

DKIM
Internet-Draft
Obsoletes: [4871](#), [5672](#) (if approved)
Intended status: Standards
Expires: July 18, 2011

D. Crocker, Ed.
Brandenburg InternetWorking
M. Kucherawy, Ed.
Cloudmark
January 14, 2011

DomainKeys Identified Mail (DKIM) Signatures - Over DOSETA
draft-crocker-dkim-rfc4871bis-doseta-00

Abstract

DomainKeys Identified Mail (DKIM) permits a person, role, or organization that owns the signing domain to claim some responsibility for a message by associating the domain with the message. This can be an author's organization, an operational relay or one of their agents. DKIM separates the question of the identity of the signer of the message from the purported author of the message. Assertion of responsibility is validated through a cryptographic signature and querying the signer's domain directly to retrieve the appropriate public key. Message transit from author to recipient is through relays that typically make no substantive change to a message or its content and thus preserve the DKIM signature.

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 July 18, 2011.

Copyright Notice

Copyright (c) 2011 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

Internet-Draft

RFC4871bis

January 2011

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.	Signing Identity {rfc4871bis-02 1.1}	4
1.2.	Terminology and Definitions	4
2.	Signing and Verifying Protocol {added}	5
3.	Extensions to DOSETA Template {rfc4871bis-02 - adapted for Doseta overlay}	6
3.1.	Signature Data Structure {rfc4871bis-02 3.5, subset} . . .	6
3.2.	Relationship between SDID and AUID {rfc4871bis-02 3.9} . .	13
3.3.	Stored Key Data {rfc4871bis-02 3.6.1, subset}	14
4.	Considerations	16
4.1.	Security Considerations {rfc4871bis-02 8, subset}	16
4.2.	IANA Considerations {rfc4871bis-02 7, subset}	17
5.	References	19
5.1.	Normative References	19
5.2.	Informative References	19
Appendix A.	MUA Considerations {rfc4871bis-02 D}	20
Appendix B.	End-to-End Scenario Example {rfc4871bis-02 A}	21
B.1.	The User Composes an Email	21
B.2.	The Email is Signed	21
B.3.	The Email Signature is Verified	22
Appendix C.	Types of Use {rfc4871bis-02 B}	23
C.1.	Alternate Submission Scenarios	24
C.2.	Alternate Delivery Scenarios	26
Appendix D.	Acknowledgements	28
	Authors' Addresses	28

Internet-Draft

RFC4871bis

January 2011

1. Introduction

DomainKeys Identified Mail (DKIM) permits a person, role, or organization to claim some responsibility for a message by associating a domain name [[RFC1034](#)] with the message [[RFC5322](#)]. This can be an author's organization, an operational relay or one of their agents. Assertion of responsibility is validated through a cryptographic signature and querying the signer's domain directly to retrieve the appropriate public key. Message transit from author to recipient is through relays that typically make no substantive change to the message content and thus preserve the DKIM signature. A message can contain multiple signatures, from the same or different organizations involved with the message.

The DKIM service is described in detail in [[RFC5585](#)] and [[RFC5863](#)].

This version of DKIM is based on a split between DKIM-specific details, versus an underlying component mechanism, called DOSETA, that can be used for other services. [[I-D.Doseta](#)].

NOTE: Much of the text for this draft is taken from the DKIM working group [draft-ietf-DKIM-rfc4871bis-02](#) revision. Sections in this document cross reference their source with the notation:

{rfc4871bis-02 xx}

where "xx" indicates the section number. It might also indicate that a subset is taken, such as when a portion is applied to the DKIM-over-DOSETA draft and a portion to the DOSETA draft. In some cases the base text also has been enhanced.

The approach taken by DKIM differs from previous approaches to message signing (for example., Secure/Multipurpose Internet Mail Extensions (S/MIME) [[RFC1847](#)], OpenPGP [[RFC4880](#)]) in that:

- o the message signature is written as a message header field so that

neither human recipients nor existing MUA (Mail User Agent) software is confused by signature-related content appearing in the message body;

- o there is no dependency on public and private key pairs being issued by well-known, trusted certificate authorities;
- o there is no dependency on the deployment of any new Internet protocols or services for public key distribution or revocation;
- o signature verification failure does not force rejection of the message;

- o no attempt is made to include encryption as part of the mechanism;
- o message archiving is not a design goal.

DKIM:

- o is compatible with the existing email infrastructure and transparent to the fullest extent possible;
- o requires minimal new infrastructure;
- o can be implemented independently of clients in order to reduce deployment time;
- o can be deployed incrementally;
- o allows delegation of signing to third parties.

[1.1.](#) Signing Identity {rfc4871bis-02 1.1}

DKIM separates the question of the identity of the signer of the message from the purported author of the message. In particular, a signature includes the identity of the signer. Verifiers can use the signing information to decide how they want to process the message. The signing identity is included as part of the signature header field.

The signing identity specified by a DKIM signature is not required to match an address in any particular header field because of the broad

methods of interpretation by recipient mail systems, including MUAs.

[1.2.](#) Terminology and Definitions

This specification incorporates the terminology defined in [\[I-D.Doseta\]](#).

Syntax descriptions use Augmented BNF (ABNF) [\[RFC5234\]](#).

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 [\[RFC2119\]](#).

Additional terminology: {rfc4871bis-02 #2, subset}

Signing Domain Identifier (SDID): This augments the definition provided by DOSETA [\[I-D.Doseta\]](#). A single domain name that is the mandatory payload output of DKIM and that refers to the identity claiming responsibility for introduction of a message into the mail stream. For DKIM processing, the name has only basic domain name semantics; any possible owner-specific semantics are outside the scope of DKIM. It is specified in [Section 3.1](#).

