MIMI Content Format

draft-ietf-mimi-content-01

Rohan Mahy, IETF 118, 8–Nov–2023

What’s new?

- new abstract format for attachments, instead of message/external-body
- added discussion of encrypting external content
- clarified the difference between render and inline dispositions
- created a way for the messageId and timestamp to be shared in the MLS additional authenticated data field
- expanded discussion of what can and should be rendered when a mention is encountered; discussed how to prevent confusion attacks with mentions.
- added a lastSeen field used to ensure a more consistent sort order of messages in a room.

Attachments via External Content

- Created new binary format modeled on message/external-body (RFC 4483)

```
body.disposition = attachment;
body.contentType = "video/mp4";
body.URL = "https://example.com/storage/bigfile.mp4";
body.size = 708234961;
body.encAlg = 0x0001;    // AES-128-GCM
body.key = "\x21399320958a6f4c745dde670d95e0d8";
body.nonce = "\xc86cf2c33f21527d1dd76f5b";
body.aad = "";
body.expires = 0;
body.description = "2 hours of key signing video";
```

- Sending client encrypts and uploads (usually to local provider); sends external content fields in room

- Receiving client downloads according to their local policy; decrypts

External Content encryption

- External content is encrypted:
  - with an **ephemeral symmetric key and nonce**
  - using an IANA-registered Authenticated Encryption with Additional Data (AEAD) algorithm as described in [RFC5116]
    - MUST implement **AES-128-GCM**
  - The key, nonce, and additional authenticated data (aad) values are set to the values used during the encryption.
  - Unless modified by an extension, aad is empty.

Sharing messageId and timestamp with providers

- Content format has a messageId and timestamp chosen by the encrypting client
  - expose a copy of this in the MLS Additional Authenticated Data field (AAD)
  - local or hub provider can reject a provisional message with a timestamp too far in the past or future (ex: one hour)
  - clients are primarily responsible for detecting duplicate messages IDs among messages they have received
  - local or hub provider can reject a message with a duplicate message ID, but are not required to.
    - UUID + owning provider domain.
      - Q: why include owning provider domain?
      - A: owning or hub provider can check if the domain part purports to be from the wrong domain; owning provider can check if user part duplicates prior messages it has a record for; 100% elimination of duplicate messages is not possible in high availability architecture.
- Concerns about extra metadata?
  - There is not a tremendous amount of data that can be gleaned here without access to some decrypted messages, but the domain name could be valuable information for traffic analysis when a provider has a small amount of volume

Sort order of messages

- Requirements question
  - Is consistent rendering of sort order a required feature of MIMI?

- If not, we can remove this section.
- If so, added a lastSeen header with the messageId of the last message the sender has seen.
- If multiple messages arrive with the same lastSeen header, the receiver will list all of those messages in its lastSeen header (“merging” the graph)
- The client still needs to guard against attacks where this header is maliciously generated

Mentions

- Currently we use links to im: URLs inside Markdown or HTML

```
body.contentType = "text/markdown;charset=utf-8";
body.content = "Kudos to [@Alice Smith](im:alice-smith@example.com)"
             + "for making the release happen!”;

body.contentType = "text/html;charset=utf-8";
body.content = "<p>Kudos to <a href='im:alice-smith@example.com'>" +
               "@Alice Smith</a> for making the release happen!</p>"
```

- Improved text. Use your local policy to decide if display text is a valid/consistent representation of the IM URI

- Please send text!

Encoding

- What **binary** encoding do we want to use and why?
- What are our requirements?
  - fast to parse and encode
  - not too many ways to encode the same thing
  - extensible
  - stale reference
- Schema-less or Schema required?
  - perhaps schema for required elements; schema-less for extensions

- Options
  - TLS presentation language
    - weird syntax
    - no typedefs
    - no parsers for Javascript and some other languages
  - CBOR
    - emphasis on small rather than fast
    - require schema or not?
    - basically requires CBOR playground to design with
  - Protobuf
    - which version?
    - wire format not 100% consistent
    - nesting issues
    - stable reference?
  - msgpack
    - not widely implemented