-ID of these messages is different, because these are all distinct messages from different senders. To avoid being notified more than a single time about the same event the user writes the following script:

```
require ["duplicate", "variables", "imap4flags",
        "fileinto"];

if header :matches "subject" "ALERT: *" {
    if duplicate :seconds 60 :uniqueid "${1}" {
        setflag "\\seen";
    }
    fileinto "Alerts";
}
```

The subjects of the notification message are structured with a predictable pattern which includes a description of the event. In the script above, the "duplicate" test is used to detect duplicate alert events. The message subject is matched against a pattern and the event description is extracted using the "variables" [VARIABLES] extension. If a message with that event in the subject was received before, but more than a minute ago, it is not detected as a duplicate due to the specified ":seconds" argument. In the the event of a duplicate, the message is marked as "seen" using the "imap4flags" [IMAP4FLAGS] extension. All alert messages are put into the "Alerts"

mailbox irrespective of whether those messages are duplicates or not.

### 5.3. Example 3

This example shows how the "duplicate" test can be used to limit the frequency of notifications sent using the "enotify" [NOTIFY] extension. Consider the following scenario: a mail user receives XMPP notifications [NOTIFY-XMPP] about new mail through Sieve, but sometimes a single contact sends many messages in a short period of time. Now the user wants to prevent being notified of all of those messages. The user wants to be notified about messages from each person at most once per 30 minutes and writes the following script:

```
require ["variables", "envelope", "enotify", "duplicate"];

if envelope :matches "from" "*" { set "sender" "${1}"; }
if header :matches "subject" "*" { set "subject" "${1}"; }

if not duplicate :seconds 1800 :uniqueid "${sender}"
{
    notify :message "[SIEVE] ${sender}: ${subject}"
        "xmpp:user@im.example.com";
}
```

The example shown above uses the message envelope sender rather than the Message-ID header as the unique ID for duplicate tracking.

The example can be extended to allow more messages from the same sender in close succession as long as the discussed subject is different. This can be achieved as follows:

```
require ["variables", "envelope", "enotify", "duplicate"];

if envelope :matches "from" "*" { set "sender" "${1}"; }
if header :matches "subject" "*" { set "subject" "${1}"; }

# account for 'Re:' prefix
if string :comparator "i;ascii-casemap"
    :matches "${subject}" "Re:*"
{
    set "subject" "${1}";
}

if not duplicate :seconds 1800
    :uniqueid "${sender} ${subject}"
{
    notify :message "[SIEVE] ${sender}: ${subject}"
        "xmpp:user@im.example.com";
}
```

This uses a combination of the message envelope sender and the subject of the message as the unique ID for duplicate tracking.

#### 5.4. Example 4

For this example, the mail user uses the "duplicate" test for two separate applications: for discarding duplicate events from a notification system and to mark certain follow-up messages in a software support mailing as "seen" using the "imap4flags" [IMAP4FLAGS] extension.

The two "duplicate" tests in the following example each use a different header to identify messages. However, these "X-Event-ID" and "X-Ticket-ID" headers can have similar values in this case (e.g. both based on a time stamp), meaning that one "duplicate" test can erroneously detect duplicates based on ID values tracked by the other. Therefore, the user wants to prevent the second "duplicate" test from matching ID values tracked by the first "duplicate" test and vice versa. This is achieved by specifying different ":handle" arguments for these tests.

```
require ["duplicate", "imap4flags"];

if duplicate :header "X-Event-ID" :handle "notifier" {
    discard;
}
if allof (
    duplicate :header "X-Ticket-ID" :handle "support",
    address "to" "support@example.com",
    header :contains "subject" "fileservers")
{
    setflag "\\seen";
}
```

#### 6. Security Considerations

A flood of unique messages could cause the duplicate tracking list to grow indefinitely. Therefore, implementations SHOULD limit the number of entries in the duplicate tracking list. When limiting the number of entries, implementations SHOULD discard the oldest ones first.

Scripts using the duplicate test evaluation should be aware that Message-IDs are not necessarily unique, either through the fault of benign generators or attackers injecting a message with the properties used by the duplicate Sieve filter at some point prior to the Sieve filter. Therefore, scripts are well advised to be

conservative with respect to actions taken when duplicate messages are identified only by Message-ID.

The list of unique IDs used for duplicate tracking can include privacy-sensitive information, such as Message-ID values, content of subject lines, and content extracted from message bodies. Implementations SHOULD protect that information, by obscuring it through hashing (see the note at the end of Section 3.2) and/or by storing it with a level of access control equivalent to that of the messages themselves.

These measures will not prevent an entity that has access to the duplicate tracking list from querying whether messages with certain unique ID values were received. As this operation is the essence of the "duplicate" test, this cannot be prevented, and may violate the expectations of the user. For example, a user who deletes a message from the server may expect that no record of it remains on the server, but that will not be true if its Message-ID is persisted on the server in the duplicate tracking list.

It's notable, however, that server logs will often store the information present on the duplicate tracking list anyway, and probably would expose plaintext Message-IDs for a much longer period than this mechanism would. Users of email services that intentionally delete such logs with the intent of limiting traceability should be made aware that use of the duplicate tracking mechanism re-exposes this information for the duration of the expiry interval. Therefore, a shorter default expiry interval may be appropriate in those situations.

## 7. IANA Considerations

The following template specifies the IANA registration of the Sieve extension specified in this document:

```
To: iana@iana.org
Subject: Registration of new Sieve extension

Capability name: duplicate
Description:     Adds test 'duplicate' that can be used to test
                  whether a particular message is a duplicate;
                  i.e., whether a copy of it was seen before by the
                  delivery agent that is executing the Sieve
                  script.
RFC number:     this RFC
Contact address: Sieve mailing list <sieve@ietf.org>
```

This information should be added to the list of sieve extensions given on <http://www.iana.org/assignments/sieve-extensions>.

## 8. Acknowledgements

Thanks to Brian Carpenter, Cyrus Daboo, Arnt Gulbrandsen, Tony Hansen, Kristin Hubner, Barry Leiba, Alexey Melnikov, Subramanian Moonesamy, Tom Petch, Hector Santos, Robert Sparks, Aaron Stone, and Stefan Winter for reviews and suggestions. With special thanks to Ned Freed for his guidance and support.

## 9. References

### 9.1. Normative References

#### [DATE-INDEX]

Freed, N., "Sieve Email Filtering: Date and Index Extensions", RFC 5260, July 2008.

#### [IMAIL]

Resnick, P., Ed., "Internet Message Format", RFC 5322, October 2008.

#### [IMAPSIEVE]

Leiba, B., "Support for Internet Message Access Protocol (IMAP) Events in Sieve", RFC 6785, November 2012.

#### [INCLUDE]

Daboo, C. and A. Stone, "Sieve Email Filtering: Include Extension", RFC 6609, May 2012.

#### [KEYWORDS]

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

#### [SIEVE]

Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language", RFC 5228, January 2008.

#### [SIEVE-MIME]

Hansen, T. and C. Daboo, "Sieve Email Filtering: MIME Part Tests, Iteration, Extraction, Replacement, and Enclosure", RFC 5703, October 2009.

#### [VARIABLES]

Homme, K., "Sieve Email Filtering: Variables Extension", RFC 5229, January 2008.

## 9.2. Informative References

- [IMAP]      Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.
- [IMAP4FLAGS]      Melnikov, A., "Sieve Email Filtering: Imap4flags Extension", RFC 5232, January 2008.
- [MAILBOX]      Melnikov, A., "The Sieve Mail-Filtering Language -- Extensions for Checking Mailbox Status and Accessing Mailbox Metadata", RFC 5490, March 2009.
- [NOTIFY]      Melnikov, A., Leiba, B., Segmuller, W., and T. Martin, "Sieve Email Filtering: Extension for Notifications", RFC 5435, January 2009.
- [NOTIFY-XMPP]      Saint-Andre, P. and A. Melnikov, "Sieve Notification Mechanism: Extensible Messaging and Presence Protocol (XMPP)", RFC 5437, January 2009.
- [VACATION]      Showalter, T. and N. Freed, "Sieve Email Filtering: Vacation Extension", RFC 5230, January 2008.

## Author's Address

Stephan Bosch  
Enschede  
NL

Email: [stephan@rename-it.nl](mailto:stephan@rename-it.nl)



Network Working Group  
Internet-Draft  
Obsoletes: 4395 (if approved)  
Intended status: Best Current Practice  
Expires: October 10, 2015

D. Thaler, Ed.  
Microsoft  
T. Hansen  
AT&T Laboratories  
T. Hardie  
Google  
L. Masinter  
Adobe  
April 8, 2015

Guidelines and Registration Procedures for URI Schemes  
draft-ietf-appsawg-uri-scheme-reg-06

Abstract

This document updates the guidelines and recommendations, as well as the IANA registration processes, for the definition of Uniform Resource Identifier (URI) schemes. It obsoletes RFC 4395.

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 <http://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 October 10, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |    |
|---|----|
| 1. Introduction . . . . .                                       | 2  |
| 1.1. URIs and IRIs . . . . .                                    | 3  |
| 2. Terminology . . . . .  | 3  |
| 3. Requirements for Permanent Scheme Definitions . . . . .      | 4  |
| 3.1. Demonstrable, New, Long-Lived Utility . . . . .            | 4  |
| 3.2. Syntactic Compatibility . . . . .                          | 5  |
| 3.3. Well-Defined . . . . .                                     | 5  |
| 3.4. Definition of Operations . . . . .                         | 6  |
| 3.5. Context of Use . . . . .                                   | 7  |
| 3.6. Internationalization and Character Encoding . . . . .      | 7  |
| 3.7. Clear Security and Privacy Considerations . . . . .        | 7  |
| 3.8. Scheme Name Considerations . . . . .                       | 8  |
| 3.9. Interoperability Considerations . . . . .                  | 9  |
| 4. Guidelines for Provisional URI Scheme Registration . . . . . | 9  |
| 5. Guidelines for Historical URI Scheme Registration . . . . .  | 10 |
| 6. Guidelines for Private URI Scheme Use . . . . .              | 10 |
| 7. URI Scheme Registration Procedure . . . . .                  | 10 |
| 7.1. General . . . . .  | 10 |
| 7.2. Registration Procedures . . . . .                          | 11 |
| 7.3. Change Control . . . . .                                   | 13 |
| 7.4. URI Scheme Registration Template . . . . .                 | 13 |
| 8. The "example" URI Scheme . . . . .                           | 14 |
| 8.1. "Example" URI Scheme Registration Request . . . . .        | 15 |
| 9. IANA Considerations . . . . .                                | 15 |
| 10. Security Considerations . . . . .                           | 16 |
| 11. Acknowledgements . . . . .                                  | 16 |
| 12. References . . . . .  | 16 |
| 12.1. Normative References . . . . .                            | 16 |
| 12.2. Informative References . . . . .                          | 17 |
| Appendix A. Changes Since RFC 4395 . . . . .                    | 18 |
| Authors' Addresses . . . . .                                    | 19 |

## 1. Introduction

The Uniform Resource Identifier (URI) protocol element and generic syntax is defined by [RFC3986]. Each URI begins with a scheme name, as defined by Section 3.1 of RFC 3986, that refers to a specification for identifiers within that scheme. The URI syntax provides a federated and extensible naming system, where each scheme's specification can further restrict the syntax and define the semantics of identifiers using that scheme.

This document obsoletes [RFC4395], which in turn obsoleted [RFC2717] and [RFC2718]. Recent documents have used the term "URI" for all resource identifiers, avoiding the term "URL" and reserving the term "URN" explicitly for those URIs using the "urn" scheme name ([RFC2141]). URN "namespaces" ([RFC3406]) are specific to the "urn" scheme and are not covered explicitly by this specification.

This document provides updated guidelines for the definition of new schemes, for consideration by those who are defining, registering, or evaluating those definitions, as well as a process and mechanism for registering schemes within the IANA URI Schemes registry. There is a single namespace for registered schemes. The intent of the registry is to:

- o provide a central point of discovery for established URI scheme names, and easy location of defining documents for schemes;
- o discourage multiple separate uses of the same scheme name;
- o help those proposing new scheme names to discern established trends and conventions, and avoid names that might be confused with existing ones;
- o encourage registration by setting a low barrier for registration.

### 1.1. URIs and IRIs

As originally defined, URIs only allowed a limited repertoire of characters chosen from US-ASCII. An Internationalized Resource Identifier (IRI), as defined by [RFC3987], extends the URI syntax to allow characters from a much greater repertoire, to accommodate resource identifiers from the world's languages. RFC 3987 [RFC3987] also defined a mapping between URIs and IRIs. IRIs use the same scheme names as URIs. Thus, there is no separate, independent registry or registration process for IRI schemes: the URI Schemes registry is used for both URIs and IRIs. Those who wish to describe resource identifiers that are useful as IRIs should define the corresponding URI syntax, and note that the IRI usage follows the rules and transformations defined in [RFC3987].

## 2. Terminology

Within this document, the key words MUST, MAY, SHOULD, REQUIRED, RECOMMENDED, and so forth are used within the general meanings established in [RFC2119], as requirements on future registrations.

This document distinguishes between a "scheme specification", being a document defining the syntax and semantics of a scheme, vs. a "scheme

registration request" being the completed template submitted to IANA. The term "scheme definition" refers generically to the syntax and semantics of a scheme, typically documented in a scheme specification.

### 3. Requirements for Permanent Scheme Definitions

This section gives considerations for new schemes. Meeting these guidelines is REQUIRED for 'permanent' scheme registration. 'Permanent' status is appropriate for, but not limited to, use in standards. For URI schemes defined or normatively referenced by IETF Standards-Track documents, 'permanent' registration status is REQUIRED.

