

Network Working Group
Internet Draft
Intended status: Informational
Expires: WRONG SYNTAX FOR MONTH

T. Hansen
AT&T Laboratories
P. Hallam-Baker
VeriSign Inc.
D. Crocker
Brandenburg InternetWorking
2008

DomainKeys Identified Mail (DKIM) Development, Deployment and Operations

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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".

The list of current Internet-Drafts can be accessed at <<http://www.ietf.org/ietf/lid-abstracts.txt>>.

The list of Internet-Draft Shadow Directories can be accessed at <<http://www.ietf.org/shadow.html>>.

This Internet-Draft will expire in WRONG SYNTAX FOR MONTH.

Copyright Notice

Copyright © The IETF Trust (2008). All Rights Reserved.

Abstract

DomainKeys Identified Mail (DKIM) associates a "responsible" identity with a message and provides a means of verifying that the association is legitimate. [RFC4871] DKIM defines a domain-level authentication framework for email using public-key cryptography and key server technology. This permits verifying the source or intermediary for a message, as well as the contents of messages. The ultimate goal of this framework is to permit a signing domain to assert responsibility for sending a message, thus proving and protecting the identity associated with the message and the integrity of the messages itself, while retaining the functionality of Internet email as it is known today. Such protection of email identity may assist in the global control of "spam" and "phishing". This document provides implementation, deployment, operational and migration considerations for DKIM.

Table of Contents

1 Introduction	3
2 Development	4
2.1 DKIM Signing Software	4
2.2 General Coding Criteria for Cryptographic Applications	4
2.3 Mailing List Manager Software	4
2.4 Email Infrastructure Agents	5
2.5 Filtering Software	5
2.6 DNS Server Software	5
3 Deployment	6
3.1 Signing	6
3.2 Verifying	6
3.3 Key Management Deployment	7
3.4 Mailing Lists	8
3.5 Mail User Agent	8
3.6 Transition Strategy	9
3.7 Migrating from DomainKeys	10
4 On-going Operations	11
4.1 DNS Signature Record Installation Considerations	11
4.2 Private Key Management	12
5 Example Uses	13
5.1 Protection of Internal Mail	13
5.2 Recipient-based Assessments	13
5.3 DKIM Support in the Client	13
5.4 Per user signatures	13
6 Security Considerations	15
7 IANA Considerations	16
8 Acknowledgements	17
9 Informative References	18
Authors' Addresses	19
Intellectual Property and Copyright Statements	20

1. Introduction

There are many areas to be considered when deploying DomainKeys Identified Mail (DKIM). This document provides practical tips for: those who are developing DKIM software, mailing list managers, filtering strategies based on the output from DKIM verification, and DNS servers; those who are deploying DKIM software, keys, mailing list software, and migrating from DomainKeys; and those who are responsible for the on-going operations of an email infrastructure that has deployed DKIM.

2. Development

2.1 DKIM Signing Software

Signer implementations **SHOULD** provide a convenient means of generating DNS key records corresponding to the signer configuration. Support for automatic insertion of key records into the DNS is also highly desirable. If supported however, such mechanism(s) **MUST** be properly authenticated.

A means of verifying that the signer configuration is compatible with the signature policy is also highly desirable.

Disclosure of a private signature key component to a third party allows that third party to impersonate the sender. The protection of private signature key data is therefore a critical concern. Signers **SHOULD** support use of cryptographic hardware providing key management features.

2.2 General Coding Criteria for Cryptographic Applications

NOTE: This section could possibly be changed into a reference to something else, such as another rfc.

Correct implementation of a cryptographic algorithm is a necessary but not a sufficient condition for the coding of cryptographic applications. Coding of cryptographic libraries requires close attention to security considerations that are unique to cryptographic applications.

In addition to the usual security coding considerations, such as avoiding buffer or integer overflow and underflow, implementers should pay close attention to management of cryptographic private keys and session keys, ensuring that these are correctly initialized and disposed of.

Operating system mechanisms that permit the confidentiality of private keys to be protected against other processes **SHOULD** be used when available. In particular, great care **MUST** be taken when releasing memory pages to the operating system to ensure that private key information is not disclosed to other processes.

