DomainKeys Identified Mail   T. Hansen
Internet-Draft   AT&T Laboratories
Intended status: Informational   D. Crocker
Expires: April 26, 2007   Brandenburg InternetWorking
                                                           P. Hallam-Baker
                                                             VeriSign Inc.
                                                          October 23, 2006

DomainKeys Identified Mail (DKIM) Service Overview
draft-ietf-dkim-overview-03

Abstract

DomainKeys Identified Mail (DKIM) associates a "responsible" identity with a message and provides a means of verifying that the association is legitimate.[I-D.ietf-dkim-base]. 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 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 an overview of DKIM and describes how it can fit into a messaging service, how it relates to other IETF message signature technologies. It also includes implementation and migration considerations.

Note

This document is being discussed on the DKIM mailing list, ietf-dkim@mipassoc.org.

Table of Contents

1. Introduction

This document provides an overview of DomainKeys Identified Mail (DKIM). It is intended for those who are adopting, developing, or deploying DKIM. It also will be helpful for those who are considering extending DKIM, either into other areas or to support additional features. This Overview does not provide information on threats to DKIM or email, or details on the protocol specifics, which can be found in [I-D.ietf-dkim-base] and [I-D.ietf-dkim-threats], respectively. The document assumes a background in basic network security technology and services.

NOTE: It must be stressed that neither this document nor DKIM attempt to provide solutions to the world's problems with spam, phish, virii, worms, joe jobs, etc. DKIM creates one, basic tool in what needs to be a large arsenal of tools, for improving the safety of Internet mail. However by itself, DKIM is not sufficient to that task and this Overview does not pursue the issues of integrating DKIM into these larger efforts. Rather, it is a basic introduction to the technology and its deployment.

NOTE: A number of sections in this document are just placeholders, for now.

There have been four other efforts at standardizing an email signature scheme:

- o Privacy Enhanced Mail (PEM) was first published in 1987 [RFC0989] and eventually transformed into MIME Object Security Services in 1995 [RFC1848]. Today, they are only of historical interest.

- o Pretty Good Privacy (PGP) was developed by Phil Zimmerman and first released in 1991.[RFC1991] A later version was standardized as OpenPGP. [RFC3156]

- o RSA Security, the holder of the patent rights to the principle public key cryptography algorithm, independently developed Secure MIME (S/MIME) to transport a PKCS #7 data object. [RFC3851]

Development of S/MIME and OpenPGP has continued. While both have achieved a significant user base neither has achieved ubiquity in deployment or use and their goals differ from those of DKIM.

In principle the S/MIME protocol can support semantics such as domain level signatures or make use of keys stored in the DNS. However the currently deployed base does not and modifying it to do so would require extensive effort.

Unlike all four previous IETF email security initiatives, DKIM employs a key centric Public Key Infrastructure (PKI) as opposed to one that is based on a certificate in the style of Kohnfelder (X.509) or Zimmerman (web of trust). That is, the owner of a key asserts its validity, rather than relying on having a broader semantic implication of the assertion, such as a quality assessment of the key's owner. DKIM treats quality assessment as an independent, value-added service, beyond the initial work of deploying a validating signature service.

NOTE: It would be useful to include a citation to a general discussion about PKI issues, including their long history of difficulties with respect to the Internet.

Further, DKIM's PKI is supported as additional information records to the existing Domain Name Service, rather than requiring deployment of a new query infrastructure. This approach also has significant performance advantages as DNS is layered on UDP and key retrieval is typically achieved in a single round trip.

2. The DKIM Value Proposition

Spam can be understood as two separate problems. The first is the problem of the companies that are inappropriately aggressive, in sending out unsolicited marketing email. This accounts for, perhaps, 5% of the spam volume and is in any case usually handled by existing spam filters. The second problem is rogue spam -- often involving criminal activities -- mostly sent from coerced botnets of compromised machines. For this latter set of mail, the origins of a message are often falsely stated.

Even without the addition of independent accreditation services, DKIM allows pair-wise sets of (possibly large) email providers and spam filtering companies to distinguish mail that is associated with a known organization, from mail that might deceptively purport to have the affiliation. This in turn allows the development of 'whitelist' schemes whereby authenticated mail from a known source with good reputation is allowed to bypass some spam filters.

In effect the email receiver is using their set of known relationships to generate their own accreditation/reputation data. This works particularly well for traffic between large sending providers and large receiving providers. However it also works well for any operator, public or private, that has mail traffic dominated by exchanges among a stable set of organizations.

NOTE: Perhaps add some citations to detailed discussions about spam and phishing and the role of authentication and accreditation in fighting them?

DKIM used in conjunction with a real-time blacklist allows phishing emails to be preemptively blocked. The value of a cousin domain, that could be mistaken for the legitimate domain, is significantly reduced if the number of emails that can be successfully sent from it is small.

Phishing attacks are typically made against trusted brands, that is, names that are closely affiliated with well-known organizations. A DKIM-based accreditation service can enforce a basic separation between domains used by such known organizations and domains used by others.

Receivers who successfully validate a signature can use information about the signer as part of a program to limit spam, spoofing, phishing, or other undesirable behavior, although the DKIM specification itself does not prescribe any specific actions by the recipient.

3. DKIM's Goals

DKIM lets an organization take responsibility for a message. The organization taking responsibility typically is a handler of the message, either as its originator or as an intermediary. It can also be an independent service, providing assistance to a handler of the message. Their reputation is the basis for evaluating whether to trust the message for delivery.

