Workgroup:	openpgp
Internet-Draft:	draft-autocrypt-lamps-protected-headers-01
Published:	4 November 2019
Intended Status:	Informational
Expires:	7 May 2020

Authors:	B.R. Einarsson	. juga	D.K. Gillmor
	Mailpile ehf	*Independent*	*ACLU*

Protected Headers for Cryptographic E-mail

Abstract

This document describes a common strategy to extend the end-to-end cryptographic protections provided by PGP/MIME, etc. to protect message headers in addition to message bodies. In addition to protecting the authenticity and integrity of headers via signatures, it also describes how to preserve the confidentiality of the Subject header.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

E-mail end-to-end security with OpenPGP and S/MIME standards can provide integrity, authentication, non-repudiation and confidentiality to the body of a MIME e-mail message. However, PGP/MIME ([RFC3156]) alone does not protect message headers. And the structure to protect headers defined in S/MIME 3.1 ([RFC3851]) has not seen widespread adoption.

This document defines a scheme, "Protected Headers for Cryptographic E-mail", which has been adopted by multiple existing e-mail clients in order to extend the cryptographic protections provided by PGP/MIME to also protect the message headers.

This document describes how these protections can be applied to cryptographically signed messages, and also discusses some of the challenges of encrypting many transit-oriented headers.

It offers guidance for protecting the confidentiality of non-transit-oriented headers like Subject, and also offers a means to preserve backwards compatibility so that an encrypted Subject remains available to recipients using software that does not implement support for the Protected Headers scheme.

The document also discusses some of the compatibility constraints and usability concerns which motivated the design of the scheme, as well as limitations and a comparison with other proposals.

While the document (and the authors') focus is primarily PGP/MIME, we believe the technique is broadly applicable and would also apply to other MIME-compatible cryptographic e-mail systems, including S/MIME ([RFC8551]). Furthermore, this technique has already proven itself as a useful building block for other improvements to cryptographic e-mail, such as the Autocrypt Level 1.1 ([Autocrypt]) "Gossip" mechanism.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Terminology

For the purposes of this document, we define the following concepts:

- *MUA* is short for Mail User Agent; an e-mail client.
- *Protection* of message data refers to cryptographic encryption and/or signatures, providing confidentiality, authenticity or both.
- *Cryptographic Layer*, *Cryptographic Envelope* and *Cryptographic Payload* are defined in Section 3
- *Original Headers* are the [RFC5322] message headers as known to the sending MUA at the time of message composition.
- *Protected Headers* are any headers protected by the scheme described in this document.
- *Exposed Headers* are any headers outside the Cryptographic Payload (protected or not).
- *Obscured Headers* are any Protected Headers which have been modified or removed from the set of Exposed Headers.
- *Legacy Display Part* is a MIME construct which provides visibility for users of legacy clients of data from the Original Headers which may have been removed or obscured from the Exposed Headers. It is defined in Section 5.
- *User-Facing Headers* are explained and enumerated in Section 1.2.1.
- *Structural Headers* are documented in Section 1.2.2.

1.2.1. User-Facing Headers

Of all the headers that an e-mail message may contain, only a handful are typically presented directly to the user. The user-facing headers are:

- `Subject`
- `From`
- `To`

- `Cc`
- `Date`
- `Reply-To`
- `Followup-To`

The above is a complete list. No other headers are considered "user-facing".

Other headers may affect the visible rendering of the message (e.g., `References` and `In-Reply-To` may affect the placement of a message in a threaded discussion), but they are not directly displayed to the user and so are not considered "user-facing" for the purposes of this document.

1.2.2. Structural Headers

A message header whose name begins with `Content-` is referred to in this document as a "structural" header.

These headers indicate something about the specific MIME part they are attached to, and cannot be transferred or copied to other parts without endangering the readability of the message.

This includes (but is not limited to):

- `Content-Type`
- `Content-Transfer-Encoding`
- `Content-Disposition`

Note that no "user-facing" headers (Section 1.2.1) are also "structural" headers. Of course, many headers are neither "user-facing" nor "structural".

FIXME: are there any non-`Content-*` headers we should consider as structural?

2. Protected Headers Summary

The Protected Headers scheme relies on three backward-compatible changes to a cryptographically-protected e-mail message:

- Headers known to the composing MUA at message composition time are (in addition to their typical placement as Exposed Headers on the outside of the message) also present in the MIME header of the root of the Cryptographic Payload. These Protected Headers share cryptographic properties with the rest of the Cryptographic Payload.
- When the Cryptographic Envelope includes encryption, any Exposed Header MAY be *obscured* by a transformation (including deletion).
- If the composing MUA intends to obscure any user-facing headers, it MAY add a decorative "Legacy Display" MIME part to the Cryptographic Payload which additionally duplicates the original values of the obscured user-facing headers.

When a composing MUA encrypts a message, it SHOULD obscure the `Subject:` header, by using the literal string `...` (three U+002E FULL STOP characters) as the value of the exposed `Subject:` header.

When a receiving MUA encounters a message with a Cryptographic Envelope, it treats the headers of the Cryptographic Payload as belonging to the message itself, not just the subpart. In particular, when rendering a header for any such message, the renderer SHOULD prefer the header's Protected value over its Exposed value.

A receiving MUA that understands Protected Headers and discovers a Legacy Display part SHOULD hide the Legacy Display part when rendering the message.

The following sections contain more detailed discussion.

3. Cryptographic MIME Message Structure

Implementations use the structure of an e-mail message to protect the headers. This section establishes some conventions about how to think about message structure.

3.1. Cryptographic Layers

"Cryptographic Layer" refers to a MIME substructure that supplies some cryptographic protections to an internal MIME subtree. The internal subtree is known as the "protected part" though of course it may itself be a multipart object.

For PGP/MIME [RFC3156] there are two forms of Cryptographic Layers, signing and encryption.

In the diagrams below, "↧" (DOWNWARDS ARROW FROM BAR, U+21A7) indicates "decrypts to".

3.1.1. PGP/MIME Signing Cryptographic Layer (multipart/signed)

```
└┬╴multipart/signed
 ├─╴[protected part]
 └─╴application/pgp-signature
```

3.1.2. PGP/MIME Encryption Cryptographic Layer (multipart/encrypted)

```
└┬╴multipart/encrypted
 ├─╴application/pgp-encrypted
 └─╴application/octet-stream
  ↧ (decrypts to)
  └─╴[protected part]
```

3.2. Cryptographic Envelope

The Cryptographic Envelope is the largest contiguous set of Cryptographic Layers of an e-mail message starting with the outermost MIME type (that is, with the Content-Type of the message itself).

If the Content-Type of the message itself is not a Cryptographic Layer, then the message has no cryptographic envelope.

"Contiguous" in the definition above indicates that if a Cryptographic Layer is the protected part of another Cryptographic Layer, the layers together comprise a single Cryptographic Envelope.

Note that if a non-Cryptographic Layer intervenes, all Cryptographic Layers within the non-Cryptographic Layer *are not* part of the Cryptographic Envelope (see the example in Section 3.3.3).

Note also that the ordering of the Cryptographic Layers implies different cryptographic properties. A signed-then-encrypted message is different than an encrypted-then-signed message.

3.3. Cryptographic Payload

The Cryptographic Payload of a message is the first non-Cryptographic Layer - the "protected part" - within the Cryptographic Envelope. Since the Cryptographic Payload itself is a MIME part, it has its own set of headers.

Protected headers are placed on (and read from) the Cryptographic Payload, and should be considered to have the same cryptographic properties as the message itself.

3.3.1. Simple Cryptographic Payloads

As described above, if the "protected part" identified in Section 3.1.1 or Section 3.1.2 is not itself a Cryptographic Layer, that part *is* the Cryptographic Payload.

If the application wants to generate a message that is both encrypted and signed, it MAY use the simple MIME structure from Section 3.1.2 by ensuring that the [RFC4880] Encrypted Message within the `application/octet-stream` part contains an [RFC4880] Signed Message.

3.3.2. Multilayer Cryptographic Envelopes

It is possible to construct a Cryptographic Envelope consisting of multiple layers for PGP/MIME, typically of the following structure:

```
A └┬╴multipart/encrypted
B  ├─╴application/pgp-encrypted
C  └─╴application/octet-stream
D    ↧ (decrypts to)
E    └┬╴multipart/signed
F     ├─╴[Cryptographic Payload]
G     └─╴application/pgp-signature
```

When handling such a message, the properties of the Cryptographic Envelope are derived from the series `A`, `E`.

As noted in Section 3.3.1, PGP/MIME applications also have a simpler MIME construction available with the same cryptographic properties.

3.3.3. A Baroque Example

Consider a message with the following overcomplicated structure:

```
H └┬╴multipart/encrypted
I  ├─╴application/pgp-encrypted
J  └─╴application/octet-stream
K   ↧ (decrypts to)
L   └┬╴multipart/signed
M    ├┬╴multipart/mixed
N    │├┬╴multipart/signed
O    ││├─╴text/plain
P    ││└─╴application/pgp-signature
Q    │└─╴text/plain
R    └─╴application/pgp-signature
```

The 3 Cryptographic Layers in such a message are rooted in parts `H`, `L`, and `N`. But the Cryptographic Envelope of the message consists only of the properties derived from the series `H`, `L`. The Cryptographic Payload of the message is part `M`.

It is NOT RECOMMENDED to generate messages with such complicated structures. Even if a receiving MUA can parse this structure properly, it is nearly impossible to render in a way that the user can reason about the cryptographic properties of part `O` compared to part `Q`.

3.4. Exposed Headers are Outside

The Cryptographic Envelope fully encloses the Cryptographic Payload, whether the message is signed or encrypted or both. The Exposed Headers are considered to be outside of both.