Certain implementations of public key algorithms such as RSA may be vulnerable to a timing analysis attack.

Support for cryptographic hardware providing key management capabilities is strongly encouraged. In addition to offering performance benefits, many cryptographic hardware devices provide robust and verifiable management of private keys.

Fortunately appropriately designed and coded cryptographic libraries are available for most operating system platforms under license terms compatible with commercial, open source and free software license terms.

Use of standard cryptographic libraries is strongly encouraged. These have been extensively tested, reduce development time and support a wide range of cryptographic hardware.

2.3 Mailing List Manager Software

A mailing list often provides facilities to its administrator to manipulate parts of the mail messages that flow through the list. The desired goal is that messages flowing through the mailing list will be verifiable by the recipient as being from the list, or failing that, as being from the individual list members.

In most cases, the list and/or its mail host **SHOULD** add its own DKIM signature to list mail. This could be done in the list management software, in an outgoing MSA or MTA, or both. List management software often makes modifications to messages that will break incoming signatures, such as adding subject tags, adding message headers or footers, and adding, deleting, or reordering MIME parts. By adding its own signature after these modifications, the list provides a verifiable, recognizable signature for list recipients.

In some cases, the modifications made by the mailing list software are simple enough that signatures on incoming messages will still be verifiable after being remailed by the list. It is still preferable that the list sign its mail so that recipients can distinguish between mail sent through the list and mail sent directly to a list member. In the absence of a list signature, a recipient may still be able to recognize and use the original signatures of the list members.

2.4 Email Infrastructure Agents

It is expected that the most common venue for a DKIM implementation will be within the infrastructure of an organization's email service, such as a department or a boundary MTA.

- Outbound:** An MSA or Outbound MTA should allow for the automatic verification of the MTA configuration such that the MTA can generate an operator alert if it determines that it is (1) an edge MTA, and (2) configured to send email messages that do not comply with the published DKIM sending policy.
- An outbound MTA should be aware that users may employ MUAs that add their own signatures and be prepared to take steps necessary to ensure that the message sent is in compliance with the advertised email sending policy.
- Inbound:** An inbound MTA or an MDA that does not support DKIM should avoid modifying messages in ways that prevent verification by other MTAs, or MUAs to which the message may be forwarded.
- An inbound MTA or an MDA may incorporate an indication of the verification results into the message, such as using an Authentication-Results header. [I-D.kucherawy-sender-auth-header]
- Intermediaries:** An email intermediary is both an inbound and outbound MTA. Each of the requirements outlined in the sections relating to MTAs apply. If the intermediary modifies a message in a way that breaks the signature, the intermediary
- SHOULD deploy abuse filtering measures on the inbound mail, and
 - MAY remove all signatures that will be broken
- In addition the intermediary MAY:
- Verify the message signature prior to modification.
 - Incorporate an indication of the verification results into the message, such as using an Authentication-Results header. [I-D.kucherawy-sender-auth-header]
 - Sign the modified message including the verification results (e.g., the Authentication-Results header).

2.5 Filtering Software

Developers of filtering schemes designed to accept DKIM authentication results as input should be aware that their implementations will be subject to counter-attack by email abusers. The efficacy of a filtering scheme cannot therefore be determined by reference to static test vectors alone; resistance to counter attack must also be considered.

Naive learning algorithms that only consider the presence or absence of a verified DKIM signature, without considering more information about the message, are vulnerable to an attack in which a spammer or other malefactor signs all their mail, thus creating a large negative value for presence of a DKIM signature in the hope of discouraging widespread use.

If heuristic algorithms are employed, they should be trained on feature sets that sufficiently reveal the internal structure of the DKIM responses. In particular the algorithm should consider the domains purporting to claim responsibility for the signature, rather than the existence of a signature or not.

Unless a scheme can correlate the DKIM signature with accreditation or reputation data, the presence of a DKIM signature SHOULD be ignored.

2.6 DNS Server Software

At a minimum, a DNS server that handles queries for DKIM key records must allow the server administrators to add free-form TXT records. It would be better if the the DKIM records could be entered using a structured form, supporting the DKIM-specific fields.

