A new cryptographic signature method for DKIM

draft-ietf-dcrup-dkim-crypto-14

Abstract

This document adds a new signing algorithm, ed25519-sha256, to DKIM
[RFC6376]. DKIM verifiers are required to implement this algorithm.

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 December 23, 2018.

Copyright Notice

Copyright (c) 2018 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.
1. Introduction

Discussion Venue: Discussion about this draft is directed to the dcrup8ietf.org [1] mailing list.

DKIM [RFC6376] signs e-mail messages, by creating hashes of the message headers and body and signing the header hash with a digital signature. Message recipients fetch the signature verification key from the DNS. The defining documents specify a single signing algorithm, RSA [RFC3447].

This document adds a new stronger signing algorithm, Edwards-Curve Digital Signature Algorithm using the Curve25519 curve (ed25519), which has much shorter keys than RSA for similar levels of security.

2. Conventions Used in This Document

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], and only when, they appear in all capitals, as shown here.
Syntax descriptions use Augmented BNF (ABNF) [RFC5234]. The ABNF tokens sig-a-tag-k and key-k-tag-type are imported from [RFC6376].

3. Ed25519-SHA256 Signing Algorithm

The ed25519-sha256 signing algorithm computes a message hash as defined in section 3 of [RFC6376] using SHA-256 [FIPS-180-4-2015] as the hash-alg, and signs it with the PureEdDSA variant Ed25519, as defined in in RFC 8032 section 5.1 [RFC8032]. Example keys and signatures in Appendix A below are based on the test vectors in RFC 8032 section 7.1 [RFC8032].

The DNS record for the verification public key has a "k=ed25519" tag to indicate that the key is an Ed25519 rather than RSA key.

This is an additional DKIM signature algorithm added to Section 3.3 of [RFC6376] as envisioned in Section 3.3.4 of [RFC6376].

Note: since Ed25519 public keys are 256 bits long, the base64 encoded key is only 44 octets, so DNS key record data will generally fit in a single 255 byte TXT string, and will work even with DNS provisioning software that doesn’t handle multi-string TXT records.

4. Signature and key syntax

The syntax of DKIM signatures and DKIM keys are updated as follows.

4.1. Signature syntax

The syntax of DKIM algorithm tags in section 3.5 of [RFC6376] is updated by adding this rule to the existing rule for sig-a-tag-k:

ABNF:

    sig-a-tag-k =/ "ed25519"

4.2. Key syntax

The syntax of DKIM key tags in section 3.6.1 of [RFC6376] is updated by adding this rule to the existing rule for key-k-tag-type:

ABNF:

    key-k-tag-type =/ "ed25519"

The p= value in the key record is the ed25519 public key encoded in base64. Since the key is 256 bits long, the base64 text is 44 octets.
long. See Appendix A.2 for a sample key record using the public key in [RFC8032] Section 7.1, Test 1.

5. Key and algorithm choice and strength

Section 3.3 of [RFC6376] describes DKIM’s hash and signature algorithms. It is updated as follows:

Signers SHOULD implement and verifiers MUST implement the ed25519-sha256 algorithm.

6. Transition Considerations

For backward compatibility, signers can add multiple signatures that use old and new signing algorithms. Since there can only be a single key record in the DNS for each selector, the signatures have to use different selectors, although they can use the same d= and i= identifiers.

The example message in Appendix A has two signatures with the same d= and i= identifiers but different a= algorithms and s= selectors.

7. Security Considerations

All of the security advice in [RFC6376] continues to apply except that the security advice about ED25519 in Section 8 of [RFC8032] supplants the advice about RSA threats.

8. IANA Considerations

IANA is requested to update registries as follows.

8.1. DKIM Key Type registry

The following value is added to the DKIM Key Type Registry

| +-------------------------+-----------+--------+
| | TYPE | REFERENCE | STATUS |
| +-------------------------+-----------+--------+
| | ed25519 | [RFC8032] | active |

Table 1: DKIM Key Type Registry Added Values
9. References

9.1. Normative References


9.2. Informative References


9.3. URIs

[1] mailto:dcrup@ietf.org
Appendix A. Example of a signed message

This is a small message with both rsa-sha256 and ed25519-sha256 DKIM signatures. The signatures are independent of each other, so either signature would be valid if the other were not present.

A.1. Secret keys

Ed25519 secret key in base64. This is the secret key from [RFC8032] section 7.1 test 1, converted from hex to base64.

nWGxne/9WmC6hEr0kuwswxERJxWL7MmkZCdUsAxyuf2A=

RSA secret key in PEM format.