[RFC3986] defines the overall syntax for URIs as:

```
URI = scheme ":" hier-part [ "?" query ] [ "#" fragment ]
```

A scheme definition cannot override the overall syntax for URIs. For example, this means that fragment identifiers cannot be re-used outside the generic syntax restrictions, and fragment identifiers are not scheme-specific. A scheme definition must specify the scheme name and the syntax of the scheme-specific part, which is clarified as follows:

```
URI = scheme ":" scheme-specific-part [ "#" fragment ]  
  
scheme-specific-part = hier-part [ "?" query ]
```

#### 3.1. Demonstrable, New, Long-Lived Utility

In general, the use and deployment of new schemes in the Internet infrastructure can be costly; some parts of URI processing are often scheme-dependent. Introducing a new scheme might require additional software, not only for client software and user agents but also in additional parts of the network infrastructure (gateways, proxies, caches) [W3CWebArch]. Since scheme names share a single, global namespace, it is desirable to avoid contention over use of short, mnemonic scheme names. New schemes ought to have utility to the Internet community beyond that available with already registered schemes. The scheme specification SHOULD discuss the utility of the scheme being registered.

### 3.2. Syntactic Compatibility

[RFC3986] defines the generic syntax for all URI schemes, along with the syntax of common URI components that are used by many URI schemes to define hierarchical identifiers. [RFC3987] extended this generic syntax to cover IRIs. All scheme specifications **MUST** define their own URI <scheme-specific-part> syntax. Care must be taken to ensure that all strings matching their scheme-specific syntax will also match the <absolute-URI> grammar described in [RFC3986].

New schemes **SHOULD** reuse the common URI components of [RFC3986] for the definition of hierarchical naming schemes. If there is a strong reason for a scheme not to use the hierarchical syntax, then the new scheme definition **SHOULD** follow the syntax of similar previously registered schemes.

Schemes that are not intended for use with relative URIs **SHOULD** avoid use of the forward slash "/" character, in order to avoid unintended processing such as resolution of "." and ".." (dot-segments).

Schemes **SHOULD** avoid improper use of "//". The use of double slashes in the first part of a URI is not a stylistic indicator that what follows is a URI: Double slashes are intended for use **ONLY** when the syntax of the <scheme-specific-part> contains a hierarchical structure. In URIs from such schemes, the use of double slashes indicates that what follows is the top hierarchical element for a naming authority. (Section 3.2 of RFC 3986 has more details.) Schemes that do not contain a conformant hierarchical structure in their <scheme-specific-part> **SHOULD NOT** use double slashes following the "<scheme>:" string.

New schemes **SHOULD** clearly define the role of reserved characters (see [RFC3986], Section 2.2) in URIs of the scheme being defined. The syntax of the new scheme should be clear about which of the "reserved" set of characters are used as delimiters within the URIs of the new scheme, and when those characters must be escaped, versus when they can be used without escaping.

### 3.3. Well-Defined

While URIs might or might not be defined as locators in practice, a scheme definition itself **MUST** be clear as to how it is expected to function. Schemes that are not intended to be used as locators **SHOULD** describe how the resource identified can be determined or accessed by software that obtains a URI of that scheme.

For schemes that function as locators, it is important that the mechanism of resource location be clearly defined. This might mean different things depending on the nature of the scheme.

In many cases, new schemes are defined as ways to translate between other namespaces or protocols and the general framework of URIs. For example, the "ftp" scheme translates into the FTP protocol, while the "mid" scheme translates into a Message-ID identifier of an email message. For such schemes, the description of the mapping SHOULD be complete, and in sufficient detail so that the mapping in both directions is clear: how to map from a URI into an identifier or set of protocol actions or name in the target namespace, and how legal values in the base namespace, or legal protocol interactions, are represented in a valid URI. See Section 3.6 for guidelines for encoding strings or sequences of bytes within valid character sequences in a URI. If not all legal values or protocol interactions of the base standard can be represented using the scheme, the definition SHOULD be clear about which subset is allowed, and why.

### 3.4. Definition of Operations

As part of the definition of how a URI identifies a resource, a scheme definition SHOULD define the applicable set of operations that can be performed on a resource using the URI as its identifier. A model for this is HTTP methods; an HTTP resource can be operated on by GET, POST, PUT, and a number of other methods available through the HTTP protocol. The scheme definition SHOULD describe all well-defined operations on the resource identifier, and what they are supposed to do.

Some schemes don't fit into the "information access" paradigm of URIs. For example, "telnet" provides location information for initiating a bi-directional data stream to a remote host; the only operation defined is to initiate the connection. In any case, the operations appropriate for a scheme SHOULD be documented.

Note: It is perfectly valid to say that "no operation apart from GET is defined for this URI". It is also valid to say that "there's only one operation defined for this URI, and it's not very GET-like". The important point is that what is defined on this scheme is described.

Scheme definitions SHOULD define a "default" operation for when a URI is invoked (or "dereferenced") by an application. For example, a common "default" operation today is to launch an application associated with the scheme name, and let it use the other URI components as inputs to do something. The default invocation, or dereferencing, of a URI SHOULD be "safe" in the sense described by

section 3.4 of [W3CWebArch]; i.e., performing such an invocation should not incur any additional obligations by doing so.

### 3.5. Context of Use

In general, URIs are used within a broad range of protocols and applications. For example, URIs are commonly used within hypertext documents as references to other resources. In some cases, a scheme is intended for use within a different, specific set of protocols or applications. If so, the scheme definition SHOULD describe the intended use and include references to documentation that define the applications and/or protocols cited. This does not obviate the need for documentation on applications and/or protocols to discuss URI schemes relevant to them.

### 3.6. Internationalization and Character Encoding

When describing schemes in which (some of) the elements of the URI are actually representations of human-readable text, care should be taken not to introduce unnecessary variety in the ways in which characters are encoded into octets and then into URI characters; see [RFC3987] and Section 2.5 (especially the last paragraph) of [RFC3986] for guidelines. If URIs of a scheme contain any text fields, the scheme definition MUST describe the ways in which characters are encoded and any compatibility issues with IRIs of the scheme.

The scheme specification SHOULD be as restrictive as possible regarding what characters are allowed in the URI, because some characters can create several different security issues (see, for example [RFC4690]).

Percent-encoded character sequences are automatically included by definition for characters given in IRI productions. This means that if you want to restrict the URI percent-encoded forms in some way, you must restrict the Unicode forms that would lead to them. In most cases, it is advisable to define the actual characters allowed in an IRI production, to allow the 'pct-encoded' definition from Section 2.1 of [RFC3986] at the same places, and to add prose that limits percent-escapes to those that can be created by converting valid UTF-8 character sequences to percent-encoding.

### 3.7. Clear Security and Privacy Considerations

Definitions of schemes MUST be accompanied by a clear analysis of the security and privacy implications for systems that use the scheme; this follows the practice of Security Consideration sections within IANA registrations [RFC5226].

In particular, Section 7 of RFC 3986 [RFC3986] describes general security considerations for URIs, while [RFC3987] gives those for IRIs. The definition of an individual scheme should note which of these apply to the specified scheme, in addition to any more scheme-specific concerns. For example, if the scheme-specific part is privacy sensitive then that should be documented.

### 3.8. Scheme Name Considerations

Section 3.1 of RFC 3986 defines the syntax of a URI scheme name; this syntax remains the same for IRIs. New scheme registrations MUST follow this syntax, which only allows a limited repertoire of characters (taken from US-ASCII). Although the syntax for the scheme name in URIs is case insensitive, the scheme name itself MUST be registered using lowercase letters.

Scheme names SHOULD be short, but also sufficiently descriptive and distinguished to avoid problems.

Schemes SHOULD NOT use names or other symbols that might cause problems with rights to use the name in IETF specifications and Internet protocols. For example, be careful with trademark and service mark names. (See Section 7.4 of [RFC3978].)

Schemes SHOULD NOT use names that are either very general purpose or associated in the community with some other application or protocol. Schemes also SHOULD NOT use names that are overly general or grandiose in scope (e.g., that allude to their "universal" or "standard" nature.)

A scheme name is not a "protocol." However, like a service name as defined in section 5 of [RFC6335], it often identifies a particular protocol or application. If a scheme name has a one-to-one correspondence with a service name, then the names SHOULD be the same.

Some organizations desire their own namespace for URI scheme names for private use (see Section 6). In doing so, it is important to prevent collisions, and to make it possible to identify the owner of a private use scheme. To accomplish these two goals, such organizations SHOULD use a prefix based on their domain name, expressed in reverse order. For example, a URI scheme name of com.example.mything might be used by the organization that owns the example.com domain name. Care must be taken, however, if the organization later loses the domain name embedded in their scheme names, since domain name registrations are not permanent. To associate the private use scheme name with the original organization,

the private use scheme can be registered using the registration procedure in Section 7.

Furthermore, to prevent collisions with private use scheme names, new scheme names registered MUST NOT contain a "." unless actually constructed from a reversed domain name.

### 3.9. Interoperability Considerations

If the person or group registering the scheme is aware of any details regarding the scheme that might impact interoperability, identify them. For example: proprietary or uncommon encoding methods; incompatibility with types or versions of any underlying protocol.

## 4. Guidelines for Provisional URI Scheme Registration

'Provisional' registration can be used for schemes that are not part of any standard, but that are intended for use (or observed to be in use) that is not limited to a private environment within a single organization. 'Provisional' registration can also be used as an intermediate step on the way to 'permanent' registration, e.g., before the scheme specification is finalized as a standard.

For a 'provisional' registration, the following apply:

- o The scheme name must meet the syntactic requirements of Section 3.8.
- o There must not already be an entry with the same scheme name. In the unfortunate case that there are multiple, different uses of the same scheme name, the Designated Expert can approve a request to modify an existing entry to note the separate use.
- o Contact information identifying the person supplying the registration must be included. Previously unregistered schemes discovered in use can be registered by third parties (even if not on behalf of those who created the scheme). In this case, both the registering party and the scheme creator SHOULD be identified.
- o If no permanent, citable specification for the scheme definition is included, credible reasons for not providing it SHOULD be given.
- o The scheme definition SHOULD include clear security considerations (Section 3.7) or explain why a full security analysis is not available (e.g., in a third-party scheme registration).

- o If the scheme definition does not meet the guidelines laid out in Section 3, the differences and reasons SHOULD be noted.

## 5. Guidelines for Historical URI Scheme Registration

In some circumstances, it is appropriate to note a scheme that was once in use or registered but for whatever reason is no longer in common use or whose use is not recommended. In this case, it is possible for an individual to request that the URI scheme be registered (newly, or as an update to an existing registration) as 'historical'. Any scheme that is no longer in common use MAY be designated as 'historical'; the registration SHOULD contain some indication as to where the scheme was previously defined or documented.

## 6. Guidelines for Private URI Scheme Use

Unregistered schemes can cause problems if use is not limited to a private environment within a single organization, since the use could leak out beyond the closed environment. Even within a closed environment, other colliding uses of the same scheme name could occur. As such, a unique namespace (see Section 3.8) MUST be used, and it is strongly encouraged to do a 'provisional' registration unless the scheme name is constructed from a domain name as discussed in Section 3.8.

## 7. URI Scheme Registration Procedure

### 7.1. General

The IANA policy (using terms defined in [RFC5226]) for 'provisional' registration was formerly Expert Review. The policy for 'permanent' and 'historical' registration continues to be Expert Review, but this document changes the policy for 'provisional' registration to First Come First Served.

The registration procedure is intended to be very lightweight for non-contentious registrations. For the most part, we expect the good sense of submitters and reviewers, guided by these procedures, to achieve an acceptable and useful consensus for the community.

In exceptional cases, where the negotiating parties cannot form a consensus, the final arbiter of any contested registration shall be the IESG.

If standardization is anticipated, the working group or individuals concerned are advised to submit an early 'permanent' registration request, rather than waiting until the standardization process has

run its course. IANA will pass this to the Designated Expert who may recommend 'provisional' registration until the specification is approved as a standard. This will provide opportunity for feedback while specification development and review is still active, and the submitter(s) are still in a position to respond to any issues that might be raised. If and when the specification is approved as a standard, the submitters should submit a request to change the registration status to 'permanent'.

The role of the Designated Expert in the procedure for 'permanent' registrations described here is to ensure that the normal open review process has been properly followed, and to raise possible concerns about wider implications of proposals for the use and deployment of URIs. Nothing in the procedure empowers the Designated Expert to override properly arrived-at IETF or working group consensus.

## 7.2. Registration Procedures

Someone wishing to register a new scheme MUST:

1. Check the IANA URI Schemes registry to see whether there is already an entry for the desired name. If there is already an entry under the name, choose a different scheme name, or update the existing scheme specification.
2. Prepare a scheme registration request using the template specified in Section 7.4. The scheme registration request can be contained in an Internet Draft, submitted alone, or as part of some other permanently available, stable, protocol specification. The scheme registration request can also be submitted in some other form (as part of another document or as a stand-alone document), but the scheme registration request will be treated as an "IETF Contribution" under the guidelines of [RFC3978].
3. If the registration request is for a 'permanent' registration (or, optionally, for any other registration if desired):
  1. Review the requirements in Section 3.
  2. Send a copy of the scheme registration request or a pointer to the containing document (with specific reference to the section with the scheme registration request) to the mailing list [uri-review@ietf.org](mailto:uri-review@ietf.org), requesting review. In addition, request review on other relevant mailing lists as appropriate. For example, general discussion of URI syntactical issues can be discussed on [uri@w3.org](mailto:uri@w3.org); schemes for a network protocol can be discussed on a mailing list for

that protocol. Allow a reasonable time for discussion and comments. Four weeks is reasonable for a 'permanent' registration request.

3. Respond to review comments and make revisions to the proposed registration as needed to bring it into line with the guidelines given in this document.
4. Submit the (possibly updated) scheme registration request (or pointer to document containing it) to IANA at [iana@iana.org](mailto:iana@iana.org).

