dkim
Internet-Draft
Intended status: Standards Track
Expires: 17 May 2026

R. Latimer
Inveigle.net

13 November 2025

DKIM2 Recipient and Next Domain Signing
draft-latimer-dkim2-rcpt-nd-signing-00

Abstract

This document proposes using the DKIM2 ESMTP extension to pass a signature for each intended recipient through the SMTP session rather than splitting email at the time of signing. This approach meets the DKIM2 objective of preventing email replay, while also preserving existing SMTP delivery logic, maintaining compatibility with DKIM and avoiding multiple calls to the DKIM2 filter.

Also proposed is a method of signing the next domain in the DKIM2 chain of custody independently from the DKIM2-Signature.

As per RFC5321, "... each and every extension, regardless of its benefits, must be carefully scrutinized with respect to its implementation, deployment, and interoperability costs". The requirements outlined herein ensure the majority of DKIM2 logic can be implemented within filters or supporting code, with only minimal and isolated changes in SMTP code.

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 17 May 2026.

Table of Contents

1. Introduction

DomainKeys Identified Mail v2 (DKIM2) [I-D.ietf-dkim-dkim2-motivation] requires a signature for each recipient as a means to prevent message replay. Because recipients may not appear in the headers and the need to avoid revealing their existence if they are not disclosed, the DKIM approach of creating a single signature for each email is insufficient.

This deficiency can be addressed by splitting messages prior to signing, however, that approach would lead to sub-optimal implementations. By generating a separate signature for each recipient to be passed through the SMTP session, it is possible to verify the intent to send to the final recipient while preserving existing SMTP delivery logic, without having to create multiple instances of the email prior to signing or needing to invoke the DKIM2 filter more than once.

Deliveries between DKIM2-capable SMTP implementations can continue using existing delivery logic to optimize delivery based on the route. All DKIM2-specific logic can be isolated within the DKIM2 filters and the delivery phase of SMTP.

Passing signatures through the SMTP session preserves the ability of systems to sign and forward, as is possible with DKIM. This is necessary for systems that for security or policy reasons must sign messages independently from other domain email, or systems that always forward email via another host or third party for delivery, including MUAs and systems sending signed transactional or notification emails.

DKIM2 also proposes a method for signing the next domain handling the email. By decoupling this record from the DKIM2-Signature and signing it independently, this information can be added to the headers once routing details are known by way of a lightweight DKIM2 stub that only performs basic header hashing, signing and addition. This approach avoids additional DNS lookups attempting to relate domains to specific hosts, eliminates the need for DNS records that may leak information about how a domain routes email and only requires relays to sign using a single key (in typical configurations), regardless of how many domains (or selectors) a system provides service for.

2. The DKIM2 service extension

The extension mechanism for SMTP is defined in Section 2.2 of the current SMTP specification [RFC5321].

The name of the extension is DKIM2. Servers implementing this extension advertise DKIM2 as a keyword in the response to EHLO.

2.1. SMTP Command Line Length

This extension increases the minimum SMTP server command line length of 1000 octets, in accordance with [RFC5321], section 4.5.3.1.4. This is sufficient to accommodate 4096 bit RSA keys while leaving reasonable space for existing extensions. RSA keys larger than 2048 bits, the maximum DKIM verifiers are required to support, are rare.

2.2. The DKIM2RCPT RCPT parameter

A DKIM2RCPT parameter is calculated for each recipient and passed between DKIM2-capable systems via the RCPT command. This value is calculated the first time an email signed and is typically retained until delivery, provided the message remains within the DKIM2 ecosystem. During email delivery or on handover to a system which isn't DKIM2-capable, this value is transferred to the email headers. The requirements for transferring this value to headers are described in Section 2.3.

Tags defined for the DKIM2RCPT parameter are below:

i=

Sequence number (from 1 to N) corresponding to the current (highest) i= value, matching that of the DKIM2-Signature(s) added to the email by the current domain (REQUIRED).

ABNF:

```
dkim2rcpt-i-tag = *DIGIT
```

s=

The name of the selector used to sign the DKIM2RCPT (REQUIRED).

ABNF:

```
dkim2rcpt-s-tag = sub-domain *("." sub-domain)
```

bcc

Indicates that the recipient is included via BCC. This tag MUST be present if the recipient is not listed in the To or Cc header fields.