4. Message Composition

This section describes the composition of a cryptographically-protected message with Protected Headers.

We document legacy composition of cryptographically-protected messages (without protected headers) in Section 4.4, and then describe a revised version of that algorithm in Section 4.5 that produces conformant Protected Headers.

4.1. Copying All Headers

All non-structural headers known to the composing MUA are copied to the MIME header of the Cryptographic Payload. The composing MUA SHOULD protect all known non-structural headers in this way.

If the composing MUA omits protection for some of the headers, the receiving MUA will have difficulty reasoning about the integrity of the headers (see Section 11.2).

4.2. Confidential Subject

When a message is encrypted, the Subject should be obscured by replacing the Exposed Subject with three periods: `...`

This value (`...`) was chosen because it is believed to be language agnostic and avoids communicating any potentially misleading information to the recipient (see Section 7.1 for a more detailed discussion).

4.3. Obscured Headers

Due to compatibility and usability concerns, a Mail User Agent SHOULD NOT obscure any of: `From`, `To`, `Cc`, `Message-ID`, `References`, `Reply-To`, `In-Reply-To`, (FIXME: MORE?) unless the user has indicated they have security constraints which justify the potential downsides (see Section 7 for a more detailed discussion).

Aside from that limitation, this specification does not at this time define or limit the methods a MUA may use to convert Exposed Headers into Obscured Headers.

4.4. Message Composition without Protected Headers

This section roughly describes the steps that a legacy MUA might use to compose a cryptographically-protected message *without* Protected Headers.

The message composition algorithm takes three parameters:

- `origbody`: the traditional unprotected message body as a well-formed MIME tree (possibly just a single MIME leaf part). As a well-formed MIME tree, `origbody` already has structural headers present (see Section 1.2.2).
- `origheaders`: the intended non-structural headers for the message, represented here as a table mapping from header names to header values.. For example, `origheaders['From']` refers to the value of the `From` header that the composing MUA would typically place on the message before sending it.
- `crypto`: The series of cryptographic protections to apply (for example, "sign with the secret key corresponding to OpenPGP certificate X, then encrypt to OpenPGP certificates X and Y"). This is a routine that accepts a MIME tree as input (the Cryptographic Payload), wraps the input in the appropriate Cryptographic Envelope, and returns the resultant MIME tree as output,

The algorithm returns a MIME object that is ready to be injected into the mail system:

- Apply `crypto` to `origbody`, yielding MIME tree `output`
- For header name `h` in `origheaders`:
    - Set header `h` of `output` to `origheaders[h]`
- Return `output`

4.5. Message Composition with Protected Headers

A reasonable sequential algorithm for composing a message *with* protected headers takes two more parameters in addition to `origbody`, `origheaders`, and `crypto`:

- `obscures`: a table of headers to be obscured during encryption, mapping header names to their obscuring values. For example, this document recommends only obscuring the subject, so that would be represented by the single-entry table `obscures = {'Subject': '...'}`. If header `Foo` is to be deleted entirely, `obscures['Foo']` should be set to the special value `null`.
- `legacy`: a boolean value, indicating whether any recipient of the message is believed to have a legacy client (that is, a MUA that is capable of decryption, but does not understand protected headers).

The revised algorithm for applying cryptographic protection to a message is as follows:

- if `crypto` contains encryption, and `legacy` is `true`, and `obscures` contains any user-facing headers (see Section 1.2.1), wrap `orig` in a structure that carries a Legacy Display part:
  - Create a new MIME leaf part `legacydisplay` with header `Content-Type: text/rfc822-headers; protected-headers="v1"`
  - For each obscured header name `obh` in `obscures`:
    - If `obh` is user-facing:
      - Add `obh: origheaders[ob]` to the body of `legacydisplay`. For example, if `origheaders['Subject']` is `lunch plans?`, then add the line `Subject: lunch plans?` to the body of `legacydisplay`
  - Construct a new MIME part `wrapper` with `Content-Type: multipart/mixed`
  - Give `wrapper` exactly two subarts: `legacydisplay` and `origbody`, in that order.
  - Let `payload` be MIME part `wrapper`
- Otherwise:
  - Let `payload` be MIME part `origbody`
- For each header name `h` in `origheaders`:
  - Set header `h` of MIME part `payload` to `origheaders[h]`
- FIXME: Enigmail adds `protected-headers="v1"` parameter to `payload` here. Is this necessary?
- Apply `crypto` to `payload`, producing MIME tree `output`

- If `crypto` contains encryption:
    - For each obscured header name `obh` in `obscures`:
        - If `obscures[obh]` is `null`:
            - Drop `obh` from `origheaders`
        - Else:
            - Set `origheaders[obh]` to `obscures[obh]`
- For each header name `h` in `origheaders`:
    - Set header `h` of `output` to `origheaders[h]`
- return `output`

Note that both new parameters, `obscured` and `legacy`, are effectively ignored if `crypto` does not contain encryption. This is by design, because they are irrelevant for signed-only cryptographic protections.

5. Legacy Display

MUAs typically display user-facing headers (Section 1.2.1) directly to the user. An encrypted message may be read by a decryption-capable legacy MUA that is unaware of this standard. The user of such a legacy client risks losing access to any obscured headers.

This section presents a workaround to mitigate this risk by restructuring the Cryptographic Payload before encrypting to include a "Legacy Display" part.

5.1. Message Generation: Including a Legacy Display Part

A generating MUA that wants to make an Obscured Subject (or any other user-facing header) visible to a recipient using a legacy MUA SHOULD modify the Cryptographic Payload by wrapping the intended body of the message in a `multipart/mixed` MIME part that prefixes the intended body with a Legacy Display part.

The Legacy Display part MUST be of Content-Type `text/rfc822-headers`, and MUST contain a `protected-headers` parameter whose value is `v1`. It SHOULD be marked with `Content-Disposition: inline` to encourage recipients to render it.

The contents of the Legacy Display part MUST be only the user-facing headers that the sending MUA intends to obscure after encryption.

The original body (now a subpart) SHOULD also be marked with `Content-Disposition: inline` to discourage legacy clients from presenting it as an attachment.

5.1.1. Legacy Display Transformation

Consider a message whose Cryptographic Payload, before encrypting, that would have a traditional `multipart/alternative` structure:

```
X └┬╴multipart/alternative
Y  ├─╴text/plain
Z  └─╴text/html
```

When adding a Legacy Display part, this structure becomes:

```
V └┬╴multipart/mixed
W  ├─╴text/rfc822-headers ("Legacy Display" part)
X  └┬╴multipart/alternative ("original body")
Y   ├─╴text/plain
Z   └─╴text/html
```

Note that with the inclusion of the Legacy Display part, the Cryptographic Payload is the `multipart/mixed` part (part `V` in the example above), so Protected Headers should be placed at that part.

5.1.2. When to Generate Legacy Display

A MUA SHOULD transform a Cryptographic Payload to include a Legacy Display part only when:

- The message is going to be encrypted, and
- At least one user-facing header (see Section 1.2.1) is going to be obscured

Additionally, if the sender knows that the recipient's MUA is capable of interpreting Protected Headers, it SHOULD NOT attempt to include a Legacy Display part. (Signalling such a capability is out of scope for this document)

5.2. Message Rendering: Omitting a Legacy Display Part

A MUA that understands Protected Headers may receive an encrypted message that contains a Legacy Display part. Such an MUA SHOULD avoid rendering the Legacy Display part to the user at all, since it is aware of and can render the actual Protected Headers.

If a Legacy Display part is detected, the Protected Headers should still be pulled from the Cryptographic Payload (part `V` in the example above), but the body of message SHOULD be rendered as though it were only the original body (part `X` in the example above).

5.2.1. Legacy Display Detection Algorithm

A receiving MUA acting on a message SHOULD detect the presence of a Legacy Display part and the corresponding "original body" with the following simple algorithm:

- Check that all of the following are true for the message:
- The Cryptographic Envelope must contain an encrypting Cryptographic Layer

- The Cryptographic Payload must have a `Content-Type` of `multipart/mixed`
- The Cryptographic Payload must have exactly two subparts
- The first subpart of the Cryptographic Payload must have a `Content-Type` of `text/rfc822-headers`
- The first subpart of the Cryptographic Payload's `Content-Type` must contain a property of `protected-headers`, and its value must be `v1`.
- If all of the above are true, then the first subpart is the Legacy Display part, and the second subpart is the "original body". Otherwise, the message does not have a Legacy Display part.

5.3. Legacy Display is Decorative and Transitional

As the above makes clear, the Legacy Display part is strictly decorative, for the benefit of legacy decryption-capable MUAs that may handle the message. As such, the existence of the Legacy Display part and its `multipart/mixed` wrapper are part of a transition plan.

As the number of decryption-capable clients that understand Protected Headers grows in comparison to the number of legacy decryption-capable clients, it is expected that some senders will decide to stop generating Legacy Display parts entirely.

A MUA developer concerned about accessiblity of the Subject header for their users of encrypted mail when Legacy Display parts are omitted SHOULD implement the Protected Headers scheme described in this document.

6. Message Interpretation

This document does not currently provide comprehensive recommendations on how to interpret Protected Headers. This is deliberate; research and development is still ongoing. We also recognize that the tolerance of different user groups for false positives (benign conditions misidentified as security risks), vs. their need for strong protections varies a great deal and different MUAs will take different approaches as a result.

Some common approaches are discussed below.

6.1. Reverse-Copying

One strategy for interpreting Protected Headers on an incoming message is to simply ignore any Exposed Header for which a Protected counterpart is available. This is often implemented as a copy operation (copying header back out of the Cryptographic Payload into the main message header) within the code which takes care of parsing the message.