3. Deployment

This section describes the basic steps for introducing DKIM into an organization's email service operation. The considerations are divided between the generating DKIM signatures (Signing) and the processing of messages that contain a DKIM signature (Verifying).

3.1 Signing

Creating messages that have DKIM signatures requires a modification to only two portions of the email service:

- Addition of relevant DNS information.
- Addition of the signature by a trusted module within the organization's email handling service.

The signing module uses the appropriate private key to create a signature. The means by which the signing module obtains the private key is not specified by DKIM. Given that DKIM is intended for use during email transit, rather than for long-term storage, it is expected that keys will be changed regularly. Clearly this means that key information should not be hard-coded into software.

3.1.1 DNS Records

A receiver attempting to verify a DKIM signature must obtain the public key that is associated with the signature for that message. The DKIM-Signature header in the message will specify the basic domain name doing the signing and the selector to be used for the specific public key. Hence, the relevant `<selector>._domainkey.<domain-name>` DNS record needs to contain a DKIM-related resource record (RR) that provides the public key information.

The administrator of the zone containing the relevant domain name adds this information. Initial DKIM DNS information is contained within TXT RRs. DNS administrative software varies considerably in its abilities to add new types of DNS records.

3.1.2 Signing Module

The module doing signing can be placed anywhere within an organization's trusted Administrative Management Domain (ADMD); common choices are expected to be department-level posting and delivering agents, as well as boundary MTAs to the open Internet. (Note that it is entirely acceptable for MUAs to perform signing and verification.) Hence the choice among the modules depends upon software development and administrative overhead tradeoffs. One perspective that helps resolve this choice is the difference between the flexibility of use by systems at (or close to) the MUA, versus the centralized control that is more easily obtained by implementing the mechanism "deeper" into the organization's email infrastructure, such as at its boundary MTA.

3.1.3 Signing Policies and Practices

Every organization (ADMD) will have its own policies and practices for deciding when to sign messages and with what domain name and key (selector). Examples include signing all mail, signing mail from particular email addresses, or signing mail from particular sub-domains. Given this variability, and the likelihood that signing practices will change over time, it will be useful to have these decisions represented in some sort of configuration information, rather than being more deeply coded into the signing software.

3.2 Verifying

3.2.1 Verifier

Verifiers SHOULD treat the result of the verification step as an input to the message evaluation process rather than as providing a final decision. The knowledge that a message is authentically sent by a domain does not say much about the legitimacy of the message, unless the characteristics of the domain claiming responsibility are known.

In particular, verifiers **SHOULD NOT** automatically assume that an email message that does not contain a signature, or that contains a signature that does not verify, is forged. Verifiers should treat a signature that fails to verify the same as if no signature were present. **NOTE: THE ABOVE MAY BE MODIFIED BY SSP/ASP**

Verification is performed within an ADMD that wishes to make assessments based upon the DKIM signature's domain name. Any component within the ADMD that handles messages, whether in transit or being delivered, can do the verifying and subsequent assessments. Verification and assessment might occur within the same software mechanism, such as a Boundary MTA, or an MDA. Or they may be separated, such as having verification performed by the Boundary MTA and assessment performed by the MDA.

As with the signing process, choice of service venues for verification and assessment -- such as filtering or presentation to the recipient user -- depend on trade-offs for flexibility, control, and operational ease. An added concern is that the linkage between verification and assessment entails essential trust: the assessment module **MUST** have a strong basis for believing that the verification information is correct.

3.2.2 DNS Client

The primary means of publishing DKIM key information, initially, is initially through DNS TXT records. Some DNS client software might have problems obtaining these records; as DNS client software improves this will not be a concern.

3.2.3 Boundary Enforcement

In order for an assessment module to trust the information it receives about verification (e.g., Authentication-Results headers), it **MUST** eliminate verification information originating from outside the ADMD in which the assessment mechanism operates. As a matter of friendly practice, it is equally important to make sure that verification information generated within the ADMD not escape outside of it.

In most environments, the easiest way to enforce this is to place modules in the receiving and sending Boundary MTA(s) that strip any existing verification information.

3.3 Key Management Deployment

More to be added

3.3.1 First Party Key Management

Selectors Selectors are assigned according to the administrative needs of the signing domain, such as for rolling over to a new key or for delegating of the right to authenticate a portion of the namespace to a trusted third party.