The owner of the domain name being used for a DKIM signature is declaring that they are accountable for the message. This means that their reputation is at stake.

By design, DKIM purposely:

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 does not require the use of trusted third parties (e.g., certificate authorities) that might impose significant costs or

introduce delays to deployment

- o can be deployed incrementally, with separate deployment of signers and verifiers in either order

- o allows delegation of signing to third parties

- o is not intended be used for archival purposes

DKIM authentication provides one link in a chain of responsibility, hopefully leading to better accountability by the senders.

3.1. Treat verification failure as if unsigned.

PGP and S/MIME were both designed for strong cryptographic protection. This included treating validation failure as message failure, at least warning the user that the message was unsigned. In a small number of cases the application went even further by 'warning' the user whenever a signed message was received. This approach has proved problematic. Hence for DKIM, the guidance is that an email signature verifier is to treat messages with signatures that fail as if they were unsigned.

It is highly unlikely that an attacker is going to add a digital signature to a message unless doing so causes the message to be treated more favorably than an unsigned one. Any messages that carry signatures that fail verification are thus much more likely to be a genuine message that has been damaged in transit than an attempted forgery. It makes no sense to warn the recipient unless it is known that the sender always signs email messages and that there is a high probability that a forgery would be attempted.

3.2. Domain-level assurance

PGP and S/MIME apply the end-to-end principle in terms of individual originators and recipients, notably using full email addresses. DKIM seeks accountability at the more coarse grain of an organization or, perhaps, a department. A deployed construct that enables this granularity is the domain name, to which the signing key record is bound.

3.3. Incremental adoption

DKIM can immediately provide benefits between any two organizations that exchange email and implement DKIM. In the usual manner of "network effects", the benefits of DKIM increase dramatically as its adoption increases.

Over time, DKIM adoption might become sufficiently widespread to permit special, negative handling of messages that are not signed. However early benefits do not require this more-stringent enforcement.

3.4. Minimal infrastructure

DKIM can be implemented at a variety of places within an organization's email service. This permits the organization to choose how much or how little they want DKIM to be part of their infrastructure, rather than part of a more localized operation. Similarly, DKIM's reliance on the Domain Name Service greatly reduces the amount of new administrative infrastructure that must be deployed over the open Internet.

Even with use of the DNS, one challenge is that it is usually operated by different administrative staff than 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.

3.5. Transparent signature

S/MIME and PGP both modify the message body. Hence, their presence is visible to all email recipients and their user software must be able to process the associated constructs. In order to facilitate incremental adoption, DKIM is designed to be transparent to recipients that do not support it.

3.6. Security policy

DKIM is pursuing an incremental innovation, over basic identity authentication, through the publication of security policies associated with one or of the identities presented in a message. For example, a valid DKIM signature allows the signing party to assert responsibility for a message. For a recipient to interpret an unsigned message it is necessary to know whether it should expect a message from that source to be signed and if so the signature characteristics to expect. It would, therefore, be helpful for a potential signer to be able to publish whether they sign all of their message. Once this is published, recipients can choose to handle the receipt of unsigned messages with added caution.

4. A Quick Overview of DKIM

DKIM has a very constrained set of capabilities, primarily targeting email while it is in transit, from an originator to one or more

recipients. DKIM defines a mechanism by which email messages can be cryptographically signed, permitting a signing domain to claim responsibility for the presence of a message in the mail stream. A responsible organization adds a digital signature to the message, associating it with a domain name of that organization. Typically, signing will be done by a service agent within the authority of the message originator's Administrative Management Domain (ADMD). (Signing might be performed by any of the functional components in that environment, including a Mail User Agent (MUA), a Mail Submission Agent (MSA), or an Internet Boundary MTA. DKIM also permits signing to be performed by authorized third-parties.)

4.1. What is a DKIM signature?

A signature covers the message body and selected header fields. The signer computes a hash of the selected header fields and another hash of the body. They then use a private key to cryptographically encode this information, along with other signing parameters. The signature information is placed into a new header field of the RFC2822 [RFC2822] message.

4.2. The Selector construct

In order to allow a domain name to support multiple, simultaneous keys, a particular signature is identified by the combination of the domain name and an added field, called the "selector". Both of these are coded into the DKIM-Signature header field. 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.

4.3. Who validates the signature?

After a message has been signed, any agent in the message transit path can choose to validate the signature. Validation of the signature, by a later agent in the path, demonstrates that the signing identity took responsibility for the message

Message recipients can verify the signature by querying the signer's

domain directly to retrieve the appropriate public key, and thereby confirm that the message was attested to by a party in possession of the private key for the signing domain. Typically, validation will be done by an agent in the ADMD of the message recipient. Again, this may be done by any functional component within that environment. (Notably this means that the signature can be used by the recipient ADMD's filtering software, rather than requiring the recipient end-user to make an assessment.)

4.4. What does DKIM NOT do?

DKIM does not:

- offer any assertions about the behaviors of the identity doing the signing.

- prescribe any specific actions for receivers to take upon successful (or unsuccessful) signature validation.

- provide protection after message delivery.

- protect against re-sending (replay of) a message that already has a valid signature; therefore a transit intermediary or a recipient can re-post the message in such a way that the signature would remain valid, although the new recipient(s) would not have been specified by the originator.

