



Interoperable privacy preserving user identity and discovery for E2EE messaging

Draft proposal for Mimi
Giles Hogben, Femi Olumofin

Functional Requirements

For a given messaging service identity handle (phone number or alphanumeric UserID):

1. **Endpoint discovery:** discover receiver service IDs to retrieve public key material and send message payload e.g. [Platform1.org/send/](#), [Platform1.org/kds](#)
2. **Default service discovery:** Discover optional default receiver service ID user preference for a given PN/UserID (e.g. `default:Platform1.org`)
3. **Global uniqueness:** Fully-qualified service identifiers should be globally unique
4. **(P1) Key verification:** Provide an independently trusted party to assert and verify the association between a public key and a UserID

Privacy Requirements

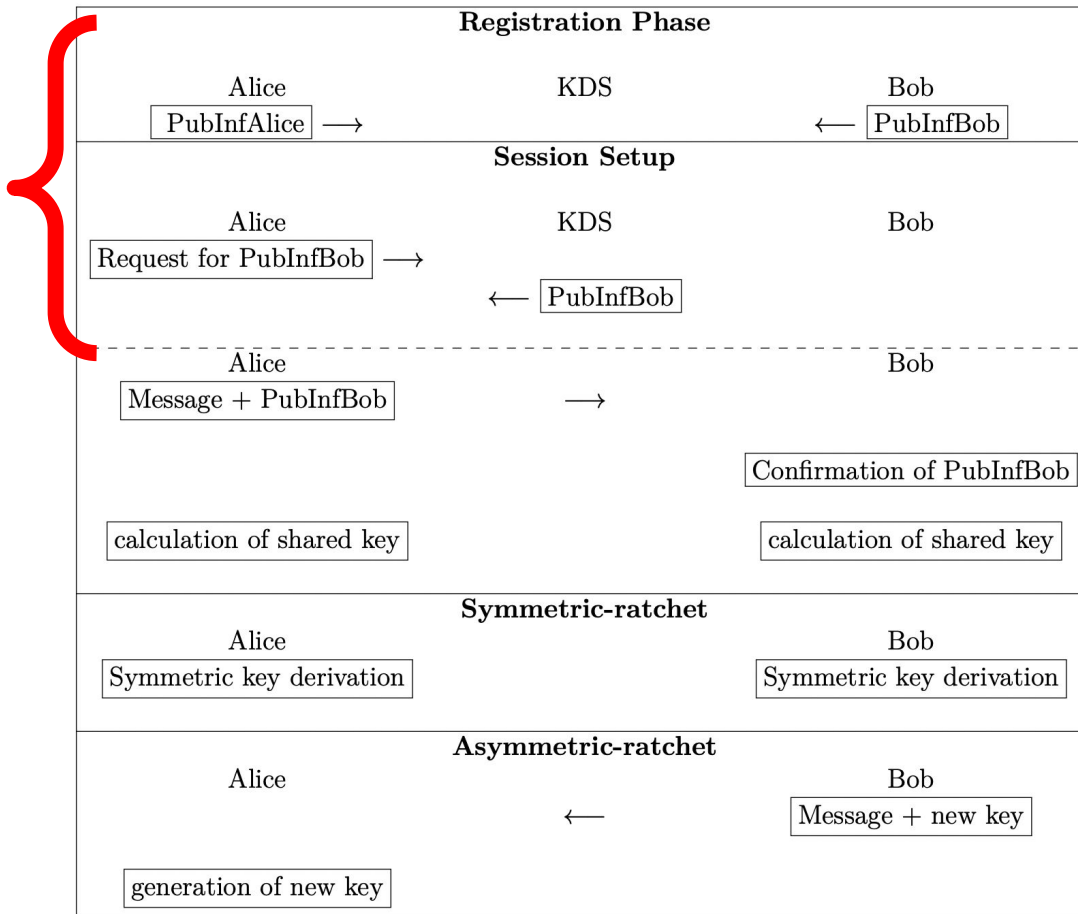
1. **Social graph:** Resolver or discovery services should not learn the PN/UserID a client is querying for (i.e. who is sending a message to who)
2. **Querying user identity:** A resolver service should not default to sharing the querying user identity with other resolver services when it requires their help for discovery
3. **Metadata:** Resolver service should not learn the exact timing of when a message is sent