A DKIM2-capable implementation MAY use this tag to split recipients when handing over to a system that does not support DKIM2, in lieu of performing its own header parsing.

b=

Signature over hash value of strings.

ABNF:

```
dkim2rcpt-b-tag = base64-string
```

2.2.1. Creating and verifying the DKIM2RCPT signature

A signer will nominate a DKIM2-Signature added by the domain and use the corresponding key to sign the DKIM2RCPT parameter. The i= and s= tags in DKIM2RCPT shall match those of the nominated DKIM2-Signature. A signer SHOULD NOT select an RSA-signed signature if an alternative is available. A signer MAY use different keys when sending to multiple recipients.

A signer MUST NOT set the DKIM2RCPT parameter when signing with an i= value greater than 1 unless it has declared itself to have exploded the message in the corresponding DKIM2-Signature or it has explicit signing authority granted by the DKIM2-NextDomain header.

The following steps will be applied, in order, to generate or verify the DKIM2RCPT signature:

1. Determine the hashing algorithm to use, based on that used by the nominated DKIM2-Signature.

2. Hash the chosen DKIM2-Signature header after applying "relaxed" canonicalization.

3. Add the email address of the intended recipient, enclosed in angled brackets (RCPT syntax).

4. Create the DKIM2RCPT record (including the DKIM2RCPT= prefix), with all required tags, leaving the value of the b= tag empty, and add the resulting record to the hash.

A signer would perform the following additional steps:

1. Sign the hash using the same key as the nominated DKIM2-Signature.

2. Base64 encode the signature and insert the resulting string into the b= tag of the DKIM2RCPT record.

A verifier would:

1. Decode the base64 signature from the b= tag of the DKIM2RCPT record.

2. Use the resulting signature to verify the hash.

2.3. Transfer to headers

In order to preserve the ability to perform end-to-end verification of an email, the DKIM2RCPT parameter along with the email address it signs, must be transferred to email headers under certain conditions.

A DKIM2 implementation MUST transfer DKIM2RCPT to a DKIM2-Recipient header on:

1. Handover for delivery

2. Handover to systems which do not support DKIM2

3. Generating a bounce message

A DKIM2 implementation SHOULD transfer DKIM2RCPT to a DKIM2-Recipient when on-sending with a new RCPT TO value, such as when:

1. Forwarding

2. Sending to mailing list recipients

An implementation MAY include multiple DKIM2-Recipient headers where the DKIM2RCPT bcc tag is not specified, but MUST create a separate instance of a message for each recipient where the bcc tag is specified.

Example of the resulting header:

```
DKIM2-Recipient: <user@example.com> DKIM2RCPT=i=1;s=selector;bcc;b=xXBUvPKVXejLi8mdvCeccD
0gFlzWzBe2JM/enQ13xJKAK+VbPHSvuvKa0WEwXgdDR1nSqw2/D1NIwatzf2rRAg==
```

The DKIM2RCPT parameter MUST NOT be sent to any upstream host that does not advertise DKIM2 support.

3. The DKIM2-NextDomain header

A signer will insert a DKIM2-NextDomain header to sign its intent to send email to the next domain. The next domain will be obtained by removing the hostname from the A/AAAA record referencing the host, or a statically configured value if the host is configured to relay via another server.

Tags defined for the DKIM2-NextDomain header are below:

i=

Sequence number (from 1 to N) corresponding to the current (highest) i= value, matching that of the DKIM2-Signature(s) added to the email by the current domain (REQUIRED).

ABNF:

```
dkim2nd-i-tag = *DIGIT
```

s=

The name of the selector used to sign the DKIM2-NextDomain header (REQUIRED).

ABNF:

```
dkim2nd-i-tag = sub-domain *("." sub-domain)
```

sa=

This tag explicitly grants signing authority for the specified domain to sign on behalf of the sender. This value may only be added to a DKIM2-NextDomain signature with an i= value of 1 and may have a value of either 1 or 0 (true or false).

If the sa flag is set, a signer MUST generate new DKIM2RCPT signatures for each recipient and verifier MUST accept the DKIM2-Signature as authoritative if the DKIM2-NextDomain signature can be verified.

```
dkim2nd-sa-tag = DIGIT
```

b=

Signature over hash value of strings.