4.5. Does DKIM eliminate anonymity for email?

The ability to send a message that does not identify its author is considered to be a valuable quality of the current email system. It turns out that DKIM is particularly helpful to this goal, because a DKIM signature will typically be used to identity an email system operator, rather than a content author. Knowing that a mail definitely came from example.com does not threaten the anonymity of the user, if it is still possible to obtain effectively anonymous accounts at example.com and other web mail providers.

4.6. Outline potential DKIM applications

TBD

4.7. What is a DKIM policy?

TBD

5. DKIM Within Existing Internet Email

5.1. Review of Internet Mail Service Architecture

Internet Mail has a simple split between the user world, in the form of Mail User Agents (MUA), and the transmission world, in the form of the Mail Handling Service (MHS) composed of Mail Transfer Agents (MTA). The MHS is responsible for accepting a message from one User and delivering it to one or more other users, creating a virtual MUA-to-MUA exchange environment. The first MTA is called the Mail Submission Agent (MSA) and the final MTA is called the Mail Delivery Agent (MDA)

```
                                 +--------+
               +---------------->|  User  |
               |                 +--------+
               |                      ^
   +--------+  |          +--------+  .
   |  User  +--+--------->|  User  |  .
   +--------+  |          +--------+  .
       .       |               ^      .
       .       |   +--------+  .      .
       .       +-->|  User  |  .      .
       .           +--------+  .      .
       .                ^      .      .
       .                .      .      .
       V                .      .      .
   +---+----------------+------+------+---+
   |   .                .      .      .   |
   |   +...............>+      .      .   |
   |   .                       .      .   |
   |   +......................>+      .   |
   |   .                              .   |
   |   +.............................>+   |
   |                                      |
   |     Mail Handling Service (MHS)      |
   +--------------------------------------+
```

Figure 1: Basic Internet Mail Service Model

Modern Internet Mail service is marked by many independent operators, many different components for providing users with service and many other components for performing message transfer. Consequently, it is necessary to distinguish administrative boundaries that surround sets of functional components.

5.1.1. Administrative Actors

Operation of Internet Mail services is apportioned to different providers (or operators). Each can be composed of an independent ADministrative Management Domain (ADMD). An ADMD operates with an independent set of policies and interacts with other ADMDs according to differing types and amounts of trust. Examples include an end-user operating their desktop client, a department operating a local Relay, an IT department operating an enterprise Relay and an ISP operating a public shared email service. These can be configured into many combinations of administrative and operational relationships, with each ADMD potentially having a complex arrangement of functional components. Figure 2 depicts the relationships among ADMDs. Perhaps the most salient aspect of an ADMD is the differential trust that determines its policies for activities within the ADMD, versus those involving interactions with other ADMDs.

Basic components of ADMD distinction include:

Edge -- Independent transfer services, in networks at the edge of the Internet Mail service.

User -- End-user services. These might be subsumed under an Edge service, such as is common for web-based email access.

Transit -- These are Mail Service Providers (MSP) offering value-added capabilities for Edge ADMDs, such as aggregation and filtering.

Note that Transit services are quite different from packet-level transit operation. Whereas end-to-end packet transfers usually go through intermediate routers, email exchange across the open Internet is often directly between the Edge ADMDs, at the email level.

```
   +-------+                          +-------+    +-------+
   | ADMD1 |                          | ADMD3 |    | ADMD4 |
   | ----- |                          | ----- |    | ----- |
   |       |   +---------------------->|       |    |       |
   | User  |   |                      |-Edge--+--->|-User  |
   |  |    |   |                 +--->|       |    |       |
   |  V    |   |                 |    +-------+    +-------+
   | Edge--+---+                 |
   |       |   |    +---------+  |
   +-------+   |    |  ADMD2  |  |
               |    |  -----  |  |
               |    |         |  |
               +--->|-Transit-+--+
                    |         |
                    +---------+
```

Figure 2: ADministrative Management Domains (ADMD) Example

The distinction between Transit network and Edge network transfer services is primarily significant because it highlights the need for concern over interaction and protection between independent administrations. The interactions between functional components within an ADMD are subject to the policies of that domain.

Common ADMD examples are:

Enterprise Service Providers -- Operating an organization's internal data and/or mail services.

Internet Service Providers -- Operating underlying data communication services that, in turn, are used by one or more Relays and Users. It is not necessarily their job to perform email functions, but they can, instead, provide an environment in which those functions can be performed.

Mail Service Providers -- Operating email services, such as for end-users, or mailing lists.

5.1.2. Field Referencing Convention

In this document, references to structured fields of a message use a two-part dotted notation. The first part cites the document that contains the specification for the field and the second is the name of the field. Hence <RFC2822.From> is the From field in an email content header [RFC2822] and <RFC2821.MailFrom> is the address in the SMTP "Mail From" command. [RFC2821]

5.2. Where to Place DKIM Functions

DKIM associates a "responsible" identity with a message and provides a means of verifying that the association is legitimate. Deciding which ADMD shall perform signing or verifying, which identity to assign and which functional components to use for DKIM processing depends upon the nature of the trust/reputation that is of interest and the most convenient or efficient way to use it.

Messages may be signed or verified by any functional component within an ADMD, as that domain wishes to arrange. Examples include:

Outbound -- MUA, MSA or boundary MTA.