Privacy non-requirement

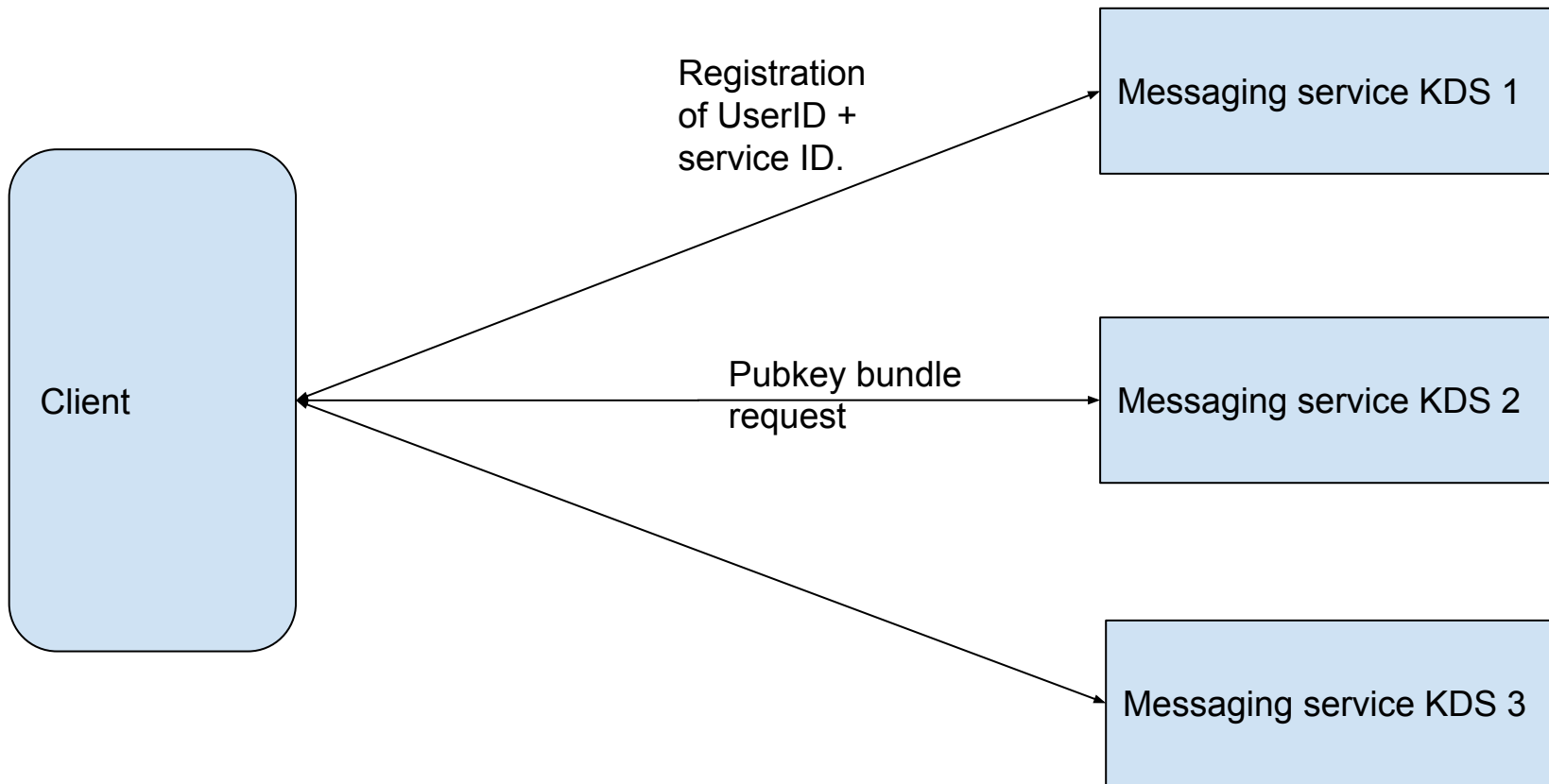
1. **Hiding service reachability:** (the link between a UserID and a service). E.g. +16501234567, reachable on Messages
 - a. All major E2EE messaging services already publish unACL'd reachability information without opt-out, Whatsapp, Telegram (not including name or any other info)
2. **Hiding the value of UserIDs or public keys:** e.g. the existence of the PN, +16501234567
3. **Hiding the association between a public key and a UserID:** e.g. PN +16501234567 has pubkey x
4. **Contact lookup by name** (or anything except username)