-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDkHlOQoBtzWRiGs5V6npP3idY6wk08a5qhdR6wy5b0dk2jLQiY/J16YiQvxqbyZCNb39W1y33UtACDfzwQ/BC/e/8uBsCR+yz1Lxj+PL61HvqMNkrM3rG4hstT5QjvH09PzoxZyVVLzBf02EeC3Ip3G+2kryOTIKT+1/K4w3QIDAQAB AoGAH0cxOhFZDgzWXdhNnAJDw5s4roOXN40hjiXa8W7Y3rhX3FJqjmJSpuC89NvQm6SVbaLA4SG5mLMueHlh4KXXfEpELEInp9s304YfLQiQbRgE77m5xKjvyQoZ2ZeZhorimoaChRLit47iuWxxzSiRmv4c+j70G1WdxNnxe4UoECQDzJ8B/JOsU8W7Ry6enGVjzkWF732CoWFZwiiFicudrBFoy63Qwcowp0CazKtvZGMN1PWnC7x/60BGc2s0ga2xAkEA8C7IpFm1/1fTRQvjl0/dDmzp243044ZNyjx/+OPNO0WcbXIGxyWvmZbXrI0WoSALJTjExEgrAEgnXsuk7QJBAL151CsYM6hMx073gnfNayNgPxdWFV6ZULhKyV7HSVYFOghYOHeYe9gaMtijYo0zGN+L3AAtNP9huqKlWzECQElalicEtvoleq+J6MgqroQ7Aa7falZ448ccsSFYEPD60FxiiO199ye9IHZKhKfCst07DUw/hZ2Cl4N5JrgUCQQCyKveNvjzkkd8HjYs0SwM0fPjk16//5qDZ2UiDgnOeuEzxBDAr518ZvFBfR41ln3W4Y3yCDgQ1LcETrS+zYcL
-----END RSA PRIVATE KEY-----

A.2. Public key DNS records

The public key p= value in the first record is the public key from [RFC8032] section 7.1 test 1, converted from hex to base64.

brisbane._domainkey.football.example.com. IN TXT ("v=DKIM1; k=ed25519; p=1qYXKCrFvS/77yWfHoq7vHapiMr1w1aaPcHUr0e=")
test._domainkey.football.example.com. IN TXT ("v=DKIM1; k=rsa; p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDkHlOQoBtzWR"+"iGs5V6npP3idY6wk08a5qhdR6wy5bd0Kb2jLQiY/J16YI10qvxqbyZCNb39W1y33UtAC"+"DfzwQ/BC/e/8uBzCR+yz1Lxj+PL61HvqMKrM3rG4hstT5QjvH09PzoxZyVVLzBF02EeC3"+"Ip3G+2kryOTIKT+1/K4w3QIDAQAB")
A.3. Signed Message

The text in each line of the message starts at the first position except for the continuation lines on the DKIM-Signature headers which start with a single space. A blank line follows the "Joe." line.

DKIM-Signature: v=1; a=ed25519-sha256; c=relaxed/relaxed;
d=football.example.com; i=@football.example.com;
g=dns/txt; s=brisbane; t=1528637909; h=from : to :
subject : date : message-id : from : subject : date;
bh=2jUSOH9NhtVGCQWNr9BrIAPreKQj06Sn7XIkfJVOzv8=;
b=AgCrinpQOoIfuHNQIbq4pg9h9kyIK3AQUdt9OdqQehSwIug4D11BusFa3bT3FY5osU7ZbnKElqeXdp1QIDw==

DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
d=football.example.com; i=@football.example.com;
g=dns/txt; s=test; t=1528637909; h=from : to :
subject : date : message-id : from : subject : date;
bh=2jUSOH9NhtVGCQWNr9BrIAPreKQj06Sn7XIkfJVOzv8=;
b=F45dVWDfMbQDGHFlXUNB2HKfbcLryhDxPpEL8Gwpser01eIixNTe3DhCV1UrsjV4BwvCvO6F+FF3zo9Rpo1tF0eS9mYQ7NtGdaGSggeefOsk2JzdA+L10TeYt9BgDfQNZtKdN1W0//KgIqXP70dEFE4LjFYncUx2Q4FADY+8=
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.

Appendix B. Change log

13 to 14 Editorial nits.
12 to 13 Made example even less wrong.
11 to 12 Made example less wrong.
10 to 11 New example with both signatures, minor nits.
09 to 10 Improve abstract, minor nits.
08 to 09 Specify sha-256 for the extremely literal minded. Take out the prehash stuff. Add example.
07 to 08  Specify base64 key records. Style edits per Dave C.