Inbound -- Boundary MTA, MDA or MUA.

By having an MUA do the signing or verifying, there is no dependence upon implementation by an email service infrastructure. By having an MHS component do signing or verifying, there is no dependence upon user training or the upgrade of potentially large numbers of user applications.

For implementation by an ADMD's email service operators, perhaps the most obvious choices within the MHS are the MSA or MDA, and the outbound or inbound (boundary) MTA. By signing or verifying at the MSA and MDA, respectively, this outermost portion of the MHS provides true end-to-end service, and requires the smallest amount of trust of the intervening service infrastructure. By signing or verifying at a boundary, the smallest number of systems need modifying within an ADMD and the signature is subject to the smallest amount of handling that can break the signature. Note, however, that this will eliminate DKIM signing for mail that stays within the ADMD.

The choice of identity to use might not be obvious. Examples include:

Author -- The domain associated with the RFC2822.From field provides basic authorization for the author to generate mail. Because the organization controlling that domain is closest to the author, they well might be in the best position to offer their reputation as a basis for asserting that the content is "safe".

Operator -- Email recipient services have long-used the IP Address of a client SMTP server as the basis for assessing whether to permit relay or delivery of a message. These Addresses identify the operator of an email service, rather than necessarily indicating the author of messages being sent by that service. Use of an operator's domain name is a natural extension of this model.

Third-party -- An independent service might wish to certify an author, a message or an operator, by providing its own signature to a message. Hence, evaluation of the message will be based upon the identity of that third-party, rather than any of the identities involved in creating or transferring the message. Indeed, this model is already emerging among a number of reputation-vetting services and is similar to the way a credit card permits a customer to make purchases, based upon the reputation of the credit card company -- and its willingness to issue the card.

Ultimately, deciding where to sign a message will depend upon both the identity being used and tradeoffs among flexibility of uses, administrative control, and operational control. Deciding where to verify a message will depend upon the intended use and similar concerns about flexibility and control. A typical choice for reputation-related verification will be to place the signature verification function "close" to the message-filtering engine.

5.3. Impact on Email Activities

5.3.1. Resources

Although cryptographic authentication is considered to be computationally expensive, the real impact of a mechanism like DKIM is remarkably small. Direct impact on CPU-load has been measured to be 10-15%. Mail handling tends to be I/O-intensive, so dedicated email platforms tend to have unused computational capacity. It is therefore likely that no new hardware will be required for these systems to be able to add DKIM support.

5.3.2. Operations

The costs to deploy, administer and operate DKIM vary greatly, depending upon the placement of DKIM-related modules. The greatest flexibility comes from placing the modules as close as possible to the end user. However this also imposes the greatest costs. The most common scenarios are likely to be:

Boundary MTA -- Here, DKIM is used only for email external to the ADMD. Administration and operation is the simplest, but could cause problems for mobile users who are associated with the organization but must send mail using facilities that are independent of their home ADMD. It also provides no assistance for inter-departmental accountability within the ADMD..

MSA/MDA (Department) -- Placing DKIM support at the points of submission and delivery increases the deployment costs but still keeps control within the ADMD's operational staff. It avoids the considerable, added costs of having to enhance all of the MUAs. This does not improve the lot of mobile users who submit from independent MSAs, but does provide full protection within the ADMD.

MUA -- Obviously this can provide the most complete protection, but at the cost of considerable added administrative effort. Worse, there is extensive evidence that email infrastructure services often perform changes to message content that can break a message signature. Examples include transformation by the MSA to ensure that the message is in full conformance with Internet standards and transformation by Boundary MTAs, to ensure conformance with organizational policies about external communications.

Placing DKIM support into the MUA is the only way to ensure that a highly mobile user retains all of its benefits, in spite of these concerns and having to send mail through independent MSAs.

TBD ... Challenge of mobile users. Server-resident folders -- web or imap -- no problem. Laptop-resident folders, requires remote MSA access or per-user keying and mobile-author awareness.

Key creation and replacement. Update DNS and signing component

5.3.3. Users

TBD ... Challenge of mobile users. Server-resident folders -- web or imap -- no problem. Laptop-resident folders, requires remote MSA access or per-user keying and mobile-author awareness.

Challenge of mailing lists. Different list styles warrant different signature policies.

Can be hidden from end-user; used by filter engine. Method and benefits for displaying to users unknown.

5.4. Migrating from DomainKeys

5.4.1. Signing

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

Boundary MTA: Enforce signature policies and practices

>

5.4.2. Verifying

DNS Client: TBD

Verifying module: See "Signing Module".

5.4.3. Boundary MTA

Strip "local" signatures that are not local?