Other non-functional requirements

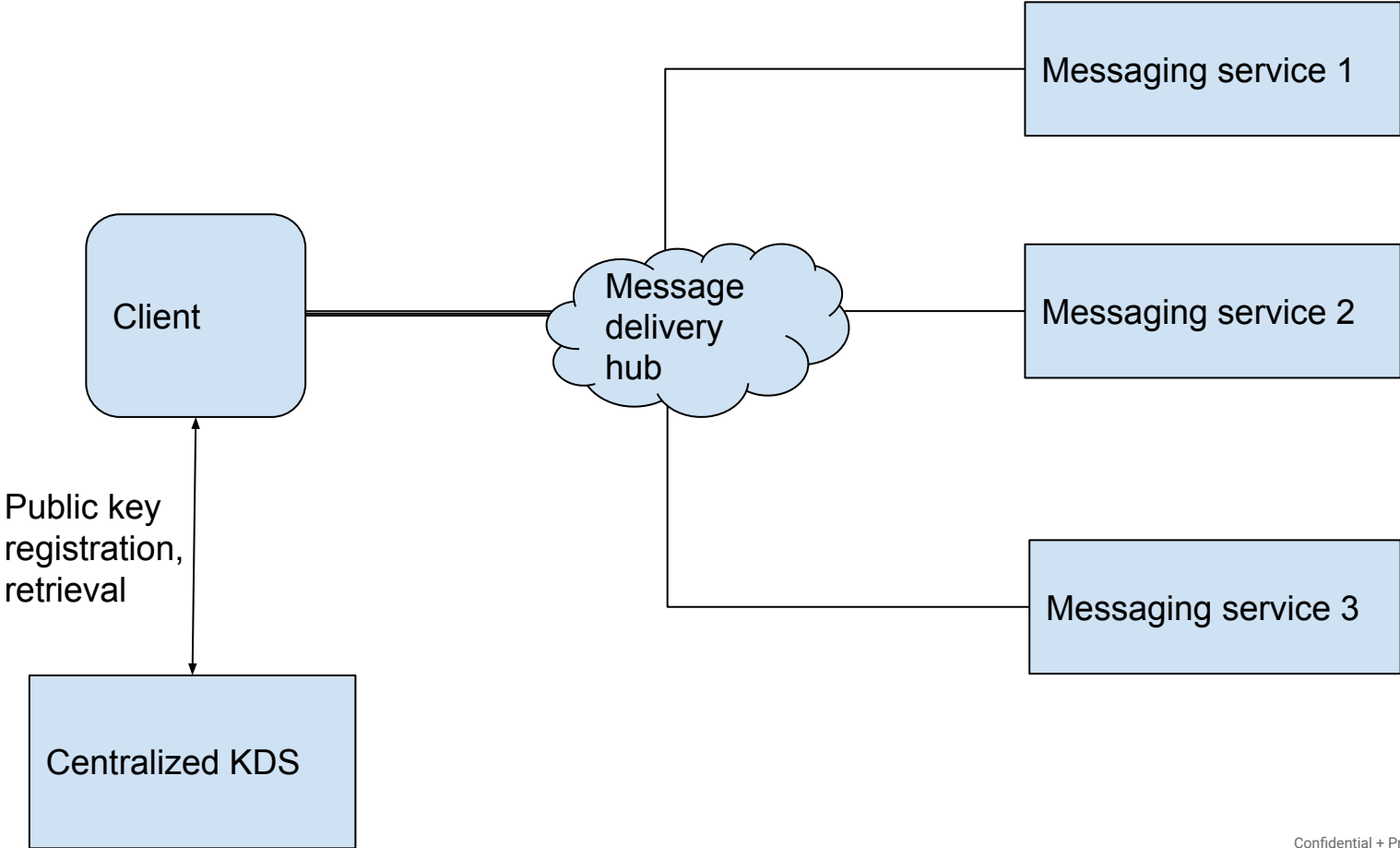
1. No single entity should be financially responsible for resolving all identity queries (e.g. even within a geographical region)
2. Costs for each participating entity of storing and querying key records should be proportional to their number of participating users.
3. Performance should support each client querying each of their contacts at least once every 24 hours