A MUA implementing this strategy should pay special attention to any user facing headers (Section 1.2.1). If a message has Protected Headers, and a user-facing header is among the Exposed Headers but missing from the Protected Headers, then an MUA implementing this strategy SHOULD delete the identified Exposed Header before presenting the message to the user.

This strategy does not risk raising a false alarm about harmless deviations, but conversely it does nothing to inform the user if they are under attack. This strategy does successfully mitigate and thwart some attacks, including signature replay attacks (Section 11.2) and participant modification attacks (Section 11.3).

6.2. Signature Invalidation

An alternate strategy for interpreting Protected Headers is to consider the cryptographic signature on a message to be invalid if the Exposed Headers deviate from their Protected counterparts.

This state should be presented to the user using the same interface as other signature verification failures.

A MUA implementing this strategy MAY want to make a special exception for the `Subject:` header, to avoid invalidating the signature on any signed and encrypted message with a confidential subject.

Note that simple signature invalidation may be insufficient to defend against a participant modification attack (Section 11.3).

6.3. The Legacy Display Part

This part is purely decorative, for the benefit of any recipient using a legacy decryption-capable MUA. See Section 5.2 for details and recommendations on how to handle the Legacy Display part.

6.4. Replying to a Message with Obscured Headers

When replying to a message, many MUAs copy headers from the original message into their reply.

When replying to an encrypted message, users expect the replying MUA to generate an encrypted message if possible. If encryption is not possible, and the reply will be cleartext, users typically want the MUA to avoid leaking previously-encrypted content into the cleartext of the reply.

For this reason, an MUA replying to an encrypted message with Obscured Headers SHOULD NOT leak the cleartext of any Obscured Headers into the cleartext of the reply, whether encrypted or not.

In particular, the contents of any Obscured Protected Header from the original message SHOULD NOT be placed in the Exposed Headers of the reply message.

7. Common Pitfalls and Guidelines

Among the MUA authors who already implemented most of this specification, several alternative or more encompasing specifications were discussed and sometimes tried out in practice. This section highlights a few "pitfalls" and guidelines based on these discussions and lessons learned.

7.1. Misunderstood Obscured Subjects

There were many discussions around what text phrase to use to obscure the `Subject:`. Text phrases such as `Encrypted Message` were tried but resulted in both localization problems and user confusion.

If the natural language phrase for the obscured `Subject:` is not localized (e.g. just English `Encrypted Message`), then it may be incomprehensible to a non-English-speaking recipient who uses a legacy MUA that renders the obscured `Subject:` directly.

On the other hand, if it is localized based on the sender's MUA language settings, there is no guarantee that the recipient prefers the same language as the sender (consider a German speaker sending English text to an Anglophone). There is no standard way for a sending MUA to infer the language preferred by the recipient (aside from statistical inference of language based on the composed message, which would in turn leak information about the supposedly-confidential message body).

Furthermore, implementors found that the phrase `Encrypted Message` in the subject line was sometimes understood by users to be an indication from the MUA that the message was actually encrypted. In practice, when some MUA failed to encrypt a message in a thread that started off with an obscured `Subject:`, the value `Re: Encrypted Message` was retained even on those cleartext replies, resulting in user confusion.

In contrast, using `...` as the obscured `Subject:` was less likely to be seen as an indicator from the MUA of message encryption, and it also neatly sidesteps the localization problems.

7.2. Reply/Forward Losing Subjects

When the user of a legacy MUA replies to or forwards a message where the Subject has been obscured, it is likely that the new subject will be `Fwd: ...` or `Re: ...` (or the localized equivalent). This breaks an important feature: people are used to continuity of subject within a thread. It is especially unfortunate when a new participant is added to a conversation who never saw the original subject.

At this time, there is no known workaround for this problem. The only solution is to upgrade the MUA to support Protected Headers.

The authors consider this to be only a minor concern in cases where encryption is being used because confidentiality is important. However, in more opportunistic cases, where encryption is being used routinely regardless of the sensitivity of message contents, this cost becomes higher.

7.3. Usability Impact of Reduced Metadata

Many mail user agents maintain an index of message metadata (including header data), which is used to rapidly construct mailbox overviews and search result listings. If the process which generates this index does not have access to the encrypted payload of a message, or does not implement Protected Headers, then the index will only contain the obscured versions Exposed Headers, in particular an obscured Subject of `...`.

For sensitive message content, especially in a hosted MUA-as-a-service situation ("webmail") where the metadata index is maintained and stored by a third party, this may be considered a feature as the subject is protected from the third-party. However, for more routine communications, this harms usability and goes against user expectations.

Two simple workarounds exist for this use case:

1. If the metadata index is considered secure enough to handle confidential data, the protected content may be stored directly in the index once it has been decrypted.
2. If the metadata index is not trusted, the protected content could be re-encrypted and encrypted versions stored in the index instead, which are then decrypted by the client at display time.

In both cases, the process which decrypts the message and processes the Protected Headers must be able to update the metadata index.

FIXME: add notes about research topics and other non-simple workarounds, like oblivious server-side indexing, or searching on encrypted data.

7.4. Usability Impact of Obscured Message-ID

Current MUA implementations rely on the outermost Message-ID for message processing and indexing purposes. This processing often happens before any decryption is even attempted. Attempting to send a message with an obscured Message-ID header would result in several MUAs not correctly processing the message, and would likely be seen as a degradation by users.

Furthermore, a legacy MUA replying to a message with an obscured `Message-ID:` would be likely to produce threading information (`References:`, `In-Reply-To:`) that would be misunderstood by the original sender. Implementors generally disapprove of breaking threads.

7.5. Usability Impact of Obscured From/To/Cc

The impact of obscuring `From:`, `To:`, and `Cc:` headers has similar issues as discussed with obscuring the `Message-ID:` header in Section 7.4.

In addition, obscuring these headers is likely to cause difficulties for a legacy client attempting formulate a correct reply (or "reply all") to a given message.

7.6. Mailing List Header Modifications

Some popular mailing-list implementations will modify the Exposed Headers of a message in specific, benign ways. In particular, it is common to add markers to the `Subject` line, and it is also common to modify either `From` or `Reply-To` in order to make sure replies go to the list instead of directly to the author of an individual post.

Depending on how the MUA resolves discrepancies between the Protected Headers and the Exposed Headers of a received message, these mailing list "features" may either break or the MUA may incorrectly interpret them as a security breach.

Implementors may for this reason choose to implement slightly different strategies for resolving discrepancies, if a message is known to come from such a mailing list. MUAs should at the very least avoid presenting false alarms in such cases.

8. Comparison with Other Header Protection Schemes

Other header protection schemes have been proposed (in the IETF and elsewhere) that are distinct from this mechanism. This section documents the differences between those earlier mechanisms and this one, and hypothesizes why it has seen greater interoperable adoption.

The distinctions include:

- backward compatibility with legacy clients
- compatibility across PGP/MIME and S/MIME
- protection for both confidentiality and signing

8.1. S/MIME 3.1 Header Protection

S/MIME 3.1 ([RFC3851]) introduces header protection via `message/rfc822` header parts.

The problem with this mechanism is that many legacy clients encountering such a message were likely to interpret it as either a forwarded message, or as an unreadable substructure.

For signed messages, this is particularly problematic - a message that would otherwise have been easily readable by a client that knows nothing about signed messages suddenly shows up as a message-within-a-message, just by virtue of signing. This has an impact on *all* clients, whether they are cryptographically-capable or not.

For encrypted messages, whose interpretation only matters on the smaller set of cryptographically-capable legacy clients, the resulting message rendering is awkward at best.

Furthermore, Formulating a reply to such a message on a legacy client can also leave the user with badly-structured quoted and attributed content.

Additionally, a message deliberately forwarded in its own right (without preamble or adjacent explanatory notes) could potentially be confused with a message using the declared structure.

The mechanism described here allows cryptographically-incapable legacy MUAs to read and handle cleartext signed messages without any modifications, and permits cryptographically-capable legacy MUAs to handle encrypted messages without any modifications.

In particular, the Legacy Display part described in {#legacy-display} makes it feasible for a conformant MUA to generate messages with obscured Subject lines that nonetheless give access to the obscured Subject header for recipients with legacy MUAs.

8.2. The Content-Type Property "forwarded=no" {forwarded=no}

Section A.1.2 of [I-D.draft-ietf-lamps-header-protection-requirements-01] refers to a proposal that attempts to mitigate one of the drawbacks of the scheme described in S/MIME 3.1 (Section 8.1).

In particular, using the Content-Type property `forwarded="no"` allows *non-legacy* clients to distinguish between deliberately forwarded messages and those intended to use the defined structure for header protection.

However, this fix has no impact on the confusion experienced by legacy clients.

8.3. pEp Header Protection

[I-D.draft-luck-lamps-pep-header-protection-03] is applicable only to signed+encrypted mail, and does not contemplate protection of signed-only mail.

In addition, the pEp header protection involved for "pEp message format 2" has an additional `multipart/mixed` layer designed to facilitate transfer of OpenPGP Transferable Public Keys, which seems orthogonal to the effort to protect headers.

Finally, that draft suggests that the exposed Subject header be one of "=?utf-8?Q?p=E2=89=A1p?=", "pEp", or "Encrypted message". "pEp" is a mysterious choice for most users, and see Section 7.1 for more commentary on why "Encrypted message" is likely to be problematic.

8.4. DKIM

[RFC6736] offers DKIM, which is often used to sign headers associated with a message.