Agent or User Identifier (AUID): A single identifier that refers to the agent or user on behalf of whom the Signing Domain Identifier (SDID) has taken responsibility. The AUID comprises a domain name and an optional <local-part>. The domain name is the same as that used for the SDID or is a sub-domain of it. For DOSETA processing, the domain name portion of the AUID has only basic domain name semantics; any possible owner-specific semantics are outside the scope of DOSETA. It is specified in [Section 3.1](#) .

Identity Assessor: This augments the definition of Assessor in [\[I-D.Doseta\]](#). A module that consumes DOSETA's mandatory payload, which is the responsible Signing Domain Identifier

(SDID). The module is dedicated to the assessment of the delivered identifier. Other DOSETA (and non-DOSETA) values can also be delivered to this module as well as to a more general message evaluation filtering engine. However, this additional activity is outside the scope of the DKIM signature specification.

2. Signing and Verifying Protocol {added}

DKIM uses the DOSETA "Generic Header/Content Signing Service Template" [[I-D.Doseta](#)] as its base.

The DOSETA template specifies TEMPLATE information that is required to tailor the signing service:

Signature Association: The DOSETA-Signature data are stored in a DKIM-Signature header field that is part of the Header of the message being signed. This contains all of the signature and key-fetching data, per [[I-D.Doseta](#)].

Semantics Signaling: The presence of a DKIM-Signature header fields signals the use of DKIM.

Semantics: A DKIM signature means that the owner of the signing domain is taking "some" responsibility for the message. Hence, the payload, or output, of DKIM is:

- + A validated domain name, specifically the d= parameter in the DKIM-Signature header field
- + An indication that its use has been validated

The nature and extent of a DKIM signer's responsibility can vary widely and is beyond the scope of this specification.

Header/Content Mapping: DKIM maps the DOSETA Header processing to an email header and the DOSETA Content to an email body, per [\[RFC5322\]](#)

[3.](#) Extensions to DOSETA Template {rfc4871bis-02 - adapted for Doseta overlay}

This section contains specifications that are added to the basic DOSETA H/C Signing Template.

[3.1.](#) Signature Data Structure {rfc4871bis-02 3.5, subset}

These are DKIM-specific tags: (

i= The Agent or User Identifier (AUID) on behalf of which the SDID is taking responsibility (DOSETA-quoted-printable; OPTIONAL, default is an empty <local-part> followed by an "@" followed by the domain from the "d=" tag).

The syntax is a standard email address where the <local-part> MAY be omitted. The domain part of the address MUST be the same as, or a subdomain of, the value of the "d=" tag.

Internationalized domain names MUST be converted using the steps listed in [Section 4 of \[RFC5890\]](#) using the "ToASCII" function.

ABNF:
sig-i-tag = %x69 [FWS] "=" [FWS]
 [local-part] "@" domain-name

The AUID is specified as having the same syntax as an email address, but is not required to have the same semantics. Notably, the domain name is not required to be registered in the DNS -- so it might not resolve in a query -- and the <local-part> MAY be drawn from a namespace unrelated to any mailbox. The details of the structure and semantics for the namespace are determined by the Signer. Any knowledge or use of those details by verifiers or assessors is outside the scope of the DOSETA Signing specification. The Signer MAY choose to use the same namespace for its AUIDs as its users' email addresses or MAY choose other means of representing its users. However, the signer SHOULD use the same AUID for each message intended to be evaluated as being within the same sphere of responsibility, if it wishes to offer receivers the option of using the AUID as a stable identifier that is finer grained than the SDID.

NOTE: The <local-part> of the "i=" tag is optional because in some cases a signer might not be able to establish a verified individual identity. In such cases, the signer might wish to assert that although it is willing to go as far as signing for the domain, it is unable or unwilling to commit to an individual user name within their domain. It can do so by including the domain part but not the <local-part> of the identity.

NOTE: Absent public standards for the semantics of an AUID, an assessment based on AUID requires a non-standardized basis.

NOTE: This specification does not require the value of the "i=" tag to match the identity in any Header field. This is considered to be an assessment-time policy issue. Constraints between the value of the "i=" tag and other identities in other Header fields might seek to apply basic authentication into the semantics of trust associated with a role such as content author. Trust is a broad and complex topic and trust mechanisms are subject to highly creative attacks. The real-world efficacy of any but the most basic bindings between the "i=" value and other identities is not

well established, nor is its vulnerability to subversion by

an attacker. Hence reliance on the use of these options needs to be strictly limited. In particular, it is not at all clear to what extent a typical end-user recipient can rely on any assurances that might be made by successful use of the "i=" options.

l= Content length count (plain-text unsigned decimal integer; OPTIONAL, default is entire Content). This tag informs the verifier of the number of octets in the Content of the data after canonicalization included in the cryptographic hash, starting from 0 immediately following the CRLF preceding the Content. This value MUST NOT be larger than the actual number of octets in the canonicalized Content.

ABNF:

```
sig-l-tag    = %x6c [FWS] "=" [FWS]
              1*76DIGIT
```

NOTE: Use of the "l=" tag might allow display of fraudulent content without appropriate warning to end users. The "l=" tag is intended for increasing signature robustness when sending to intermediaries that append data to Content, such as mailing lists that both modify their content and do not sign their messages. However, using the "l=" tag enables attacks in which an intermediary with malicious intent modifies a message to include content that solely benefits the attacker. It is possible for the appended content to completely replace the original content in the end recipient's eyes and to defeat duplicate message detection algorithms. Examples are described in Security Considerations [Section 4.1](#). To avoid this attack, signers need be extremely wary of using this tag, and verifiers might wish to ignore the tag or remove text that appears after the specified content length.