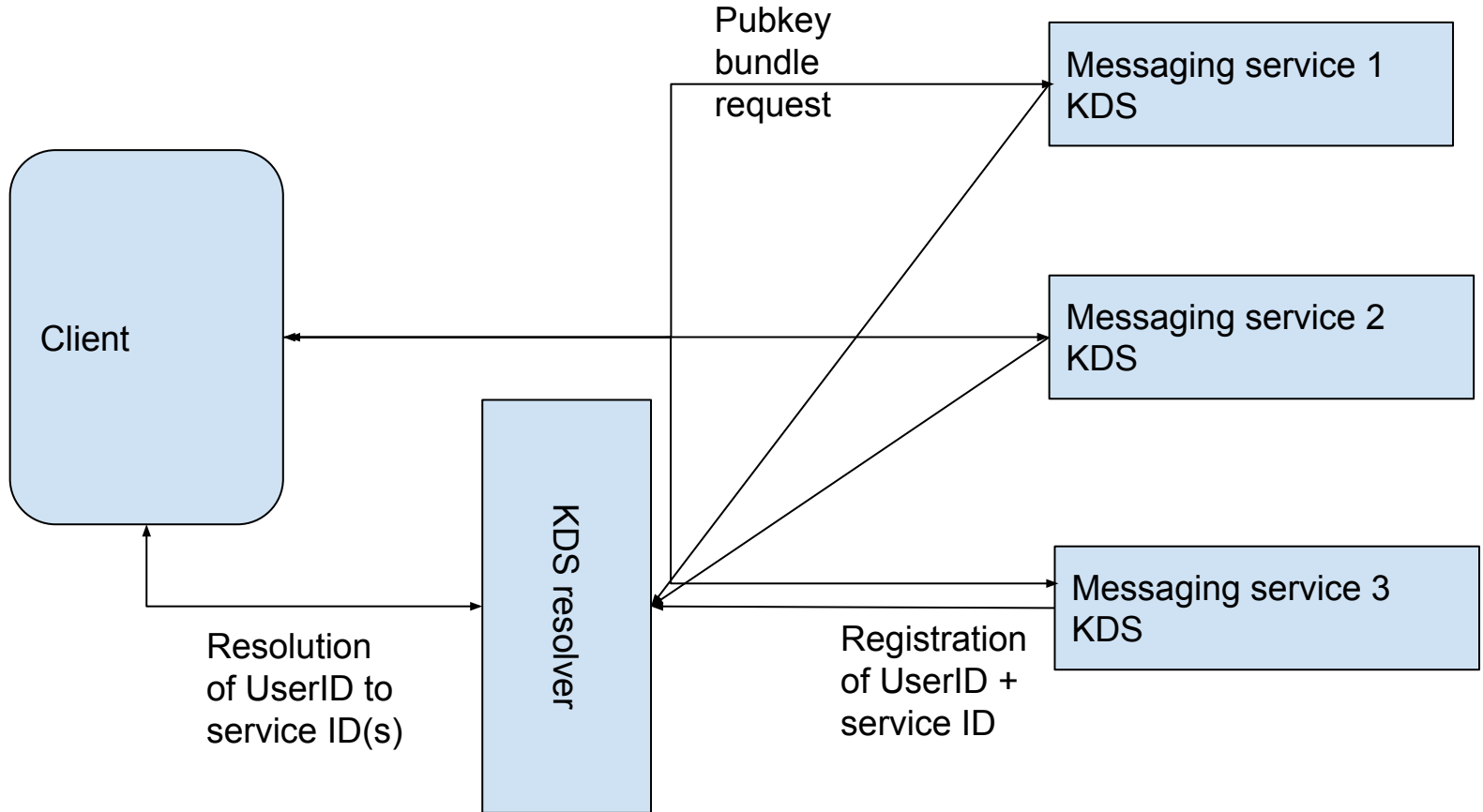
Option 1: brute force query - too expensive and leaks private info