6. DKIM Service Architecture

DKIM provides an end-to-end service for signing and verifying messages that are in transit. It is divided into components that can be performed using different, external services, such as for key retrieval, although the basic DKIM operation provides an initial set.

```
                                            |
                                            | - RFC2822 Message
                                            V
                        +----------------------------------------------+
    +-----------+       |         ORIGINATING OR RELAYING ADMD         |
    |           |       |         ============================         |
    |  Signer   |       |                                              |
    | Practices +......>|  SIGN MESSAGE                                |
    |           |       |   ...> ADD A SIGNATURE HEADER FIELD          |
    +-----+-----+ .....>|   .     GET (Domain, Selector, Priv-Key)     |
          .       .     |   ...   COMPUTE SIGNATURE                    |
          .       V     +----------------+-----------------------------+
          .   +-------+                     | - Message
          .   |       |                     |      1*(Domain, Selector,
          .   |  Key  |                     |         Sig)
          .   | Store |                 [Internet]
          .   |       |                     |
          .   +---+---+                     V
          .       .     +----------------------------------------------+
          .       .     |          RELAYING OR DELIVERING ADMD         |
          .       .     |          ===========================         |
          .       .     |                                              |
          .       .     |  VERIFY MESSAGE (Verifier Practices)         |
          .       .     |   ...> VERIFY A SIGNATURE HEADER FIELD       |
          .       .     |   .     GET A SIGNATURE                      |
          .       .....>|   .     LOOKUP PUB-KEY (Domain, Selector)    |
          .             |   .     VERIFY SIGNATURE VALUE               |
          .             |   ...   EVALUATE SIGNATURE CONSTRAINTS       |
          .             +-------------------+--------------------------+
          .                                 |  - Verified Domain(s)
          .                                 V  - [Report]
          .             +----------------------------------------------+
          .             |                                              |
          .             |   MESSAGE DISPOSITION                        |
          .............>|       SIGNER PRACTICES                       |
          .............>|       REPUTATION                             |
          .             |                                              |
    +-----+------+      +----------------------------------------------+
    |            |
    | Reputation |
    |            |
    +------------+>
```

Figure 3: DKIM Service Architecture

Basic message processing divides between signing the message, validating the signature, and then performing further decision-making based upon the validated signature. The component doing the signing

applies whatever signing policies are in force, including ones that determine what private key to use. Validation may be performed by any functional component along the relay and delivery path. The public key is retrieved, based upon the parameters stored in the message. The example shows use of the validated identity for assessing an associated reputation. However it could be applied for other tasks, such as management tracking of mail.

7. Implementation Considerations

7.1. Development

7.1.1. Coding Criteria for Cryptographic Applications

Correct implementation of a cryptographic algorithm is a necessary but not a sufficient condition for 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.

On multiple processor and dual core architectures, 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.

7.1.1.1. Signer

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.

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. Protection of private signature key data is therefore a critical concern. Signers SHOULD support use of cryptographic hardware providing key management features.

The import and export of private keys, and the ability to generate a Certificate Signing Request (CSR) for certificate registration are highly desirable.

7.1.1.2. 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 validate, is forged. Verifiers should treat a signature that fails to validate the same as if no signature were present.

7.1.2. Email Handlers

7.1.2.1. 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 MTA, 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 MTA. In this case the signature applied by the MUA may be in addition to or in place of the MTA 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. An MUA MAY perform DKIM signature verification directly. Such an MUA SHOULD allow for the case where mail is modified in the inbound MTA path.

7.1.3. Mail Transfer Agent

It is expected that the most common venue for a DKIM implementation wil be a department or a boundary MTA.

Outbound: An Outbound MTA should allow for 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, (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 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.

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

* remove all signatures that will be broken

In addition the intermediary MAY:

* Verify the message signature prior to modification

* Incorporate an Authentication-Results header to report the verification result.

* Sign the modified message including the Authentication-Results header

7.2. Filtering

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 valid DKIM signature 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.

7.3. DNS Server

TBD

7.4. Accreditation service

TBD

8. Deployment

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

8.1. Signing

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

- o Addition of relevant DNS information.

- o 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 this 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.

8.1.1. DNS Records

A receiver attempting to validate 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>._domainkeys.<domain-name> DNS record needs to contain a DKIM-related RR that provides the public key information.

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

8.1.2. 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. However, note that it is entirely acceptable to have signing and validation be done by MUAs. 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.

8.1.3. 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.

8.2. Verifying

Verification is performed within an ADMD that wishes to make assessments based upon the domain name used for a DKIM signature. Any component within the ADMD that handles messages, whether in transit or being delivered, can be appropriate to do the verifying. It must communicate the results of verification to another component within the ADMD that performs the desired 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.

8.2.1. DNS Client

The primary means of publishing DKIM key information, initially, is 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.

8.2.2. Boundary Enforcement

In order for an assessment module to trust the information it receives about verification, it is essential to 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 stripping of verification information is to place modules in the receiving and sending Boundary MTA(s). For incoming mail, check for known means of communicating verification information and remove it. The same applies for outgoing mail.

8.3. 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.

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.

8.3.1. 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 a new signing algorithm, 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, first with algorithm A and algorithm B

    A. The signer tests new signing configuration

    B. Signer advertises that it signs with algorithm A and 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

8.3.2. 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:

- o 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 making use of the signature algorithm for choosing the order.
- o 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 ignores signatures created using the algorithm A and references to algorithm A in policy records are treated as if the algorithm were not implemented.
- o 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.

9. Operations

This section describes the basic steps for the continued operation of email systems that use DKIM. This section discusses keeping DKIM going, as opposed to getting DKIM started. The primary considerations are: the upkeep of the selector files, and the security of the private keys.

9.1. DNS Signature Record Deployment Considerations

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; 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

9.1.1. 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.

9.1.2. Server Clusters

TBD

9.1.3. 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 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. optionally back up the old private key in a secure location

6. remove the old private key from your servers

7. 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.

9.1.4. Subdomain Considerations

TBD

9.1.5. Third party Signature Delegations

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.

9.2. 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 private key files remain secure.