NOTE: The value of the "l=" tag is constrained to 76 decimal digits. This constraint is not intended to predict the size of future data or to require implementations to use an integer representation large enough to represent the maximum possible value, but is intended to remind the implementer to check the length of this and all other tags during verification and to test for integer overflow when decoding the value. Implementers might need to limit the actual value expressed to a value smaller than 10^{76} , for example, to allow a message to fit within the available storage

space.

z= Copied Header fields (DOSETA-quoted-printable, but see description; OPTIONAL, default is null). A vertical-bar-separated list of selected Header fields present when the message was signed, including both the field name and value. It is not required to include all Header fields present at the time of signing. This field need not contain the same Header fields listed in the "h=" tag. The Header field text itself MUST encode the vertical bar ("|", %x7C) character. That is, vertical bars in the "z=" text are meta-characters, and any actual vertical bar characters in a copied header field MUST be encoded. Note that all whitespace MUST be encoded, including whitespace between the colon and the header field value. After encoding, FWS MAY be added at arbitrary locations in order to avoid excessively long lines; such whitespace is NOT part of the value of the header field, and MUST be removed before decoding.

The Header fields referenced by the "h=" tag refer to the fields in the [[RFC5322](#)] Header, not to any copied fields in the "z=" tag. Copied header field values are for diagnostic use.

Header fields with characters requiring conversion (perhaps from legacy MTAs that are not [[RFC5322](#)] compliant) SHOULD be converted as described in MIME Part Three [[RFC2047](#)].

ABNF:

```
sig-z-tag      = %x7A [FWS] "=" [FWS]
                  sig-z-tag-copy
                  *( "|" [FWS] sig-z-tag-copy )
sig-z-tag-copy = hdr-name [FWS] ":"
                  qp-hdr-value
```

Internet-Draft

RFC4871bis

January 2011

EXAMPLE of a signature header field spread across multiple continuation lines:

```
DKIM-Signature: v=1; a=rsa-sha256; d=example.net;
s=brisbane; c=simple; q=dns/txt; i=@eng.example.net;
t=1117574938; x=1118006938;
h=from:to:subject:date;
z=From:foo@eng.example.net|To:joe@example.com|
Subject:demo=20run|
Date:July=205,=202005=203:44:08=20PM=20-0700;
bh=MTIzNDU2Nzg5MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTI=;
b=dzdVy0fAKCdLXdJOc9G2q8LoXSlEniSbav+yuU4zGeeruD00lszZVoG4ZHRNiYzR
```

[3.1.1.](#) Content Length Limits {rfc4871bis-02 3.4.5}

A text length count MAY be specified to limit the signature calculation to an initial prefix of an ASCII text data portion, measured in octets. If the Content length count is not specified, the entire Content is signed.

This capability is provided because it is very common for intermediate data handling services to add trailers to text (for example, instructions how to get off a mailing list). Until such data is signed by the intermediate handler, the text length count can be a useful tool for the verifier since it can, as a matter of policy, accept messages having valid signatures that do not cover the additional data.

NOTE: Using text length limits enables an attack in which an attacker modifies a message to include content that solely benefits the attacker. It is possible for the appended content to completely replace the original content in the end recipient's eyes and to defeat duplicate message detection algorithms. To avoid this attack, signers need to be wary of using this tag, and verifiers might wish to ignore the tag or remove text that appears after the specified content length, perhaps based on other criteria.

The text length count allows the signer of text to permit data to be appended to the end of the text of a signed message. The text length

count MUST be calculated following the canonicalization algorithm; for example, any whitespace ignored by a canonicalization algorithm is not included as part of the Content length count. Signers of MIME messages that include a Content length count SHOULD be sure that the length extends to the closing MIME boundary string.

NOTE: A creator wishing to ensure that the only acceptable modifications are to add to a MIME postlude would use a text length count encompassing the entire final MIME boundary string, including the final "--CRLF". A signer wishing to allow additional MIME parts but not modification of existing parts would use a Content length count extending through the final MIME boundary string, omitting the final "--CRLF". Note that this only works for some MIME types, such as, multipart/mixed but not multipart/signed.

A text length count of zero means that the text is completely unsigned.

Creators wishing to ensure that no modification of any sort can occur will specify the "simple" canonicalization algorithm for all data portions and will and omit the text length counts.

[3.1.2.](#) Recommended Signature Content {rfc4871bis-02 5.5}

In order to maximize compatibility with a variety of verifiers, it is recommended that signers follow the practices outlined in this section when signing a message. However, these are generic recommendations applying to the general case; specific senders might wish to modify these guidelines as required by their unique situations. Verifiers MUST be capable of verifying signatures even if one or more of the recommended Header fields is not signed (with the exception of From:, which MUST always be signed) or if one or more of the dis-recommended Header fields is signed. Note that verifiers do have the option of ignoring signatures that do not cover a sufficient portion of the header or Content, just as they might ignore signatures from an identity they do not trust.