06 to 07:  Remove RSA fingerprints. Change Pure to hashed eddsa.

05 to 06:  Editorial changes only.

04 to 05:  Remove deprecation cruft and inconsistent key advice. Fix p= and k= text.

03 to 04:  Change eddsa to ed25519. Add Martin’s key regeneration issue. Remove hashed ed25519 keys. Fix typos and clarify text. Move syntax updates to separate section. Take out SHA-1 stuff.

01 to 02:  Clarify EdDSA algorithm is ed25519 with Pure version of the signing. Make references to tags and fields consistent.

Author’s Address

John Levine
Taughannock Networks
PO Box 727
Trumansburg, NY  14886

Phone: +883.5100.01196712
Email: standards@taugh.com
Abstract

DomainKeys Identified Mail (DKIM) uses digital signature to associate a message with a given sending domain. Currently, there is only one cryptography algorithm defined for use with DKIM (RSA). This document defines four new elliptic curve cryptography algorithms for use with DKIM. This will allow for algorithm agility if a weakness is found in RSA, and allows for smaller key length to provide the same digital signature strength.

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 23, 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
1. Introduction

DomainKeys Identified Mail (DKIM) [RFC6376] uses digital signatures to associate a sending domain with a given message. Each DKIM signed email message as a digital signature in its header, that can be validated by a receiver by obtaining the appropriate public key stored in the DNS. Currently, DKIM has only one cryptographic algorithm defined for use (RSA) and two digital signature algorithms (RSA/SHA-1 and RSA/SHA-256). In the past, 1024-bit RSA keys were common, equating to (roughly) a security key strength of 80 bits [NIST.800-57.2016]. Today, a minimum of 112 bits is recommended, which equates to 2048 bit RSA keys.

The public portion of 2048 bit RSA keys are still small enough to fit into a DNS TXT RR without issues in performance. The encoded public key is too large to fit into the maximum allowed characters in a single string, but a DNS TXT RR allows for multiple strings, so the key can be broken into "chunks" to allow it to be served. However, some code components may not correctly handle TXT RRs with multiple strings which will result in errors in validation.

Elliptic Curve Cryptography (ECC) has shown to have the same (roughly) equivalent key strength with smaller sizes. A 224 to 255 bit ECDSA key has (roughly) the same key strength as a 2048 bit RSA key (112 bits of strength). This means smaller keys can be used to achieve the same DKIM security strength, as well as being easier to manage in the DNS.
Having additional digital signature algorithms defined for use with DKIM also permits algorithm agility. If a weakness is discovered in one digital signature algorithm, email senders can quickly migrate to another algorithm without waiting for a standards action and subsequent software update.

This document defines a ECDSA as a new algorithms for DKIM. This document also defines a new hash algorithm for use with DKIM signatures. This document updates the IANA registry with new values for the algorithms. This document does not change the DKIM key or signature formats, but only defines new algorithm values using those formats.

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].

2. Defining New ECC algorithms for Use with DKIM

This document defines a new digital signature algorithm for use with DKIM:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECDSA P-256</td>
<td>ecdsa256</td>
</tr>
</tbody>
</table>

For ECDSA, the SHA-1 hash algorithm MUST NOT be used.

3. Changes to ABNF Definitions of DKIM Keys and Signatures

The original definition of DKIM signatues and keys are defined in [RFC6376]. The following are changes to the definition to include the new digital signature algorithm and secure hash algorithm.

3.1. Changes to DKIM Key Definition

The original definition of the textual representation of DKIM keys is found in section 3.6.1 of [RFC6376]. The only changes to the definition is below. The entire key:tag definition is included for clarity. All other tags:value pairs are unchanged. References to the definitions below have also been updated to reflect the current state of the art.

h= Acceptable hash algorithms (plain-text; OPTIONAL, defaults to "sha256"). A colon-separated list of hash algorithms that might
be used. Unrecognized algorithms MUST be ignored. Refer to [RFC6376]Section 3.3 for a discussion of the hash algorithms implemented by Signers and Verifiers. The set of algorithms listed in this tag in each record is an operational choice made by the Signer.

**ABNF:**

```plaintext
code�.key-h-tag = %x68 [FWS] "=" [FWS] key-h-tag-alg
   *( [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
```

**k=** Key type (plain-text; OPTIONAL, default is "rsa"). Signers and Verifiers MUST support the "rsa" key type. The "rsa" key type indicates that an ASN.1 DER-encoded [UTI.X680.2002] RSAPublicKey (see [RFC8017], Sections 3.1 and A.1.1) is being used in the "p=" tag. The "ecdsa256" key type indicates an ASN.1 DER-encoded [UTI.X680.2002] PublicKey (see [RFC5480], Section 2.2) is being used in the "p=" tag. (Note: the "p=" tag further encodes the value using the base64 algorithm.) Unrecognized key types MUST be ignored.

