Workgroup: Network Working Group
Internet-Draft:
draft-dweekly-wrong-recipient-04
Published: 10 February 2024
Intended Status: Informational
Expires: 13 August 2024
Authors: D. Weekly

Adding a Wrong Recipient URL for Handling Misdirected Emails

Abstract

This document describes a mechanism for an email recipient to indicate to a sender that they are not the intended recipient.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://dweekly.github.io/ietf-wrong-recipient/draft-dweekly-wrong-recipient.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-dweekly-wrong-recipient/.

Source for this draft and an issue tracker can be found at https://github.com/dweekly/ietf-wrong-recipient.

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 13 August 2024.

Table of Contents

1. Introduction

Many users with common names and/or short email addresses receive transactional emails from service providers intended for others. These emails can't be unsubscribed (as they are transactional) but neither are they spam. These emails commonly are from a noreply@ email address; there is no standards-based mechanism to report a "wrong recipient" to the sender. Doing so is in the interest of all three involved parties: the inadvertent recipient (who does not want the email), the sender (who wants to be able to reach their customer and who does not want the liability of transmitting PII to a third party), and the valid recipient.

This document proposes a structured mechanism for the reporting of such misdirected email via either HTTPS POST or email inbox, directly mirroring the List-Unsubscribe and List-Unsubscribe-Post mechanisms of [RFC2369] and [RFC8058] respectively.

2. Proposal

There ought be a mechanism whereby a service can indicate it has an endpoint to indicate a "wrong recipient" of an email. If this header field is present in an email message, the user can select an option to indicate that they are not the intended recipient.

Similar to one-click unsubscription [RFC8058], the mail service can perform this action in the background as an HTTPS POST to the provided URL without requiring the user's further attention to the matter. A mailto: URI may also be included for non-HTTP MUAs, akin to List-Unsubscribe from [RFC2369].

Since it's possible the user may have a separate valid account with the sending service, it may be important that the sender be able to tie *which* email was sent to the wrong recipient. For this reason, the sender may also include an opaque blob in the header field to specify the account ID referenced in the email; this is included in the POST.

Note that this kind of misdelivery shouldn't be possible if a service has previously verified the user's email address for the account.

3. Conventions and Definitions

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.

4. High-Level Goals

Allow a recipient to stop receiving emails intended for someone else.

Allow a service to discover when they have the wrong email for a user.

5. Out of Scope

This document does not propose a mechanism for automatically discovering whether a given user is the correct recipient of an email, though it is possible to use some of the signals in an email, such as the intended recipient name, to infer a possible mismatch between actual and intended recipients.

6. Implementation

6.1. Mail Senders When Sending

Mail Senders that wish to be notified when a misdelivery has occurred **SHOULD** include a Wrong-Recipient header field with an HTTPS URI to which the recipient's mail client can POST and/or a mailto: URI to which an email should be sent. If this header field is included, the mail sender **MUST** ensure these endpoints are valid for a period of at least one year after sending.

The sender **MUST** encode a mapping to the underlying account identifier in the URI in order to allow the service to know which of their accounts has an incorrect email.

The URI **SHOULD** include an opaque identifier or another hard-to-forge component in addition to, or instead of, the plaintext recipient email address and user ID in order to prevent a malicious party from exercising the endpoint on a victim's behalf. Possible examples include using a signature parameter to the URI or UUID with a sender-local database lookup to retrieve the email and user ID referenced.

6.2. Mail Recipients

When a mail client receives an email that includes a Wrong-Recipient header field, an option **SHOULD** be exposed in the user interface that allows a recipient to indicate that the mail was intended for another user, if and only if the email is reasonably assured to not be spam, e.g. if both DKIM and SPF are passing with a valid DMARC record.

If the user selects this option, the mail client **MUST** perform an HTTPS POST to the first https URI in the Wrong-Recipient header field, or send an empty message to the first referenced mailto: address.

The POST request **MUST NOT** include cookies, HTTP authorization, or any other context information. The "wrong recipient" reporting operation is logically unrelated to any previous web activity, and context information could inappropriately link the report to previous activity.

The POST body **MUST** include only "Wrong-Recipient=true".

If the response is a HTTP 500 type error indicating server issue, the client **MAY** retry. If the HTTP response to the POST is a 200, the client **SHOULD NOT** retry. No feedback to the user as to the success or failure of this operation is proposed or required.

6.3. Mail Senders After Wrong Sender Notification

When a misdelivery has been indicated by a POST to the HTTPS URI or email to the given mailto: URI, the sender **MUST** make a reasonable effort to cease emails to the indicated email address for that user account.

The POST endpoint **MUST NOT** issue an HTTP redirect and **SHOULD** return a 200 OK status; the content body will be ignored.