The following Header fields SHOULD be included in the signature, if they are present in the message being signed:

- o From (REQUIRED in all signatures)
- o Sender, Reply-To
- o Subject
- o Date, Message-ID
- o To, Cc
- o MIME-Version

- o Content-Type, Content-Transfer-Encoding, Content-ID, Content-Description
- o Resent-Date, Resent-From, Resent-Sender, Resent-To, Resent-Cc, Resent-Message-ID
- o In-Reply-To, References
- o List-Id, List-Help, List-Unsubscribe, List-Subscribe, List-Post, List-Owner, List-Archive

The following Header fields SHOULD NOT be included in the signature:

- o Return-Path
- o Received
- o Comments, Keywords
- o Bcc, Resent-Bcc
- o DOSETA-Signature

Optional Header fields (those not mentioned above) normally SHOULD NOT be included in the signature, because of the potential for

additional Header fields of the same name to be legitimately added or reordered prior to verification. There are likely to be legitimate exceptions to this rule, because of the wide variety of application-specific Header fields that might be applied to a message, some of which are unlikely to be duplicated, modified, or reordered.

Signers SHOULD choose canonicalization algorithms based on the types of data they process and their aversion to risk. For example, e-commerce sites sending primarily purchase receipts, which are not expected to be processed by intermediaries such as mailing lists or other software likely to modify data, will generally prefer "simple" canonicalization. Sites sending primarily person-to-person data will likely prefer to be more resilient to modification during transport by using "relaxed" canonicalization.

Signers SHOULD NOT use "l=" unless they intend to accommodate intermediate data processors that append text to a message. For example, many mailing list processors append "unsubscribe" information to message bodies. If signers use "l=", they SHOULD include the entire Content existing at the time of signing in computing the count. In particular, signers SHOULD NOT specify a Content length of 0 since this might be interpreted as a meaningless signature by some verifiers.

[3.1.3.](#) Signature Verification {rfc4871bis-02 6.1.3, subset}

A Content length specified in the "l=" tag of the signature limits the number of bytes of the Content passed to the verification algorithm. All data beyond that limit is not validated by DOSETA. Hence, verifiers might treat a message that contains bytes beyond the indicated Content length with suspicion, such as by truncating the message at the indicated Content length, declaring the signature invalid (for example, by returning PERMFAIL (unsigned content)), or conveying the partial verification to the policy module.

NOTE: Verifiers that truncate the Content at the indicated Content length might pass on a malformed MIME message if the signer used the "N-4" trick (omitting the final "--CRLF") described in the informative note in [Section 3.1.1](#). Such verifiers might wish to check for this case and include a trailing "--CRLF" to avoid breaking the MIME structure. A simple way to achieve this might be to append "--CRLF" to any "multipart" message with a Content

length; if the MIME structure is already correctly formed, this will appear in the postlude and will not be displayed to the end user.

[3.2.](#) Relationship between SDID and AUID {rfc4871bis-02 3.9}

DOSETA's primary task is to communicate from the Signer to a recipient-side Identity Assessor a single Signing Domain Identifier (SDID) that refers to a responsible identity. DOSETA MAY optionally provide a single responsible Agent or User Identifier (AUID).

Hence, DOSETA's mandatory output to a receive-side Identity Assessor is a single domain name. Within the scope of its use as DOSETA output, the name has only basic domain name semantics; any possible owner-specific semantics are outside the scope of DOSETA. That is, within its role as a DOSETA identifier, additional semantics cannot be assumed by an Identity Assessor.

Upon successfully verifying the signature, a receive-side DOSETA verifier MUST communicate the Signing Domain Identifier (d=) to a consuming Identity Assessor module and MAY communicate the Agent or User Identifier (i=) if present.

To the extent that a receiver attempts to intuit any structured semantics for either of the identifiers, this is a heuristic function that is outside the scope of DOSETA's specification and semantics. Hence, it is relegated to a higher-level service, such as a delivery handling filter that integrates a variety of inputs and performs heuristic analysis of them.

This document does not require the value of the SDID or AUID to match an identifier in any other message header field. Such a requirement is, instead, an assessor policy issue. The purpose of such a linkage could be to authenticate the value in that other header field. This, in turn, is the basis for applying a trust assessment based on the identifier value. Trust is a broad and complex topic and trust mechanisms are subject to highly creative attacks. The real-world efficacy of any but the most basic bindings between the SDID or AUID and other identities is not well established, nor is its vulnerability to subversion by an attacker. Hence, reliance on the use of such bindings SHOULD be strictly limited. In particular, it

is not at all clear to what extent a typical end-user recipient can rely on any assurances that might be made by successful use of the SDID or AUID.

[3.3.](#) Stored Key Data {rfc4871bis-02 3.6.1, subset}

This section defines additions to the DOSETA Library, concerning stored key data.

`g=` Granularity of the key (plain-text; OPTIONAL, default is "*"). This value MUST match the Local-part of the "i=" tag of the DKIM- Signature header field (or its default value of the empty string if "i=" is not specified), with a single, optional "*" character matching a sequence of zero or more arbitrary characters ("wildcarding"). An email with a signing address that does not match the value of this tag constitutes a failed verification. The intent of this tag is to constrain which signing address can legitimately use this selector, for example, when delegating a key to a third party that should only be used for special purposes. Wildcarding allows matching for addresses such as "user+*" or "*-offer". An empty "g=" value never matches any addresses.

ABNF:

```
key-g-tag      = %x67 [FWS] "=" [FWS] key-g-tag-lpart
                key-g-tag-lpart = [dot-atom-text]
                                ["*" [dot-atom-text] ]
```

`h=` Acceptable hash algorithms (plain-text; OPTIONAL, defaults to allowing all algorithms). A colon-separated list of hash algorithms that might be used. Signers and Verifiers MUST support the "sha256" hash algorithm. Verifiers MUST also support the "sha1" hash algorithm. Unrecognized hash

algorithms MUST be ignored.