Examples include: jun2005.eng._domainkey.example.com
widget.promotion._domainkey.example.com

NOTE: It is intended that assessments of DKIM identities be based on the domain name, and not include the selector. This permits the selector to be used only for key administration, rather than having an effect on reputation assessment.

3.3.2 Third Party Key Management

???????????????? 3rd party generates the public / private key pair and sends the public key to be published in the DNS.

3.3.3 Signer Actions

All Signers **SHOULD**:

- Include any existing Sender header field in the signed header field list, if the Sender header field exists.

Signers wishing to avoid the use of Third-Party Signatures **SHOULD** do everything listed above, and also:

- Include the Sender header field name in the header field list ("h=" tag) under all circumstances, even if the Sender header field does not exist in the header block. This prevents another entity from adding a Sender header field.
- Publish Signing Practices that do not sanction the use of Third-Party Signatures.

3.4 Mailing Lists

There are several forms of mailing lists, which interact with signing in different ways.

- "Verbatim" mailing lists send messages without modification whatsoever. They are often implemented as MTA-based aliases. Since they do not modify the message, signatures are unaffected and will continue to verify. It is not necessary for the forwarder to re-sign the message; however, some may choose to do so in order to certify that the message was sent through the list.
- "Digesting" mailing lists collect together one or more postings and then retransmit them, often on a nightly basis, to the subscription list. These are essentially entirely new messages which must be independently authored (that is, they will have a "From" header field referring to the list, not the submitters) and signed by the Mailing List Manager itself, if they are signed at all.
- "Resending" mailing lists receive a message, modify it (often to add "unsubscribe" information or advertising), and immediately resend that message to the subscription list. They are problematic because they usually do not change the "From" header field of the message, but they do invalidate the signature in the process of modifying the message.

The first two cases act in obvious ways and do not require further discussion. The remainder of this session applies only to the third case.

3.4.1 Mailing List Manager Actions

Mailing List Managers should make every effort to ensure that messages that they relay and which have Valid Signatures upon receipt also have Valid Signatures upon retransmission. In particular, Mailing List Managers that modify the message in ways that break existing signatures SHOULD:

- Verify any existing DKIM Signatures. A DKIM-aware Mailing List Manager MUST NOT re-sign an improperly signed message in such a way that would imply that the existing signature is acceptable.
- Apply regular anti-spam policies. A Mailing List Manager SHOULD apply message content security policy just as they would messages destined for an individual user's mailbox. In fact, a Mailing List Manager might apply a higher standard to messages destined to a mailing list than would normally be applied to individual messages.

NON-NORMATIVE RATIONALE: Since reputation will accrue to signers, Mailing List Managers should verify the source and content of messages before they are willing to sign lest their reputation be sullied by nefarious parties.

- Add a Sender header field using a valid address pointing back to the Mailing List Administrator or an appropriate agent (such as an "owner-" or a "-request" address).
- Sign the resulting message with a signature that is valid for the Sender header field address. The Mailing List Manager SHOULD NOT sign messages for which they are unwilling to accept responsibility.

Mailing List Managers MAY:

- Reject messages with signatures that do not verify or are otherwise Suspicious.

3.5 Mail User Agent

DKIM is designed to support deployment and use in email components other than an MUA. However an MUA MAY also implement DKIM features directly.

Outbound: If an MUA is configured to send email directly, rather than relayed through an outbound MSA, the MUA SHOULD be considered as if it were an outbound MTA for the purposes of DKIM. An MUA MAY support signing even if mail is to be relayed through an outbound

MSA. In this case the signature applied by the MUA may be in addition to any MSA signature.

Inbound: An MUA MAY rely on a report of a DKIM signature verification that took place at some point in the inbound MTA path (e.g., an Authentication-Results header), or an MUA MAY perform DKIM signature verification directly. A verifying MUA SHOULD allow for the case where mail is modified in the inbound MTA path.

It is common for components of an ADMD's email infrastructure to do violence to a message, such as to render a DKIM signature invalid. Hence, users of MUAs that support DKIM signing and/or verifying need a basis for knowing that their associated email infrastructure will not break a signature.