DKIM is orthogonal to the work described in this document, since it is typically done by the domain operator and not the end user generating the original message. That is, DKIM is not "end-to-end" and does not represent the intent of the entity generating the message.

Furthermore, a DKIM signer does not have access to headers inside an encrypted Cryptographic Layer, and a DKIM verifier cannot effectively use DKIM to verify such confidential headers.

8.5. S/MIME "Secure Headers"

[RFC7508] describes a mechanism that embeds message header fields in the S/MIME signature using ASN.1.

The mechanism proposed in that draft is undefined for use with PGP/MIME. While all S/MIME clients must be able to handle CMS and ASN.1 as well as MIME, a standard that works at the MIME layer itself should be applicable to any MUA that can work with MIME, regardess of whether end-to-end security layers are provided by S/MIME or PGP/MIME.

That mechanism also does not propose a means to provide confidentiality protection for headers within an encrypted-but-not-signed message.

Finally, that mechanism offers no equivalent to the Legacy Display described in Section 5. Instead, sender and receiver are expected to negotiate in some unspecified way to ensure that it is safe to remove or modify Exposed Headers in an encrypted message.

8.6. Triple-Wrapping

[RFC2634] defines "Triple Wrapping" as a means of providing cleartext signatures over signed and encrypted material. This can be used in combination with the mechanism described in [RFC7508] to authenticate some headers for transport using S/MIME.

But it does not offer confidentiality protection for the protected headers, and the signer of the outer layer of a triple-wrapped message may not be the originator of the message either.

In practice on today's Internet, DKIM ([RFC6736] provides a more widely-accepted cryptographic header-verification-for-transport mechanism than triple-wrapped messages.

9. Test Vectors

The subsections below provide example messages that implement the Protected Header scheme.

The secret keys and OpenPGP certificates from [I-D.draft-bre-openpgp-samples-00] can be used to decrypt and verify them.

They are provided in textual source form as [RFC5322] messages.

9.1. Signed Message with Protected Headers

This shows a clearsigned message. Its MIME message structure is:

```
└┬╴multipart/signed
 ├─╴text/plain ← Cryptographic Payload
 └─╴application/pgp-signature
```

Note that if this message had been generated without Protected Headers, then an attacker with access to it could modify the Subject without invalidating the signature. Such an attacker could cause Bob to think that Alice wanted to cancel the contract with BarCorp instead of FooCorp.

```
Received: from localhost (localhost [127.0.0.1]);
 Sun, 20 Oct 2019 09:18:28 -0400 (UTC-04:00)
MIME-Version: 1.0
Content-Type: multipart/signed; boundary="1790868a14";
 protocol="application/pgp-signature"; micalg="pgp-sha512"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Sun, 20 Oct 2019 09:18:11 -0400
Subject: The FooCorp contract
Message-ID: <signed@protected-headers.example>

--1790868a14
Content-Type: text/plain; charset="us-ascii"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Sun, 20 Oct 2019 09:18:11 -0400
Subject: The FooCorp contract
Message-ID: <signed@protected-headers.example>

Bob, we need to cancel this contract.

Please start the necessary processes to make that happen today.

Thanks, Alice
--
Alice Lovelace
President
OpenPGP Example Corp

--1790868a14
content-type: application/pgp-signature

-----BEGIN PGP SIGNATURE-----

wnUEARYKAB0FAl2sXpMWIQTrhbtfozp14V6UTmPyMVUMT0fjjgAKCRDyMVUMT0fj
jq3uAP4/K66bZXT4jFsmKLztz2Ihxjftgf3TaeD2uL05yWdJAQEAjRdWIh35C6MP
utqkLnFeLpkTwrMnncdF/G+so/yXvQA=
=UMd4
-----END PGP SIGNATURE-----

--1790868a14--
```

9.2. Signed and Encrypted Message with Protected Headers

This shows a simple encrypted message with protected headers. The encryption also contains an signature in the OpenPGP Message structure. Its MIME message structure is:

```
└┬╴multipart/encrypted
 ├─╴application/pgp-encrypted
 └─╴application/octet-stream
   ↧ (decrypts to)
   └─╴text/plain ← Cryptographic Payload
```

The `Subject:` header is successfully obscured.

Note that if this message had been generated without Protected Headers, then an attacker with access to it could have read the Subject. Such an attacker would know details about Alice and Bob's business that they wanted to keep confidential.

The protected headers also protect the authenticity of subject line as well.

The session key for this message's crypto layer is an AES-256 key with value `8df4b2d27d5637138ac6de46415661be0bd01ed12ecf8c1db22a33cf3ede82f2` (in hex).

If Bob's MUA is capable of interpreting these protected headers, it should render the `Subject:` of this message as `BarCorp contract signed, let's go!`.

```
Received: from localhost (localhost [127.0.0.1]);
 Mon, 21 Oct 2019 07:18:39 -0700 (UTC-07:00)
MIME-Version: 1.0
Content-Type: multipart/encrypted; boundary="bcde3ce988";
 protocol="application/pgp-encrypted"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Message-ID: <signed+encrypted@protected-headers.example>
Subject: ...

--bcde3ce988
content-type: application/pgp-encrypted

Version: 1

--bcde3ce988
content-type: application/octet-stream

-----BEGIN PGP MESSAGE-----

wV4DR2b2udXyHrYSAQdAifmSGlN6dUG8WjtsDsVf3RoFUu69cEhUQyVMaUBEaSAw
EAtGxmoM2YY6y/87UXI2USJMj9PiFn7RuV0pAFVT6NwMAY1JgLX5qoSdKXuLZ9CA
wcDMA3wvqk35PDeyAQv9HNVhvGMSyCXZjsu5LlLGPF/6XHnk3PtunCo8GpUd7Mg9
zVDS0zK+dtePYHNgKZ47KLDBgu6XInVBWeeSkImaWjFirTmqp/GP20urKQ/phSkC
vI88cEH+fCqeFxDcL5tb0RLm3/iv707CHvoOM2qCbV8WDSSvNY2FGlJZqqGO3mkE
VhZFytVop12c/L5+PltIS0/P25KMoSuIIb9xenAncyLZ1a2M/NsgZjBqWeXFfQnZ
ssMK1xOvNIYNxUzEws+U6un74OE5sBZeZCvM/nIf50iXvEQMxoc/MX2XFUA9Scid
+bmy9nZCit0KQNk4ikrshgtxmG6xJfMv1IpnscQwMy9KfOAhnrVWFVHpzr+K7mXb
yHHF4Ov1Cl2FvwHU6DujaoApkn/xg5BjbRZxfRfVF7LvZ3UJJ/v1XzGLv5LTL8Fr
1S+Ql69M8yvftMiZ799dNgOT7jc4CY5yN7P2YQn5Z3Nm/gUWcGwuqwQecw0hs/87
yCQzkDHAC62LL6+zHqc20sHbAeuQHcGttI9Vu8rEO+5OeDr3WjTB/UXvLKr/G9ty
LUpaYYwFtNgMaRAg0niMV9xfwTFjLBmNkq/8N0mAOsZSO9lMZyUIfBiFbw5yWNzx
TuKxZymZ3ts6ywvKOgzLNgF+AdtTQk5nkNIsh7Fd02RSl9heF3t47FXVSvBSo5KI
FXuznjzK7VNl8fTp9MpBwp00Dai3jtKGQ3/XGiD4l/wa/QxfffojPAZ9UZpgA2Xx
Uw3W4+zCNZNJ35QME6I2ysKwbgAQGFeKM57lLXrmIJWU7KEIDnc1MCBwsSt50yB8
kIdSPXxK/Jon2wbATUN8Uuo3oLA2dpH8XncjrkqTooNjkK3uPrGNphDBVSMA5W5Z
deHc9NmzETXLBPysc0LHWMUO8g4YnWB4sLq9ZBxTYYX9CYRJvdB8EZN4Dq+IUDVK
W7Hu8oFkPRqU7oVa+utiZq5YvTXbIMJBWdUa8r8zlwz0jVsUJGBIPDWhs8Yse2JX
54dNJRAy2X5M3KM1S2Aat1gHl35cft5pLYLp5/gs7GYgybhYfgXbcbBHE6/XTAtg
L7ZbzN+AEDu24uPQaTN5jUA8MfQIkksRgIhZN3N8NBVltv4t+tbtIiaLLaQ/7Wdd
X0BINwZxhBZHEtjljqf4VE4RlWpMriW+ezcrPU3zEcM62knjeCLCh9iseAuz1J1o
R1o4DKwlVY9dJZigguO9kzz+K9n1/mpn8orV9kn5FyH9vs9ZF+RQiSHgpoZ3TKER
iy4T7WPV1WzyPSTmlKktOGjgJ5nszKw8YarMjtXYiPNOShBWuBTxBeSyjCLhZ85m
YAhS1znrJ9CzX3jjaZTHTd/5gYN7wVByUlw9OkyN2QQRFl6fg1xN6Tb79oGxDqh/
BHb6PBgDtwnGmHdDmw==
=rTjd
-----END PGP MESSAGE-----

--bcde3ce988--
```

Unwrapping the Cryptographic Layer yields the following content:

```
Content-Type: text/plain; charset="us-ascii"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Subject: BarCorp contract signed, let's go!
Message-ID: <signed+encrypted@protected-headers.example>

Hi Bob!

I just signed the contract with BarCorp and they've set us up with
an account on their system for testing.

The account information is:

        Site: https://barcorp.example/
    Username: examplecorptest
    Password: correct-horse-battery-staple

Please get the account set up and apply the test harness.

Let me know when you've got some results.

Thanks, Alice
--
Alice Lovelace
President
OpenPGP Example Corp
```

9.3. Signed and Encrypted Message with Protected Headers and Legacy Display Part

If Alice's MUA wasn't sure whether Bob's MUA would know to render the obscured `Subject:` header correctly, it might include a legacy display part in the cryptographic payload.