ABNF:

```
key-h-tag      = %x68 [FWS] "=" [FWS]
                  key-h-tag-alg
                  0*( [FWS] ":" [FWS]
                      key-h-tag-alg )
key-h-tag-alg  = "sha1" / "sha256" /
                  x-key-h-tag-alg
x-key-h-tag-alg = hyphenated-word
                  ; for future extension
```

s= Service Type (plain-text; OPTIONAL; default is "*"). A colon-separated list of service types to which this record applies. Verifiers for a given service type MUST ignore this record if the appropriate type is not listed. Unrecognized service types MUST be ignored. Currently defined service types are as follows:

* matches all service types

email electronic mail (not necessarily limited to SMTP)

This tag is intended to constrain the use of keys for other purposes, if use of DOSETA is defined by other services in the future.

ABNF:

```
key-s-tag      = %x73 [FWS] "=" [FWS]
                  key-s-tag-type
                  0*( [FWS] ":" [FWS]
                      key-s-tag-type )
key-s-tag-type  = "email" / "*" /
                  x-key-s-tag-type
x-key-s-tag-type = hyphenated-word
                  ; for future extension
```

- t= Flags, represented as a colon-separated list of names (plain-text; OPTIONAL, default is no flags set). Unrecognized flags MUST be ignored. The defined flags are as follows:
- y This domain is testing DOSETA. Verifiers MUST NOT treat data from signers in testing mode differently from unsigned data, even if the signature fails to verify. Verifiers MAY wish to track testing mode results to assist the signer.
 - s Any DOSETA-Signature Header fields using the "i=" tag MUST have the same domain value on the right-hand side of the "@" in the "i=" tag and the value of the "d=" tag. That is, the "i=" domain MUST NOT be a subdomain of "d=". Use of this flag is RECOMMENDED unless subdomaining is required.

ABNF:

```
key-t-tag      = %x74 [FWS] "=" [FWS]
                  key-t-tag-flag
                  0*( [FWS] ":" [FWS]
                     key-t-tag-flag )
key-t-tag-flag = "y" / "s" /
                  x-key-t-tag-flag
x-key-t-tag-flag = hyphenated-word
                  ; for future extension
```

[4. Considerations](#)

[4.1. Security Considerations {rfc4871bis-02 8, subset}](#)

[4.1.1. Misuse of Content Length Limits \("l=" Tag\)](#)

Content length limits (in the form of the "l=" tag) are subject to several potential attacks.

[4.1.1.1. Addition of New MIME Parts to Multipart/*](#)

If the Content length limit does not cover a closing MIME multipart section (including the trailing "--CRLF" portion), then it is possible for an attacker to intercept a properly signed multipart message and add a new Content part. Depending on the details of the MIME type and the implementation of the Consumer, this could allow an attacker to change the information displayed to an end user from an

apparently trusted source.

For example, if attackers can append information to a "text/html" Content part, they might be able to exploit a bug in some MUAs that continue to read after a "</html>" marker, and thus display HTML text on top of already displayed text. If a message has a "multipart/alternative" Content part, they might be able to add a new Content part that is preferred by the displaying MUA.

[4.1.1.2.](#) Addition of new HTML content to existing content

Several receiving MUA implementations do not cease display after a "</html>" tag. In particular, this allows attacks involving overlaying images on top of existing text.

EXAMPLE: Appending the following text to an existing, properly closed message will in many MUAs result in inappropriate data being rendered on top of existing, correct data:

```
<div style="position: relative; bottom: 350px; z-index: 2;">
 </div>
```

[4.2.](#) IANA Considerations {rfc4871bis-02 7, subset}

DKIM uses registries now assigned to DOSETA [[I-D.Doseta](#)]. This section specifies additions to the registries that were in the original DKIM Signing specification. They are not part of the DOSETA specification, but are now specific to DKIM.

[4.2.1.](#) DKIM-Signature Tag Specifications

These values are added to the registry that is now defined in [[I-D.Doseta](#)]:

+-----+-----+-----+-----+-----+-----+	
TYPE	REFERENCE
+-----+-----+-----+-----+-----+-----+	
i	(this document, Section 3.1)
l	(this document, Section 3.1)
z	(this document, Section 3.1)
+-----+-----+-----+-----+-----+-----+	

Table 1: DKIM-Signature Tag Specification Registry Initial Values

4.2.2. _domainkey DNS TXT Record Tag Specifications

These values are added to the registry that is now defined in [\[I-D.Doleta\]](#):

Crocker & Kuchera

Expires July 18, 2011

[Page 17]

Internet-Draft

RFC4871bis

January 2011

TYPE	REFERENCE
g	(this document, Section 3.3)
h	(this document, Section 3.3)
s	(this document, Section 3.3)
t	(this document, Section 3.3)

DKIT _domainkey DNS TXT Record Tag Specification Registry

Initial Values

4.2.3. DKIM Service Types Registry

The "s=" <key-s-tag> tag (specified in [Section 3.3](#)) provides for a list of service types to which this selector might apply.

IANA has established the DKIM Service Types Registry for service types.

The initial entries in the registry comprise:

TYPE	REFERENCE
email	(this document, Section 3.3 , "s=")
*	(this document, , Section 3.3 , "s=")

DKIM Service Types Registry Initial Values

4.2.4. DKIM Service Flags Registry

Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

- [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", [RFC 4234](#), January 2008.
- [RFC5322] Resnick, P., "Internet Message Format", [RFC 5322](#), October 2008.
- [RFC5672] Crocker, D., Ed., "[RFC 4871](#) DomainKeys Identified Mail (DKIM) Signatures: Update", [RFC 5672](#), August 2009.
- [RFC5890] Klensin, J., "Internationalizing Domain Names in Applications (IDNA): Definitions and Document Framework", [RFC 5890](#), August 2010.

