Matrix for MIMI

matthew@matrix.org | travisr@matrix.org
@matthew:matrix.org | @travis:t2l.io
Matrix is an open network for secure, decentralised real-time communication.

- Interoperable chat
- Interoperable VoIP
- Open comms for VR/AR
- Real-time IoT data fabric
Matrix’s mission:
To build the real-time communication layer of the open Web.
Matrix as a Framework

I-D.ralston-mimi-matrix-framework
Matrix’s capabilities

- Matrix is already used for interoperable chat!
- Rooms are replicated* across all participating servers
- ACLs, history retention, etc are all defined by the room version
- Events carry information between users in a room
- Extensible support for current & future chat features
- Supports end-to-end encryption, including (D)MLS
- HTTP+JSON APIs for ease of use, but can be (nearly) anything

*uses a DAG normally, but Linearized Matrix fixes that…
Eventual consistency

- Allows any server in the room to go offline and receive events later
- Offline servers only affect users on that server
- Uses a Directed Acyclic Graph (DAG) to store events, like Git
- “Gaps” in the DAG are healed when servers come back online
- Plenty of rules to govern how the DAG accepts events
- Not exactly the easiest thing to implement in DMA’s short timescales
# What MIMI looks like on Matrix

<table>
<thead>
<tr>
<th>MIMI</th>
<th>Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messaging framework</td>
<td>Room versions</td>
</tr>
<tr>
<td>ACLs, permissions, basis for interoperability</td>
<td>Contains the algorithms needed to run the room</td>
</tr>
<tr>
<td>Messaging Content/Format</td>
<td>Extensible Events</td>
</tr>
<tr>
<td>What is ultimately encrypted or sent over the wire</td>
<td>Uses “content blocks” to reuse schemas</td>
</tr>
<tr>
<td>Transport</td>
<td>HTTP+JSON primarily</td>
</tr>
<tr>
<td>Between backend servers</td>
<td>Could be (nearly) anything, though</td>
</tr>
<tr>
<td>User discovery</td>
<td>Identity servers</td>
</tr>
<tr>
<td>Including resolving a phone number to a user ID</td>
<td>We’d rather replace these with something else</td>
</tr>
<tr>
<td>MLS for E2EE</td>
<td>DMLS for E2EE, or optionally Olm/Megolm</td>
</tr>
</tbody>
</table>
What if we didn’t have to use a DAG though?
Linearized Matrix

I-D.ralston-mimi-linearized-matrix
Linearized Matrix

- Partially centralizes rooms around an owner for a given conversation
- Still uses Matrix’s access control semantics and rules
- Still uses Matrix’s event format for interoperability
- Represents rooms as a single array rather than a DAG
- Significantly easier to implement and reason about
- Put it over whatever transport you like
- Doesn’t shut the door on decentralisation; can interop with normal DAG-capable Matrix.
Room owner

Linearized Matrix (for a given conversation)

Homeserver

Existing client
Linearized Matrix (for a given conversation)

Wire

iMessage

Public API

DAG-Capable Matrix

WhatsApp

Telegram

Public API
Thank you!

@matthew:matrix.org | @travis:t2l.io
matthew@matrix.org | travisr@matrix.org
https://matrix.org