This message is structured in the following way:

```
└┬╴multipart/encrypted
 ├─╴application/pgp-encrypted
 └─╴application/octet-stream
   ↧ (decrypts to)
   └┬╴multipart/mixed ← Cryptographic Payload
    ├─╴text/rfc822-headers ← Legacy Display Part
    └─╴text/plain
```

The example below shows the same message as Section 9.2.

If Bob's MUA is capable of handling protected headers, the two messages should render in the same way as the message in Section 9.2, because it will know to omit the Legacy Display part as documented in Section 5.2.

But if Bob's MUA is capable of decryption but is unaware of protected headers, it will likely render the Legacy Display part for him so that he can at least see the originally-intended `Subject:` line.

For this message, the session key is an AES-256 key with value `95a71b0e344cce43a4dd52c5fd01deec5118290bfd0792a8a733c653a12d223e` (in hex).

```
Received: from localhost (localhost [127.0.0.1]);
 Mon, 21 Oct 2019 07:18:39 -0700 (UTC-07:00)
MIME-Version: 1.0
Content-Type: multipart/encrypted; boundary="8f1c37571f";
 protocol="application/pgp-encrypted"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Message-ID: <sign+enc+legacy-display@protected-headers.example>
Subject: ...

--8f1c37571f
content-type: application/pgp-encrypted

Version: 1

--8f1c37571f
content-type: application/octet-stream

-----BEGIN PGP MESSAGE-----

wV4DR2b2udXyHrYSAQdARLfz+1WBB1rOgBFbyrPQXZkCoiK/aA7SpG8mY39S8Tow
cuEVQ1/a4B0VfwiKMyXomehg4GMo7akIAd7nh1LIG26eW+JeEjOJLhjrcg4x5Cg/
wcDMA3wvqk35PDeyAQv9Hu30CZtCMGeHCVyvPeZZuYUWtHDADt4Wo3rg5va5bUu1
nZCV/7vo9worPUvhN+qqLP0t4l0KbdklNofLKggJt/+LgJ/IvJv4KhwK6PR10Cba
Lu2uyzUJK33WKCnvPzqsgEuE4OmbGcIZki3Bo+hKLgr0wS1sNi5okybM5JMmrqTw
GXEmHdtohx4/YFsAJ++b4WEWb26jflBbj7NwyXdAESb/lcxi5ZKqXerRJiaN2X/x
O/CiwZwSw3LA7VlCwN8Jb9AR4KjjFHIi6pUOp5S7Iz0Hs0juA6862gsuOrfGN8q8
1KkTUPwAw0lQSnSpMxsnRS3+zv1aeWnm8K+bt1Q0E/Nl1E0GYtwiEBLVWX1ZQYCr
DgrgFBl3/kvx8e+L+b6bEF9GVckZSGrkzJJeMx1JzGaR5MtkEJThsZAlyrJVpMuf
un4N1Xy11G3IWNMCl8SfvPdnaSrytVej2s3ItL+0sxy3wi4hhCXle/YJuFwPTbEP
G8jkjJknuVd/6kxf85mT0sI1AfS//hCeieoyi9cjeBVGh39z7bonD2bSp5RfYKI5
ANj5ANV+hWeB8TGmI7Ka6OOU/43MuilIRAu79M+XnFjMqDQWmRLhydgkThdc63+l
LTt4jZRnUI2IjxsZ5Bgc13agpWzsStJcjRYz8QWOoANc+A74MCX75gsFn8NbQknR
xa/rXpMEF6TulvgCtV/tDCXOv2hnpu+JhIqwLgKIspJih60R8oSIr5qzX3B4AAcc
8Lr3cGrlohVtMDUYUkQF81+KsBWKJZWEvhZdQZC2nSzJSx5hgmw0D6ybYSGuCh9Z
MyZbH38HJnwkZQWUYPyg4ui8XFi0PVY1WignaF6l0D0DhklzgkzO0Ey1BvEu4Zdg
jkfUjYD4VnXNd4UyIwycfo8myrx3fqd5WcZRJmX9Njhlwn3a4l0adZlTIG9S0ytP
VW9jijjGQ+IhizH+Q4jErcEuHJhNDCD0xOIpjQz68/NDm94BDmI2dyr07YOrQEQa
ahDl7vMfMFQVncGp4zY0kYmNDOPSG3djCU5OhKA6dRz8cmigxvW0/CzMrOArMso3
oW+EjldvkQIgeDwodARO8OLKKdQBQhcWIV4G3R8oaLXDxbP/3XAx7eU53jPi0ahW
PbcD7IfHdrVVTyKLcolb0MqnP12gtnCmOwqWSA3D0aeuRGxIKCLnMVMID3I7OVjb
1PMpXs4EsgIuVxWbm0qibVrw9yYd/4xRKKdZqYP+PCSo4aQEMzW7U+mWiZUmDE07
4xzZlTd1qBRUgBKdteNjOcZ859hPZGREuG++JKBrL5Yr/kVBf8UFGLPES+8vslg3
zMQ9K2FO50o4LxYyaKZEW9ihk2BbGB60+hiimtbpWjqZ79qZZ3PJqzd2Au7da7x4
jKhOSvFAoLyze+8l2m+8uzGAQTh/1k6e3O6UcwdrV5Z4i41LZp2qdD7WBSfZD1tv
IdvtbwnZ7YlLr/X0ESERPW4WWrDlHq4SDt5H16hgAbXVfYwmHxgAPawnIRLYVqZ6
ViIf7Hfaqg==
=QAR/
-----END PGP MESSAGE-----

--8f1c37571f--
```

Unwrapping the Cryptographic Layer yields the following content:

```
Content-Type: multipart/mixed; boundary="6ae0cc9247"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Subject: BarCorp contract signed, let's go!
Message-ID: <sign+enc+legacy-display@protected-headers.example>

--6ae0cc9247
content-type: text/rfc822-headers; protected-headers="v1"
Content-Disposition: inline

Subject: BarCorp contract signed, let's go!

--6ae0cc9247
Content-Type: text/plain; charset="us-ascii"

Hi Bob!

I just signed the contract with BarCorp and they've set us up with
an account on their system for testing.

The account information is:

        Site: https://barcorp.example/
    Username: examplecorptest
    Password: correct-horse-battery-staple

Please get the account set up and apply the test harness.

Let me know when you've got some results.

Thanks, Alice
--
Alice Lovelace
President
OpenPGP Example Corp

--6ae0cc9247--
```

9.4. Multilayer Message with Protected Headers

Some mailers may generate signed and encrypted messages with a multilayer cryptographic envelope. We show here how such a mailer might generate the same message as Section 9.2.

A typical message like this has the following structure:

```
└┬╴multipart/encrypted
 ├─╴application/pgp-encrypted
 └─╴application/octet-stream
  ↧ (decrypts to)
  └┬╴multipart/signed
   ├─╴text/plain ← Cryptographic Payload
   └─╴application/pgp-signature
```

For this message, the session key is an AES-256 key with value `5e67165ed1516333daeba32044f88fd75d4a9485a563d14705e41d31fb61a9e9` (in hex).

```
Received: from localhost (localhost [127.0.0.1]);
 Mon, 21 Oct 2019 07:18:39 -0700 (UTC-07:00)
MIME-Version: 1.0
Content-Type: multipart/encrypted; boundary="15d01ebd43";
 protocol="application/pgp-encrypted"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Message-ID: <multilayer@protected-headers.example>
Subject: ...

--15d01ebd43
content-type: application/pgp-encrypted

Version: 1

--15d01ebd43
content-type: application/octet-stream

-----BEGIN PGP MESSAGE-----

wV4DR2b2udXyHrYSAQdAOgQDEkyc6EDXP9maqDSnaxSKQ5Cli2idlkJr/fiRJUkw
FBc7t5vaz9x2HIE1M87W8fljvfK9HQIcLRxLo4kba3ZI7wLbDUSQP5SXzV2agnf5
wcDMA3wvqk35PDeyAQv7BFf4oXdwgK7+GaFykpweiQV9PtdzyQUyAZKTjblmH53S
bURXXxQaJVs1v5sqM85WMwgBbCQw2Gjs2K9l4JBWubC/ROO2AKG8odPaj1XA+FW4
cW3jP1G/hoHRhTsWFOYQm/+1lfa7DRt5WVPkIBSHECHP7NW5slLB0uGJaeopU4bY
ZY+65r3ZV3ieTkexwEVkcAdLHGzgpCXyYfj1JwLWWHAuJv96K137Q37J36g9T8wR
hlkIDRqIorY2IexI2lv/PsEHXrzUw4RT4HllriGmHmRJA45QoijnFA3ei+IuhIPm
OcQmlyICZL40fznOaRWYHqp2oLaJ8OVHTU/ZAYurVj+0vsc7qcfxF69S9LvTSInu
CtcamqybdH56wd575OdFKKcng75M19ttIXNguejwMJR0ERL/4xh0y5oN9v5fYzUM
LiK1HIBTjY9JW/jbeqr+InuwTAEvh7Vfzjg+8bMhJMVnTgjea3FSdcfxsrnsZp30
JY6SC70on74Di/zmBg1Z0sIxAVYh7Vc++W0eUIeEj+Azc4mIfaDZ5U3hHk1OV8Lt
XCJz6r/KzUuy3bogwhVUL76kMvuKw/3zQ5zI2YYDpAybsXtUhVA6hg6Zy4JTtJEU
+Z0H0a2EU3CYPBG+ic0PzxAdTz7iDb9AvwpRgWJrgBQmZ5J8bWjgvRTKdt7e2cz8
0ESrfetg+VSEJLWWipNZNzNGaHlUO7ypgwjYYKfX0VAq5rhWCk8079/n4Xzcn9mt
9UaqfjvaV6FuRDFTW1YVkVJdndnC9vQzkHVb6MPFA4fp5H3aY/j3yvMa5YaePv1v
3zA70nuFbe6j1RQO6KhiJBJA7x+MtnZFt6xByhdImVloSr7c9kfuRaFQ83YbwM5I
vjrz29jB8+jG9msFeJ75ajFKpUiN1yVOltTQg+WS28osD3irb461X5YtJCCuD8+d
i6EA7W9P/Hr1YJsaH1wFxYqEpvSClpHWUD/nMbUUWmhvTQ75yJyF1BDfEPmaHhsd
vRBVkZgKdSUo8uNRsSakVWe+4D0U92P0kPyZog6LOOq5EILXnmtZpri6zGt0evgV
qEc316nfQeWRism2KJot83TXIov6KIliB4THBo1Chnp/eCs634B4KF2Z1K2N4AHf
8nIIfpJw60VqPrmOzUUvyabiqrebEkhJ7ZHesZJI+OL8UbaAFklaHMHv6PYWDyBl
7XEwRV8MxqMADd094p5sPXOhj4kbCvHCAY08NFPGIPFVUuwE0YRvRhtVaqMVwf/o
AHO6lGMdQqw1NhmRHkcdLK9qVdZvg5MPwm5w6n8/JvvsHkAVDpsBmvX9jeajI1pq
X6b2cn/G9uNCM1K8zsYIbM/RMM1ILmTh1rgQjFc8S1xE2pQNydegk0JaQz/IqbAa
GZy153vaUNzWSku5Ef3AjFP7YTyB+WRR+AHkAg2UawJq8FXR+KYMjWkg0BPBmhE+
TXXt8IYUE0uudIAHplt4RWXfr1dfZH2UODdl2ZNyQExtPfTE4VUYtpCIrgSAERKD
QBjq
=ME+d
-----END PGP MESSAGE-----

--15d01ebd43--
```