Upon receipt of a scheme registration request, the following steps MUST be followed:

1. IANA checks the submission for completeness; if required sections of the scheme registration request are missing or any citations are not correct, IANA will reject the registration request. A registrant can resubmit a corrected request if desired.
2. If the request is for 'provisional' registration and no entry already exists in the current registry for the same name, IANA adds the registration to the registry, under the First Come First Served policy.
3. Otherwise, IANA enters the registration request in the IANA registry, with status marked as "Pending Review", and the remainder of this section applies.
4. IANA requests Expert Review of the registration request against the corresponding guidelines from this document.
5. The Designated Expert will evaluate the request against the criteria of the requested status.
6. In the case of a 'permanent' registration request, the Designated Expert may:
  - \* Accept the specification of the scheme for 'permanent' registration.
  - \* Suggest 'provisional' registration instead.
  - \* Request IETF review and IESG approval; in the meanwhile, suggest 'provisional' registration.
  - \* Request additional review or discussion, as necessary.

7. If an entry already exists for the same name, the Designated Expert will determine whether the request should be rejected, or whether the existing entry should be modified to note the separate use. This conflict process applies regardless of the requested status or the status of the existing entry.
8. Once Expert Review approves registration for a given status, IANA updates the registration to indicate the approved status. If Expert Review instead rejects the registration, the "Pending Review" request is removed from the registry.

Either based on an explicit request or independently initiated, the Designated Expert or the IESG can request the upgrade of a 'provisional' registration to a 'permanent' one. In such cases, IANA will update the status of the corresponding entry. Typically this would only occur if the use is considered a standard (not necessarily an IETF standard).

### 7.3. Change Control

Registrations can be updated in the registry by the same mechanism as required for an initial registration. In cases where the original definition of the scheme is contained in an IESG-approved document, update of the specification also requires IESG approval.

'Provisional' registrations can be updated by the original registrant or anyone designated by the original registrant. In addition, the IESG can reassign responsibility for a 'provisional' registration scheme, or can request specific changes to a scheme registration. This will enable changes to be made to schemes where the original registrant is out of contact, or unwilling or unable to make changes.

Transition from 'provisional' to 'permanent' status can be requested and approved in the same manner as a new 'permanent' registration. Transition from 'permanent' to 'historical' status requires IESG approval. Transition from 'provisional' to 'historical' can be requested by anyone authorized to update the 'provisional' registration.

### 7.4. URI Scheme Registration Template

This template describes the fields that **MUST** be supplied in a scheme registration request, suitable for adding to the registry:

Scheme name:

See Section 3.8 for guidelines.

Status:

This reflects the status requested, and must be one of 'permanent', 'provisional', or 'historical'.

Applications/protocols that use this scheme name:  
See Section 3.5.

Contact:

Person (including contact information) to contact for further information.

Change controller:

Organization or person (often the author), including contact information, authorized to change this.

References:

Include full citations for all referenced documents. Scheme registration requests for 'provisional' registration can be included in an Internet Draft; when the documents expire or are approved for publication as an RFC, the registration will be updated. A scheme specification is only required for 'permanent' registration.

The previous version of this specification required the following additional fields in a scheme registration request. These fields are no longer part of the template. The answers instead belong in the scheme specification.

Scheme syntax:

See Section 3.2 for guidelines.

Scheme semantics:

See Section 3.3 and Section 3.4 for guidelines.

Encoding considerations:

See Section 3.3 and Section 3.6 for guidelines.

Interoperability considerations:

See Section 3.9 for guidelines.

Security considerations:

See Section 3.7 for guidelines.

## 8. The "example" URI Scheme

There is a need for a scheme name that can be used for examples in documentation without fear of conflicts with current or future actual schemes. The scheme "example" is hereby registered as a 'permanent' scheme for that purpose.

The "example" scheme is specified as follows:

**Scheme syntax:** The entire range of allowable syntax specified in [RFC3986] is allowed for "example" URIs. Similarly, the entire range of allowable syntax specified in [RFC3987] is allowed for "example" IRIs. For example, <example:foo>, <example:/foo>, and <example://foo> are all valid.

**Scheme semantics:** URIs in the "example" scheme are to be used for documentation purposes only. The use of "example" URIs must not be used as locators, identify any resources, or specify any particular set of operations.

**Encoding considerations:** See Section 2.5 of [RFC3986] for guidelines.

**Interoperability considerations:** None.

**Security considerations:** None.

#### 8.1. "Example" URI Scheme Registration Request

**Scheme name:** example

**Status:** permanent

**Applications/protocols that use this scheme name:** An "example" URI is to be used for documentation purposes only. It MUST NOT be used for any protocol.

**Contact:** N/A

**Change controller:** IETF

**References:** Section 8 of this RFC XXXX.

RFC Editor Note: Replace XXXX with this RFC's reference.

#### 9. IANA Considerations

Previously, the former "URL Scheme" registry was replaced by the "Uniform Resource Identifier (URI) Schemes" registry. The process was based on [RFC5226] "Expert Review" with an initial (optional) mailing list review.

The updated template has an additional field for the status of the scheme, and the procedures for entering new name schemes have been augmented. Section 7 establishes the process for new scheme registration.

IANA is requested to do the following:

- o Update the URI Schemes registry to point to this document.
- o Combine the "Permanent URI Schemes", "Provisional URI Schemes", and "Historical URI Schemes" sub-registries into a single common registry with an additional "Status" column containing the status (Permanent, Provisional, Historical, or Pending Review), and an additional "Notes" column which is normally empty, but may contain notes approved by the Designated Expert.
- o Add the "example" URI scheme to the registry (see the template in Section 8.1 for registration).

## 10. Security Considerations

All registered values are expected to contain clear security considerations as discussed in Section 3.7. However, information concerning possible security vulnerabilities of a protocol might change over time. Consequently, claims as to the security properties of a registered scheme might change as well. As new vulnerabilities are discovered, information about such vulnerabilities might need to be attached to existing documentation, so that users are not misled as to the true security properties of a registered scheme.

## 11. Acknowledgements

Thanks to Mark Nottingham and Graham Klyne and other members of the apps-discuss@ietf.org mailing list for their comments on this document.

Many thanks to Patrik Faltstrom, Paul Hoffmann, Ira McDonald, Roy Fielding, Stu Weibel, Tony Hammond, Charles Lindsey, Mark Baker, and other members of the uri@w3.org mailing list for their comments on earlier versions.

Parts of this document are based on [RFC2717], [RFC2718] and [RFC3864]. Some of the ideas about use of URIs were taken from the "Architecture of the World Wide Web" [W3CWebArch].

## 12. References

### 12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997.

- [RFC3978] Bradner, S., "IETF Rights in Contributions", RFC 3978, March 2005.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, August 2011.

## 12.2. Informative References

- [RFC2717] Petke, R. and I. King, "Registration Procedures for URL Scheme Names", BCP 35, RFC 2717, November 1999.
- [RFC2718] Masinter, L., Alvestrand, H., Zigmond, D., and R. Petke, "Guidelines for new URL Schemes", RFC 2718, November 1999.
- [RFC3406] Daigle, L., van Gulik, D., Iannella, R., and P. Faltstrom, "Uniform Resource Names (URN) Namespace Definition Mechanisms", BCP 66, RFC 3406, October 2002.
- [RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, September 2004.
- [RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.
- [RFC4395] Hansen, T., Hardie, T., and L. Masinter, "Guidelines and Registration Procedures for New URI Schemes", BCP 35, RFC 4395, February 2006.
- [RFC4690] Klensin, J., Faltstrom, P., Karp, C., and IAB, "Review and Recommendations for Internationalized Domain Names (IDNs)", RFC 4690, September 2006.
- [W3CWebArch] W3C Technical Architecture Group, "Architecture of the World Wide Web, Volume One", December 2004, <<http://www.w3.org/TR/webarch/>>.

## Appendix A. Changes Since RFC 4395

1. Combined the Historical, Permanent, and Provisional URI Schemes registries into one registry with a status column. This is done to make it easier to prevent duplicates and see existing conventions.
2. Added a Notes column in the registry for notes approved by the Designated Expert.
3. Moved the following fields out of the scheme registration request template and into the requirements for a scheme specification: Scheme syntax, Scheme semantics, Encoding considerations, Interoperability considerations, and Security considerations.
4. Simplified the process for 'provisional' registration significantly: changed from Expert Review to First Come First Served, and clarified that mailing list review is not required.
5. Updated process for handling of scheme name conflicts, so that adding a note can be approved by the Designated Expert rather than the IESG.
6. Clarified that a "URI scheme name" and an "IRI scheme name" are the same thing and thus use the same IANA registry.
7. Clarified that a registration request falls under the "IETF Contribution" rules, but the scheme's specification need not.
8. Added the "example:" URI scheme.
9. Added text about when to use 'provisional' registration.
10. Updated convention for Private use schemes to use "." (instead of "-") between domain name labels, to reduce chance of collision, and recommended use of a reverse domain name prefix to allow identifying the owning organization.
11. Recommended that scheme definitions define a "default" operation for when a URI is invoked.
12. Recommended that a scheme name be the same as the service name, when there exists a 1:1 correspondence.
13. Elaborated on when a 'provisional' request should be upgraded to Permanent.

14. Clarified that since, per RFC 3986, fragment identifiers are not scheme-specific and hence are not part of the scheme definition.
15. Added text about getting early review and registration for schemes intended to become standards.
16. Added text about privacy as an example of what should be covered in security considerations in a scheme definition.
17. Clarified the role of a Designated Expert in reviewing Standards-Track registrations.

#### Authors' Addresses

Dave Thaler (editor)  
Microsoft  
One Microsoft Way  
Redmond, WA 98052  
US

Phone: +1 425 703 8835  
Email: dthaler@microsoft.com

Tony Hansen  
AT&T Laboratories  
200 Laurel Ave.  
Middletown, NJ 07748  
USA

Email: tony+uriereg@maillennium.att.com

Ted Hardie  
Google

Phone: +1 408 628 5864  
Email: ted.ietf@gmail.com

Larry Masinter  
Adobe  
345 Park Ave.  
San Jose, CA 95110  
US

Phone: +1 408 536 3024  
Email: masinter@adobe.com  
URI: <http://larry.masinter.net>

Individual submission  
Internet-Draft  
Updates: 7001 (if approved)  
Intended status: Standards Track  
Expires: December 19, 2014

M. Kucherawy  
June 17, 2014

A Property Types Registry for the Authentication-Results Header Field  
draft-kucherawy-authres-ptypes-registry-00

Abstract

[RFC7001] describes a header field called Authentication-Results for use with electronic mail messages to indicate the results of message authentication efforts. Any receiver-side software, such as mail filters or Mail User Agents (MUAs), can use this header field to relay that information in a convenient and meaningful way to users, or make sorting and filtering decisions.

One portion of the definition in that document limits the types of authentication properties about a message to a small, fixed set. This document updates the specification to allow new property types to be declared and used.

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 <http://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 December 19, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |   |
|---|---|
| 1. Introduction . . . . .               | 3 |
| 2. Updated 'ptype' Definition . . . . . | 3 |
| 3. IANA Considerations . . . . .        | 4 |
| 4. Security Considerations . . . . .    | 4 |
| 5. Normative References . . . . .       | 5 |
| Appendix A. Acknowledgements . . . . .  | 5 |

## 1. Introduction

[RFC7001] describes a header field called Authentication-Results for electronic mail messages that presents the results of a message authentication effort in a machine-readable format. The intent of the header field is to create a place to collect such data when message authentication mechanisms are in use so that a Mail User Agent (MUA) and downstream filters can make filtering decisions and/or provide a recommendation to the user as to the validity of the message's origin and possibly the safety and integrity of its content.

The specification in that document enumerated a small set of types of properties that can be reported using this mechanism. There has emerged a desire to report types of properties about a message through this mechanism. Accordingly, this document updates the specification to allow for additional property types ("ptypes") beyond the original set, and creates a registry where new ones can be listed and their defining documents referenced.

## 2. Updated 'ptype' Definition

Advanced Backus Naur Form (ABNF) is defined in [RFC5234].

The ABNF in Section 2.2 of [RFC7001] is updated as follows:

```
ptype = Keyword
      ; indicates whether the property being evaluated was
      ; a parameter to an [SMTP] command, was a value taken
      ; from a message header field, was some property of
      ; the message body, or was some other property evaluated by
      ; the receiving MTA
```

The ABNF token "Keyword" is defined in Section 4.1.2 of [RFC5321].

Legal values of "ptype" are as defined in this document, or in the IANA "Email Authentication Property Types" registry (see Section 3). The initial values are as follows, matching those defined in [RFC7001]:

body: Indicates information that was extracted from the body of the message. This might be an arbitrary string of bytes, a hash of a string of bytes, a Uniform Resource Identifier, or some other content of interest.

header: Indicates information that was extracted from the header of the message. This might be the value of a header field or some portion of a header field.

policy: As defined in Section 2.3 of [RFC7001].

smtp: Indicates information that was extracted from an SMTP command that was used to relay the message.

A consumer of this header field encountering a "ptype" it does not understand simply ignores the result it is reporting.

### 3. IANA Considerations

IANA is requested to create the Email Authentication Property Types registry. Entries in this registry are subject to the Expert Review rules as described in [RFC5226]. Each entry in the registry requires the following values:

- o The "ptype" token to be registered, which must fit within the ABNF described in Section 2.
- o A brief description of what sort of information this "ptype" is meant to cover.
- o A reference to the defining document, if any.

The initial entries in this table are enumerated in Section 2. This document should be listed as their defining document values.

For new entries, the Designated Expert simply needs to assure that the description provided for the new entry adequately describes the intended use. An example would be helpful to include, although entries in the Email Authentication Methods registry or the Email Authentication Result Names registry might also serve as examples of intended use.

### 4. Security Considerations

A consumer of this header field might be confused by a result bearing a "ptype" it does not understand. The advice is simply to ignore such a result since its semantics are unknown to such a consumer. It is unknown how legacy code, which expects one of a fixed set of "ptype" tokens, will handle new tokens as they begin to appear. This could conceivably result in undesirable deliveries for consumers that have been implemented to "fail open".

## 5. Normative References

- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.
- [RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008.
- [RFC7001] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7001, September 2013.

