MIMI Content Format

draft-mahy-mimi-content-02

Rohan Mahy, IETF 116, 30-Mar-2023
Goal

• If content is end-to-end encrypted, we need a standard format for common messaging features
  • plain text and rich text messaging
  • mentions
  • replies
  • reactions
  • edit or delete previously sent messages
  • expiring messages
  • delivery notifications/read receipts
  • shared files/audio/videos
  • calling / conferencing
  • message threading

• Addresses a MIMI charter item

• Should be able to extend this format and also send proprietary formats alongside of or instead of the standard format when appropriate
Approach

• Current version has an abstract syntax to focus on the semantics
  • Look! No CPIM!
• Semantics have been stable for most features since draft debuted at IETF113
• Introduces a **Message Container** type
  • each Message Container MUST have a **timestamp** (when was this encrypted) and a **unique message ID** (UUID properties)
  • Messages can refer to other messages (by the target message’s message ID)
    • Reply or Reaction. A reaction uses a reaction disposition
  • Messages can be edited by providing a new message which updates the old one, or deleted by updating with zero length content
  • Messages can have an expiration; messages can be part of a thread
  • most Messages have bodies, which can be nested. Deletions do not have bodies
• ... and a **Message Report** type
  • Message Reports can update the status of a list of messages (ex: delivered, read, unread, error)
• When used with MLS, the Message Container does not duplicate information integrity protected by MLS (ex: group ID, and sender).
  (Corresponding fields could be included when using another protocol.)
Issues from the list
Values already in MLS

• The message container does not have a “To” address. The MLS group is already specified and integrity protected in an MLS application message.

• Likewise the Sender (client), which is like a “From” address is integrity protected in the MLS application message, and the user identity would typically already be known to all the clients in the group through the client’s Credential.

• There are other fields which the client can derive from the MLS state. Which depends on how we define the MLS profile.

• **Q:** Should we send fields with these semantics in the message container anyway?

• **Propose:** we can create relevant fields when not using MLS, but that these fields are omitted when MLS is used.
How does client know what formats are OK?

- For MLS this is covered in Section 2.3 of draft-ietf-mls-extensions (content advertisement) and related to Sections 7.2, 11.1, 12.1.7 of draft-ietf-mls-protocol (MLS core protocol)

- In brief:
  - **supported** media types are listed for each member of the group. are updated periodically in long-lived groups (after client upgrade very likely)
  - **supported** media types are advertised in KeyPackages (used to add clients). clients update these periodically and very likely after upgrade.
  - creator can list **required** media types for a group. All clients need to have support for these.
  - the required media types can be **updated** with a GroupContextExtensions Proposal, as long as the resulting clients
Threads vs replies 1/2

• **inReplyTo** says that a single message is in reply to a single previous message. *It should not be used for selecting the order of messages in a thread.*

• **inReplyTo** is also used for reactions (likes and unlikes), because the reaction is directly in response to a single specific previous message.

• You can reply to a reply, or like a reply. The composer of the reply cannot edit the replied message. (Currently most messaging systems just quote the most recent message in the reply).

• Replies do not affect rendering order (you can reply to a message days, weeks, or months old). Indeed this is often used to bump a conversation.
Threads vs replies 2/2

- Threading is a feature of some enterprise IM systems like Slack and Teams. All messages in the thread are rendered linearly. There is no indentation as in email or netnews.

- threadId identifies a single ancestor message ID. All messages with the same threadId would likely be rendered in a single list of messages.

- The only time that inReplyTo seems appropriate in a threaded message is when sending a reaction about an earlier message in the thread.

- Q: Does the content format need to specify a specific rendering order?

- Propose: No. Use the timestamp
Report on multiple messages

• The draft as written has reports which can update status of multiple messages.

• Assumed requirements:
  
  • Especially during federation and interop, small amounts of processing delay and clock skew means saying “I read everything from time x or message y” won’t work. (would result in marking an unread message as read and vice versa.)

  • Is it possible to mark a single message as read which is not the most recent message? Is it possible to mark a message unread before a message which is read?

  • Do we want to be able to mark multiple messages read in a group at the same time?

• OK?
Mentions

• the draft describes mentions using a link to a URI with the `im:` scheme type.
  
  • Markdown: Kudos to [@Alice Smith](im:alice-smith@example.com) for...
  
  • HTML: Kudos to `<a href="im:alice-smith@example.com">@Alice Smith</a>` for...
  
• Q: Can we do better than this?
What else?

• Several things about the behavior should be more specified
  • Content-Disposition meaning and behavior
  • Sanitizing inputs
• Privacy and Security Considerations need to be fleshed out.
• **Q:** What about such and such thing that was mentioned on the list before the plenary?
  • **Yes.** These will be included in the **next version** of the draft.
This page intentionally left blank
Next Steps

• **Q:** Are the semantics of this approach a reasonable start?

• Assume we will revisit concrete syntax as the transfer protocol matures

• **Q:** Can we adopt this draft as a WG item supporting the content format work item?