[5.2](#). Informative References

- [RFC1847] Galvin, J., Murphy, S., Crocker, S., and N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", [RFC 1847](#), October 1995.
- [RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration

Procedures for Message Header Fields", [BCP 90](#), [RFC 3864](#), September 2004.

- [RFC4870] Delany, M., "Domain-Based Email Authentication Using Public Keys Advertised in the DNS (DomainKeys)", [RFC 4870](#), May 2007.
- [RFC4871] Allman, E., Callas, J., Delany, M., Libbey, M., Fenton, J., and M. Thomas, "DomainKeys Identified Mail (DKIM) Signatures", [RFC 4871](#), May 2007.
- [RFC4880] Callas, J., Donnerhake, L., Finney, H., and R. Thayer, "OpenPGP Message Format", [RFC 4880](#), November 2007.
- [RFC5585] Hansen, T., Crocker, D., and P. Hallam-Baker, "DomainKeys Identified Mail (DKIM) Service Overview", June 2009.
- [RFC5863] Hansen, T., Siegel, H., Hallam-Baker, P., and D. Crocker, "DomainKeys Identified Mail (DKIM): Development,

[Appendix A](#). MUA Considerations {rfc4871bis-02 D}

When a DKIM signature is verified, the processing system sometimes makes the result available to the recipient user's MUA. How to present this information to the user in a way that helps them is a matter of continuing human factors usability research. The tendency is to have the MUA highlight the SDID, in an attempt to show the user the identity that is claiming responsibility for the message. An MUA might do this with visual cues such as graphics, or it might include the address in an alternate view, or it might even rewrite the original From address using the verified information. Some MUAs might indicate which header fields were protected by the validated DKIM signature. This could be done with a positive indication on the signed header fields, with a negative indication on the unsigned header fields, by visually hiding the unsigned header fields, or some combination of these. If an MUA uses visual indications for signed header fields, the MUA probably needs to be careful not to display unsigned header fields in a way that might be construed by the end user as having been signed. If the message has an l= tag whose value does not extend to the end of the message, the MUA might also hide or mark the portion of the message body that was not signed.

The aforementioned information is not intended to be exhaustive. The MUA may choose to highlight, accentuate, hide, or otherwise display any other information that may, in the opinion of the MUA author, be deemed important to the end user.

[Appendix B](#). End-to-End Scenario Example {rfc4871bis-02 A}

This section shows the complete flow of an email from submission to final delivery, demonstrating how the various components fit together. The key used in this example is shown in [[I-D.Doseta](#)], "Creating a Public Key".

[B.1](#). The User Composes an Email

From: Joe SixPack <joe@football.example.com>
To: Suzie Q <suzie@shopping.example.net>
Subject: Is dinner ready?

Date: Fri, 11 Jul 2003 21:00:37 -0700 (PDT)
Message-ID: <20030712040037.46341.5F8J@football.example.com>

Hi.

We lost the game. Are you hungry yet?

Joe.

Figure 1: The User Composes an Email

[B.2.](#) The Email is Signed

This email is signed by the example.com outbound email server and now looks like this:

DOSETA&nbhy;Signature: v=1; a=rsa-sha256; s=brisbane; d=example.com;
c=simple/simple; q=dns/txt; i=joe@football.example.com;
h=Received : From : To : Subject : Date : Message-ID;

bh=2jUSOH9NhtVGCQWnr9BrIAPreKQj06Sn7XIkfJV0zv8=;
b=AuUoFEfDxTDkHLLXSZEpZj79LICEps6eda7W3deTVF0k4yAUoqOB
4nujc7YopdG5dWLSdNg6xNAZp0Pr+kHxt1IrE+NahM6L/LbvaHut
KVdkLLkpVaVVQPzeRDI009S02Il5Lu7rDNH6mZckBdrIx0orEtZV
4bmp/YzhwvcubU4=;
Received: from client1.football.example.com [192.0.2.1]
by submitserver.example.com with SUBMISSION;
Fri, 11 Jul 2003 21:01:54 -0700 (PDT)
From: Joe SixPack <joe@football.example.com>
To: Suzie Q <suzie@shopping.example.net>
Subject: Is dinner ready?
Date: Fri, 11 Jul 2003 21:00:37 -0700 (PDT)
Message-ID: <20030712040037.46341.5F8J@football.example.com>

Hi.

We lost the game. Are you hungry yet?

Joe.

The Email is Signed

The signing email server requires access to the private key associated with the "brisbane" selector to generate this signature.

[B.3.](#) The Email Signature is Verified

The signature is normally verified by an inbound SMTP server or possibly the final delivery agent. However, intervening MTAs can also perform this verification if they choose to do so. The verification process uses the domain "example.com" extracted from the "d=" tag and the selector "brisbane" from the "s=" tag in the DOSETA-Signature header field to form the DNS DOSETA query for: brisbane._domainkey.example.com

Signature verification starts with the physically last Received header field, the From header field, and so forth, in the order listed in the "h=" tag. Verification follows with a single CRLF followed by the body (starting with "Hi."). The email is canonically prepared for verifying with the "simple" method. The result of the query and subsequent verification of the signature is stored (in this example) in the X-Authentication-Results header field line. After successful verification, the email looks like this:

X-Authentication-Results: shopping.example.net
header.from=joe@football.example.com; DOSETA=pass
Received: from mout23.football.example.com (192.168.1.1)
by shopping.example.net with SMTP;
Fri, 11 Jul 2003 21:01:59 -0700 (PDT)
DOSETA&nbhy;Signature: v=1; a=rsa-sha256; s=brisbane; d=example.com;
c=simple/simple; q=dns/txt; i=joe@football.example.com;
h=Received : From : To : Subject : Date : Message-ID;
bh=2jUSOH9NhtVGCQWnr9BrIAPreKQj06Sn7XIkfJV0zv8=;
b=AuUoFEfDxTDkHLLXSZEj79LICEps6eda7W3deTVF0k4yAUoq0B
4nujc7YopdG5dWLSdNg6xNAZpOPr+kHxt1IrE+NahM6L/LbvaHut
KVdkLLkpVaVVQPzeRDI009S02Il5Lu7rDNH6mZckBdrIx0orEtZV
4bmp/YzhwvcubU4=;
Received: from client1.football.example.com [192.0.2.1]
by submitserver.example.com with SUBMISSION;
Fri, 11 Jul 2003 21:01:54 -0700 (PDT)
From: Joe SixPack <joe@football.example.com>
To: Suzie Q <suzie@shopping.example.net>
Subject: Is dinner ready?
Date: Fri, 11 Jul 2003 21:00:37 -0700 (PDT)
Message-ID: <20030712040037.46341.5F8J@football.example.com>

Hi.

We lost the game. Are you hungry yet?

Joe.

Successful verification

[Appendix C](#). Types of Use {rfc4871bis-02 B}

DOSETA signing and validating can be used in different ways, for different operational scenarios. This Appendix discusses some common examples.

NOTE: Descriptions in this Appendix are for informational purposes only. They describe various ways that DOSETA can be used, given particular constraints and needs. In no case are these examples intended to be taken as providing explanation or guidance concerning DOSETA specification details, when creating an implementation.

Internet-Draft

RFC4871bis

January 2011

[C.1.](#) Alternate Submission Scenarios

In the most simple scenario, a user's MUA, MSA, and Internet (boundary) MTA are all within the same administrative environment, using the same domain name. Therefore, all of the components involved in submission and initial transfer are related. However, it is common for two or more of the components to be under independent administrative control. This creates challenges for choosing and administering the domain name to use for signing, and for its relationship to common email identity Header fields.

[C.1.1.](#) Delegated Business Functions

Some organizations assign specific business functions to discrete groups, inside or outside the organization. The goal, then, is to authorize that group to sign some mail, but to constrain what signatures they can generate. DOSETA selectors (the "s=" signature tag) facilitate this kind of restricted authorization. Examples of these outsourced business functions are legitimate email marketing providers and corporate benefits providers.

Here, the delegated group needs to be able to send messages that are signed, using the email domain of the client company. At the same time, the client often is reluctant to register a key for the provider that grants the ability to send messages for arbitrary addresses in the domain.

There are multiple ways to administer these usage scenarios. In one case, the client organization provides all of the public query service (for example, DNS) administration, and in another it uses DNS delegation to enable all ongoing administration of the DOSETA key record by the delegated group.

If the client organization retains responsibility for all of the DNS administration, the outsourcing company can generate a key pair, supplying the public key to the client company, which then registers it in the query service, using a unique selector. The client company retains control over the use of the delegated key because it retains the ability to revoke the key at any time.

If the client wants the delegated group to do the DNS administration, it can have the domain name that is specified with the selector point

to the provider's DNS server. The provider then creates and maintains all of the DKIM signature information for that selector. Hence, the client cannot provide constraints on the <local-part> of addresses that get signed, but it can revoke the provider's signing rights by removing the DNS delegation record.

[C.1.2.](#) PDAs and Similar Devices

PDAs demonstrate the need for using multiple keys per domain. Suppose that John Doe wanted to be able to send messages using his corporate email address, `jdoe@example.com`, and his email device did not have the ability to make a Virtual Private Network (VPN) connection to the corporate network, either because the device is limited or because there are restrictions enforced by his Internet access provider. If the device was equipped with a private key registered for `jdoe@example.com` by the administrator of the `example.com` domain, and appropriate software to sign messages, John could sign the message on the device itself before transmission through the outgoing network of the access service provider.

[C.1.3.](#) Roaming Users

Roaming users often find themselves in circumstances where it is convenient or necessary to use an SMTP server other than their home server; examples are conferences and many hotels. In such circumstances, a signature that is added by the submission service will use an identity that is different from the user's home system.

Ideally, roaming users would connect back to their home server using either a VPN or a SUBMISSION server running with SMTP AUTHentication on port 587. If the signing can be performed on the roaming user's laptop, then they can sign before submission, although the risk of further modification is high. If neither of these are possible, these roaming users will not be able to send mail signed using their own domain key.

[C.1.4.](#) Independent (Kiosk) Message Submission

Stand-alone services, such as walk-up kiosks and web-based information services, have no enduring email service relationship with the user, but users occasionally request that mail be sent on

their behalf. For example, a website providing news often allows the reader to forward a copy of the article to a friend. This is typically done using the reader's own email address, to indicate who the author is. This is sometimes referred to as the "Evite problem", named after the website of the same name that allows a user to send invitations to friends.

A common way this is handled is to continue to put the reader's email address in the From header field of the message, but put an address owned by the email posting site into the Sender header field. The posting site can then sign the message, using the domain that is in the Sender field. This provides useful information to the receiving email site, which is able to correlate the signing domain with the

initial submission email role.