3.6 Transition Strategy

Deployment of a new signature algorithm without a 'flag day' requires a transition strategy such that signers and verifiers can phase in support for the new algorithm independently, and (if necessary) phase out support for the old algorithm independently.

[Note: this section assumes that a security policy mechanism exists. It is subject to change.]

DKIM achieves these requirements through two features: First a signed message may contain multiple signatures created by the same signer. Secondly the security policy layer allows the signing algorithms in use to be advertised, thus preventing a downgrade attack.

3.6.1 Signer transition strategy

Let the old signing algorithm be A, the new signing algorithm be B. The sequence of events by which a Signer may introduce the new signing algorithm B, without disruption of service to legacy verifiers, is as follows:

1. Signer signs with algorithm A
 - A. Signer advertises that it signs with algorithm A
2. Signer signs messages twice, with both algorithm A and algorithm B
 - A. The signer tests new signing configuration
 - B. Signer advertises that it signs with either algorithm A or algorithm B
3. Signer determines that support for Algorithm A is no longer necessary
4. Signer determines that support for algorithm A is to be withdrawn
 - A. Signer removes advertisement for Algorithm A
 - B. Signer waits for cached copies of earlier signature policy to expire
 - C. Signer stops signing with Algorithm A

3.6.2 Verifier transition strategy

The actions of the verifier are independent of the signer's actions and do not need to be performed in a particular sequence. The verifier may make a decision to cease accepting algorithm A without first deploying support for algorithm B. Similarly a verifier may be upgraded to support algorithm B without requiring algorithm A to be withdrawn. The decisions of the verifier must make are therefore:

- The verifier MAY change the degree of confidence associated with any signature at any time, including determining that a given signature algorithm provides a limited assurance of authenticity at a given key strength.

A verifier MAY evaluate signature records in any order it chooses, including using the signature algorithm to choose the order.

- The verifier MAY make a determination that Algorithm A does not offer a useful level of security, or that the cost of verifying the signature is less than the value of doing so.

In this case the verifier would ignore signatures created using algorithm A and references to algorithm A in policy records would be treated as if the algorithm were not implemented.

- The verifier MAY decide to add support for additional signature algorithms at any time.
The verifier MAY add support for algorithm B at any time.

3.7 Migrating from DomainKeys

3.7.1 Signing

DNS Records: DKIM is upwardly compatible with existing DomainKeys (DK) [RFC4870] DNS records, so that a DKIM module does not automatically require additional DNS administration! However DKIM has enhanced the DomainKeys DNS key record format by adding several additional optional parameters.

Boundary MTA: The principle use of DomainKeys is at Boundary MTAs. Because no operational transition is ever instantaneous, it is not advisable for existing DomainKeys signers to switch to DKIM without continuing to perform DomainKeys signing. A signer should add a DKIM signature to a message that also has a DomainKeys signature, until such time as DomainKeys receive-side support is sufficiently reduced. With respect to signing policies, a reasonable, initial approach is to use DKIM signatures in the same way as DomainKeys signatures are already being used.

3.7.2 Verifying

DNS Client: DNS queries for the DKIM key record, use the same Domain Name naming conventions as were used for DomainKeys, and the same basic record format. No changes to the DNS client should be required.

Verifying module: See the section on Signing above.

4. On-going Operations

This section describes the basic steps for operation of email systems that use DKIM, after the capability has initially been deployed. The primary considerations are:

- the upkeep of the selector files, and
- the security of the private keys.

4.1 DNS Signature Record Installation Considerations

Even with use of the DNS, one challenge is that DNS record management is usually operated by an administrative staff that is different from those who operate an organization's email service. In order to ensure that DKIM DNS records are accurate, this imposes a requirement for careful coordination between the two operations groups.

The key point to remember is that the DNS DKIM selectors WILL and SHOULD be changed over time. Some reasons for changing DKIM selectors are well understood, while others are still theoretical. There are several schemes that may be used to determine the policies for changing DKIM selectors:

- time based
- associations with clusters of servers
- the use of third party signers
- security considerations

4.1.1 Time Basis and Security Considerations

The reason for changing the selector periodically is usually related to the security exposure of a system. When the potential exposure of the private keys associated with the DKIM selector have reached sufficient levels, the selector should be changed. (It is unclear currently what kinds of metrics can be used to aid in deciding when the exposure has reached sufficient levels to warrant changing the selector.)