## Appendix A. Acknowledgements

The author wishes to acknowledge the following for their review and constructive criticism of this update: (names)

### Author's Address

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

EMail: [superuser@gmail.com](mailto:superuser@gmail.com)



Network Working Group  
Internet-Draft  
Updates: 7001 (if approved)  
Intended status: Standards Track  
Expires: June 22, 2015

F. Martin, Ed.  
LinkedIn  
December 19, 2014

Authentication-Results Registration for TLS  
draft-martin-authentication-results-tls-03

Abstract

This memo updates the registry of properties in Authentication-Results: message header fields to allow relaying of the results of an email sent using STARTTLS [RFC3207] or not.

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 <http://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 June 22, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |   |
|---|---|
| 1. Introduction . . . . .                             | 2 |
| 1.1. Requirements Language . . . . .                  | 2 |
| 1.2. Discussion . . . . .                             | 2 |
| 2. Definitions . . . . .                              | 3 |
| 2.1. results meaning . . . . .                        | 3 |
| 2.2. properties . . . . .                             | 4 |
| 3. IANA Considerations . . . . .                      | 5 |
| 4. Security Considerations . . . . .                  | 7 |
| 5. References . . . . .                               | 7 |
| 5.1. Normative References . . . . .                   | 7 |
| 5.2. Informative References . . . . .                 | 8 |
| Appendix A. Authentication-Results Examples . . . . . | 8 |
| A.1. TLS Results . . . . .                            | 8 |
| Author's Address . . . . .                            | 9 |

## 1. Introduction

STARTTLS [RFC3207] defines how to send an email over an SMTP [RFC5321] encrypted session between two mail servers.

This memo thus registers an additional reporting property allowing a TLS result to be relayed as an annotation in a message header.

## 1.1. Requirements Language

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 RFC 2119 [RFC2119].

## 1.2. Discussion

STARTTLS [RFC3207] defines how to send an email over an encrypted session between two mail servers, Message Transfer Agent (MTA), using the TLS [RFC5246] protocol.

Most of these exchanges are opportunistic, meaning a best effort is done to establish an encrypted message exchange regardless of the strength of the cipher or the validity of the certificates used. However, the results of this negotiation should be recorded in the message via the Authentication-Results header [RFC7001] to indicate to other message processing algorithms, including Messaging User Agents (MUA), how securely this message was transmitted from the MTA client to the MTA server.

The concept of authentication here is related to the presentation of a certificate which is verified valid by a set of trusted Certificate

Authorithies (CA), via DANE [RFC6698] or by local policy. This does not indicate that any string in the certificate is related to any string in the email.

The usage and usefulness of the Authentication-Results header is discussed in [RFC7001].

## 2. Definitions

This memo adds to the "Email Authentication Methods" registry, created by IANA upon publication of [RFC7001], the following:

- o The method "tls"; and
- o Associated with that method, the properties (reporting items) "cert.client", "cert.server", "cert.verif", "tls.v", "key.ciphersuite", "key.fingerprint", "key.length" and "key.strength".

### 2.1. results meaning

The "tls" method can have the following results:

none: the message was sent in clear.

pass: the message was sent encrypted and the client certificate was verified valid either using a trusted CA, via DANE [RFC6698] or via a local policy and host identity was verified.

selfsigned: the message was sent encrypted but the client certificate is self signed.

invalidhost: the message was sent encrypted and the client certificate is verified valid but the host identity is invalid.

fail: the message was sent encrypted but the client certificate is not valid. It is advised to use comments to indicate the nature of the problem like certificate expired, not linked to a trusted CA,...

temperror: the message was sent encrypted and the server was not able to verify the client certificate at this time. This may indicate for instance that the server could not fetch the CRL.

permerror: the message was sent encrypted and the client certificate was not verified by the MTA server. MTA should always attempt to verify the client certificate.

## 2.2. properties

cert.client: the subject of the X.509 certificate used by the client to initiate TLS.

cert.server: the subject of the X.509 certificate used by the server to initiate TLS (optional).

cert.clientalt: the subject alternative name of the X.509 certificate used by the client to initiate TLS (optional).

cert.serveralt: the subject alternative name of the X.509 certificate used by the server to initiate TLS (optional).

cert.clientissuer: the issuer of the X.509 certificate used by the client to initiate TLS (optional).

cert.serverissuer: the issuer of the X.509 certificate used by the server to initiate TLS (optional).

cert.verif: the type of certification performed: CA, DANE [RFC6698], LOCAL (optional).

tls.v: the protocol version used to encrypt SSL2.0, SSL3.0, TLS1.0, TLS1.1,... (optional)

key.ciphersuite: the description of the TLS cipher suite used as defined in the IANA cipher suite registry.

key.fingerprint: the fingerprint of the key used (optional).

key.length: the length in bits of the key used (optional).

key.strength: as many SMTP TLS are opportunistic in nature this property is an arbitrary value set by the MTA server to indicate the strength of the encryption (optional).

While ciphers strength vary overtime, and key length in bits does not indicate a comparable strength between various cyphers, it may be difficult for all the processors of the authentication-results header to redo the analysis based on the cipher used and all to arrive to the same conclusion. It seems, therefore, best if the receiving MTA does that analysis and communicate it to the other layers. This is the purpose of the key.strength. For instance This value could be used by the MUA to indicate to the end user some quality of the encryption channel.

### 3. IANA Considerations

Per [IANA], the following items have been added to the "Email Authentication Methods" registry:

| Method | Defined  | ptype | property     | value   |
|--------|----------|-------|--------------|---|
| tls    | RFC 3207 | cert  | client       | subject of<br>client<br>certificate<br>section 4.1.2.6<br>of RFC 5280           |
| tls    | RFC 3207 | cert  | server       | subject of<br>server<br>certificate<br>section 4.1.2.6<br>of RFC 5280           |
| tls    | RFC 3207 | cert  | clientalt    | alternate subject<br>of client<br>certificate<br>section 4.2.1.6<br>of RFC 5280 |
| tls    | RFC 3207 | cert  | serveralt    | alternate subject<br>of server<br>certificate<br>section 4.2.1.6<br>of RFC 5280 |
| tls    | RFC 3207 | cert  | clientissuer | issuer of<br>client<br>certificate<br>section 4.1.2.4<br>of RFC 5280            |
| tls    | RFC 3207 | cert  | serverissuer | issuer of<br>server<br>certificate<br>section 4.1.2.4<br>of RFC 5280            |
| tls    | RFC 3207 | cert  | verif        | CA<br>DANE<br>LOCAL   |

|     |          |     |             |   |
|-----|----------|-----|-------------|---|
| tls | RFC 3207 | tls | v           | protocol<br>version<br>description<br>from RFC 5246           |
| tls | RFC 3207 | key | ciphersuite | IANA cipher<br>suite registry<br>description<br>from RFC 5246 |
| tls | RFC 3207 | key | fingerprint | key<br>fingerprint<br>from RFC 5246                           |
| tls | RFC 3207 | key | length      | in bits   |
| tls | RFC 3207 | key | strength    | low<br>medium<br>high   |

Also, the following items have been added to the "Email Authentication Result Names" registry:

| Code        | Existing/New | Defined In | Method         | Meaning   |
|-------------|--------------|------------|----------------|-----------|
| none        | existing     | RFC 7001   | tls<br>(added) | this memo |
| pass        | existing     | RFC 7001   | tls<br>(added) | this memo |
| selfsigned  | existing     | RFC 7001   | tls<br>(added) | this memo |
| invalidhost | existing     | RFC 7001   | tls<br>(added) | this memo |
| fail        | existing     | RFC 7001   | tls<br>(added) | this memo |
| temperror   | existing     | RFC 7001   | tls<br>(added) | this memo |
| permerror   | existing     | RFC 7001   | tls<br>(added) | this memo |

#### 4. Security Considerations

This memo creates a mechanism for relaying STARTTLS [RFC3207] results using the structure already defined by [RFC7001]. The Security Considerations sections of those documents should be consulted.

By this mechanism, some identifiers of the client certificates gets to live pass the receiving MTA. This is a change in the sender expectation on where the client certificate is used

#### 5. References

##### 5.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3207] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, February 2002.
- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.
- [RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008.
- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.
- [RFC7001] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7001, September 2013.

## 5.2. Informative References

- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

## Appendix A. Authentication-Results Examples

This section presents an example of the use of this new header field to indicate TLS results.

### A.1. TLS Results

A message that went over a successful TLS session:

```
Authentication-Results: mail-router.example.net;
  dkim=pass (good signature) header.d=newyork.example.com
  header.b=oINEO8hg;
  tls=pass (verified, expires 20140505)
  cert.verif=CA
  cert.client="CN=smtp.example.com,O=ACME,L=ToonTown,
    ST=CA,C=US"
  cert.clientalt="DNS:smtp.example.com, DNS:newyork.example.com"
  cert.clientissuer="C=US, O=AcmeCert Inc, CN=AcmeCert CA"
  key.ciphersuite=TLS_RSA_WITH_RC4_128_SHA
  tls.v=TLS1.0
  key.fingerprint="68:B3:29:DA:98:93:E3:40:99:C7:D8:
    AD:5C:B9:C9:40"
  key.length=128
  key.strength=MEDIUM;
Received: from newyork.example.com
  (newyork.example.com [192.0.2.250])
  by mail-router.example.net (8.11.6/8.11.6)
  for <recipient@example.net>
  with ESMTPS id i7PK0sH7021929;
  Fri, Feb 15 2002 17:19:22 -0800
DKIM-Signature: v=1; a=rsa-sha256; s=rashani;
  d=newyork.example.com;
  t=1188964191; c=relaxed/simple;
  h=From:Date:To:VBR-Info:Message-Id:Subject;
  bh=sEu28nfs9fuZGD/pSr7ANysbY3jtdaQ3Xv9xPQtS0m7=;
  b=oINEO8hgn/gnunsg ... 9n9ODSNFSDij3=
From: sender@newyork.example.com
Date: Fri, Feb 15 2002 16:54:30 -0800
To: meetings@example.net
Message-Id: <12345.abc@newyork.example.com>
Subject: here's a sample
```

Author's Address

Franck Martin (editor)  
LinkedIn  
Mountain View, CA  
US  
  
Email: [fmartin@linkedin.com](mailto:fmartin@linkedin.com)

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: January 29, 2015

M. Nottingham  
Akamai  
E. Wilde  
UC Berkeley  
July 28, 2014

Problem Details for HTTP APIs  
draft-nottingham-http-problem-07

Abstract

This document defines a "problem detail" as a way to carry machine-readable details of errors in a HTTP response, to avoid the need to invent new error response formats for HTTP APIs.

Note to Readers

This draft should be discussed on the apps-discuss mailing list [1].

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 <http://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 January 29, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|  |    |
|--|----|
| 1. Introduction . . . . .                                      | 2  |
| 2. Requirements . . . . .                                      | 3  |
| 3. The Problem Details JSON Object . . . . .                   | 3  |
| 3.1. Problem Details Object Members . . . . .                  | 4  |
| 3.2. Extension Members . . . . .                               | 5  |
| 4. Defining New Problem Types . . . . .                        | 5  |
| 4.1. Example . . . . .   | 6  |
| 4.2. Pre-Defined Problem Types . . . . .                       | 7  |
| 5. Security Considerations . . . . .                           | 7  |
| 6. IANA Considerations . . . . .                               | 8  |
| 7. Acknowledgements . . . . .                                  | 9  |
| 8. References . . . . .  | 9  |
| 8.1. Normative References . . . . .                            | 9  |
| 8.2. Informative References . . . . .                          | 10 |
| Appendix A. HTTP Problems and XML . . . . .                    | 10 |
| Appendix B. Using Problem Details with Other Formats . . . . . | 12 |
| Authors' Addresses . . . . .                                   | 12 |

## 1. Introduction

HTTP [RFC2616] status codes are sometimes not sufficient to convey enough information about an error to be helpful. While humans behind Web browsers can be informed about the nature of the problem with an HTML [W3C.REC-html401-19991224] response body, non-human consumers of so-called "HTTP APIs" are usually not.

This specification defines simple JSON [RFC4627] and XML [W3C.REC-xml-20081126] document formats to suit this purpose. They are designed to be reused by HTTP APIs, which can identify distinct "problem types" specific to their needs.

Thus, API clients can be informed of both the high-level error class (using the status code) and the finer-grained details of the problem (using one of these formats).

For example, consider a response that indicates that the client's account doesn't have enough credit. The 403 Forbidden status code might be deemed most appropriate to use, as it will inform HTTP-generic software (such as client libraries, caches and proxies) of the general semantics of the response.

However, that doesn't give the API client enough information about why the request was forbidden, the applicable account balance, or how to correct the problem. If these details are included in the response body in a machine-readable format, the client can treat it appropriately; for example, triggering a transfer of more credit into the account.

This specification does this by identifying a specific type of problem (e.g., "out of credit") with a URI [RFC3986]; HTTP APIs can do this by nominating new URIs under their control, or by reusing existing ones.

Additionally, problems can contain other information, such as a URI that identifies the specific occurrence of the problem (effectively giving an identifier to the concept "The time Joe didn't have enough credit last Thursday"), which may be useful for support or forensic purposes.

The data model for problem details is a JSON [RFC4627] object; when formatted as a JSON document, it uses the "application/problem+json" media type. Appendix A defines how to express them in an equivalent XML format, which uses the "application/problem+xml" media type.

Note that problem details are (naturally) not the only way to convey the details of a problem in HTTP; if the response is still a representation of a resource, for example, it's often preferable to accommodate describing the relevant details in that application's format. Likewise, in many situations, there is an appropriate HTTP status code that does not require extra detail to be conveyed.

Instead, the aim of this specification is to define common error formats for those applications that need one, so that they aren't required to define their own, or worse, tempted to re-define the semantics of existing HTTP status codes.

## 2. Requirements

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].

## 3. The Problem Details JSON Object

The canonical model for problem details is a JSON [RFC4627] object.

When serialised as a JSON document, that format is identified with the "application/problem+json" media type.

