

# Some Design Questions for MIMI

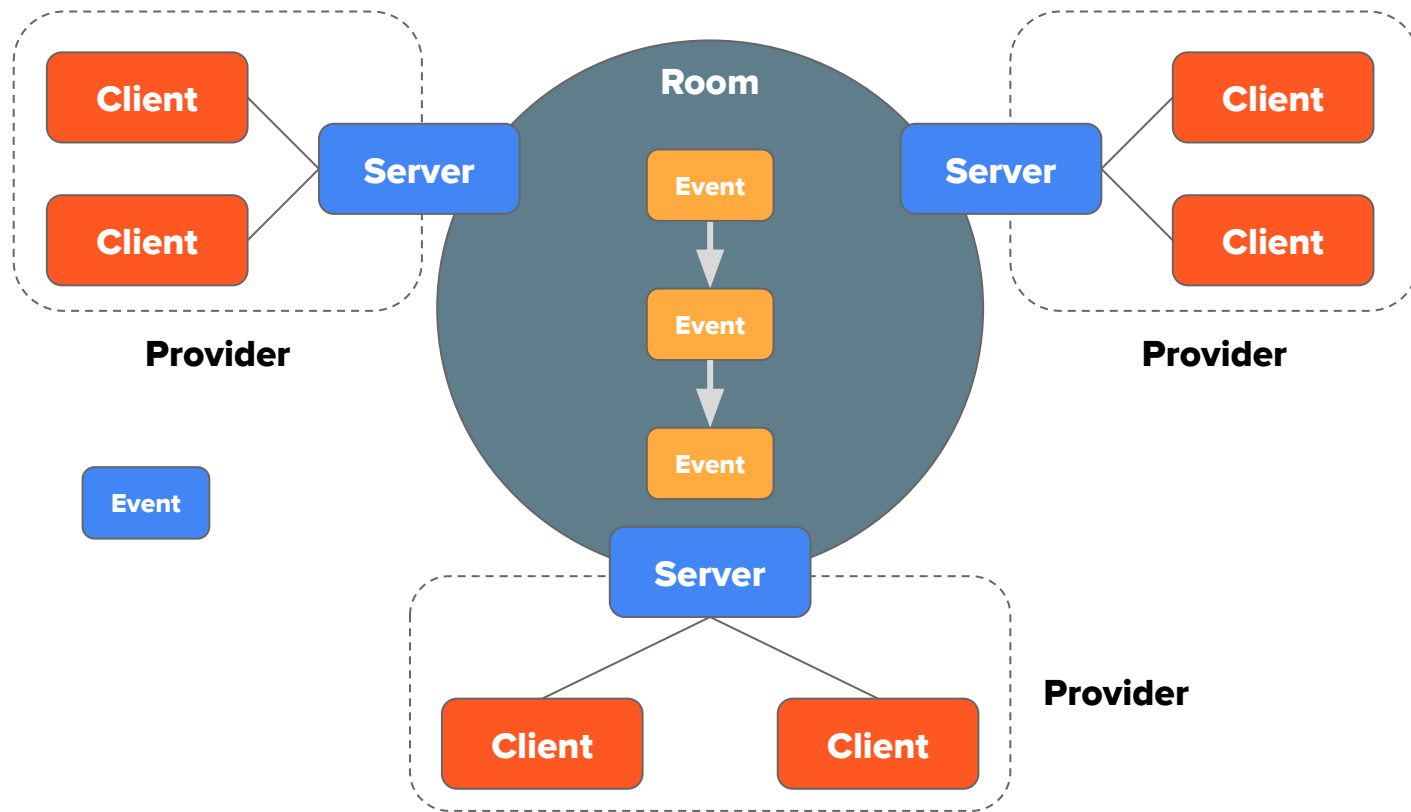
---

Richard Barnes  
IETF 117, July 2023

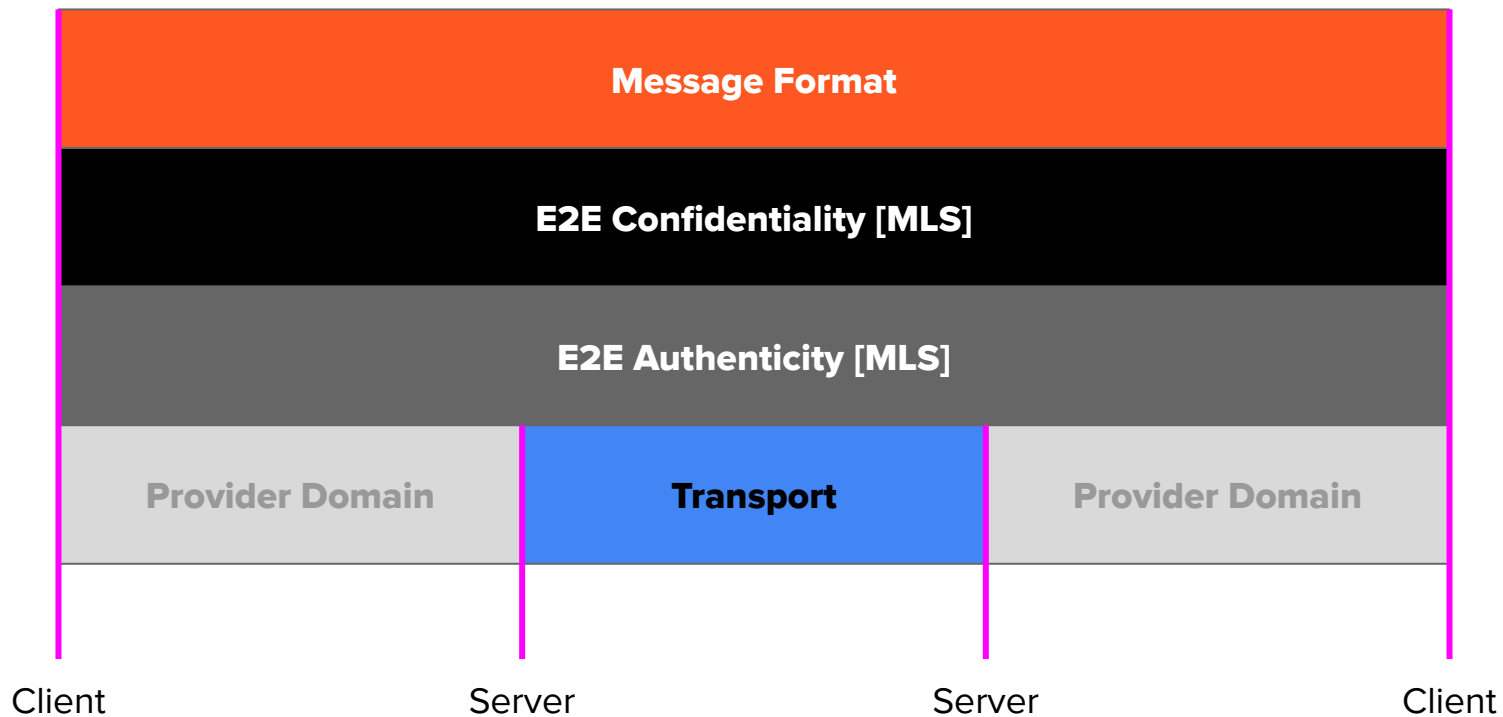
# Architecture Recap\*

**\* someone should probably  
write an architecture draft**

# Logical Structure



# Protocol Stack



# Policy for a Room = questions such as...

Can a room be joined without authorization from a current member?

Who is entitled to...

Admit users?

Kick / ban users?

Modify the room's policy?

**“Owning Server”**

# Possible Notions of “Ownership”

1. “Hub” — Central distribution point for a room’s events
2. “Policy enforcer” — Filters out events that are not compliant with policy
3. “Policy controller” — Sets policy for the room / bounds acceptable policy

These are not totally orthogonal:

- Policy enforcer probably needs to be a hub
- Hubs are in a natural position to do policy enforcement
- Policy enforcement naturally bounds policy, at least to the degree that the enforcer needs to understand the policy

# Policy Control

Assumption that part of the state of the room is an authorization policy (as above)

Someone needs to set that policy — clients and/or servers / providers

Some entities may also provide bounds on what policies are acceptable

Client / server / provider distinction is not totally clean — Providers could reflect their policy constraints via their clients, using intra-provider coordination

# Questions

Does a room always have a hub server?