**ABNF:**

```plaintext
code�.key-k-tag = %x76 [FWS] "=" [FWS] key-k-tag-type
key-k-tag-type = "rsa" / "ecdsa256" / x-key-k-tag-type
x-key-k-tag-type = hyphenated-word ; for future extension
```

### 3.2. Changes to DKIM Signature Definition

The original definition of the textual representation of DKIM signatures is found in section 3.5 of [RFC6376]. The only changes to the definition is below. The entire key:tag definition is included for clarity. All other tags:value pairs are unchanged. References to the definitions below have also been updated to reflect the current state of the art.

**a=** The algorithm used to generate the signature (plain-text; REQUIRED). Verifiers MUST support "rsa-sha1" and "rsa-sha256" and SHOULD support "ecdsa256-sha256"; Signers MUST NOT use "sha1" with "ecdsa256". See [RFC6376] Section 3.3 for a description of RSA and [FIPS.186-4.2013] Section 6 for a brief description of ECDSA.
ABNF:

```
sig-a-tag     = %x61 [FWS] "=" [FWS] sig-a-tag-alg
sig-a-tag-alg = sig-a-tag-k "-" sig-a-tag-h
sig-a-tag-k   = "rsa" / "ecdsa256" / x-sig-a-tag-k
sig-a-tag-h   = "sha1" / "sha256" / x-sig-a-tag-h
x-sig-a-tag-k = ALPHA *(ALPHA / DIGIT)
    ; for later extension
x-sig-a-tag-h = ALPHA *(ALPHA / DIGIT)
    ; for later extension
```

4. Sender Considerations

New algorithms for an established protocols take some time to gain wide deployment. There will be a period of time where new algorithms are in operation side by side with older algorithms. There will also be a sizable percentage of DKIM validators that will not understand new algorithms until they are upgraded. This will lead to a period of time where multiple DKIM signature algorithms are in use for a sender. Email administrators MAY want to also sign with RSA/SHA-1 or RSA/SHA-256 for a period of time. This period of time is difficult to measure, but DMARC [RFC7960] aggregate reports could provide a view on DKIM validation rates by receivers.

5. Receiver Considerations

These requirements are for DKIM verifiers (as defined in [RFC6376]). These entities would be the consumers of any end-to-end email security policy and would be the entity responsible for validating DKIM signatures.

DKIM verifiers claiming conformance to this document MUST implement all of the above cryptographic algorithms.

This document does NOT change the behavior of the core DKIM specification in that verifiers MUST ignore unknown algorithms in DKIM signatures.

6. Security Considerations

This document defines the use of new elliptic curve cryptographic algorithms for use with DomainKey Identified Mail (DKIM). This document is not a discussion of the relative strengths or weaknesses of these algorithms, but only defines their use.

There is a risk for mail receivers that do not understand or implement the new algorithms. Attackers could modify or spoof messages from sending zones using one of the newly defined algorithms.
and it would not be detectable as an attack by ECC-ignorant receivers. Likewise, ECC-ignorant receivers may mark valid DKIM signed email messages as invalid due to unknown algorithms.

7. IANA Considerations

This draft defines the use of a new algorithm for DKIM. This draft updates the "DKIM Key Tag" registry to include the following new value:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mnemonic</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECDSA P-256</td>
<td>ecdsa256</td>
<td>This document</td>
</tr>
</tbody>
</table>

The current DKIM Key Tag registry is located at https://www.iana.org/assignments/dkim-parameters/dkim-parameters.xhtml#dkim-parameters-6

8. References

8.1. Normative References


8.2. Informative References


Author’s Address

Scott Rose
NIST
100 Bureau Dr.
Gaithersburg, MD  20899
USA

Phone: +1 301-975-8439
Email: scott.rose@nist.gov
Abstract

The cryptographic algorithm and key size requirements included when DKIM was designed in the last decade are functionally obsolete and in need of immediate revision. This document updates DKIM requirements to those minimally suitable for operation with currently specified algorithms.

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 May 7, 2018.