9.3. Mailing List Management

[Note: this section may be controversial.]

A mailing list often provides facilities to its administrator to manipulate parts of the mail messages that are flowing through. The desired goal is that messages flowing through the mailing list will be verifiable by the recipient, which means that either the mailing list (or its MSA) must sign the message, or the mailing list must not perform actions on the messages that will break existing DKIM signatures. To avoid breaking existing signatures, a mailing list system has these choices:

- A mailing list may add its own DKIM signature. If it does this, it must make sure that it adds its signature after it performs any content transformations to the message, such as adding a footer to the body, adding a prefix to the body, modifying the subject header, etc.

- If a mailing list does not add its own DKIM signature, it must not modify any existing headers that are listed in an h= parameter of any existing DKIM-Signature headers, nor may it add any footer content to the body if there is no l= in any existing DKIM-Signature headers.

- If a mailing list cannot add its own DKIM signature, and must modify the headers or body in a way that will break verification of existing DKIM-Signature headers, it should remove any existing DKIM-Signature headers.

10. Outline Future Extensions

The design of DKIM is unapologetically focused on overcoming the need for immediate deployment and achieving ubiquitous use in the near future. As such, the original design discussions generally tok the approach of 'if in doubt, leave it out'.

But the need to exclude consideration of these features in the near term was in no way intended to preclude their development at a later date. Nor is the lack of a specification describing the use of DKIM with a different PKI infrastructure intended to indicate an intention to develop similar capabilities within the DKIM framework at a future date.

DKIM is an example of 'Design for Deployment', and the need to support rapid deployment has been the overriding priority. It has traditionally been asserted that deployment of a flawed cryptographic

protocol is an almost unacceptable risk, and that bad security is worse than no security; experience demonstrates otherwise. Informing users that email is insecure does not cause them to modify their behavior to avoid dependence thereupon. Deployment of flawed cryptographic security systems such as SSL and WEP has been rectified far more quickly than deployment of protocols such as IPSEC or DNSSEC where caution has prevailed.

One possible future for DKIM is that it becomes the starting point for a new cryptographic infrastructure that eventually replaces legacy systems including S/MIME and PGP. While this future is certainly preferable to never achieving ubiquitous deployment of strong cryptographic security in the Internet, it would certainly take a long time to re-invent this particular wheel. Moreover the deployment of the proposed DKIM enhancements would face political opposition from the adherents to existing formats to be rendered historical. A likely outcome of such a strategy is that the existing two way standards stalemate between S/MIME and PGP would become a three way stalemate.

Alternatively, another possible future is that DKIM provides the 'bootstrap' that enables ubiquitous use of the legacy infrastructure, including the deployed base of PGP and S/MIME capable email clients and the existing business infrastructure of commercial Certificate Authorities. Such a strategy would make use of DKIM in conjunction with existing PKI standards such as PKIX and XKMS and leverage features of PGP and S/MIME where appropriate.

10.1. Introducing a new signing algorithm

Regardless of whether extension of the DKIM feature set is desirable, the need to replace the signature algorithm is practically a certainty. The RSA signature algorithm at best provides equivalent security to an 80 bit symmetric cipher when used with a 1024 bit key [cite]. Extending the key size to 2048 bits improves the cipher strength to only 112 bit equivalence. Achieving 128 bit security requires a minimum of 3072 bits. Achieving equivalent cipher strength to a 192 bit symmetric algorithm requires a prohibitive key size.

The choice of cryptographic algorithm affects the DKIM algorithm in two important ways: First there is the difficulty of storing keys in the DNS. Secondly there is the problem of handling larger signatures.

The default DNS response packet limit of 512 bytes places an effective upper bound of 4096 bits on a DKIM key. In practice the need for packaging, meta-data and the desirability of using DNSSEC to

sign the record reduces the upper bound to no more than 2048 bits.

The size of the DKIM signature itself is a weaker constraint. Even so, while 1024 and even 2048 bit signatures are likely to be acceptable in most implementations larger signature sizes may become prohibitive, particularly since the signature must be Base 64 encoded.

10.2. Possible future signature algorithm choices

ECC cryptography offers a means of implementing public key cryptography using a key size and signature size that are each only twice the size of the equivalent symmetric key algorithm.

While ECC offers clear technical advantages over algorithms based on the difficulty on solving the discrete log problem in a finite field, it is not possible at this point to be confident that a means of applying ECC has been found that is consistent with the position on intellectual property adopted by the DKIM working group.

The DSA signature algorithm based on the discrete log problem faces the same key size limitations as RSA. Importantly for the design of DKIM and DNSSEC however, the signature size is much smaller, the same size as for ECC algorithms.

It is likely that DSA would have received greater attention during the design of DSA if key sizes greater than 512 bits and use of hash functions stronger than SHA-1 had been supported at the time. In March 2006 a proposed revision of the DSA signature algorithm answered these objections permitting larger key sizes and specifying the use of stronger hash functions (including SHA-256 and SHA 512). While the advantages offered by DSA are not sufficient to warrant an immediate transition to the new algorithm at this late stage, it is highly probably that the algorithm will be employed by DNSSEC when finally deployed.

10.3. Linkage to Other PKIs

The principle limitations in DKIM are the lack of support for end-user keying, the lack of support for long term verification of signatures, and the lack of support for trusted third party issued assertions. Each of these limitations is determined by the key distribution mechanism rather than the signature format.