Unwrapping the encryption Cryptographic Layer yields the following content:

```
Content-Type: multipart/signed; boundary="a6b911f1d1";
 protocol="application/pgp-signature"; micalg="pgp-sha512"

--a6b911f1d1
Content-Type: text/plain; charset="us-ascii"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Subject: BarCorp contract signed, let's go!
Message-ID: <multilayer@protected-headers.example>

Hi Bob!

I just signed the contract with BarCorp and they've set us up with
an account on their system for testing.

The account information is:

        Site: https://barcorp.example/
    Username: examplecorptest
    Password: correct-horse-battery-staple

Please get the account set up and apply the test harness.

Let me know when you've got some results.

Thanks, Alice
--
Alice Lovelace
President
OpenPGP Example Corp

--a6b911f1d1
content-type: application/pgp-signature

-----BEGIN PGP SIGNATURE-----

wnUEARYKAB0FAl2tviMWIQTrhbtfozp14V6UTmPyMVUMT0fjjgAKCRDyMVUMT0fj
jv/lAP95zG/boihWaRRYusB5KInnMqz8DM9CrxCO/Z67FoZvQAD/WJKfIW/UaBaG
TvwLcfdYDnHVFi/sLCPzP7/+Rp/prQU=
=X47R
-----END PGP SIGNATURE-----

--a6b911f1d1--
```

Note the placement of the Protected Headers on the Cryptographic Payload specifically, which is not the immediate child of the encryption Cryptographic Layer.

9.5. Multilayer Message with Protected Headers and Legacy Display Part

And, a mailer that generates a multilayer cryptographic envelope might want to provide a Legacy Display part, if it is unsure of the capabilities of the recipient's MUA. We show here how sucha mailer might generate the same message as Section 9.2.

Such a message might have the following structure:

```
└┬╴multipart/encrypted
 ├─╴application/pgp-encrypted
 └─╴application/octet-stream
   ↧ (decrypts to)
   └┬╴multipart/signed
    ├┬╴multipart/mixed ← Cryptographic Payload
    │├─╴text/rfc822-headers ← Legacy Display Part
    │└─╴text/plain
    └─╴application/pgp-signature
```

For this message, the session key is an AES-256 key with value `b346a2a50fa0cf62895b74e8c0d2ad9e3ee1f02b5d564c77d879caaee7a0aa70` (in hex).

```
Received: from localhost (localhost [127.0.0.1]);
 Mon, 21 Oct 2019 07:18:39 -0700 (UTC-07:00)
MIME-Version: 1.0
Content-Type: multipart/encrypted; boundary="750bb87f7c";
 protocol="application/pgp-encrypted"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Message-ID: <multilayer+legacy-display@protected-headers.example>
Subject: ...

--750bb87f7c
content-type: application/pgp-encrypted

Version: 1

--750bb87f7c
content-type: application/octet-stream

-----BEGIN PGP MESSAGE-----

wV4DR2b2udXyHrYSAQdAl9YvLLNZzswNHPuBf0LHXgrp7l6MvJ4bc1tgPZD8XGww
mbzTgolXvZe/1NewcfrKpEr2dxQikm9XqvzdODcunsca++c+6sgDGNMNEzSgivaO
wcDMA3wvqk35PDeyAQv/ZKJLN7S79WnezPjzy6RKJi6qPQgKR3X8zfZsnGCw7ooA
Bx5zk+sO2XHM+ho8YJ0HAULkBvzXbDgRoe4VO1kn06nwYBzMnyotNcNf7p6KSfkB
ypiBZ3Orr/0fVaXoStNZfTFp+UqPNw0fVtbTyZRZ0AXmmxVbGPjxjb6m/qRWj26k
0sNb/ruYPzpBEkBdMlK+wYlJHtwyV9gyXU7U33o0UrSf/CcnQcXmJ+OkJbEjUNW/
MHN69jVY8WC9nOgL98qGLtqQwFaxBEemRCoh3PU4Qw52HHpSJBRJuWb/WjACQ9Ds
wGjg5Q2lBUosnaFUvIFg+eP+aqshSEtSYMXHmERysA7hY91R9YSncPpAjTeb298N
XTKlBmvM6JCT21Ur3y2mi8NmQdmn6J3Pa88MwNpUnJ3yWjNPJZVvbFUkseD3+sDL
oLmxil75U8GoB1YxHoX7TTrkkkHPEJ6jlz3sjOXWByOEfuarSjlwn+QiFFGCMpSJ
0TMye28sCTMs4X6eJSqi0sJ9AU7ecIHNwq9IhMtYcK+6xnY9C9uBoNfnHpigzHj/
vq0mBnpvEMf9GkUNbkrzwwMu6wFaTSrcvAQjPN+llgvfI1B+lFhOloQJU3Rpuqop
aOoj7LWoocdeCNQINUkflbX0nFf3sLs4lOT/RwfHauwr2PMb2umBNi4ML0gKfj+D
eSoHqiKhDT2USVt1Kt/KnRC1KSd7lAf6U9rvyWA++w8V/gqt7PNVBREem9Ek8AEA
o9uM37nBJuyJSlA6Tqo2GDw603izKbz8A+JlvWyUQWE106nqBX/LMkkm8zhl045+
EUfKJGIMHFhEWaayPtLFtU1cDvFh2OeZftF1qN451RpWRDwEIVeA6IngotWAaejU
QPLXtDvXKC8O2vIcdI95M+x9yq3or40KS0stZVQAgLZWiXFvvqwyTc+fiby2LYzv
/JPVH3f+F3Vz229u/iob6mgLe3O1Xa2bhcwFqFG1AlpMx2f/ZJsBvYUJ4MMBM/S9
xJ4QPna6oHilBfs72Y2pyCrG6KIBIeWkVd6XhLKaFq5QtKM/rO8IOFtgU7iiJYwD
ZIyVqaV8weaRSF5uGWH2Mc+6/hSeQ+yx8h4sa26KkIwooHbSnx3sjefAB29h013G
8n7u/T375w5Y3J3bHpM888BXUNJh0J+Yiey9PNIEljp577PLBv8sKP0FVpxxfxPO
BFaSoJGiba1GqjJfLRsf3ExeA+ocrnuFfo6x+kyZ7zd0+4+jIQ6fQtF5dnoBbHLA
iTyFZm24994qSOoOoZGEBA5DFsGktAEDfrD8mNYQR9ubY14zlhcOZblQ34w4WsTS
C7olDgoWjos3UQggh+HN+ulp5BO+xTwCVCB85VoVH6pEIZ2IWcAo+R21OMIjyX5d
aE8p3tcqQAGbdPsDR/WRTd/fvNLmEzLDv18ZuglY6b+f0qErG5ce1AJpEhsFZuiX
2oCxVpmURf0T7j7EdrCC8Bhjaq5fw1PPp9Azqv7csYidhmeAw9NetwVo2+fg0H1z
m7sB3QI2qqw4/5ErrKZ1CV109eMOUFMuM+fiJEu+vuXBayvviCPkz0pWHUmjexWS
ISKPpt8ok3hLpojbNf96lDxChlpqaILSL6SopTicnw==
=h5ce
-----END PGP MESSAGE-----

--750bb87f7c--
```

