Design Team
Design Team

- At IETF 117, no consensus to adopt any protocol drafts
- WG suggested a smaller team get together and work up a common proposal
  - Basel Al-Naffouri
  - Richard Barnes
  - Matthew Hodgson
  - Konrad Kohbrok
  - Rohan Mahy
  - Travis Ralston
  - Raphael Robert
- Plan to have drafts ready for WG review in mid-Sept, adoption call late Sept
Progress So Far

- Met F2F after MIMI at IETF 117
- 3 calls since, discussing architecture / approach, some example cases
- General boundaries / desiderata
  - E2E security based on MLS
  - Object-level authentication of protocol messages, including server-originated ones
  - Incremental adoptability / transition story from current protocols
- draft-barnes-mimi-arch is a working document
  - Capture areas of clarity & agreement
  - Drive discussion by making proposals for resolving ambiguity / disagreement
  - Live copy at [https://github.com/bifurcation/mimi-arch](https://github.com/bifurcation/mimi-arch)
- Roughly on schedule for September deadlines
Architecture

* someone should probably write an architecture draft
draft-barnes-mimi-arch WIP
- Each room is hosted at a hub server
- All communications go via the hub
  - Messages
  - State changes
  - Ancillary things like KeyPackage fetches
- Anticipate that not every server will be willing to talk to every other server
  - A and B might not be willing to talk directly
  - But if users from A and B are in a room hosted at C...
  - A needs to be able to talk to B via C
Room Operations

- **Creation**
  - Provider-local operation, no protocol operation needed to enable it
  - However, creation initializes the room, possibly resulting in information distributed later

- **Messaging**
  - Servers forward messages on behalf of clients
  - Origin server submits to hub, hub fans out to other servers
  - Servers may also apply authorization policy

- **State maintenance**
  - Rooms have state aside from messaging: authz policy, E2EE keys/credentials, etc.
  - Servers facilitate making updates to this state
  - ... and keep the state synchronized across all servers
Protocol Stack

Client -> Provider (Provider) -> Follower (Hub) -> Follower (Messaging) -> Provider (Control) -> Follower (Transport) -> (Provider) -> Client
Transport

POST /events HTTP/1.1
Content-Type: application/mimi+json

{
    "eventType": "keyPackageRequest",
    "from": "@alice@providera.example",
    "user": "@bob@providerb.example",
    "signature": "0Iz7AZYAh..."
}

Transport protocol defines:
- General event format
- Event signing / origin authentication
- Event distribution between servers

Analogous to:
- Matrix PDUs / transactions
- MIMI-DS signed DSRequest
Control Protocols

POST /events HTTP/1.1
Content-Type: application/mimi+json

{
    "eventType": "keyPackageRequest",
    "from": "@alice@providera.example",
    "user": "@bob@providerb.example",
    "signature": "0Iz7AZAh..."
}

- Rooms have a few (mostly) orthogonal types of state, for example:
  - E2E encryption state – which clients have the keys to decrypt messages?
  - Authorization policy – which users / servers are allowed to take which actions
- For each type of state, there is a corresponding control protocol
  - E2E encryption control protocol – Fetch key packages, add/remove clients
  - Policy control protocol – Grant/deny permissions to users
- Control protocols use the transport protocol to move events around
“Members” without Clients?

- Users interact with a room via clients
  - A client needs to be part of the E2EE state of the room to participate
  - A client’s presence in the E2EE state indicates who its user is
- The E2EE state of the room implies a notion of user-level membership
  - A user is a member if they have at least one client joined
- Is this the only notion of user-level membership? Or is can a user meaningfully be a “member” without having any clients joined?
  - Yes: Other users shouldn’t need to be aware of a user’s clients coming and going
  - No: A user with no clients can’t read messages or take any action
- Potential middle way: “policy subjects”
  - Agreement that we need policy independent of E2EE
  - “Richard isn’t here right now, but if he shows up he can moderate the room”
DT Next Steps

- Get to agreement on the last few architectural points
- Start drafting real protocols that implement the roles in the architecture
  - Transport protocol
  - Message forwarding protocol
  - Control protocols (pending the above discussion)
- Transport protocol already in progress
  - Combining ideas from MIMI-DS and Matrix (not as shown above!)