For example, a HTTP response carrying JSON problem details:

```
HTTP/1.1 403 Forbidden
Content-Type: application/problem+json
Content-Language: en

{
  "type": "http://example.com/probs/out-of-credit",
  "title": "You do not have enough credit.",
  "detail": "Your current balance is 30, but that costs 50.",
  "instance": "http://example.net/account/12345/msgs/abc",
  "balance": 30,
  "accounts": ["http://example.net/account/12345",
               "http://example.net/account/67890"]
}
```

Here, the out-of-credit problem (identified by its type URI) indicates the reason for the 403 in "title", gives a reference for the specific problem occurrence with "instance", gives occurrence-specific details in "detail", and adds two extensions; "balance" conveys the account's balance, and "accounts" gives links where the account can be topped up.

### 3.1. Problem Details Object Members

A problem details object MAY have the following members:

- o "type" (string) - An absolute URI [RFC3986] that identifies the problem type. When dereferenced, it SHOULD provide human-readable documentation for the problem type (e.g., using HTML [W3C.REC-html401-19991224]). When this member is not present, its value is assumed to be "about:blank".
- o "title" (string) - A short, human-readable summary of the problem type. It SHOULD NOT change from occurrence to occurrence of the problem, except for purposes of localisation.
- o "status" (number) - The HTTP status code ([RFC2616], Section 6) generated by the origin server for this occurrence of the problem.
- o "detail" (string) - An human readable explanation specific to this occurrence of the problem.
- o "instance" (string) - An absolute URI that identifies the specific occurrence of the problem. It may or may not yield further information if dereferenced.

Consumers MUST use the type string as the primary identifier for the problem type; the title string is advisory, and included only for users who are not aware of the semantics of the URI, and don't have the ability to discover them (e.g., offline log analysis). Consumers SHOULD NOT automatically dereference the type URI.

The status member, if present, is only advisory; it conveys the HTTP status code used for the convenience of the consumer. Generators MUST use the same status code in the actual HTTP response, to assure that generic HTTP software that does not understand this format still behaves correctly. See Section 5 for further caveats regarding its use.

The detail member, if present, SHOULD focus on helping the client correct the problem, rather than giving debugging information.

Consumers SHOULD NOT parse the detail member for information; extensions are more suitable and less error-prone ways to obtain such information.

### 3.2. Extension Members

Problem type definitions MAY extend the problem details object with additional members.

For example, our "out of credit" problem above defines two such extensions, "balance" and "accounts" to convey additional, problem-specific information.

Clients consuming problem details MUST ignore any such extensions that they don't recognise; this allows problem types to evolve and include additional information in the future.

## 4. Defining New Problem Types

When an HTTP API needs to define a response that indicates an error condition, it might be appropriate to do so by defining a new problem type.

Before doing so, it's important to understand what they are good for, and what's better left to other mechanisms.

Problem details are not a debugging tool for the underlying implementation; rather, they are a way to expose greater detail about the HTTP interface itself. New problem types need to carefully consider the Security Considerations (Section 5); in particular the risk of exposing attack vectors by exposing implementation internals through error messages.

Likewise, truly generic problems - i.e., conditions that could potentially apply to any resource on the Web - are usually better expressed as plain status codes. For example, a "write access disallowed" problem is probably unnecessary, since a 403 Forbidden status code in response to a PUT request is self-explanatory.

Finally, an application may have a more appropriate way to carry an error in a format that it already defines. Problem details are intended to avoid the necessity of establishing new "fault" or "error" document formats, not to replace existing domain-specific formats.

That said, it is possible to add support for problem details to existing HTTP APIs using HTTP content negotiation (e.g., using the Accept request header to indicate a preference for this format).

New problem type definitions MUST document:

1. A type URI (typically, with the "http" scheme),
2. A title that appropriately describes it (think short), and
3. The HTTP status code for it to be used with.

Problem types MAY specify the use of the Retry-After response header in appropriate circumstances.

A problem's type URI SHOULD resolve to HTML [W3C.REC-html401-19991224] documentation that explains how to resolve the problem.

A problem type definition MAY specify additional members on the Problem Details object. For example, an extension might use typed links [RFC5988] to another resource that can be used by machines to resolve the problem.

If such additional members are defined, their names SHOULD start with a letter (ALPHA, as per [RFC5234]) and SHOULD consist of characters from ALPHA, DIGIT, and "\_" (so that it can be serialized in formats other than JSON), and SHOULD be three characters or longer.

#### 4.1. Example

For example, if you are publishing an HTTP API to your online shopping cart, you might need to indicate that the user is out of credit (our example from above), and therefore cannot make the purchase.

If you already have an application-specific format that can accommodate this information, it's probably best to do that. However, if you don't, you might consider using one of the problem details formats; JSON if your API is JSON-based, or XML if it uses that format.

To do so, you might look for an already-defined type URI that suits your purposes. If one is available, you can reuse that URI.

If one isn't available, you could mint and document a new type URI (which ought to be under your control and stable over time), an appropriate title and the HTTP status code that it will be used with, along with what it means and how it should be handled.

In summary: an instance URI will always identify a specific occurrence of a problem. On the other hand, type URIs can be reused if an appropriate description of a problem type is already available someplace else, or they can be created for new problem types.

#### 4.2. Pre-Defined Problem Types

This specification reserves the use of one URI as a problem type:

The "about:blank" URI [RFC6694], when used as a problem type, indicates that the problem has no additional semantics beyond that of the HTTP status code.

When "about:blank" is used, the title SHOULD be the same as the recommended HTTP status phrase for that code (e.g., "Not Found" for 404, and so on), although it MAY be localized to suit client preferences (expressed with the Accept-Language request header).

Please note that according to how the "type" member is defined (Section 3.1), the "about:blank" URI is the default value for that member. Consequently, any problem details object not carrying an explicit "type" member implicitly uses this URI.

#### 5. Security Considerations

When defining a new problem type, the information included must be carefully vetted. Likewise, when actually generating a problem - however it is serialized - the details given must also be scrutinized.

Risks include leaking information that can be exploited to compromise the system, access to the system, or the privacy of users of the system.

Generators providing links to occurrence information are encouraged to avoid making implementation details such as a stack dump available through the HTTP interface, since this can expose sensitive details of the server implementation, its data, and so on.

The "status" member duplicates the information available in the HTTP status code itself, thereby bringing the possibility of disagreement between the two. Their relative precedence is not clear, since a disagreement might indicate that (for example) an intermediary has modified the HTTP status code in transit. As such, those defining problem types as well as generators and consumers of problems need to be aware that generic software (such as proxies, load balancers, firewalls, virus scanners) are unlikely to know of or respect the status code conveyed in this member.

## 6. IANA Considerations

This specification defines two new Internet media types [RFC6838]:

```
Type name: application
Subtype name: problem+json
Required parameters: None
Optional parameters: None; unrecognised parameters
                      should be ignored
Encoding considerations: Same as [RFC4627]
Security considerations: see [this document]
Interoperability considerations: None.
Published specification: [this document]
Applications that use this media type: HTTP
Additional information:
    Magic number(s): n/a
    File extension(s): n/a
    Macintosh file type code(s): n/a
Person & email address to contact for further information:
    Mark Nottingham <mnot@mnot.net>
Intended usage: COMMON
Restrictions on usage: None.
Author: Mark Nottingham <mnot@mnot.net>
Change controller: IESG
```

Type name: application  
Subtype name: problem+xml  
Required parameters: None  
Optional parameters: None; unrecognized parameters  
should be ignored  
Encoding considerations: Same as [RFC3023]  
Security considerations: see [this document]  
Interoperability considerations: None.  
Published specification: [this document]  
Applications that use this media type: HTTP  
Additional information:  
    Magic number(s): n/a  
    File extension(s): n/a  
    Macintosh file type code(s): n/a  
Person & email address to contact for further information:  
    Mark Nottingham <mnot@mnot.net>  
Intended usage: COMMON  
Restrictions on usage: None.  
Author: Mark Nottingham <mnot@mnot.net>  
Change controller: IESG

## 7. Acknowledgements

The authors would like to thank Jan Algermissen, Mike Amundsen, Subbu Allamaraju, Roy Fielding, Eran Hammer, Sam Johnston, Mike McCall, Julian Reschke, and James Snell for review of this specification.

## 8. References

### 8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.
- [RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.
- [RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

## 8.2. Informative References

- [ISO-19757-2]  
International Organization for Standardization,  
"Information Technology --- Document Schema Definition  
Languages (DSDL) --- Part 2: Grammar-based Validation ---  
RELAX NG", ISO/IEC 19757-2, 2003.
- [RFC3023] Murata, M., St. Laurent, S., and D. Kohn, "XML Media  
Types", RFC 3023, January 2001.
- [RFC5988] Nottingham, M., "Web Linking", RFC 5988, October 2010.
- [RFC6694] Moonesamy, S., "The "about" URI Scheme", RFC 6694, August  
2012.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type  
Specifications and Registration Procedures", BCP 13, RFC  
6838, January 2013.
- [W3C.REC-html401-19991224]  
Raggett, D., Hors, A., and I. Jacobs, "HTML 4.01  
Specification", World Wide Web Consortium Recommendation  
REC-html401-19991224, December 1999,  
<<http://www.w3.org/TR/1999/REC-html401-19991224>>.
- [W3C.REC-rdfa-core-20120607]  
Adida, B., Birbeck, M., McCarron, S., and I. Herman, "RDFa  
Core 1.1", World Wide Web Consortium Recommendation REC-  
rdfa-core-20120607, June 2012,  
<<http://www.w3.org/TR/2012/REC-rdfa-core-20120607>>.
- [W3C.REC-xml-20081126]  
Bray, T., Paoli, J., Sperberg-McQueen, M., Maler, E., and  
F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth  
Edition)", World Wide Web Consortium Recommendation REC-  
xml-20081126, November 2008,  
<<http://www.w3.org/TR/2008/REC-xml-20081126>>.

## Appendix A. HTTP Problems and XML

Some HTTP-based APIs use XML [W3C.REC-xml-20081126] as their primary format convention. Such APIs MAY express problem details using the format defined in this appendix.

The OPTIONAL RELAX NG schema [ISO-19757-2] for the XML format is:

```

default namespace ns = "urn:ietf:rfc:XXXX"

start = problem

problem =
  element problem {
    ( element type           { xsd:anyURI }?
      & element title         { xsd:string }?
      & element detail         { xsd:string }?
      & element status         { xsd:positiveInteger }?
      & element instance       { xsd:anyURI }? ),
    anyNsElement
  }

anyNsElement =
  ( element ns:* { anyNsElement | text }
    | attribute * { text })*

```

The media type for this format is "application/problem+xml".

Extension arrays and objects can be serialized into the XML format by considering an element containing a child or children to represent an object, except for elements that contain only child element(s) named 'i', which are considered arrays. For example, an alternate version of the example above would appear in XML as:

```

HTTP/1.1 403 Forbidden
Content-Type: application/problem+xml
Content-Language: en

```

```

<?xml version="1.0" encoding="UTF-8"?>
<problem xmlns="urn:ietf:rfc:XXXX">
  <type>http://example.com/probs/out-of-credit</type>
  <title>You do not have enough credit.</title>
  <detail>Your current balance is 30, but that costs 50.</detail>
  <instance>
    http://example.net/account/12345/msgs/abc
  </instance>
  <balance>30</balance>
  <accounts>
    <i>http://example.net/account/12345</i>
    <i>http://example.net/account/67890</i>
  </accounts>
</problem>

```

Note that this format uses an XML Namespace. This is primarily to allow embedding it into other XML-based formats; it does not imply that it can or should be extended with elements or attributes in

other namespaces. The RELAX NG schema explicitly only allows elements from the one namespace used in the XML format. Any extension arrays and objects MUST be serialized into XML markup using only that namespace.

#### Appendix B. Using Problem Details with Other Formats

In some situations, it can be advantageous to embed Problem Details in formats other than those described here. For example, an API that uses HTML [W3C.REC-html401-19991224] might want to also use HTML for expressing its problem details.

Problem details can be embedded in other formats by either encapsulating one of the existing serializations (JSON or XML) into that format, or by translating the model of a Problem Detail (as specified in Section 3) into the format's conventions.

For example, in HTML, a problem could be embedded by encapsulating JSON in a script tag:

```
<script type="application/problem+json">
  {
    "type": "http://example.com/probs/out-of-credit",
    "title": "You do not have enough credit.",
    "detail": "Your current balance is 30, but that costs 50.",
    "instance": "http://example.net/account/12345/messages/abc",
    "balance": 30,
    "accounts": ["http://example.net/account/12345",
                 "http://example.net/account/67890"]
  }
</script>
}
```

or by inventing a mapping into RDFa [W3C.REC-rdfa-core-20120607].

This specification does not make specific recommendations regarding embedding Problem Details in other formats; the appropriate way to embed them depends both upon the format in use and application of that format.

#### Authors' Addresses

Mark Nottingham  
Akamai

Email: [mnot@mnot.net](mailto:mnot@mnot.net)  
URI: <http://www.mnot.net/>

Erik Wilde  
UC Berkeley

Email: [dret@berkeley.edu](mailto:dret@berkeley.edu)  
URI: <http://dret.net/netdret/>

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: August 19, 2017

M. Nottingham  
February 15, 2017

Home Documents for HTTP APIs  
draft-nottingham-json-home-06

Abstract

This document proposes a "home document" format for non-browser HTTP clients.

Note to Readers

The issues list for this draft can be found at  
<https://github.com/mnot/I-D/labels/json-home> .

The most recent (often, unpublished) draft is at  
<https://mnot.github.io/I-D/json-home/> .

Recent changes are listed at <https://github.com/mnot/I-D/commits/gh-pages/json-home> .

For information about implementations, see <https://github.com/mnot/I-D/wiki/json-home> .

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 <http://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 August 19, 2017.

## Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|   |    |
|---|----|
| 1. Introduction . . . . .                                   | 3  |
| 1.1. Notational Conventions . . . . .                       | 4  |
| 2. API Home Documents . . . . .                             | 4  |
| 3. API Objects . . . . .                                    | 7  |
| 4. Resource Objects . . . . .                               | 7  |
| 4.1. Resolving Templated Links . . . . .                    | 8  |
| 5. Resource Hints . . . . .                                 | 8  |
| 5.1. allow . . . . .  | 9  |
| 5.2. formats . . . . .                                      | 9  |
| 5.3. acceptPatch . . . . .                                  | 9  |
| 5.4. acceptPost . . . . .                                   | 10 |
| 5.5. acceptPut . . . . .                                    | 10 |
| 5.6. acceptRanges . . . . .                                 | 10 |
| 5.7. acceptPrefer . . . . .                                 | 11 |
| 5.8. docs . . . . .   | 11 |
| 5.9. preconditionRequired . . . . .                         | 11 |
| 5.10. authSchemes . . . . .                                 | 11 |
| 5.11. status . . . . .                                      | 12 |
| 6. Security Considerations . . . . .                        | 12 |
| 7. IANA Considerations . . . . .                            | 12 |
| 7.1. HTTP Resource Hint Registry . . . . .                  | 12 |
| 7.2. Media Type Registration . . . . .                      | 13 |
| 8. References . . . . .                                     | 13 |
| 8.1. Normative References . . . . .                         | 13 |
| 8.2. Informative References . . . . .                       | 14 |
| Appendix A. Acknowledgements . . . . .                      | 15 |
| Appendix B. Creating and Serving Home Documents . . . . .   | 15 |
| B.1. Managing Change in Home Documents . . . . .            | 15 |
| B.2. Evolving and Mixing APIs with Home Documents . . . . . | 16 |
| Appendix C. Consuming Home Documents . . . . .              | 16 |
| Appendix D. Frequently Asked Questions . . . . .            | 17 |

|  |    |
|--|----|
| D.1. Why not use (insert other service description format)?  | 17 |
| D.2. Why doesn't the format allow references or inheritance? | 17 |
| D.3. What about "Faults" (i.e., errors)?                     | 17 |
| D.4. How Do I find the schema for a format?                  | 17 |
| D.5. How do I express complex query arguments?               | 17 |
| Author's Address   | 18 |

## 1. Introduction

It is becoming increasingly common to use HTTP [RFC7230] for applications other than traditional Web browsing. Such "HTTP APIs" are used to integrate processes on disparate systems, make information available to machines across the Internet, and as part of the implementation of "micro-services."

By using HTTP, these applications realise a number of benefits, from message framing to caching, and well-defined semantics that are broadly understood and useful.

Often, these applications of HTTP are defined by documenting static URLs that clients need to know and servers need to implement. Any interaction outside of these bounds is uncharted territory.

For some applications, this approach brings issues, especially when the interface changes, either due to evolution, extension or drift between implementations. Furthermore, implementing more than one instance of interface can bring further issues, as different environments have different requirements.

The Web itself offers one way to address these issues, using links [RFC3986] to navigate between states. A link-driven application discovers relevant resources at run time, using a shared vocabulary of link relations [RFC5988] and internet media types [RFC6838] to support a "follow your nose" style of interaction - just as a Web browser does to navigate the Web.

A client can then decide which resources to interact with "on the fly" based upon its capabilities (as described by link relations), and the server can safely add new resources and formats without disturbing clients that are not yet aware of them.

Doing so can provide any of a number of benefits, including:

- o Extensibility - Because new server capabilities can be expressed as link relations, new features can be layered in without introducing a new API version; clients will discover them in the home document. This promotes loose coupling between clients and servers.

- o Evolvability - Likewise, interfaces can change gradually by introducing a new link relation and/or format while still supporting the old ones.
- o Customisation - Home documents can be tailored for the client, allowing different classes of service or different client permissions to be exposed naturally.
- o Flexible deployment - Since URLs aren't baked into documentation, the server can choose what URLs to use for a given service.
- o API mixing - Likewise, more than one API can be deployed on a given server, without fear of collisions.

Whether an application ought to use links in this fashion depends on how it is deployed; generally, the most benefit will be received when multiple instances of the service are deployed, possibly with different versions, and they are consumed by clients with different capabilities. In particular, Internet Standards that use HTTP as a substrate are likely to require the attributes described above.

This document defines a "home page" format using the JSON format [RFC7159] for APIs to use as a launching point for the interactions they offer, using links. Having a well-defined format for this purpose promotes good practice and tooling.

### 1.1. Notational Conventions

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. API Home Documents

An API Home Document (or, interchangeably, "home document") uses the format described in [RFC7159] and has the media type "application/json-home".

\*Note: this media type is not final, and will change before final publication.\*

Its content consists of a root object with:

- o A "resources" member, whose value is an object that describes the resources associated with the API. Its member names are link relation types (as defined by [RFC5988]), and their values are Resource Objects (Section 4).

- o Optionally, a "api" member, whose value is an API Object (Section 3) that contains information about the API as a whole.

For example:

```
GET / HTTP/1.1
Host: example.org
Accept: application/json-home

HTTP/1.1 200 OK
Content-Type: application/json-home
Cache-Control: max-age=3600
Connection: close

{
  "api": {
    "title": "Example API",
    "links": {
      "author": "mailto:api-admin@example.com",
      "describedBy": "https://example.com/api-docs/"
    }
  },
  "resources": {
    "tag:me@example.com,2016:widgets": {
      "href": "/widgets/"
    },
    "tag:me@example.com,2016:widget": {
      "hrefTemplate": "/widgets/{widget_id}",
      "hrefVars": {
        "widget_id": "https://example.org/param/widget"
      },
      "hints": {
        "allow": ["GET", "PUT", "DELETE", "PATCH"],
        "formats": {
          "application/json": {}
        },
        "acceptPatch": ["application/json-patch+json"],
        "acceptRanges": ["bytes"]
      }
    }
  }
}
```

Here, we have a home document for the API "Example API", whose author can be contacted at the e-mail address "api-admin@example.com", and whose documentation is at "https://example.com/api-docs/".

It links to a resource `"/widgets/"` with the relation `"tag:me@example.com,2016:widgets"`. It also links to an unknown number of resources with the relation type `"tag:me@example.com,2016:widget"` using a URI Template [RFC6570], along with a mapping of identifiers to a variable for use in that template.

It also gives several hints about interacting with the latter `"widget"` resources, including the HTTP methods usable with them, the PATCH and POST formats they accept, and the fact that they support partial requests [RFC7233] using the `"bytes"` range-specifier.

It gives no such hints about the `"widgets"` resource. This does not mean that it (for example) doesn't support any HTTP methods; it means that the client will need to discover this by interacting with the resource, and/or examining the documentation for its link relation type.

Effectively, this names a set of behaviors, as described by a resource object, with a link relation type. This means that several link relations might apply to a common base URL; e.g.:

```
{
  "resources": {
    "tag:me@example.com,2016:search-by-id": {
      "hrefTemplate": "/search?id={widget_id}",
      "hrefVars": {
        "widget_id": "https://example.org/param/widget_id"
      }
    },
    "tag:me@example.com,2016:search-by-name": {
      "hrefTemplate": "/search?name={widget_name}",
      "hrefVars": {
        "widget_name": "https://example.org/param/widget_name"
      }
    }
  }
}
```

Note that the examples above use both tag [RFC4151] and https [RFC7230] URIs; any URI scheme can be used to identify link relations and other artefacts in home documents. Typically, these are not links to be followed; they are only used to identify things.

### 3. API Objects

An API Object contains links to information about the API itself.

Two optional members are defined:

- o "title" has a string value indicating the name of the API;
- o "links" has an object value, whose member names are link relation types [RFC5988], and values are URLs [RFC3986]. The context of these links is the API home document as a whole.

No links are required to be conveyed, but APIs might benefit from setting the following:

- o author - a suitable URL (e.g., mailto: or https:) for the author(s) of the API
- o describedBy - a link to documentation for the API
- o license - a link to the legal terms for using the API

Future members of the API Object MAY be defined by specifications that update this document.

### 4. Resource Objects

A Resource Object links to resources of the type indicated in their name using one of two mechanisms; either a direct link (in which case there is exactly one resource of that relation type associated with the API), or a templated link, in which case there are zero to many such resources.

Direct links are indicated with an "href" property, whose value is a URI [RFC3986].

Templated links are indicated with an "hrefTemplate" property, whose value is a URI Template [RFC6570]. When "hrefTemplate" is present, the Resource Object MUST have a "hrefVars" property; see "Resolving Templated Links".

Resource Objects MUST have exactly one of the "href" or "href-vars" properties.

In both forms, the links that "href" and "hrefTemplate" refer to are URI-references [RFC3986] whose base URI is that of the API Home Document itself.

Resource Objects MAY also have a "hints" property, whose value is an object that uses named Resource Hints (see Section 5) as its properties.

#### 4.1. Resolving Templated Links

A URI can be derived from a Templated Link by treating the "hrefTemplate" value as a Level 3 URI Template [RFC6570], using the "hrefVars" property to fill the template.

The "hrefVars" property, in turn, is an object that acts as a mapping between variable names available to the template and absolute URIs that are used as global identifiers for the semantics and syntax of those variables.

For example, given the following Resource Object:

```
"https://example.org/rel/widget": {
  "hrefTemplate": "/widgets/{widget_id}",
  "hrefVars": {
    "widget_id": "https://example.org/param/widget"
  },
  "hints": {
    "allow": ["GET", "PUT", "DELETE", "PATCH"],
    "formats": {
      "application/json": {}
    },
    "acceptPatch": ["application/json-patch+json"],
    "acceptRanges": ["bytes"]
  }
}
```

If you understand that "https://example.org/param/widget" is a numeric identifier for a widget, you can then find the resource corresponding to widget number 12345 at "https://example.org/widgets/12345" (assuming that the Home Document is located at "https://example.org/").

#### 5. Resource Hints

Resource hints allow clients to find relevant information about interacting with a resource beforehand, as a means of optimizing communications, as well as advertising available behaviors (e.g., to aid in laying out a user interface for consuming the API).

Hints are just that - they are not a "contract", and are to only be taken as advisory. The runtime behavior of the resource always overrides hinted information.

For example, a resource might hint that the PUT method is allowed on all "widget" resources. This means that generally, the user has the ability to PUT to a particular resource, but a specific resource might reject a PUT based upon access control or other considerations. More fine-grained information might be gathered by interacting with the resource (e.g., via a GET), or by another resource "containing" it (such as a "widgets" collection) or describing it (e.g., one linked to it with a "describedBy" link relation).

This specification defines a set of common hints, based upon information that's discoverable by directly interacting with resources. See Section 7.1 for information on defining new hints.

#### 5.1. allow

- o Resource Hint Name: allow
- o Description: Hints the HTTP methods that the current client will be able to use to interact with the resource; equivalent to the Allow HTTP response header.
- o Specification: [this document]

Content MUST be an array of strings, containing HTTP methods. As per HTTP, when GET is supported, a client MAY assume that HEAD is supported.

#### 5.2. formats

- o Resource Hint Name: formats
- o Description: Hints the representation types that the resource makes available, using the GET method.
- o Specification: [this document]

Content MUST be an object, whose keys are media types, and values are objects, currently empty.

#### 5.3. acceptPatch

- o Resource Hint Name: accept-Patch
- o Description: Hints the PATCH [RFC5789] request formats accepted by the resource for this client; equivalent to the Accept-Patch HTTP response header.
- o Specification: [this document]

Content MUST be an array of strings, containing media types.

When this hint is present, "PATCH" SHOULD be listed in the "allow" hint.

#### 5.4. acceptPost

- o Resource Hint Name: acceptPost
- o Description: Hints the POST request formats accepted by the resource for this client.
- o Specification: [this document]

Content MUST be an array of strings, containing media types.

When this hint is present, "POST" SHOULD be listed in the "allow" hint.

#### 5.5. acceptPut

- o Resource Hint Name: acceptPut
- o Description: Hints the PUT request formats accepted by the resource for this client.
- o Specification: [this document]

Content MUST be an array of strings, containing media types.

When this hint is present, "PUT" SHOULD be listed in the "allow" hint.

#### 5.6. acceptRanges

- o Resource Hint Name: acceptRanges
- o Description: Hints the range-specifiers available to the client for this resource; equivalent to the Accept-Ranges HTTP response header [RFC7233].
- o Specification: [this document]

Content MUST be an array of strings, containing HTTP range-specifiers (typically, "bytes").

## 5.7. acceptPrefer

- o Resource Hint Name: acceptPrefer
- o Description: Hints the preferences [RFC7240] supported by the resource. Note that, as per that specifications, a preference can be ignored by the server.
- o Specification: [this document]

Content MUST be an array of strings, containing preferences.

## 5.8. docs

- o Resource Hint Name: docs
- o Description: Hints the location for human-readable documentation for the relation type of the resource.
- o Specification: [this document]

Content MUST be a string containing an absolute-URI [RFC3986] referring to documentation that SHOULD be in HTML format.

## 5.9. preconditionRequired

- o Resource Hint Name: preconditionRequired
- o Description: Hints that the resource requires state-changing requests (e.g., PUT, PATCH) to include a precondition, as per [RFC7232], to avoid conflicts due to concurrent updates.
- o Specification: [this document]

Content MUST be an array of strings, with possible values "etag" and "last-modified" indicating type of precondition expected.

## 5.10. authSchemes

- o Resource Hint Name: authSchemes
- o Description: Hints that the resource requires authentication using the HTTP Authentication Framework [RFC7235].
- o Specification: [this document]

Content MUST be an array of objects, each with a "scheme" property containing a string that corresponds to a HTTP authentication scheme,

and optionally a "realms" property containing an array of zero to many strings that identify protection spaces that the resource is a member of.

For example, a Resource Object might contain the following hint:

```
{
  "authSchemes": [
    {
      "scheme": "Basic",
      "realms": ["private"]
    }
  ]
}
```

#### 5.11. status

- o Resource Hint Name: status
- o Description: Hints the status of the resource.
- o Specification: [this document]

Content MUST be a string; possible values are:

- o "deprecated" - indicates that use of the resource is not recommended, but it is still available.
- o "gone" - indicates that the resource is no longer available; i.e., it will return a 404 (Not Found) or 410 (Gone) HTTP status code if accessed.

