MLS Content Negotiation

draft-mahy-mls-content-neg-00

Rohan Mahy, IETF 114, 29-July-2022

Problem

- No way to find out which application formats are **supported**, and which formats are **required** in a group

- No built-in way in MLS to signal the format of MLS application data
  - Makes switching content format in long-lived groups very difficult
  - Makes interoperability challenging

Proposed MLS Extensions

- In KeyPackage
  - `accepted_media_types`

- In GroupContext
  - `required_media_types`

- These use the extensive IANA media types registry format (straightforward registration, free vendor namespace)
  - Examples:
    text/plain
    text/html
    application/foo-im-protocol+json
    application/vnd.examplecorp.instantmessage
    multipart/alternative
    text/markdown

Application Framing

- Application Data is assumed to use Application Framing if `required_media_types` is in the `GroupContext` extensions for the group
- Application Framing contains the `media_type`, then the rest of the `application_content`

```
struct {
    MediaType media_type;
    opaque<V> application_content;
} ApplicationFraming;
```

- If the `media_type` is a zero-length vector, the first media type is assumed

Next Steps

- MLS extensions probably need to happen in the MLS WG
- Can we adopt content negotiation as a WG item? (regardless of approach)
- Is this draft a reasonable approach to the problem?
- If you don’t like the proposed approach, please comment on the mailing list
  - Make sure to specify if you have an issue with the semantics or the specific syntax

Backup: How do clients figure out the format?

- `required_media_types` is not in the GroupContext
  - Application Data (probably) does not using Application Framing. No change in message size.
- `required_media_type` is in the GroupContext. Assume Application Framing
  - if `media_type` is empty string
    - `application_content` is the first `media_type` in `required_media_types`. Message is 3-4 bytes longer.
  - if `media_type` is present
    - `application_content` is the specified `media_type`.
    - specified `media_type` is multipart/alternative.
      - multipart message includes at least one supported type for every member
      - each member that receives the message uses the first sent `media_type` they support