ABNF:

```
dkim2nd-b-tag = base64-string
```

3.1. Creating and verifying the DKIM2-NextDomain signature

A signer will use a key to sign its intent to send email to the next domain. This key will be identified by the s= value from the DKIM2-NextDomain header and the d= value from any DKIM2-Signature with a corresponding i= value. The specified selector DOES NOT have to appear in a DKIM2-Signature header.

A signer SHOULD NOT use an RSA key if an alternative is available.

The following steps will be applied, in order, to generate or verify DKIM2-NextDomain signature:

1. Hash all instances of DKIM2-Signature with the corresponding i= after applying "relaxed" canonicalization. Headers are hashed from the bottom up.

2. Create the DKIM2-NextDomain header with all required tags, leaving the value of the b= tag empty, before adding the header to the hash.

A signer would perform the following additional steps:

1. Sign the hash using the nominated key.

2. Base64 encode the signed hash and insert the resulting string into the b= tag of the DKIM2-NextDomain header.

A verifier would:

1. Decode the base64 signature from the b= tag of the DKIM2-NextDomain header.

2. Use the resulting signature to verify the hash.

4. Tag Lists

The DKIM2-NextDomain header uses the same syntax as the DKIM2-Signature as defined in [RFC6376]. The DKIM2RCPT parameter uses a modified form of the DKIM tag-list that is suitable for inclusion as a RCPT command parameter. All values are strings containing either plain text or "base64" text as defined in [RFC2045].

```
dkim2rcpt-tag-list   = dkim2rcpt-tag-spec *( ";" dkim2rcpt-tag-spec ) [ ";" ]
dkim2rcpt-tag-spec   = dkim2rcpt-tag-name / (dkim2rcpt-tag-name "=" tag-value)
dkim2rcpt-tag-name   = ALPHA *(ALPHA / DIGIT)
dkim2rcpt-tag-value  = dkim2rcpt-tag-string / dkim2rcpt-tag-selector / base64-string
dkim2rcpt-tag-string = *(ALPHA / DIGIT / "-" /".")
dkim2rcpt-selector   = sub-domain *("." sub-domain)
base64-string        = *(ALPHA / DIGIT / "+" / "/") [=[=\]]
```

White space MUST NOT be present in a DKIM2RCPT parameter.

Tags MUST be interpreted in a case-sensitive manner. Values MUST be processed as case sensitive unless the specific tag description of semantics specifies case insensitivity.

Tags with duplicate names MUST NOT occur within a single dkim2rcpt-tag-list; if a tag name does occur more than once, the entire dkim2rcpt-tag-list is invalid.

Unrecognized tags MUST be ignored.

Tags that have an empty value are not the same as omitted tags. An omitted tag is treated as having the default value; A tag with an empty value explicitly designates the empty string.

Tags for which no dkim2rcpt-tag-value is specified are boolean; true if included, false if omitted.

5. Examples

Keys used for examples

Private key (PEM)

```
-----BEGIN PRIVATE KEY-----
MC4CAQAwBQYDK2VwBCIEIKKki7NRhuvTKReBbyE019YTrQDvsRQZBmxFAYEY6NaE
-----END PRIVATE KEY-----
```

