MIMI Transport Protocol (MTP)

Jonathan Rosenberg
Cullen Jennings
Suhas Nandakumar
Strawman Decisions in this Draft

• Single provider owns the group chat
• Server-to-Server only
• Messages pulled from guest provider
• Sync on a per group chat basis
• Sync uses long poll (no one loves this)
• Properties vs. Messages
• OOB Connection Authorization for anti-spam
Properties vs. Messages

**Property**
Something for which the only thing that matters, is its current value.

- Group Name
- Topic
- isModerated
- isReadOnly

**Messages**
Content shared in the group, where I will want to see them in time order. This is the “stuff” in the content draft.

- Text
- Reaction
- Link
- Image
Owning Provider

Guest Provider

Scope of Protocol

MIMI REST

MIMI Sync (long poll)

User A

User B

Out of Band Comms (SMS, Email, etc)
MTP Sync Model

1. Guest provider learns of new group chat that its user is in. In current I-D is via OOB technique, others can be added.

Guest provider maintains list of foreign groups a user is in.
Guest provider uses REST API to fetch properties. Does this to establish initial/current view on properties for the group.
MTP Sync Model

Guest provider subscribes to changes by asking for all events from current time onwards.
MTP Sync Model

Sequence of events.

Each event has time and type.

Events indicate:
• New message
• Change in property
MTP Sync Model: Recovery

Owning Provider

Guest Provider

User A

User B

Last event: T2
MTP Sync Model: Recovery

Owning Provider

Guest Provider

Give me everything since T2

All events since T2

User A

User B
MTP Cloud Scale Model

Owning Server1
Owning Server2
Owning Server3
HTTP LB
Sub Group1
Sub Group2
Sub Group3
Sub Group4
Sub Group5
Guest Server1
Guest Server1
Guest Server1

Subscriber side sharded across servers by subscription ID
No problem if multiple subscriptions for same group – de-duped via message ID uniqueness

Subscribed side load balanced using existing HTTP LB
Fine grained subscriptions enable easy distribution of work
Subscription state bound to connection state – eliminates need for subscription DB