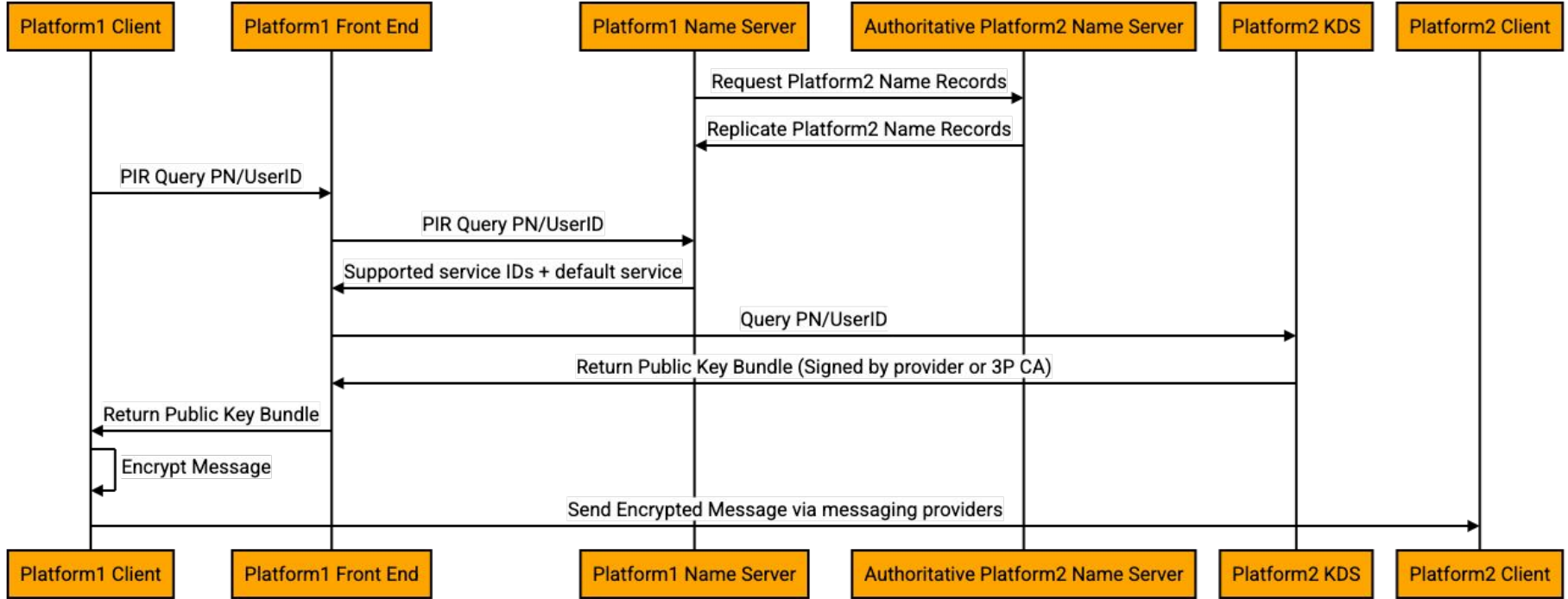
Option 2: Centralized hub - expensive and organizationally complex