Is the hub server expected to enforce policy for the room?

Are other servers involved in the room expected to enforce?

Who can influence policy for a room?

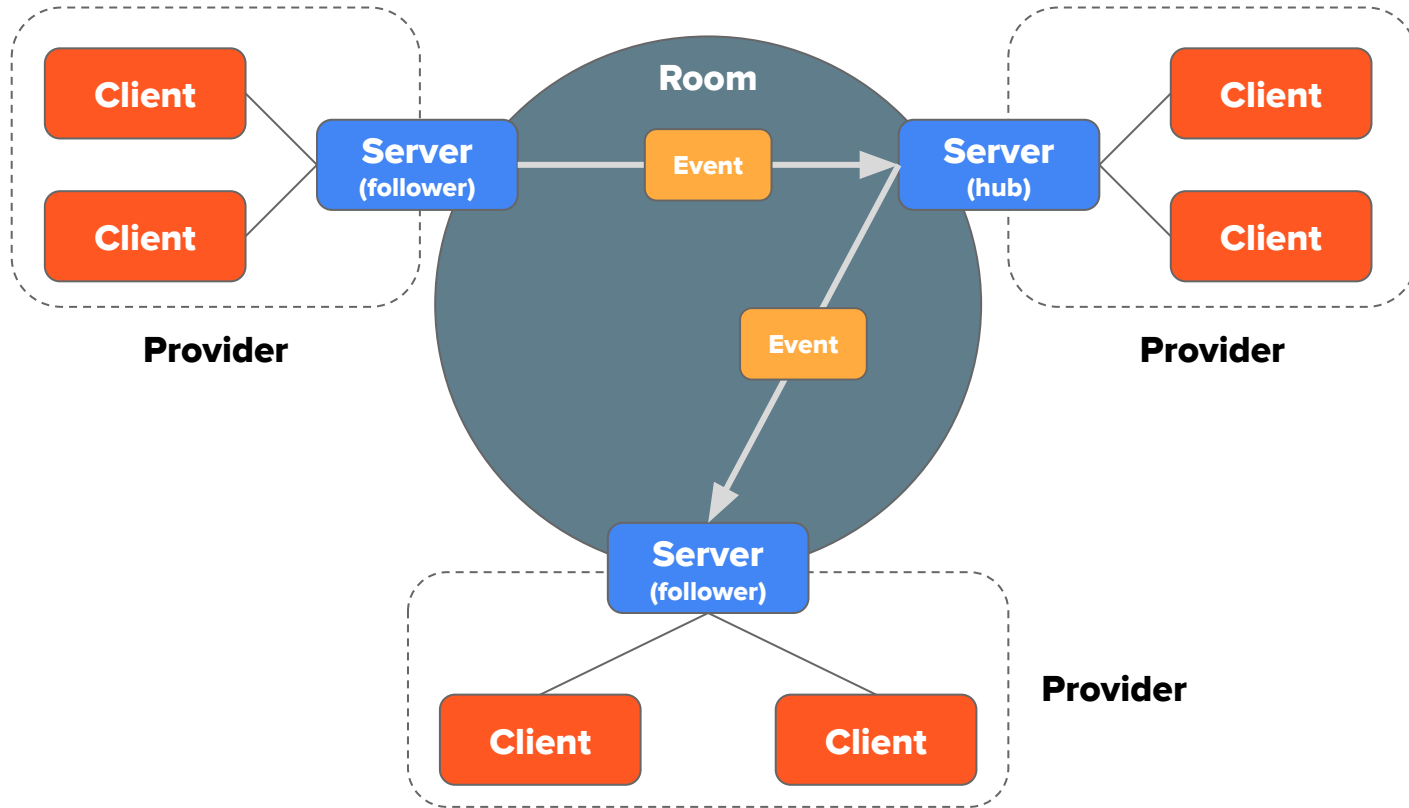
The clients in the room?

Any server involved in the room?