Unwrapping the encryption Cryptographic Layer yields the following content:

```
Content-Type: multipart/signed; boundary="4e3b9ccaba";
 protocol="application/pgp-signature"; micalg="pgp-sha512"

--4e3b9ccaba
Content-Type: multipart/mixed; boundary="6ae0cc9247"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Subject: BarCorp contract signed, let's go!
Message-ID: <multilayer+legacy-display@protected-headers.example>

--6ae0cc9247
content-type: text/rfc822-headers; protected-headers="v1"
Content-Disposition: inline

Subject: BarCorp contract signed, let's go!

--6ae0cc9247
Content-Type: text/plain; charset="us-ascii"

Hi Bob!

I just signed the contract with BarCorp and they've set us up with
an account on their system for testing.

The account information is:

        Site: https://barcorp.example/
    Username: examplecorptest
    Password: correct-horse-battery-staple

Please get the account set up and apply the test harness.

Let me know when you've got some results.

Thanks, Alice
--
Alice Lovelace
President
OpenPGP Example Corp

--6ae0cc9247--

--4e3b9ccaba
content-type: application/pgp-signature

-----BEGIN PGP SIGNATURE-----

wnUEARYKAB0FAl2tviMWIQTrhbtfozp14V6UTmPyMVUMT0fjjgAKCRDyMVUMT0fj
jj/AAQDqeRa+AaS9dHoYHE4sSGhnXfuTlB9WPbtI/3uLmpX4wgD/boo2TFUJ4VYs
KPDOt/ekjp079bvvfcSjpLNEI1sfSwA=
=Otfk
-----END PGP SIGNATURE-----

--4e3b9ccaba--
```

9.6. An Unfortunately Complex Example

For all of the potential complexity of the Cryptographic Envelope, the Cryptographic Payload itself can be complex. The Cryptographic Envelope in this example is the same as the previous example (Section 9.5). The Cryptographic Payload has protected headers and a legacy display part (also the same as Section 9.5), but in addition Alice's MUA composes a message with both plaintext and HTML variants, and Alice includes a single attachment as well.

While this message is complex, a modern MUA could also plausibly generate such a structure based on reasonable commands from the user composing the message (e.g., Alice composes the message with a rich text editor, and attaches a file to the message).

The key takeaway of this example is that the complexity of the Cryptographic Payload (which may contain a Legacy Display part) is independent of and distinct from the complexity of the Cryptographic Envelope.

This message has the following structure:

```
└┬╴multipart/encrypted
 ├─╴application/pgp-encrypted
 └─╴application/octet-stream
  ↧ (decrypts to)
  └┬╴multipart/signed
   ├┬╴multipart/mixed ← Cryptographic Payload
   │├─╴text/rfc822-headers ← Legacy Display Part
   │└┬╴multipart/mixed
   │ ├┬╴multipart/alternative
   │ │├─╴text/plain
   │ │└─╴text/html
   │ └─╴text/x-diff ← attachment
   └─╴application/pgp-signature
```

For this message, the session key is an AES-256 key with value `1c489cfad9f3c0bf3214bf34e6da42b7f64005e59726baa1b17ffdefe6ecbb52` (in hex).

```
Received: from localhost (localhost [127.0.0.1]);
 Mon, 21 Oct 2019 07:18:39 -0700 (UTC-07:00)
MIME-Version: 1.0
Content-Type: multipart/encrypted; boundary="241c1d8182";
 protocol="application/pgp-encrypted"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Message-ID: <unfortunately-complex@protected-headers.example>
Subject: ...

--241c1d8182
content-type: application/pgp-encrypted

Version: 1

--241c1d8182
content-type: application/octet-stream

-----BEGIN PGP MESSAGE-----
```

```
wV4DR2b2udXyHrYSAQdAp4ZrYIrBddsWr41zuxkG+58YgQDeKk1h+gHTz1BmVFMw
oLGI9dIR1LEgCm7FGTB61oXa4JqxSM1+h6q+UFGHjypGMj0/E+BABTgoC7CuYrAr
wcDMA3wvqk35PDeyAQv9EHLWRWMLLSkSJSEqNuywgnAN2I+i6WaCou7t/vP0Looz
/VePnARGcwi6b4RSQYaClf95SOiqzqD56hiXW5yb+2r057HSvAVZ78r0ymCFN83Y
nu9Byy3vulvqueP1PgqJmBY0u5eJjgtCGQs2YM1bb++hyPFHPNsgJuAkB8YwSmqk
aIrFRi2YZXd61Zhvdl58f/ECFMkpmSQRROxddFSXjt/nFXXimWQFP4Jp/m1VjCBF
ne5bQpOdrBjWXWds7zUnFspCtj4RinFI7UjyLR9VelOkezyc58nAIgTdjD0wrp+g
RBdNBGSpoBMBj4t6gVCNMFQL04/UhwQmwl+R0gFDwd2XdJPa9ijCyxROFP9CNcNN
x1Jq+SgkdJMJLbsyWlF8GvioNOMg0cgSEoyXtwCBZV3IpXdMt1SmMAhEv6mmWR4t
zI6BJ3i0dX/y+djz93uj0Ty2fmd/h//OaI5JMn+muhNss4tRRHhNistqyjFO6qaj
cadwj/QetMWVAR8e8lDc0sPeASPx9QMDzFWI+joVIKZ7oAvHw6WArpS+Gu9rhIB6
aa9Xn0dn4l/xYDzFvZqSgVasL7+BFj1NZtdgvdgvLd/ACfAW4G5XvrQ+dEHW/p2n
oVP58W7jKMJNwDxZva1fwNb+6eWwkGVhzI11uX6n0mtL6UpfFYLfirSD/Z/IpMos
sJ1RCnox60W1JardwXIkx5rFgtHgFb9hUyyZKC6VXstuIoSAtlc7NCRsSwuP5PGY
f0g3ttgivLMZOV9Oankqijol6jFDUrNAZJrLZKYYs0AhIkWoDlwhsK4bWSyEk7Zc
BPR033MgGpY4CCadEWPZL4n5vhUsYnBr9LihKDzDWZzdU/5YQpM8OuLqqk9mxsuo
Oim8HPkJ2z1Itw58UIW23cqVXz8uKtEsywNSv8VlM2IVG9jHvhmnK4laZN2U+1bp
KIY9giBFlCqxSjyx2Knq2C7HaBelWjqaGUkH1YOdsnCKEj/JRJYo4ogOLy4xSHEz
8gaDQZjyHLICvsrL84RzDfxx+yWid0Gzzzf69/ux0bATkUXN5tMy4h2p15Fm8LtK
9IAQjiByqf0FKvfQLt8SleNMDPvBfscTCNb+N7aLoJARto2oLHyes8AxM18c4Qb+
ihNpDwtIvXUN9dn6moylna0Y2eo6zjGWK/bxKVvlNakwxtVOLHxpj1xuNiQC5LJR
n0rHsHOUZQUWTfgp+N8vdwMOJhLyD1yTiCbzrtuw+QYRCXBNBSkc1Jtr6yCESKr/
1ef03Ygtb0G/H0I6KDLVdrrc0TjjkD98hjILMc953coF4a3yKJOWoLGOWrWup+IX
kiax2FlJ3b13PZODENVfdhQ4ACKUTrl3eZNepZmwzVK8z8CPlQbRYEo7sET0IEBp
Vo7VnLeeUZzNOqwZkyipRNRfkQzMmTjbNZeKvsCQsoZx2goo1Pm7XG093z34RcK8
HHsrEvY7kymXoU1xS2gQYQcoiq4LBY42HJ/+mXcEKqSUuwINYVhlwutFL23T1uvp
9/eY6jyn5cc+QSCZMIf5MRKKruc13xzs/WaxVFd2NfLAghtlqqZj1ziKZ3XRLlwc
pesR9415yGakbBC2C5HwUOhHvv5NMuX4S2UHOiRX+XQzzEOafBekRCHAOXPfbTEm
Xj7wPJVSXS7vCV3K+2scAZopuOJMIOkegcJAsuata2GiHr2TbcRbMAZSQzrQ/wSe
GbkgLHSthKEXVEbkYMTHSDPClpThppfD40mBIHyhw3BbC8j3lVgEZ1EeXyJuhZDu
VzPeRxYD9Yun6UOYYbjBSiWNe59DylN1ZBTICgymnff+utfW94UXs93FGRGgSpNB
c8Jc3tlKd7VP+FlEKBmqFHRzE7fdnabQ3BUBnPdBwjkFqImVOLwwKEZ8MRowDjfu
tcjpUEvROWi/FORqmkZHik7AqfuCO4cB3g5AePYfweIEONXxK7yjjpGlmfNgVLBa
uHlSSNl7/oIRP1ivCNEUmmMbqvKnjrTx7i/0XKdHeyGMpVSaksH4Nj+Wz7jA+65K
iEhVOC2QoKSlI5W7v9fAQXCtNfXWlrrVSAqxk74rpIErdip8SpJloGOvtVtApi19
=p3e5
```

```
-----END PGP MESSAGE-----

--241c1d8182--
```