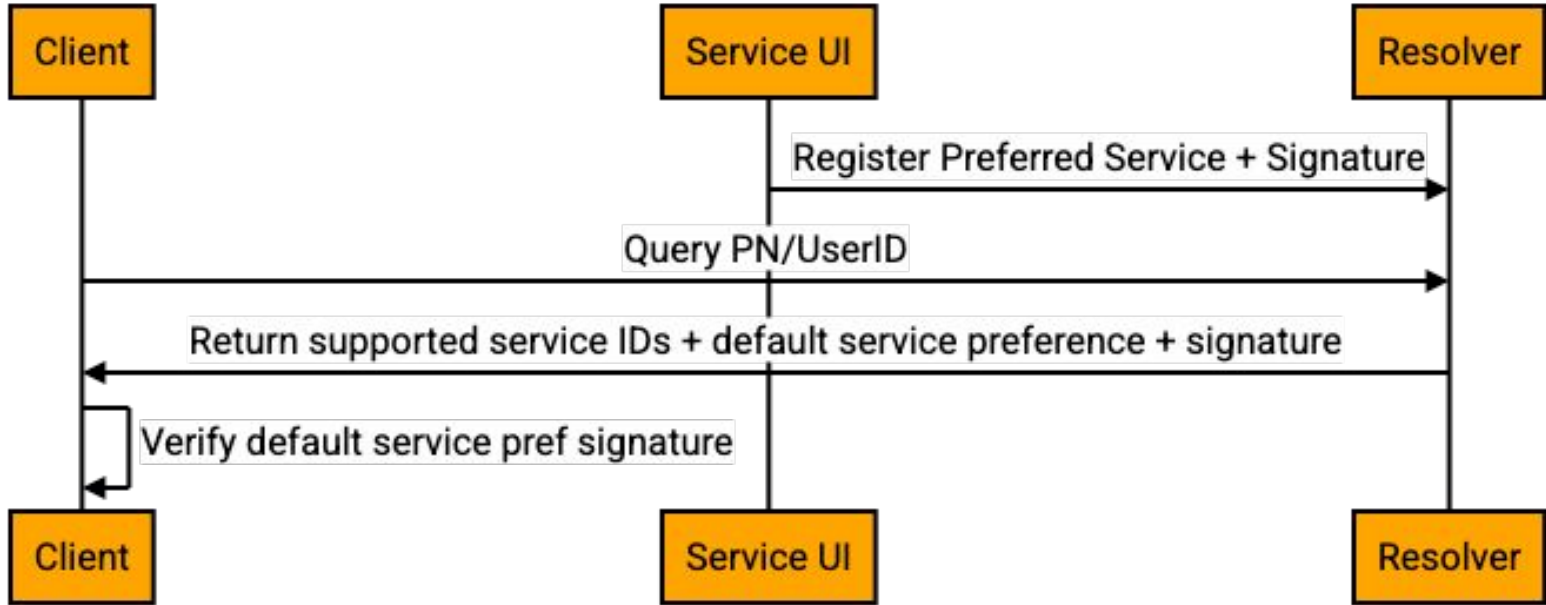
Option 3 (Preferred): Federated with KDS resolver service



Key distribution



Preferred service integrity



Privacy of resolver queries

- Goal: prevent leakage of the user's social graph to resolvers and other parties
- Setting: User may query a PN/UserID in an ad hoc manner or in a batch (e.g., key bundle download for all of a user's address book contacts)
- Our proposal: Private Information Retrieval (PIR)
 - Google's PIR framework to transform any standard lattice-based homomorphic PIR scheme into efficient keyword PIR
 - Approach is feasible with privacy - cost tradeoff that we consider as reasonable