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. Discussion Venue ............................................. 2
2. Introduction .................................................. 2
3. Conventions Used in This Document ............................ 3
4. DKIM Signing and Verification Algorithms ........................ 3
   4.1. DKIM Signing and Verification Algorithms .................... 3
   4.2. Key Sizes ............................................... 3
5. Security Considerations ......................................... 4
6. IANA Considerations ........................................... 4
7. References ................................................... 4
   7.1. Normative References ....................................... 4
   7.2. Informative References ..................................... 4
   7.3. URIs .................................................. 5
Appendix A. Acknowledgements ...................................... 5
Author’s Address ................................................ 5

1. Discussion Venue

RFC EDITOR: Please remove this section before publication.

Discussion about this draft is directed to the dcrup@ietf.org [1]
mailing list.

2. Introduction

DKIM [RFC6376] signs e-mail messages, by creating hashes of the
message headers and content and signing the header hash with a
digital signature. Message recipients fetch the signature
verification key from the DNS where it is stored in a TXT record.

The defining documents specify a single signing algorithm, RSA
[RFC8017], and recommends key sizes of 1024 to 2048 bits (but require
verification of 512 bit keys). As discussed in US-CERT VU#268267
[VULNOTE], the operational community has recognized that shorter keys
compromise the effectiveness of DKIM. While 1024 bit signatures are
common, stronger signatures are not. Widely used DNS configuration
software places a practical limit on key sizes, because the software
only handles a single 256 octet string in a TXT record, and RSA keys
significantly longer than 1024 bits don’t fit in 256 octets.

Due to the recognized weakness of the sha1 hash algorithm, see
[RFC6194], and the wide availability of the sha256 hash algorithm (it
has been required part of DKIM [RFC6376] since it was originally
standardized in 2007), the sha1 hash algorithm MUST NOT be used.
This is being done now to allow the operational community time to fully shift to sha256 in advance of any sha1 related crisis.

3. Conventions Used in This Document

The capitalized 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].

4. DKIM Signing and Verification Algorithms

Section 4.1 updates [RFC6376] Section 3.3.

Section 4.2 updates [RFC6376] Section 3.3.3.

The algorithm described in [RFC6376] Section 3.3.1 is now historic and no longer used by DKIM.

[RFC6376] Sections 3.3.2 and 3.3.4 are not affected.

4.1. DKIM Signing and Verification Algorithms

DKIM supports multiple digital signature algorithms. Two algorithms are defined by this specification at this time: rsa-sha1 and rsa-sha256. Signers MUST sign using rsa-sha256. Verifiers MUST be able to verify using rsa-sha256. rsa-sha1 MUST NOT be used for signing or verifying.

DKIM signatures identified as having been signed with historic algorithms (currently rsa-sha1) have permanently failed evaluation as discussed in [RFC6376] Section 3.9.

4.2. Key Sizes

Selecting appropriate key sizes is a trade-off between cost, performance, and risk. Since short RSA keys more easily succumb to off-line attacks, Signers MUST use RSA keys of at least 1024 bits for all keys. Signers SHOULD use RSA keys of at least 2048 bits.

Verifiers MUST be able to validate signatures with keys ranging from 1024 bits to 4096 bits, and they MAY be able to validate signatures with larger keys. Verifier policies can use the length of the signing key as one metric for determining whether a signature is acceptable. Verifiers MUST NOT consider signatures using RSA keys of less than 1024 bits as valid signatures.
DKIM signatures with insufficient key sizes (currently rsa-sha256 with less than 1024 bits) have permanently failed evaluation as discussed in [RFC6376] Section 3.9.

5. Security Considerations

This document does not change the Security Considerations of [RFC6376]. It reduces the risk of signature compromise due to weak cryptography. The SHA-1 risks discussed in [RFC6194] Section 3 are resolved due to rsa-sha1 no longer being used by DKIM.

6. IANA Considerations

IANA is requested to update the "sha1" registration in the "DKIM Hash Algorithms" as follows:

| TYPE | REFERENCE | STATUS   |
|------|-----------+----------|
| sha1 | [RFC6376] | historic |

Table 1: DKIM Hash Algorithms Changed Value

7. References

7.1. Normative References


7.2. Informative References
Appendix A. Acknowledgements

The author wishes to acknowledge the following for their review and comment on this proposal: Kurt Andersen, Murray S. Kucherawy, Martin Thomson, John Levine, Russ Housley, and Jim Fenton.

Thanks to John Levine for his DCRUP work that was the source for much of the introductory material in this draft.

Author’s Address

Scott Kitterman
Kitterman Technical Services
3611 Scheel Dr
Ellicott City, MD 21042

Phone: +1 301 325-5475
Email: scott@kitterman.com