Header Protection
I-D Status

LAMPS @ IETF-109 / Tuesday, 17 Nov 2020

draft-ietf-lamps-header-protection-01

Bernie Hoeneisen / Daniel Kahn Gillmor /
Alexey Melnikov

Summary of Changes since IETF-108

- Implemented Feedback from LAMPS WG session IETF-108
  - Editorial clean-up
  - Adding definitions for Cryptographic Layer, Cryptographic Payload, and Cryptographic Envelope (reference to new I-D dkg-lamps-e2e-mail-guidance)
  - Dropped Encrypted Only Messages
  - Updated Obfuscation recommendation
- Add DKG as co-author – Welcome!

Status of Issues (1/2)

- Backward Compatibility → Open
  - TBD later
- Protection Levels → Closed
  - Sending side: focus on “signature only” and “encrypted and signature”
  - Receiving side: decide on documenting other cases later
- MIME Format → Open
  - TBD later (after research to compare both options)

Status of Issues (2/2)

- Obfuscation of Header Fields → closed
  - Recommend only Subject and Message-ID (no objection raised on the ML)
  - May need to re-open this one (cf. next presentation)
- Rendered message → open
  - Render “Inner” Message only, but additional information made available
- Bcc handling → closed
  - Keep text regarding Bcc minimal in this document
  - refer to other documents

Next Steps Overview

- Overhaul draft to focus on **implementation guidance**
- Describe two schemes of header protection found in the wild:
  - **Wrapped Message** (S/MIME 3.1+)
  - **Injected Headers** (`draft-autocrypt`, aka “memory hole”)
- How to interpret them
- How to compose them
  - For encrypted messages, **Header Confidentiality Policy**
- Comparisons and Test Vectors

Open Questions

All about message composition:

- Which header protection scheme?
- Default Header Confidentiality Policy?

Two Header Protection Schemes

Only need to consider the Cryptographic Payload…

Wrapped Message

```
[…Cryptographic Envelope…]
A └┬message/rfc822
   │  (forwarded=no)
B  └┬multipart/alternative
C   ├─text/plain
D   └─text/html
```

Injected Headers

```
[…Cryptographic Envelope…]
D └┬multipart/alternative
   │  (protected-headers=v1)
E  ├text/plain
F  └text/html
```

(w/ legacy display)

only for some encrypted messages, not for signed-only messages

```
[…Cryptographic Envelope…]
G └┬multipart/mixed
   │  (protected-headers=v1)
H  ├text/plain [legacy display]
   │  (protected-headers=v1)
I  └┬multipart/alternative
J   ├text/plain
K   └text/html
```

Choosing a scheme for message composition

HP Scheme Evaluation

Recipient MUA Capabilities

	Legacy (no crypto)		Legacy (with crypto)		Fully Implemented	
Protections for composed Message	render	reply	render	reply	render	reply
Signed-only (multipart/signed)					good	good
Signed-only (pkcs7 signedData)	unreadable message	unreadable message			good	good
Signed & encrypted	unreadable message	unreadable message			good	good

Header Confidentiality Policy

- When composing an **encrypted** message with header protection, how should the outside header be formed, based on the inside header?
- HCP is defined as a function in pseudocode:
  - `hcp(name, val_in) → val_out`
- (If `val_out` is `null`, the header `name` will be omitted)
- Communications tool for MUA implementers and researchers to describe their plans to each other.

Default HCP recommendation?

```
hcp_minimal(name, val_in):
  if name is 'Subject':
    return '[...]'
  else:
    return val_in
```

Deliverability, Server-side threading...

```
hcp_strong(name, val_in):
  eh = ['From', 'To',
        'Cc', 'Date']
  if name in eh:
    return val_in
  elif name = 'Subject':
    return '[...]'
  elif name = 'Message-ID':
    return new_message_id()
  else:
    return null
```

Confidentiality, Metadata surveillance, ...

There are other possible HCPs...

More Subtleties...

- If your peer uses a stronger HCP, how do you reply to their encrypted message without leaking data?
- Identifying confidential protected headers under subtle obfuscation (e.g. TZ-stripping for `Date`, or dropping the *display-name* from an address header like `To`)
- Impact of HCP on:
  - IMAP threading and IMAP header search
  - server-side spamfiltering