Homomorphic Encryption

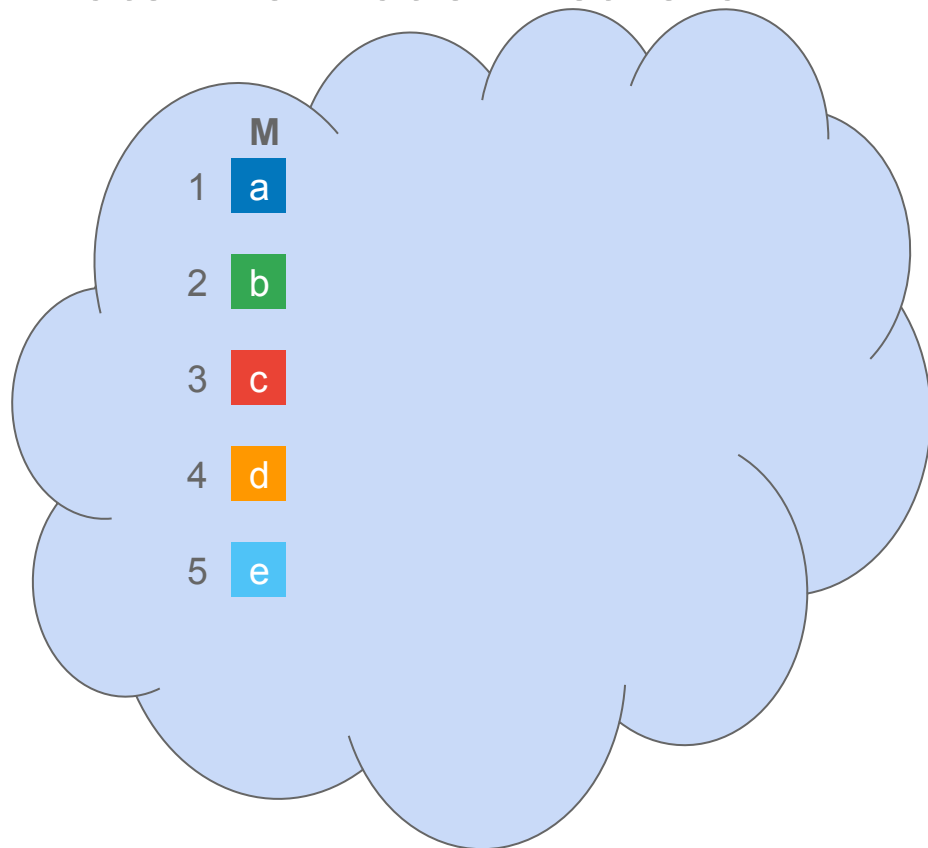
$$\text{[Red box with lock icon and 'a']} + \text{[Yellow box with lock icon and 'b']} = \text{[Orange box with lock icon and 'a+b']}$$

$$\text{[Blue box with 'a']} \times \text{[Yellow box with lock icon and 'b']} = \text{[Green box with lock icon and 'a*b']}$$

(Not encrypted)