For example,

- Selectors should be changed more frequently on systems that are widely exposed, than on systems that are less widely exposed. For example, a gateway system that has numerous externally-accessible services running on it, is more widely exposed than one that ONLY runs a mail server.
- Selectors should be changed more frequently on operating systems that are under wide attack.
- While the use of DKIM information is transient, keys with sufficient exposure do become stale and should be changed.
- Whenever you make a substantial system change, such as bringing up a new server, or making a major operating system change, you should consider changing the selector.
- Whenever there is either suspicion or evidence of the compromise of the system or the private keys, you should change the selector.

4.1.2 Deploying New Selectors

A primary consideration in changing the selector is remembering to change it. It needs to be a standard part of the operational staff Methods and Procedures for your systems. If they are separate, both the mail team and the DNS team will be involved in deploying new selectors.

When deploying a new selector, it needs to be phased in:

1. Generate the new public / private key pair and create a new selector record with the public key in it.
2. Add the new selector record to your DNS.
3. Verify that the new selector record can be used to verify signatures.
4. Turn on signing with the new private key.
5. Remove the old private key from your servers.

6. After a period of time, remove the old selector from your DNS.

The time an unused selector should be kept in the DNS system is dependent on the reason it's being changed. If the private key has definitely been exposed, the corresponding selector should be removed immediately. Otherwise, longer periods are allowable.

4.1.3 Subdomain Considerations

A Domain Name is the basis for making differential quality assessments about a DKIM-signed message. It is reasonable for a single organization to have a variety of very different activities, which warrant a variety of very different assessments. A convenient way to distinguish among such activities is to sign with different domain names. That is, the organization should sign with sub-domain names that are used for different organizational activities.

4.1.4 Delegating Signing Authority to a Third party

Allowing third parties to sign email from your domain opens your system security to include the security of the third party's systems. At a minimum, you should not allow the third parties to use the same selector and private key as your main mail system. It is recommended that each third party be given their own private key and selector. This limits the exposure for any given private key, and minimizes the impact if any given private key were exposed.

4.2 Private Key Management

The permissions of private key files must be carefully managed. If key management hardware support is available, it should be used. Auditing software should be used to periodically verify that the permissions on the private key files remain secure.

5. Example Uses

A DKIM signature tells the signature verifier that the owner of a particular domain name accepts responsibility for the message. Combining this information with information that allows the history of the domain name owner to be assessed may allow processing the message, based on the probability that the message is likely to be trustworthy, or not, without the need for heuristic content analysis.

5.1 Protection of Internal Mail

One identity is particularly amenable to easy and accurate assessment: The organization's own identity. Members of an organization tend to trust messages that purport to be from within that organization. However Internet Mail does not provide a straightforward means of determining whether such mail is, in fact, from within the organization. DKIM can be used to remedy this exposure. If the organization signs all of its mail, then its boundary MTAs can look for mail purporting to be from the organization but does not contain a verifiable signature. Such mail can be presumed to be spurious.

WHAT ABOUT MAIL TO A MAILING LIST THAT COMES BACK WITH A BROKEN SIGNATURE????

5.2 Recipient-based Assessments

Recipients of large volumes of email can internally generate reputation data for email senders. Recipients of smaller volumes of messages are likely to need to acquire reputation data from a third party. In either case the use of reputation data is intrinsically limited to email senders that have established a prior history of sending messages.

In fact, an email receiving service may be in a position to establish bilateral agreements with particular senders, such as business partners or trusted bulk sending services. Although it is not practical for each recipient to accredit every sender, the definition of core networks of explicit trust can be quite useful.

5.2.1 Third-party Reputation and Accreditation Services

For scaling efficiency, it is appealing to use Trusted Third Party reputation and accreditation services, to allow an email sender to obtain a single assessment that is then recognized by every email recipient that recognizes the Trusted Third Party.