Although the DNS infrastructure could in principle be extended to support these features, this approach would require substantial modifications that entirely negate the advantage of employing an existing infrastructure. The point of using DNS is to reuse the DNS

infrastructure, not necessarily the DNS protocol.

The preferred approach to addressing these limitations is to support use of a PKI infrastructure designed to support these requirements such as PKIX, PGP or XKMS.

10.4. Trusted Third Party Assertions

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 behavior of the domain name owner to be predicted may allow the probability that the message is abusive to be determined without the need for heuristic content analysis.

Recipients of large volumes of email can generate reputation data for email senders internally. 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.

Another commonly used technique for evaluating email senders is accreditation. Most spam sent today is sent by criminals to further a scheme that is unambiguously illegal. Spam demonstrates an Internet equivalent of Gresham's law: the bad spam drives out the merely irritating, the outright criminal drives out the bad. A message is highly unlikely to be spam if: the email sender can demonstrate that it is a legitimate business, and that it has provided a legitimate address where legal process can be served. In addition the accredited email sender may accept a legally binding undertaking not to send spam and possibly post a performance bond that is subject to forfeiture in case of default.

As with reputation data, a high volume email recipient may be in a position to establish bilateral agreements with high volume senders. Smaller recipients are not in a position to require accreditation, nor is it practical for each large sender to accredit every sender. Trusted Third Party accreditation services allow an email sender to obtain an accreditation that is recognized by every email recipient that recognizes the Trusted Third Party.

[Need use of both systems]

[Need means of advertising existence of positive reputation data]

10.5. Linkage to X.509 Certificates

The industry standard for distribution of Trusted Third Party data tied to a public key is the X.509v3/PKIX standard. X.509 based PKI is designed to support requirements such as long term archiving of signatures, end entity signing and Trusted Third Party assertions. Combining the DKIM signature format with the PKIX PKI infrastructure provides an equivalent set of capabilities to S/MIME.

Two approaches may be used to inform signature verifiers that an X.509 certificate has been issued that makes an assertion about the signing key for a DKIM signature:

1. By means of an attribute in the key record

2. By means of a signature header q= parameter

Typical commercially issued digital certificates are considerably larger (1-2 kb) than the 512 byte message size that DNS servers are required to support as a minimum. Practical large scale PKI deployment requires support for certificate chains in addition to the end entity certificate. Publication of a URL for the certificate or certificate chain is therefore a more appropriate approach than storing the certificate data itself in the DNS.

The ability to support multiple key distribution mechanisms for the same key is highly desirable. For example a signer may support DNS based key distribution (for the convenience and efficiency of transport based DKIM signature verifiers) as well as an X.509 certificate.

In other cases a signer may intentionally discourage transport verification by only providing an X.509 certificate.

An X.509 application of particular interest is the use of DKIM as a signature format for Secure Internet Letterhead (Letterhead). Letterhead employs X.509 certificates containing a LOGOTYPE attribute extension [LOGOTYPE] to identify the certificate subject and/or issuer to the user by means of a brand image such as a corporate logo. [PHB-NIST]

10.6. XKMS

XKMS is a key centric PKI that supports registration and location of keys. XKMS is layered as a Web service and the existence of XKMS service for a domain is typically advertised by means of a DNS SRV record.

XKISS, the key location component of XKMS provides a superset of the capabilities of the DKIM DNS key distribution mechanism. As XKMS is layered on SOAP over HTTP over TCP/IP the overhead incurred in retrieving keys through XKMS is substantially higher than the single UDP request/response of DNS key distribution.

Like X.509 XKMS is designed to key management over time periods of years and decades rather than days and supports the use of trusted third parties. XKMS is designed to allow complexity to be concentrated at the Web service as opposed to the client. A client interacts with an XKMS service using request of the form 'provide a key to verify signatures on messages sent by A using protocol B'.

XKMS may also be used as a gateway to one or more PKIs including an X.509 based PKI that makes use of sophisticated features such as cross certification. The verifier may at its option rely on the XKMS service to provide a trusted key or request the complete certificate path allowing offline verification.

A signer may notify signature verifiers that a key may be retrieved using XKMS by means of the q= attribute. The verifier may then discover the corresponding XKMS service using the SRV mechanism as set out in the XKMS specification.

10.7. Verification in the Client

The DKIM specification is designed to support edge to edge authentication. The specification neither supports nor prohibits verification of DKIM signatures in the client. In particular the specification does not attempt to define client semantics for signatures or provide an exhaustive list of user interface security considerations.

For client based verification to be practical, it is likely that a client needs to be capable of determining that it is able to receive messages that do not get clobbered before coming to any conclusions with respect to unsigned messages.

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.

10.8. Per user signature

Although DKIM is designed to support domain level signatures, the DKIM core design neither supports nor prohibits use of per user

signatures. A DKIM key record can specify restrictions on the email addresses it can be used to sign for. These restrictions are not intended to be exhaustive nor are detailed semantics or security considerations set out for interpreting per user signatures. The primary use that this feature is intended to support, is to allow a company to delegate the signing authority for bulk marketing communications without the risk of effectively delegating the authority to sign contracts on behalf of the CEO.

For per user signing keys to provide value beyond this limited use scenario, it is likely that additional requirements will be necessary, such as the ability to perform long term validation of the key. Linkage to some form of PKI is likely to be necessary.