### 6. Security Considerations

Clients need to exercise care when using hints. For example, a naive client might send credentials to a server that uses the auth-req hint, without checking to see if those credentials are appropriate for that server.

### 7. IANA Considerations

#### 7.1. HTTP Resource Hint Registry

This specification defines the HTTP Resource Hint Registry. See Section 5 for a general description of the function of resource hints.

In particular, resource hints are generic; that is, they are potentially applicable to any resource, not specific to one application of HTTP, nor to one particular format. Generally, they ought to be information that would otherwise be discoverable by interacting with the resource.

Hint names **MUST** be composed of the lowercase letters (a-z), digits (0-9), underscores ("\_") and hyphens ("-"), and **MUST** begin with a lowercase letter.

Hint content **SHOULD** be described in terms of JSON [RFC7159] constructs.

New hints are registered using the Expert Review process described in [RFC5226] to enforce the criteria above. Requests for registration of new resource hints are to use the following template:

- o Resource Hint Name: [hint name]
- o Description: [a short description of the hint's semantics]
- o Specification: [reference to specification document]

Initial registrations are enumerated in Section 5.

## 7.2. Media Type Registration

TBD

## 8. References

### 8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<http://www.rfc-editor.org/info/rfc3986>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.

- [RFC5988] Nottingham, M., "Web Linking", RFC 5988, DOI 10.17487/RFC5988, October 2010, <<http://www.rfc-editor.org/info/rfc5988>>.
- [RFC6570] Gregorio, J., Fielding, R., Hadley, M., Nottingham, M., and D. Orchard, "URI Template", RFC 6570, DOI 10.17487/RFC6570, March 2012, <<http://www.rfc-editor.org/info/rfc6570>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <<http://www.rfc-editor.org/info/rfc7159>>.
- [RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <<http://www.rfc-editor.org/info/rfc7234>>.

## 8.2. Informative References

- [RFC4151] Kindberg, T. and S. Hawke, "The 'tag' URI Scheme", RFC 4151, DOI 10.17487/RFC4151, October 2005, <<http://www.rfc-editor.org/info/rfc4151>>.
- [RFC5789] Dusseault, L. and J. Snell, "PATCH Method for HTTP", RFC 5789, DOI 10.17487/RFC5789, March 2010, <<http://www.rfc-editor.org/info/rfc5789>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<http://www.rfc-editor.org/info/rfc6838>>.
- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <<http://www.rfc-editor.org/info/rfc7230>>.
- [RFC7232] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests", RFC 7232, DOI 10.17487/RFC7232, June 2014, <<http://www.rfc-editor.org/info/rfc7232>>.
- [RFC7233] Fielding, R., Ed., Lafon, Y., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Range Requests", RFC 7233, DOI 10.17487/RFC7233, June 2014, <<http://www.rfc-editor.org/info/rfc7233>>.

- [RFC7235] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Authentication", RFC 7235, DOI 10.17487/RFC7235, June 2014, <<http://www.rfc-editor.org/info/rfc7235>>.
- [RFC7240] Snell, J., "Prefer Header for HTTP", RFC 7240, DOI 10.17487/RFC7240, June 2014, <<http://www.rfc-editor.org/info/rfc7240>>.
- [RFC7807] Nottingham, M. and E. Wilde, "Problem Details for HTTP APIs", RFC 7807, DOI 10.17487/RFC7807, March 2016, <<http://www.rfc-editor.org/info/rfc7807>>.

#### Appendix A. Acknowledgements

Thanks to Jan Algermissen, Mike Amundsen, Bill Burke, Sven Dietze, Graham Klyne, Leif Hedstrom, Joe Hildebrand, Jeni Tennison, Erik Wilde and Jorge Williams for their suggestions and feedback.

#### Appendix B. Creating and Serving Home Documents

When making an API home document available, there are a few things to keep in mind:

- o A home document is best located at a memorable URI, because its URI will effectively become the URI for the API itself to clients.
- o Home documents can be personalized, just as "normal" home pages can. For example, you might advertise different URIs, and/or different kinds of link relations, depending on the client's identity.
- o Home documents ought to be assigned a freshness lifetime (e.g., "Cache-Control: max-age=3600") so that clients can cache them, to avoid having to fetch it every time the client interacts with the service.
- o Custom link relation types, as well as the URIs for variables, should lead to documentation for those constructs.

##### B.1. Managing Change in Home Documents

The URIs used in API home documents MAY change over time. However, changing them can cause issues for clients that are relying on cached home documents containing old links.

To mitigate the impact of such changes, servers ought to consider:

- o Reducing the freshness lifetime of home documents before a link change, so that clients are less likely to refer to an "old" document.
- o Regarding the "old" and "new" URIs as equally valid references for an "overlap" period.
- o After that period, handling requests for the "old" URIs appropriately; e.g., with a 404 Not Found, or by redirecting the client to the new URI.

## B.2. Evolving and Mixing APIs with Home Documents

Using home documents affords the opportunity to change the "shape" of the API over time, without breaking old clients.

This includes introducing new functions alongside the old ones - by adding new link relation types with corresponding resource objects - as well as adding new template variables, media types, and so on.

It's important to realise that a home document can serve more than one "API" at a time; by listing all relevant relation types, it can effectively "mix" different APIs, allowing clients to work with different resources as they see fit.

## Appendix C. Consuming Home Documents

Clients might use home documents in a variety of ways.

In the most common case - actually consuming the API - the client will scan the Resources Object for the link relation(s) that it is interested in, and then to interact with the resource(s) referred to. Resource Hints can be used to optimize communication with the client, as well as to inform as to the permissible actions (e.g., whether PUT is likely to be supported).

Note that the home document is a "living" document; it does not represent a "contract", but rather is expected to be inspected before each interaction. In particular, links from the home document MUST NOT be assumed to be valid beyond the freshness lifetime of the home document, as per HTTP's caching model [RFC7234].

As a result, clients ought to cache the home document (as per [RFC7234]), to avoid fetching it before every interaction (which would otherwise be required).

Likewise, a client encountering a 404 (Not Found) on a link is encouraged obtain a fresh copy of the home document, to assure that it is up-to-date.

#### Appendix D. Frequently Asked Questions

##### D.1. Why not use (insert other service description format)?

There are a fair number of existing service description formats, including those that specialise in "RESTful" interactions. However, these formats generally are optimised for pairwise integration, or one-server-to-many-client integration, and less capable of describing protocols where both the server and client can evolve and be extended.

##### D.2. Why doesn't the format allow references or inheritance?

Adding inheritance or references would allow more modularity in the format and make it more compact, at the cost of considerable complexity and the associated potential for errors (both in the specification and by its users).

Since good tools and compression are effective ways to achieve the same ends, this specification doesn't attempt them.

##### D.3. What about "Faults" (i.e., errors)?

In HTTP, errors are conveyed by HTTP status codes. While this specification could (and even may) allow enumeration of possible error conditions, there's a concern that this will encourage applications to define many such "faults", leading to tight coupling between the application and its clients. See [RFC7807] for further considerations.

##### D.4. How Do I find the schema for a format?

That isn't addressed by home documents. Ultimately, it's up to the media type accepted and generated by resources to define and constrain (or not) their syntax.

##### D.5. How do I express complex query arguments?

Complex queries - i.e., those that exceed the expressive power of Link Templates or would require ambiguous properties of a "resources" object - aren't intended to be defined by a home document. The appropriate way to do this is with a "form" language, much as HTML defines.

Note that it is possible to support multiple query syntaxes on the same base URL, using more than one link relation type; see the example at the start of the document.

Author's Address

Mark Nottingham

Email: [mnot@mnot.net](mailto:mnot@mnot.net)

URI: <https://www.mnot.net/>

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: February 3, 2020

M. Nottingham  
August 2, 2019

The "safe" HTTP Preference  
draft-nottingham-safe-hint-11

Abstract

This specification defines a "safe" preference for HTTP requests that expresses a desire to avoid objectionable content, according to the definition of that term by the origin server.

This specification does not define a precise semantic for "safe". Rather, the term is interpreted by the server and within the scope of each Web site that chooses to act upon this information.

Support for this preference by clients and servers is optional.

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 February 3, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|  |   |
|--|---|
| 1. Introduction . . . . .                              | 2 |
| 2. The "safe" Preference . . . . .                     | 4 |
| 3. Implementation Status . . . . .                     | 5 |
| 4. Security Considerations . . . . .                   | 6 |
| 5. IANA Considerations . . . . .                       | 6 |
| 6. References . . . . .                                | 7 |
| Appendix A. Acknowledgements . . . . .                 | 8 |
| Appendix B. Sending "safe" from Web Browsers . . . . . | 8 |
| Appendix C. Supporting "safe" on Web Sites . . . . .   | 8 |
| Author's Address . . . . .                             | 9 |

## 1. Introduction

Many Web sites have a "safe" mode, to assist those who don't want to be exposed (or have their children exposed) to content to which they might object.

However, that goal is often difficult to achieve, because of the need to go to every Web site that might be used, navigate to the appropriate page (possibly creating an account along the way) to get a cookie [RFC6265] set in the browser, for each browser on every device used.

A more manageable approach is for the browser to proactively indicate a preference for safe content. A user agent that supports doing so (whether it be an individual browser, or through an Operating System HTTP library) need only be configured once to assure that the preference is advertised to a set of sites, or even all sites.

This specification defines how to declare this desire in requests as a HTTP Preference [RFC7240].

Note that this specification does not define what content might be considered objectionable, and so the concept of "safe" is also not precisely defined. Rather, the term is interpreted by the server and within the scope of each Web site that chooses to act upon this information.

That said, the intent of "safe" is to allow end users (or those acting on their behalf) to express a desire to avoid content that is considered objectionable within the cultural context of that site;

usually (but not always) content that is unsuitable for minors. The "safe" preference is not intended to be used for other purposes.

Furthermore, sending "safe" does not guarantee that the Web site will use it, nor that it will apply a concept of "objectionable" that is consistent with the requester's views. As such, its effect can be described as "best effort," and not to be relied upon. In other words, sending the preference is no more reliable than going to each Web site and manually selecting a "safe" mode, but it is considerably easier.

It is also important to note that the "safe" preference is not a reliable indicator that the end user is a child; other users might have a desire for unobjectionable content, and some children might browse without the preference being set.

Note also that the cultural context applies to the hosting location of a site, the content provider, and the source of the content. It cannot be guaranteed that a user-agent and origin server will have the same view of the concept of what is objectionable.

Simply put, it is a statement by (or on behalf of) the end user to the effect "If your site has a 'safe' setting, this user is hereby opting into that, according to your definition of the term."

The mechanism described in this document does not have IETF consensus and is not a standard. It is a widely deployed approach that has turned out to be useful, and is presented here so that server and browser implementations can have a common understanding of how it operates.

This mechanism was presented for publication as an IETF Proposed Standard, but was not approved for publication by the IESG because of concerns that included the vagueness of the meaning of "safe", the ability of a proxy to insert the hint outside of a user's control, the fact that there was no way to know whether the hint was or was not applied to the response returned by the server, and how the use of this preference may incentivize increased censorship and/or targeting of minors.

The specification has been updated to address those concerns, but the IESG has not approved progressing this document as an IETF Proposed Standard. As a result, it is published in the Independent Stream.

### 1.1. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 2. The "safe" Preference

When present in a request, the "safe" preference indicates that the user prefers that the origin server to not respond with content which is designated as objectionable, according to the origin server's definition of the concept.

For example, a request that includes the "safe" preference:

```
GET /foo.html HTTP/1.1
Host: www.example.org
User-Agent: ExampleBrowser/1.0
Prefer: safe
```

Typically, user agents that emit the "safe" preference will include it in all requests with the "https" URI scheme, although some might expose finer-grained controls over when it is sent; this ensures that the preference is available to the applicable resources. User agents MUST NOT emit the "safe" preference on requests with the "http" URI scheme (see Section 4). See Appendix B for more information about configuring the set of resources "safe" is sent to.

Safe MAY be implemented in common HTTP libraries (e.g., an operating system might choose to insert the preference in requests based upon system-wide configuration).

Origin servers that utilize the "safe" preference ought to document that they do so, along with the criteria that they use to denote objectionable content. If a server has more fine-grained degrees of "safety", it SHOULD select a reasonable default to use, and document that; it MAY use additional mechanisms (e.g., cookies [RFC6265]) to fine-tune.