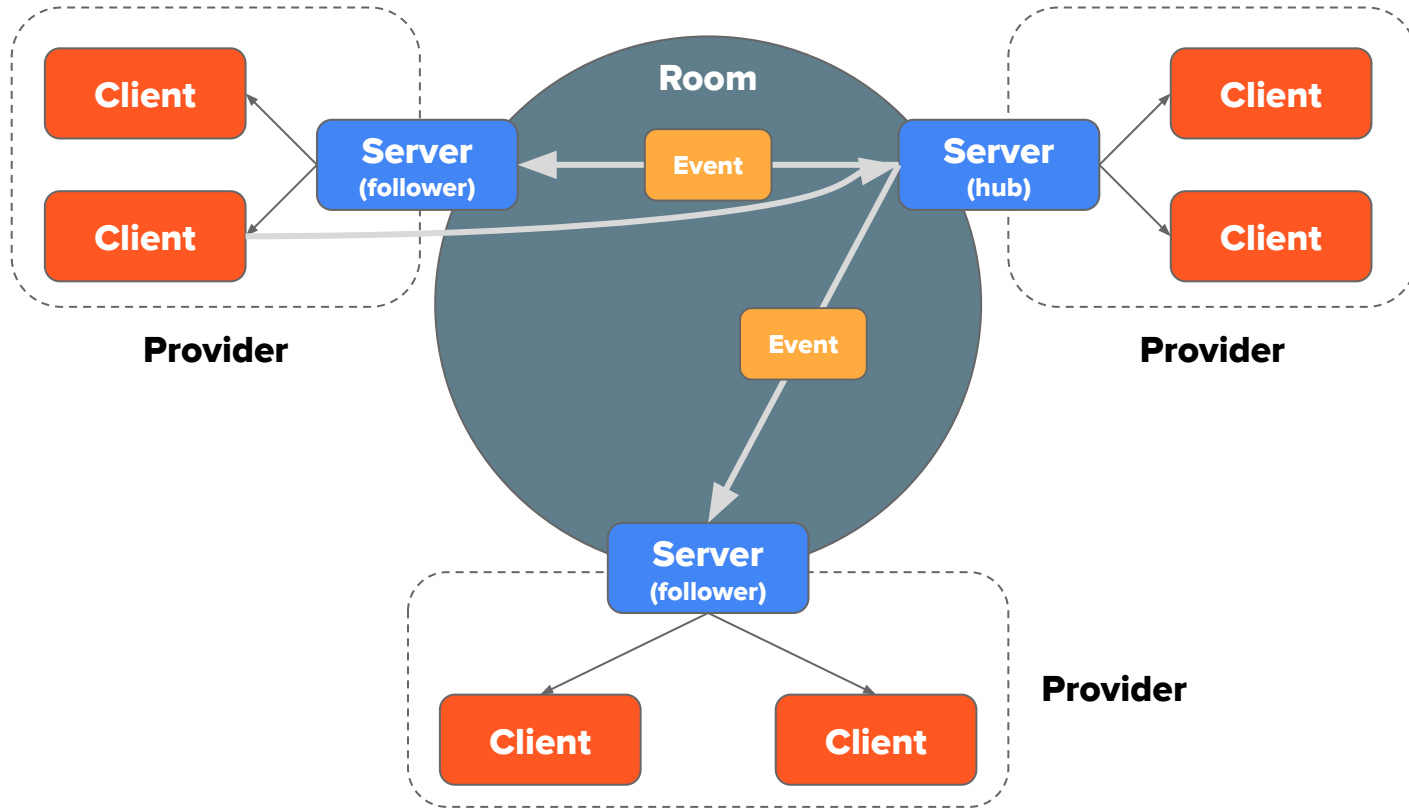
A designated server in the room? (The “owning provider”)

# Client-to-Server API

# Is everything server-mediated... [CSSSC]



# Can clients directly submit events? [CSSC]



# Do we need a client-server API?

S-S API seems clearly in scope

Do we also need a C-S API?

For the “direct connection” CSSC case, not for the CSSSC case

What would the differences be vis-à-vis a S-S API?

Transport: Servers are available, something you can send HTTP requests to

Identity / authn / authz: Server has a separate identity from the clients

Perhaps C-S API arises naturally as a minor variation / degenerate case of S-S?