Receiving sites often wish to provide their end users with information about mail that is mediated in this fashion. Although the real efficacy of different approaches is a subject for human factors usability research, one technique that is used is for the verifying system to rewrite the From header field, to indicate the address that was verified. For example: From: John Doe via news@news-site.com <jdoe@example.com>. (Note that such rewriting will break a signature, unless it is done after the verification pass is complete.)

C.2. Alternate Delivery Scenarios

Email is often received at a mailbox that has an address different from the one used during initial submission. In these cases, an intermediary mechanism operates at the address originally used and it then passes the message on to the final destination. This mediation process presents some challenges for DKIM signatures.

C.2.1. Affinity Addresses

"Affinity addresses" allow a user to have an email address that remains stable, even as the user moves among different email providers. They are typically associated with college alumni associations, professional organizations, and recreational organizations with which they expect to have a long-term relationship. These domains usually provide forwarding of incoming

email, and they often have an associated Web application that authenticates the user and allows the forwarding address to be changed. However, these services usually depend on users sending outgoing messages through their own service providers' MTAs. Hence, mail that is signed with the domain of the affinity address is not signed by an entity that is administered by the organization owning that domain.

With DOSETA, affinity domains could use the Web application to allow users to register per-user keys to be used to sign messages on behalf of their affinity address. The user would take away the secret half of the key pair for signing, and the affinity domain would publish the public half in DNS for access by verifiers.

This is another application that takes advantage of user-level keying, and domains used for affinity addresses would typically have a very large number of user-level keys. Alternatively, the affinity domain could handle outgoing mail, operating a mail submission agent that authenticates users before accepting and signing messages for them. This is of course dependent on the user's service provider not

blocking the relevant TCP ports used for mail submission.

[C.2.2.](#) Simple Address Aliasing (.forward)

In some cases a recipient is allowed to configure an email address to cause automatic redirection of email messages from the original address to another, such as through the use of a Unix .forward file. In this case, messages are typically redirected by the mail handling service of the recipient's domain, without modification, except for the addition of a Received header field to the message and a change in the envelope recipient address. In this case, the recipient at the final address' mailbox is likely to be able to verify the original signature since the signed content has not changed, and DOSETA is able to validate the message signature.

[C.2.3.](#) Mailing Lists and Re-Posters

There is a wide range of behaviors in services that take delivery of a message and then resubmit it. A primary example is with mailing lists (collectively called "forwarders" below), ranging from those that make no modification to the message itself, other than to add a

Received header field and change the envelope information, to those that add Header fields, change the Subject header field, add content to the body (typically at the end), or reformat the body in some manner. The simple ones produce messages that are quite similar to the automated alias services. More elaborate systems essentially create a new message.

A Forwarder that does not modify the body or signed Header fields of a message is likely to maintain the validity of the existing signature. It also could choose to add its own signature to the message.

Forwarders which modify a message in a way that could make an existing signature invalid are particularly good candidates for adding their own signatures (for example, mailing-list-name@example.net). Since (re-)signing is taking responsibility for the data, these signing forwarders are likely to be selective, and forward or re-sign a message only if it is received with a valid signature or if they have some other basis for knowing that the message is not spoofed.

A common practice among systems that are primarily redistributors of mail is to add a Sender header field to the message, to identify the address being used to sign the message. This practice will remove any preexisting Sender header field as required by [\[RFC5322\]](#). The forwarder applies a new DOSTA-Signature header field with the signature, public key, and related information of the forwarder.

[Appendix D](#). Acknowledgements

The previous IETF version of DKIM [\[RFC4871\]](#) was edited by: Eric Allman, Jon Callas, Mark Delany, Miles Libbey, Jim Fenton and Michael Thomas.

That specification was the result of an extended, collaborative effort, including participation by: Russ Allbery, Edwin Aoki, Claus Assmann, Steve Atkins, Rob Austein, Fred Baker, Mark Baugher, Steve Bellovin, Nathaniel Borenstein, Dave Crocker, Michael Cudahy, Dennis Dayman, Jutta Degener, Frank Ellermann, Patrik Faeltsstroem, Mark Fanto, Stephen Farrell, Duncan Findlay, Elliot Gillum, Olafur Gu[eth]mundsson, Phillip Hallam-Baker, Tony Hansen, Sam Hartman, Arvel Hathcock, Amir Herzberg, Paul Hoffman, Russ Housley, Craig

Hughes, Cullen Jennings, Don Johnsen, Harry Katz, Murray S. Kucherawy, Barry Leiba, John Levine, Charles Lindsey, Simon Longsdale, David Margrave, Justin Mason, David Mayne, Thierry Moreau, Steve Murphy, Russell Nelson, Dave Oran, Doug Otis, Shamim Pirzada, Juan Altmayer Pizzorno, Sanjay Pol, Blake Ramsdell, Christian Renaud, Scott Renfro, Neil Rerup, Eric Rescorla, Dave Rossetti, Hector Santos, Jim Schaad, the Spamhaus.org team, Malte S. Stretz, Robert Sanders, Rand Wacker, Sam Weiler, and Dan Wing.

The earlier DomainKeys was a primary source from which DKIM was derived. Further information about DomainKeys is at [[RFC4870](#)].

Authors' Addresses

D. Crocker (editor)
Brandenburg InternetWorking
675 Spruce Dr.
Sunnyvale
USA

Phone: +1.408.246.8253
Email: dcrocker@bbiw.net
URI: <http://bbiw.net>

M. Kucherawy (editor)
Cloudmark
128 King St., 2nd Floor
San Francisco, CA 94107
USA

Email: msk@cloudmark.com