In addition any scheme that involves maintenance of a significant number of public keys will require infrastructure to support that management. 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. At a minimum, a key registration protocol must be defined.

10.9. Encryption

DKIM is not designed to support encryption. A strong case can be made for applying the DKIM style of transparent security, key centric key management and domain level keying. It is not clear that re-using the DKIM signature architecture is the best way to achieve this goal.

The DKIM signature format in particular is designed to allow meta-data to be attached to a message without modifying the content. Content encryption by its very nature requires modification of the message content. The message encryption formats of PGP and S/MIME both solve the problem of message encryption perfectly adequately; there is no reason to believe that a new effort in this space would improve matters.

An architecture of this form would require development of an email receiver security policy that allows a recipient to state that encrypted email messages are acceptable and to specify the key distribution infrastructure(s) by which the necessary encryption keys may be discovered.

A policy infrastructure of this type is implicit in the XKMS standard. One drawback to this approach is that policy distribution and key distribution are conflated, an approach that is satisfactory for message level encryption schemes such as PGP and S/MIME, but less satisfactory for transport layer encryption such as SSL.

10.10. Reuse of Key Record

A number of proposals have been made that attempt to reuse the DKIM key record. Architects considering this approach should be aware of the advantages and limitations.

As a minimum each of the security considerations listed in the DKIM specification should be considered and its possible relevance to the proposed field of use carefully evaluated. Application of a security mechanism outside the context in which it was originally designed for is a principle cause of security failures.

DKIM is designed to meet the security needs of an application where the cost of individual failures is insignificant or small. A single spam in an email inbox is not a disaster, indeed it is no longer even an irritation. For the long time spam sufferer who has seen their inbox filled with hundreds or even thousands of spams, an occasional failure may even be cause for satisfaction as a reminder of a successfully vanquished foe.

One of the chief limitations of using DNS based key records is that maintenance of DNS records is typically a network operations concern. If the entity to which the public key corresponds is a network object (e.g. a mail server), the use of DNS based key management is likely to be satisfactory. If the entity is not a network related object (e.g. an end user) a significant degree of pushback from network administrators should be anticipated.

The design of DKIM is designed for rapid deployment in response to an immediate need. As such many of the design decisions are not the decisions that would be taken if the choice were unconstrained by the limitations of the current legacy DNS. In particular the use of Base 64 encoded public keys distributed through TXT records is not ideal.

Applications that do not face the same constraints as DKIM should carefully evaluate the feasibility of using the binary key record. In particular an application predicated on the use of DNSSEC to authenticate keys should assume support for DKIM binary key records.

10.11. Use of Policy Record

DKIM demonstrates the power of using the DNS to distribute security policy information. It is not possible to achieve robust security unless the parties to a conversation know the security capabilities and expectations of the other.

Any party proposing re-use of the DKIM policy record should carefully consider whether their needs would be better met by proposing a

flexible security policy architecture in the DKIM style rather than proposing ad-hoc extensions and variations.

At a minimum any proposal for new security policy formats that make use of the TXT record should employ a new policy prefix and ensure that mislabeled and wild-carded policy records are not accidentally misinterpreted.

11. Acknowledgements

TBD

12. { Misc Text -- Should go elsewhere, if used at all }

DKIM permits the signing identity to be different from the identities used for the author or the initial posting agent. This is essential, for example, to enable support of signing by authorized third-parties, and to permit signatures by email providers who are otherwise independent of the domain name of the message author.

DKIM permits restricting the use of a signature key to particular types of services, such as only for email. This is helpful when delegating signing authority, such as to a particular department or to a third-party outsourcing service.

With DKIM the signer MUST explicitly list the headers that are signed. This is an improvement because it requires the signer to use the more conservative (likely to verify correctly) mechanism and makes it considerably more robust against the handling of intermediary MTAs.

DKIM self-signs the DKIM-Signature header field, to protect against its being modified.

In order to survive the vagaries of different email transfer systems, mechanisms like DKIM must evaluate the message data in some canonical form, such as treating a string of spaces as tabs as if they were a single space. DKIM has added the "relaxed" canonicalization in place of "nofws".

MUA UI considerations

key delegation/ 3rd party

13. References

13.1. References -- Normative

[I-D.ietf-dkim-base]
Allman, E., "DomainKeys Identified Mail (DKIM) Signatures", draft-ietf-dkim-base-06 (work in progress), October 2006.

[I-D.ietf-dkim-threats]
Fenton, J., "Analysis of Threats Motivating DomainKeys Identified Mail (DKIM)", draft-ietf-dkim-threats-03 (work in progress), May 2006.

[RFC2822] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

13.2. Informative References

[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.

[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.

[RFC2821] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001.

[RFC3156] Elkins, M., Del Torto, D., Levien, R., and T. Roessler, "MIME Security with OpenPGP", RFC 3156, August 2001.

[RFC3851] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004.

Authors' Addresses

Tony Hansen
AT&T Laboratories
200 Laurel Ave.
Middletown, NJ 07748
USA

Email: tony+dkimov@maillennium.att.com

Dave Crocker
Brandenburg InternetWorking
675 Spruce Dr.
Sunnyvale, CA 94086
USA

Email: dcrocker@bbiw.net

Phillip Hallam-Baker
VeriSign Inc.

Email: pbaker@verisign.com

Acknowledgment

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