A response corresponding to the request above might have headers that look like this:

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: text/html
Preference-Applied: safe
Server: ExampleServer/2.0
Vary: Prefer
```

Here, the Preference-Applied response header ([RFC7240]) indicates that the site has applied the preference. Servers are not required to send Preference-Applied (even when they have applied the preference), but are encouraged to where possible.

Note that the Vary response header needs to be sent if the response is cacheable and might change depending on the value of the "Prefer" header. This is not only true for those responses that are "safe", but also the default "unsafe" response.

See [RFC7234] Section 4.1 for more information the interaction between Vary and Web caching.

See Appendix C for additional advice specific to Web servers wishing to use "safe".

### 3. Implementation Status

Note to RFC Editor: Please remove this section before publication.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft. 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.

- o Microsoft Internet Explorer - see <https://support.microsoft.com/en-hk/help/2980016/>
- o Microsoft Bing - see <https://developer.microsoft.com/en-us/microsoft-edge/testdrive/demos/familysearch/>
- o Mozilla Firefox - see <https://support.mozilla.org/en-US/kb/block-and-unblock-websites-parental-controls-firef>
- o Cisco - see <http://blogs.cisco.com/security/filtering-explicit-content>

#### 4. Security Considerations

The "safe" preference is not a secure mechanism; it can be inserted or removed by intermediaries with access to the request stream (e.g. for "http" URLs). Therefore, it is prohibited from being included in requests with the "http" scheme.

Its presence reveals information about the user, which may be of assistance in "fingerprinting" the user by sites and other entities in the network. This information which provides insight into the preferences of the user, might be used to make assumptions about the user and so could be used to target categories of user for purposes such as targeting (including advertising and identification of minors). Therefore, user agents SHOULD NOT include it in requests when the user has expressed a desire to avoid such attacks (e.g., some forms of "private mode" browsing).

By its nature, including "safe" in requests does not assure that all content will actually be safe; it is only when servers elect to honor it that content might be "safe".

Even then, a malicious server might adapt content so that it is even less "safe" (by some definition of the word). As such, this mechanism on its own is not enough to assure that only "safe" content is seen; those who wish to ensure that will need to combine its use with other techniques (e.g., content filtering).

Furthermore, the server and user may have differing ideas regarding the semantics of "safe." As such, the "safety" of the user's experience when browsing from site to site as well as over time might (and probably will) change.

#### 5. IANA Considerations

This specification registers the following entry in the "HTTP Preferences" registry [RFC7240]:

- o Preference: safe
- o Value: (no value)
- o Description: Indicates that "safe" / "unobjectionable" content is preferred.
- o Reference: (this document)
- o Notes:

## 6. References

### 6.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <<https://www.rfc-editor.org/info/rfc7234>>.
- [RFC7240] Snell, J., "Prefer Header for HTTP", RFC 7240, DOI 10.17487/RFC7240, June 2014, <<https://www.rfc-editor.org/info/rfc7240>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

### 6.2. Informative References

- [RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, DOI 10.17487/RFC6265, April 2011, <<https://www.rfc-editor.org/info/rfc6265>>.

## Appendix A. Acknowledgements

Thanks to Alissa Cooper, Ilya Grigorik, Emma Llanso, Jeff Hughes, Lorrie Cranor, Doug Turner and Dave Crocker for their comments.

## Appendix B. Sending "safe" from Web Browsers

As discussed in Section 2, there are many possible ways for the "safe" preference to be generated. One possibility is for a Web browser to allow its users to configure the preference to be sent.

When doing so, it is important not to misrepresent the preference as binding to Web sites. For example, an appropriate setting might be a checkbox with wording such as:

[ ] Request "safe" content from Web sites

... along with further information available upon request.

Browsers might also allow the "safe" preference to be "locked" - that is, prevent modification without administrative access, or a passcode.

Note that this specification does not require browsers to send "safe" on all requests, although that is one possible implementation; e.g., alternate implementation strategies include blacklists and whitelists.

## Appendix C. Supporting "safe" on Web Sites

Web sites that allow configuration of a "safe" mode (for example, using a cookie) can add support for the "safe" preference incrementally; since the preference will not be supported by all clients immediately, it is necessary to have another way to configure it.

When honoring the safe preference, it is important that it not be possible to disable it through the Web site's interface, since "safe" may be configured and locked down by the browser or computer's administrator (e.g., a parent). If the site has such a means of configuration (e.g., stored user preferences) and the safe preference is received in a request, the "safer" interpretation ought to be used.

The appropriate level of "safety" is a site-specific decision. When selecting it, sites ought to bear in mind that disabling the preference might be considerably more onerous than through other

means, especially if the preference is generated based upon Operating System configuration.

Sites might offer different levels of "safeness" through Web configuration, they will need to either inform their users of what level the "safe" hint corresponds to, or provide them with some means of adjusting it.

If the user expresses a wish to disable "safe" mode, the site can remind them that the safe preference is being sent, and ask them to consult their administrator (since "safe" might be set by a locked-down Operating System configuration).

As explained in Section 2, responses that change based upon the presence of the "safe" preference need to either carry the "Vary: Prefer" response header field, or be uncacheable by shared caches (e.g., with a "Cache-Control: private" response header field). This is to avoid an unsafe cached response being served to a client that prefers safe content (or vice versa).

#### Author's Address

Mark Nottingham

EMail: [mnot@mnot.net](mailto:mnot@mnot.net)

URI: <https://www.mnot.net/>

Network Working Group  
Internet-Draft  
Intended Status: Informational  
Expires: January 5, 2015

S. Leonard  
Penango, Inc.  
July 4, 2014

The text/markdown Media Type  
draft-seantek-text-markdown-00.txt

Abstract

This document registers the text/markdown media type for use with Markdown, a family of plain text formatting syntaxes that optionally can be converted to formal markup languages such as HTML.

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 <http://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."

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## 1. Introduction

In computer systems, textual data is stored and processed using a continuum of techniques. On the one end is plain text: a linear sequence of characters in some character set (code), possibly interrupted by line breaks, page breaks, or other control characters. Plain text provides /some/ fixed facilities for formatting instructions, namely codes in the character set that have meanings other than "represent this character on the output medium"; however, these facilities are not particularly extensible. Compare with [RFC6838] Section 4.2.1. (Applications may neuter the effects of these special characters by prohibiting them or by ignoring their dictated meanings, as is the case with how modern applications treat most control characters in US-ASCII.) On this end, any text reader or editor that interprets the character set can be used to see or manipulate the text. If some characters are corrupted, the corruption is unlikely to affect the ability of a computer system to process the text (even if the human meaning is changed).

On the other end is binary format: a sequence of instructions intended for some computer application to interpret and act upon. Binary formats are flexible in that they can store non-textual data efficiently (perhaps storing no text at all, or only storing certain kinds of text for very specialized purposes). Binary formats require an application to be coded specifically to handle the format; no partial interoperability is possible. Furthermore, if even one byte or bit are corrupted in a binary format, it may prevent an application from processing any of the data correctly.

Between these two extremes lies formatted text, i.e., text that includes non-textual information coded in a particular way, that affects the interpretation of the text by computer programs. Formatted text is distinct from plain text and binary format in that the non-textual information is encoded into textual characters, which are assigned specialized meanings /not/ defined by the character set. With a regular text editor and a standard keyboard (or other standard input mechanism), a user can enter these textual characters to express the non-textual meanings. For example, a character like "<" no longer means "LESS-THAN SIGN"; it means the start of a tag or element that affects the document in some way.

On the formal end of the spectrum is markup, a family of languages for annotating a document in such a way that the annotations are syntactically distinguishable from the text. Markup languages are (reasonably) well-specified and tend to follow (mostly) standardized syntax rules. Examples of markup languages include SGML, HTML, XML, and LaTeX. Standardized rules lead to interoperability between markup processors, but a skill requirement for new (human) users of the

language that they learn these rules in order to do useful work. This imposition makes markup less accessible for non-technical users (i.e., users who are unwilling or unable to invest in the requisite skill development).

```

informal          /-----formatted text-----\          formal
<-----v-----v-----v-----v-----v----->
plain text      informal markup    formal markup    binary format
                  (Markdown)        (HTML, XML, etc.)

```

Figure 1: Degrees of Formality in Data Storage Formats for Text

On the informal end of the spectrum are lightweight markup languages. In comparison with formal markup like XML, lightweight markup uses simple syntax, and is designed to be easy for humans to enter with basic text editors. Markdown, the subject of this document, is an /informal/ plain text formatting syntax that is intentionally targeted at non-technical users (i.e., users upon whom little to no skill development is imposed) using unspecialized tools (i.e., text boxes). Jeff Atwood once described these informal markup languages as /humane/. [HUMANE]

Markdown specifically is a family of syntaxes that are based on the original work of John Gruber with substantial contributions from Aaron Swartz, released in 2004. [MARKDOWN] Since its release a number of web or web-facing applications have incorporated Markdown into their text entry systems, frequently with proprietary extensions. Fed up with the complexity and security pitfalls of formal markup languages (e.g., HTML5) and proprietary binary formats (e.g., commercial word processing software), yet unwilling to be confined to the restrictions of plain text, many users have turned to Markdown for document processing. Whole toolchains now exist to support Markdown for online and offline projects.

Due to Markdown's intentional informality, there is no standard specifying the Markdown syntax, and no governing body that guides or impedes its development. Markdown works for users for two key reasons. First, the markup instructions (in text) look similar to the markup that they represent; therefore the cognitive burden to learn the syntax is very low. Second, the primary arbiter of the syntax's success is *\*running code\**. The tool that converts the Markdown to a presentable format, and not a series of formal pronouncements by a standards body, is the basis for whether syntactic elements matter.

To support identifying and conveying Markdown (as distinguished from plain text), this document defines a media type and a "flavor" parameter that indicates, in broad strokes, the author's intent on how to interpret the Markdown.

### 1.1. Requirements Terminology

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. Markdown Media Type Registration Applications

This section provides the media type registration application for the text/markdown media type (see [RFC6838], Section 5.6).

Type name: text

Subtype name: markdown

Required parameters: charset. Per Section 4.2.1 of [RFC6838], charset is REQUIRED. The default value is UTF-8. If omitted, parsers MAY reject the input; if parsers accept the input, they MUST interpret the content as UTF-8.

Optional parameters:

flavor=f; where f is an identifier that specifies the "flavor", or variation, of the Markdown syntax. The parameter represents the intent of the author, namely, that the Markdown will be interpreted "best" (i.e., as the author intended) when processed with tools associated with the identified flavor.

The flavor parameter is opaque and case-sensitive. Valid flavor values can be any sequence of characters or bytes; in practice, however, virtually all will be alphanumeric (US-ASCII) and registered in the IANA Markdown Flavors Registry, discussed in Section 4. Implementations checking flavor parameters MUST only compare them for exact equality.

Encoding considerations: Text.

Security considerations:

Markdown interpreted as plain text is relatively harmless. A text editor need only display the text. The editor SHOULD take care to handle control characters appropriately, and to limit the effect of the Markdown to the text editing area itself; malicious Unicode-based Markdown could, for example, surreptitiously change the directionality of the text. An editor for normal text would already take these control characters into consideration, however.

Markdown interpreted as a precursor to other formats, such as HTML,

carry all of the security considerations as the target formats. For example, HTML can contain instructions to execute scripts, redirect the user to other webpages, download remote content, and upload personally identifiable information. Markdown also can contain islands of formal markup, such as HTML. These islands of formal markup may be passed as-is, transformed, or ignored (perhaps because the islands are conditional or incompatible) when the Markdown is interpreted into the target format. Since Markdown may have different interpretations depending on the tool and the environment, a better approach is to analyze (and sanitize or block) the output markup, rather than attempting to analyze the Markdown.

Interoperability considerations:

Markdown flavors are designed to be broadly compatible with humans ("humane"), but not necessarily with each other. Therefore, syntax in one Markdown flavor may be ignored or treated differently in another flavor. The overall effect is a general degradation of the output, proportional to the quantity of flavor-specific Markdown used in the text. When it is desirable to reflect the author's intent in the output, stick with the flavor identified in the flavor parameter.

Published specification: This specification.

Applications that use this media type:

Markdown conversion tools, Markdown WYSIWYG editors, and plain text editors and viewers; target markup processors indirectly use Markdown (e.g., web browsers for Markdown converted to HTML).

Additional information:

Magic number(s): None  
File extension(s): .md, .markdown  
Macintosh File Type Code(s): TEXT

Person & email address to contact for further information:

Sean Leonard <dev+ietf@seantek.com>

Restrictions on usage: None.

Author: Sean Leonard <dev+ietf@seantek.com>

Intended usage: COMMON

Change controller: The IESG <iesg@ietf.org>

### 3. Example

The following is an example of Markdown as an e-mail attachment:

```
MIME-Version: 1.0
Content-Type: text/markdown; charset=UTF-8; flavor=GitHub
Content-Disposition: attachment; filename=readme.md

Sample GitHub Markdown
=====

This is some sample GitHub Flavored Markdown (*GFM*).
The generated HTML is then run through filters in the
[html-pipeline](https://github.com/jch/html-pipeline)
to perform things like [sanitization](#html-sanitization) and
[syntax highlighting](#syntax-highlighting).

Bulleted Lists
-----

Here are some bulleted lists...

* One Potato
* Two Potato
* Three Potato

- One Tomato
- Two Tomato
- Three Tomato

More Information
-----

[.markdown, .md](http://daringfireball.net/projects/markdown/)
has more information.
```

### 4. IANA Considerations

IANA is asked to register the media type text/markdown in the Standards tree using the application provided in Section 2 of this document.

IANA is also asked to establish a subtype registry called "Markdown Flavors". Entries in these registries is by Expert Review [RFC5226]. The Expert will determine whether the registration represents a bona-fide variation of the Markdown syntax (i.e., neither a duplicate of an existing registration nor a syntax that is something other than Markdown; [MARKDOWN] SHALL be treated as a normative basis), a brief description, one or more responsible parties, whether the flavor is being maintained at the time of registration, and the existence of at least one complete tool (with or without documentation) that processes the Markdown syntax into a formal document language.

A responsible party can be an individual author or maintainer, a corporate author or maintainer (plus an individual contact), or a representative of a community of interest dedicated to the Markdown syntax.

The registry shall have one initial value, "Standard", with the following data:

Description:

The Markdown syntax as it exists in the Markdown 1.0.1 Perl script at <http://daringfireball.net/projects/markdown/>, with accompanying documentation at <http://daringfireball.net/projects/markdown/syntax>.

Responsible Parties:

(individual)  
John Gruber <http://daringfireball.net/>  
[comments@daringfireball.net](mailto:comments@daringfireball.net)

Currently Maintained? No

Tool:

Name: Markdown 1.0.1  
Reference: <http://daringfireball.net/projects/markdown/>  
Purpose: Converts to HTML or XHTML circa 2004.

## 5. Security Considerations

See the answer to the Security Considerations template questions in Section 2.

## 6. References

### 6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

- [RFC5226] Narten, T., and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, May 2008.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, January 2013.

## 6.2. Informative References

- [HUMANE] Atwood, J., "Is HTML a Humane Markup Language?", WWW <http://blog.codinghorror.com/is-html-a-humane-markup-language/>, May 2008.
- [MARKDOWN] Gruber, J., "Daring Fireball: Markdown", WWW <http://daringfireball.net/projects/markdown/>, December 2004.

Author's Address

Sean Leonard  
Penango, Inc.  
5900 Wilshire Boulevard  
21st Floor  
Los Angeles, CA 90036  
USA

EMail: [dev+ietf@seantek.com](mailto:dev+ietf@seantek.com)  
URI: <http://www.penango.com/>