Public key (DNS)

```
Ipr95PB/7JnJRg5nXs9MJvADv0J6I8/f2pB/ZNt24Dg=
```

Delivery to a system with DKIM2 capabilities.

```
S: 220 smtp.nextdomain.example ESMTP
C: ehlo [127.0.0.1]
S: 250-smtp.nextdomain.example Hello [127.0.0.1] [10.0.1.24], pleased to meet you
S: 250-DKIM2
S: 250 Help
C: mail from:<sender@example.com>
S: 250 2.0.0 OK
C: rcpt to:<user@example.com> DKIM2RCPT=i=1;s=ed25519;b=I1/77hTpktZA5URnD/5WAMDnrNgtE4uzq
Naq/BvFJKeEiONDD9BDTda4OeRIP1Jsi6+0RkyDVx69WGp1JrW5Bw==
S: 250 OK
C: data
S: 354 Go
C: DKIM2-NextDomain: i=1; s=ed25519; nd=nextdomain.example;
C:      b=+fBe1mhGbmishVq1nqPENvRivXX3bH5cwDOlZaK6c4npu/N5NJn/KLvoNuGysofLuXeQ/Rulk5bY
C:      U4hANn6ZBA==
C: DKIM2-Signature: i=1; a=ed25519-sha256; t=1763059498;
C:      s=ed25519; d=example.com;
C:      h=Date:Subject:From:To:Message-ID;
C:      bh=fdkeB/A0FkbVP2k4J4pNPoeWH6vqBm9+b0C3OY87Cw8=;
C:      b=kiuvOIgXlvP0lIgcRj0CXPrz5o1cpgj6Dh8ASIUz/xQMfXZCRsCyQq/Af0M7uKnLeqFlc/7M0xqP
C:      iFuSgjMXDA==
C: Date: Thu, 13 Nov 2025 18:44:58 GMT
C: Subject: Test
C: From: sender@example.com
C: To: user@example.com
C: Message-ID: <gXjs5gIAXngZjsu6@example.com>
C: Content-Type: text/plain; charset=us-ascii
C: Content-Transfer-Encoding: 7bit
C:
C: Test
C: .
S: 250 Accepted
```

Delivery to a system without DKIM2 capabilities.

```
S: 220 smtp.nextdomain.example ESMTP
C: ehlo [127.0.0.1]
S: 250-smtp.nextdomain.example Hello [127.0.0.1] [10.0.1.24], pleased to meet you
S: 250 Help
C: mail from:<sender@example.com>
S: 250 2.0.0 OK
C: rcpt to:<user@example.com>
S: 250 OK
C: data
S: 354 Go
C: DKIM2-Recipient: <user@example.com> DKIM2RCPT=i=1;s=ed25519;b=I1/77hTpktZA5URnD/5WAMDn
rNgtE4uzqNaq/BvFJKeEiONDD9BDTda4OeRIP1Jsi6+0RkyDVx69WGp1JrW5Bw==
C: DKIM2-NextDomain: i=1; s=ed25519; nd=nextdomain.example;
C:      b=+fBe1mhGbmishVq1nqPENvRivXX3bH5cwDOlZaK6c4npu/N5NJn/KLvoNuGysofLuXeQ/Rulk5bY
C:      U4hANn6ZBA==
C: DKIM2-Signature: i=1; a=ed25519-sha256; t=1763059498;
C:      s=ed25519; d=example.com;
C:      h=Date:Subject:From:To:Message-ID;
C:      bh=fdkeB/A0FkbVP2k4J4pNPoeWH6vqBm9+b0C3OY87Cw8=;
C:      b=kiuvOIgXlvP0lIgcRj0CXPrz5o1cpgj6Dh8ASIUz/xQMfXZCRsCyQq/Af0M7uKnLeqFlc/7M0xqP
C:      iFuSgjMXDA==
C: Date: Thu, 13 Nov 2025 18:44:58 GMT
C: Subject: Test
C: From: sender@example.com
C: To: user@example.com
C: Message-ID: <gXjs5gIAXngZjsu6@example.com>
C: Content-Type: text/plain; charset=us-ascii
C: Content-Transfer-Encoding: 7bit
C:
C: Test
C: .
S: 250 Accepted
```

6. References

6.1. Normative References

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, DOI 10.17487/RFC2045, November 1996, <https://www.rfc-editor.org/info/rfc2045>.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <https://www.rfc-editor.org/info/rfc5321>.

[RFC6376] Crocker, D., Ed., Hansen, T., Ed., and M. Kucherawy, Ed., "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, DOI 10.17487/RFC6376, September 2011, <https://www.rfc-editor.org/info/rfc6376>.

6.2. Informative References

[I-D.ietf-dkim-dkim2-motivation] Gondwana, B., Clayton, R., and W. Chuang, "DKIM2 - signing the source and destination of every email", Work in Progress, Internet-Draft, draft-ietf-dkim-dkim2-motivation-02, 2 November 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-dkim-dkim2-motivation-02>.

Author’s Address

Roydon Latimer
Inveigle.net
Auckland
New Zealand
Email: cs@inveigle.net