Unwrapping the encryption Cryptographic Layer yields the following content:

```
Content-Type: multipart/signed; boundary="c72d4fa142";
 protocol="application/pgp-signature"; micalg="pgp-sha512"

--c72d4fa142
Content-Type: multipart/mixed; boundary="6ae0cc9247"
From: Alice Lovelace <alice@openpgp.example>
To: Bob Babbage <bob@openpgp.example>
Date: Mon, 21 Oct 2019 07:18:11 -0700
Subject: BarCorp contract signed, let's go!
Message-ID: <unfortunately-complex@protected-headers.example>

--6ae0cc9247
content-type: text/rfc822-headers; protected-headers="v1"
Content-Disposition: inline

Subject: BarCorp contract signed, let's go!

--6ae0cc9247
Content-Type: multipart/mixed; boundary="8dfc0e9ecf"

--8dfc0e9ecf
Content-Type: multipart/alternative; boundary="32c4d5a901"

--32c4d5a901
Content-Type: text/plain; charset="us-ascii"

Hi Bob!

I just signed the contract with BarCorp and they've set us up with
an account on their system for testing.

The account information is:

        Site: https://barcorp.example/
    Username: examplecorptest
    Password: correct-horse-battery-staple

Please get the account set up and apply the test harness.

Let me know when you've got some results.

Thanks, Alice
--
Alice Lovelace
President
OpenPGP Example Corp

--32c4d5a901
Content-Type: text/html; charset="us-ascii"

<html><head></head><body><p>Hi Bob!
</p><p>
I just signed the contract with BarCorp and they've set us up with
 an account on their system for testing.
</p><p>
The account information is:
</p><dl>
```

```
<dt>Site</dt><dd>
<a href="https://barcorp.example/">https://barcorp.example/</a>
</dd>
<dt>Username</dt><dd><tt>examplecorptest</tt></dd>
<dt>Password</dt><dd>correct-horse-battery-staple</dd>
</dl><p>
Please get the account set up and apply the test harness.
</p><p>
Let me know when you've got some results.
</p><p>
Thanks, Alice<br/>
-- <br/>
Alice Lovelace<br/>
President<br/>
OpenPGP Example Corp<br/>
</p></body></html>

--32c4d5a901--

--8dfc0e9ecf
Content-Type: text/x-diff; charset="us-ascii"
Content-Disposition: inline; filename="testharness-config.diff"

diff -ruN a/testharness.cfg b/testharness.cfg
--- a/testharness.cfg
+++ b/testharness.cfg
@@ -13,3 +13,8 @@
 endpoint = https://openpgp.example/test/
 username = testuser
 password = MJVMZlHR75mILg
+
+[barcorp]
+endpoint = https://barcorp.example/
+username = examplecorptest
+password = correct-horse-battery-staple

--8dfc0e9ecf--

--6ae0cc9247--

--c72d4fa142
content-type: application/pgp-signature

-----BEGIN PGP SIGNATURE-----

wnUEARYKAB0FAl2tviMWIQTrhbtfozp14V6UTmPyMVUMT0fjjgAKCRDyMVUMT0fj
jrR3AP9H2o1HBGLwkz5qzBgGmXsXLrc2xbluWtYmiDQcnq3e9QEA+DaBG1gEXasg
7OfAEqT4DrOivtNo18CxpIPrskgOXws=
=Ul2/
-----END PGP SIGNATURE-----

--c72d4fa142--
```

10. IANA Considerations

FIXME: register content-type parameter for legacy-display part

MAYBE: provide a list of user-facing headers, or a new "user-facing" column in some table of known RFC5322 headers?

MAYBE: provide a comparable indicator for which headers are "structural" ?

11. Security Considerations

This document describes a technique that can be used to defend against two security vulnerabilities in traditional end-to-end encrypted e-mail.

11.1. Subject Leak

While e-mail structure considers the Subject header to be part of the message metadata, nearly all users consider the Subject header to be part of the message content.

As such, a user sending end-to-end encrypted e-mail may inadvertently leak sensitive material in the Subject line.

If the user's MUA uses Protected Headers and obscures the Subject header as described in Section 4.2 then they can avoid this breach of confidentiality.

11.2. Signature Replay

A message without Protected Headers may be subject to a signature replay attack, which attempts to violate the recipient's expectations about message authenticity and integrity. Such an attack works by taking a message delivered in one context (e.g., to someone else, at a different time, with a different subject, in reply to a different message), and replaying it with different message headers.

A MUA that generates all its signed messages with Protected Headers gives recipients the opportunity to avoid falling victim to this attack.

Guidance for how a message recipient can use Protected Headers to defend against a signature replay attack are out of scope for this document.

11.3. Participant Modification

A trivial (if detectable) attack by an active network adversary is to insert an additional e-mail address in a `To` or `Cc` or `Reply-To` or `From` header. This is a staging attack against message confidentiality - it relies on followup action by the recipient.

For an encrypted message that is part of an ongoing discussion where users are accustomed to doing "reply all", such an insertion would cause the replying MUA to encrypt the replying message to the additional party, giving them access to the conversation. If the replying MUA quotes and attributes cleartext from the original message within the reply, then the attacker learns the contents of the encrypted message.

As certificate discovery becomes more automated and less noticeable to the end user, this is an increasing risk.

An MUA that rejects Exposed Headers in favor of Protected Headers should be able to avoid this attack when replying to a signed message.

12. Privacy Considerations

This document only explicitly contemplates confidentiality protection for the Subject header, but not for other headers which may leak associational metadata. For example, `From` and `To` and `Cc` and `Reply-To` and `Date` and `Message-Id` and `References` and `In-Reply-To` are not explicitly necessary for messages in transit, since the SMTP envelope carries all necessary routing information, but an encrypted [RFC5322] message as described in this document will contain all this associational metadata in the clear.

Although this document does not provide guidance for protecting the privacy of this metadata directly, it offers a platform upon which thoughtful implementations may experiment with obscuring additional e-mail headers.

13. Document Considerations

[RFC Editor: please remove this section before publication]

This document is currently edited as markdown. Minor editorial changes can be suggested via merge requests at https://github.com/autocrypt/protected-headers or by e-mail to the authors. Please direct all significant commentary to the public IETF LAMPS mailing list: spasm@ietf.org

13.1. Document History

Changes between version -00 and -01:

- Credit Randall for "correct horse battery staple".
- Adjust test vectors to ensure no line in the generated .txt format exceeds 72 chars.
- Minor formatting cleanup to appease idnits.
- Update references to more recent documents (RFC 2822 -> 5322, -00 to -01 of draft-ietf-lamps-header-protection-requirements).

14. Acknowledgements

The set of constructs and algorithms in this document has a previous working title of "Memory Hole", but that title is no longer used as different implementations gained experience in working with it.

These ideas were tested and fine-tuned in part by the loose collaboration of MUA developers known as [Autocrypt].

Additional feedback and useful guidance was contributed by attendees of the OpenPGP e-mail summit ([OpenPGP-Email-Summit-2019]).

The following people have contributed implementation experience, documentation, critique, and other feedback:

- Holger Krekel
- Patrick Brunschwig
- Vincent Breitmoser

The password example used in Section 9 comes from [xkcd936].

15. References

15.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3156] Elkins, M., Del Torto, D., Levien, R., and T. Roessler, "MIME Security with OpenPGP", RFC 3156, DOI 10.17487/RFC3156, August 2001, <https://www.rfc-editor.org/info/rfc3156>.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, DOI 10.17487/RFC4880, November 2007, <https://www.rfc-editor.org/info/rfc4880>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

15.2. Informative References

[Autocrypt] , "Autocrypt Specification 1.1", 13 October 2019, <https://autocrypt.org/level1.html>.

[I-D.draft-bre-openpgp-samples-00] Einarsson, B., juga, j., and D. Gillmor, "OpenPGP Example Keys and Certificates", Work in Progress, Internet-Draft, draft-bre-openpgp-samples-00, 15 October 2019, <http://www.ietf.org/internet-drafts/draft-bre-openpgp-samples-00.txt>.

[I-D.draft-ietf-lamps-header-protection-requirements-01]
Melnikov, A. and B. Hoeneisen, "Problem Statement and Requirements for Header Protection", Work in Progress, Internet-Draft, draft-ietf-lamps-header-

protection-requirements-01, 29 October 2019, <http://www.ietf.org/internet-drafts/draft-ietf-lamps-header-protection-requirements-01.txt>.

[I-D.draft-luck-lamps-pep-header-protection-03] Luck, C., "pretty Easy privacy (pEp): Progressive Header Disclosure", Work in Progress, Internet-Draft, draft-luck-lamps-pep-header-protection-03, 5 July 2019, <http://www.ietf.org/internet-drafts/draft-luck-lamps-pep-header-protection-03.txt>.

[OpenPGP-Email-Summit-2019] , "OpenPGP Email Summit 2019", 13 October 2019, <https://wiki.gnupg.org/OpenPGPEmailSummit201910>.

[RFC2634] Hoffman, P., Ed., "Enhanced Security Services for S/MIME", RFC 2634, DOI 10.17487/RFC2634, June 1999, <https://www.rfc-editor.org/info/rfc2634>.

[RFC3851] Ramsdell, B., Ed., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, DOI 10.17487/RFC3851, July 2004, <https://www.rfc-editor.org/info/rfc3851>.

[RFC6736] Brockners, F., Bhandari, S., Singh, V., and V. Fajardo, "Diameter Network Address and Port Translation Control Application", RFC 6736, DOI 10.17487/RFC6736, October 2012, <https://www.rfc-editor.org/info/rfc6736>.

[RFC7508] Cailleux, L. and C. Bonatti, "Securing Header Fields with S/MIME", RFC 7508, DOI 10.17487/RFC7508, April 2015, <https://www.rfc-editor.org/info/rfc7508>.

[RFC8551] Schaad, J., Ramsdell, B., and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification", RFC 8551, DOI 10.17487/RFC8551, April 2019, <https://www.rfc-editor.org/info/rfc8551>.

[xkcd936] Munroe, R., "xkcd: Password Strength", 10 August 2011, <https://www.xkcd.com/936/>.

Authors' Addresses

Bjarni Rúnar Einarsson
Mailpile ehf
Baronsstigur
Iceland
Email: bre@mailpile.is

juga
Independent
Email: juga@riseup.net

Daniel Kahn Gillmor
American Civil Liberties Union
125 Broad St.
New York, NY, 10004
United States of America
Email: dkg@fifthhorseman.net