5.3 DKIM Support in the Client

The DKIM specification is expected to be used primarily between Boundary MTAs, or other infrastructure components of the originating and receiving ADMDs. However there is nothing in DKIM that is specific to those venues. In particular, it should be possible to support signing and verifying in MUAs.

DKIM requires that all verifiers treat messages with signatures that do not verify as if they are unsigned. If verification in the client is to be acceptable to users, it is also essential that successful verification of a signature not result in a less than satisfactory user experience compared to leaving the message unsigned.

5.4 Per user signatures

Although DKIM's use of domain names is optimized for a scope of organization-level signing, it is possible to administer sub-domains and/or selectors in a way that supports per-user signing.

NOTE: As stated earlier, it is important to distinguish between the use of selectors for differential administration of keys, versus the use of sub-domains for differential reputations.

As a constraint on an authorized DKIM signing agent, their associated key record can specify restrictions on the email addresses permitted to be signed with that domain and key. A typical intent of this feature is to allow a company to delegate the signing authority for bulk marketing communications without the risk of effectively delegating the authority to sign messages purporting to come from the domain-owning organization's CEO.

NOTE: Any scheme that involves maintenance of a significant number of public keys is likely to require infrastructure enhancements, to support that management. For example, a system in which the end user is required to generate a public key pair and transmit it to the DNS administrator out of band is not likely to meet acceptance criteria for either usability or security.

6. Security Considerations

TBD

7. IANA Considerations

TBD

8. Acknowledgements

TBD

9 Informative References

- [I-D.ietf-openpgp-rfc2440bis] Callas, J., "OpenPGP Message Format", Internet-Draft draft-ietf-openpgp-rfc2440bis-22 (work in progress), April 2007.
- [I-D.kucherawy-sender-auth-header] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", Internet-Draft draft-kucherawy-sender-auth-header-11 (work in progress), February 2008.
- [RFC0989] Linn, J. and IAB Privacy Task Force, "[Privacy enhancement for Internet electronic mail: Part I: Message encipherment and authentication procedures](#)", RFC 989, February 1987.
- [RFC1034] Mockapetris, P., "[Domain names - concepts and facilities](#)", STD 13, RFC 1034, November 1987.
- [RFC1848] Crocker, S., Galvin, J., Murphy, S., and N. Freed, "[MIME Object Security Services](#)", RFC 1848, October 1995.
- [RFC1991] Atkins, D., Stallings, W., and P. Zimmermann, "[PGP Message Exchange Formats](#)", RFC 1991, August 1996.
- [RFC2440] Callas, J., Donnerhackle, L., Finney, H., and R. Thayer, "[OpenPGP Message Format](#)", RFC 2440, November 1998.
- [RFC2821] Klensin, J., "[Simple Mail Transfer Protocol](#)", RFC 2821, April 2001.
- [RFC2822] Resnick, P., "[Internet Message Format](#)", RFC 2822, April 2001.
- [RFC3156] Elkins, M., Del Torto, D., Levien, R., and T. Roessler, "[MIME Security with OpenPGP](#)", RFC 3156, August 2001.
- [RFC3164] Lonvick, C., "[The BSD Syslog Protocol](#)", RFC 3164, August 2001.
- [RFC3851] Ramsdell, B., "[Secure/Multipurpose Internet Mail Extensions \(S/MIME\) Version 3.1 Message Specification](#)", RFC 3851, July 2004.
- [RFC4686] Fenton, J., "[Analysis of Threats Motivating DomainKeys Identified Mail \(DKIM\)](#)", RFC 4686, September 2006.
- [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.

INTERNET
DRAFT

Authors' Addresses

Tony Hansen

AT&T Laboratories
200 Laurel Ave.
Middletown, NJ 07748
USA
EMail: tony+dkimov@mailennium.att.com

Phillip Hallam-Baker

VeriSign Inc.
EMail: pbaker@verisign.com

Dave Crocker

Brandenburg InternetWorking
675 Spruce Dr.
Sunnyvale, CA 94086
USA
EMail: dcrocker@bbiw.net

Full Copyright Statement

Copyright © The IETF Trust (2008).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an “AS IS” basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <<http://www.ietf.org/ipr>>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org¹.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

¹ <mailto:ietf-ipr@ietf.org>