Any GET request to the same URI **MUST NOT** be treated as an indication of a wrong recipient notification, since anti-spam software may attempt a GET request to URIs mentioned in mail headers without receiving user consent. Senders **MAY** return an error 405 Method Not Allowed in response to a GET request to the URI.

The sender **SHOULD** make a best effort to attempt to discern a correct email address for the user account, such as by using a different known email address for that user, postal mail, text message, phone call, app push, or presenting a notification in the user interface of the service. How the sender should accomplish this task is not part of this specification.

7. Additional Requirements

The email needs at least one valid authentication identifier. In this version of the specification the only supported identifier type is DKIM [RFC7489], that provides a domain-level identifier in the content of the "d=" tag of a validated DKIM-Signature header field.

The Wrong-Recipient header field needs to be included in the "h=" tag of a valid DKIM-Signature header field.

8. Header Syntax

The following ABNF imports fields and WSP from [RFC5322] and URI from [RFC3986]. Only https and mailto URIs are acceptable.

```
fields =/ wrong-recipient

wrong-recipient = "Wrong-Recipient:" 0*1WSP "<" URI ">" 0*WSP
URI = *( %x21-7E)    ; As defined in RFC 3986
```

9. Examples

9.1. Signed HTTPS URI

Header in Email:

```
Wrong-Recipient: <https://example.com/wrong-recipient?uid=12345&email=us
```

Resulting POST request

```
POST /wrong-recipient?uid=12345&email=user@example.org&sig=a29c83d HTTP/
Host: example.com
Content-Length: 20

Wrong-Recipient=true
```

9.2. UUID HTTPS URI

Header in Email:

```
Wrong-Recipient: <https://example.com/wrong-recipient?uuid=c002bd9a-e015
```

Resulting POST request

```
POST /wrong-recipient?uuid=c002bd9a-e015-468f-8621-9baf6fca12aa HTTP/1.1
Host: example.com
Content-Length: 20

Wrong-Recipient=true
```

9.3. Combined mailto: and HTTPS URIs

Header in Email:

```
Wrong-Recipient:
    <https://example.com/wrong-recipient?uuid=c002bd9a-e015-468f-8621-9b
    <mailto:wrong-recipient.c002bd9a-e015-468f-8621-9baf6fca12aa@example
```

10. Security Considerations

The Wrong-Recipient header field may contain the recipient address, but that is already exposed in other header fields like To:.

The user ID of the recipient with the sending service may be exposed by the Wrong-Recipient URI, which may not be desired but a sender can instead use an opaque blob to perform a mapping to a user ID on their end without leaking any information to outside parties, such as the UUID examples given above.

A bad actor with access to the user's email could maliciously indicate the recipient was a Wrong Recipient with any services that used this protocol, causing mail delivery and potentially account access difficulties for the user.

The Wrong-Sender POST provides a strong hint to the mailer that the address to which the message was sent was valid, and could in principle be used as a way to test whether an email address is valid. However, unlike passive methods like embedding tracking

pixels, the mechanism proposed here takes an active user action. Nonetheless, MUAs ought only expose this Wrong Recipient option if relatively confident that the email is not spam, using signals such as a valid DMARC record and passing DKIM & SPF checks.

A sender with a guessable URI structure and no use of either signed parameters or a UUID would open themselves up to a malicious party POST'ing email credentials for victims, potentially causing difficulty. Senders should be strongly encouraged to use a signature or opaque blob as suggested.

11. IANA Considerations

IANA has been requested to add a new entry to the "Provisional Message Header Field Names" registry, to be made permanent if this proposal becomes a standard.

Header field name: Wrong-Recipient
Protocol: mail
Status: Provisional
Author/Change controller: IETF
Specification document(s): *** This document ***
Related information: none

12. References

12.1. 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/rfc/rfc2119>.

[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/rfc/rfc8174>.

12.2. Informative References

[RFC2369] Neufeld, G. and J. Baer, "The Use of URLs as Meta-Syntax for Core Mail List Commands and their Transport through Message Header Fields", RFC 2369, DOI 10.17487/RFC2369, July 1998, <https://www.rfc-editor.org/rfc/rfc2369>.

[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, <https://www.rfc-editor.org/rfc/rfc3986>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/rfc/rfc5322>.

[RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <https://www.rfc-editor.org/rfc/rfc7489>.

[RFC8058] Levine, J. and T. Herkula, "Signaling One-Click Functionality for List Email Headers", RFC 8058, DOI 10.17487/RFC8058, January 2017, <https://www.rfc-editor.org/rfc/rfc8058>.

Acknowledgments

Many thanks to John Levine for helping shepherd this document as well as Oliver Deighton and Murray Kucherawy for their kind and actionable feedback on the language and first draft of the proposal. Thanks to Eliot Lear for helping guide the draft to the right hands for review. Many thanks to the members of IETF ART for vigorous discussion thereof.

Author's Address

David Weekly

Email: david@weekly.org