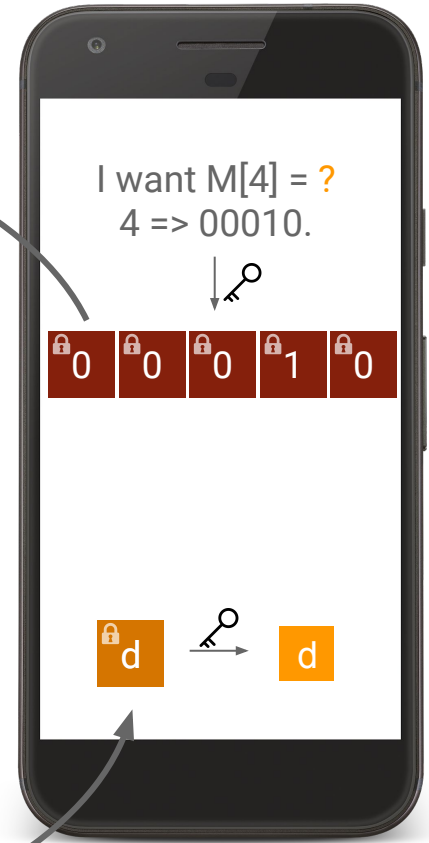
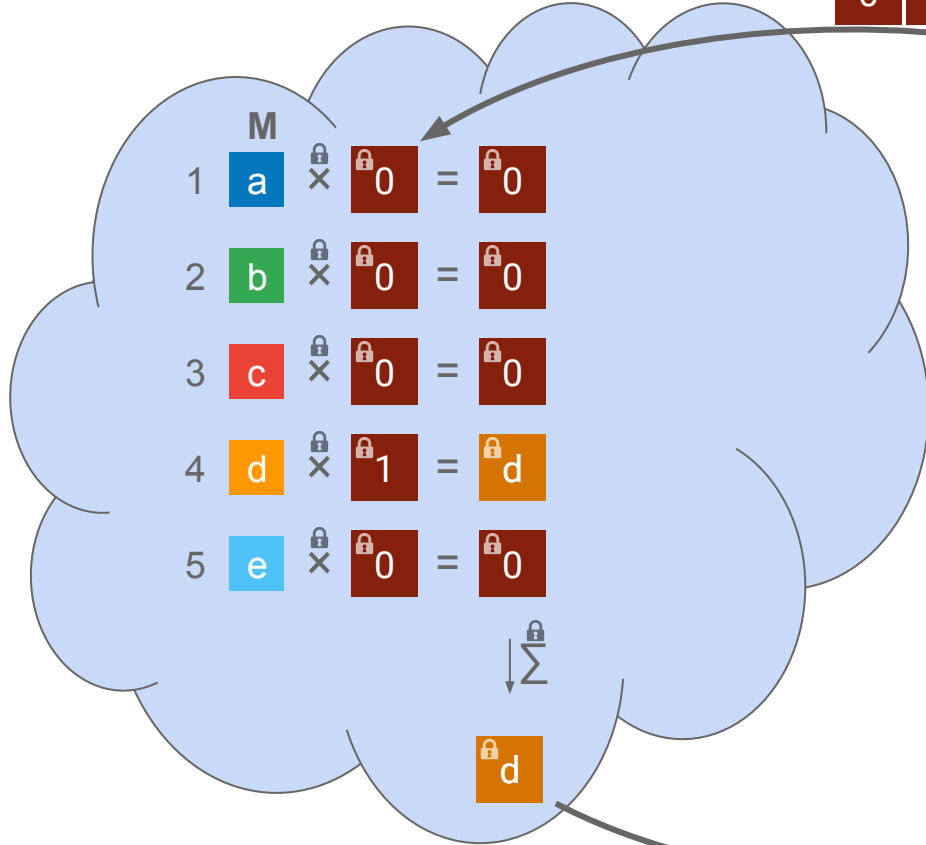
$$\text{[Dark blue box with lock icon and 'a']} \times \text{[Yellow box with lock icon and 'b']} = \text{[Green box with lock icon and 'a*b']}$$

Private Information Retrieval



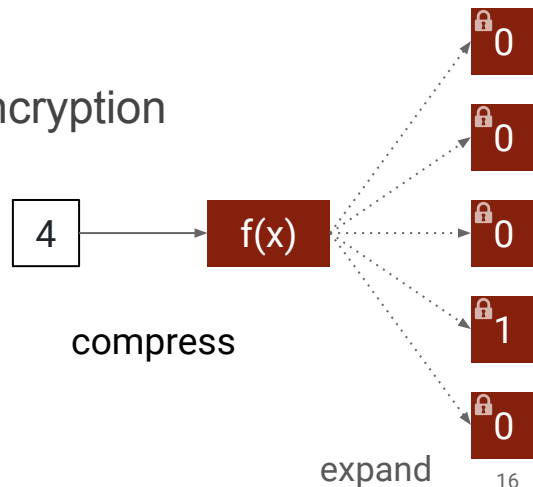
Private Information Retrieval

$\begin{matrix} \text{0} & \text{0} & \text{0} & \text{1} & \text{0} \\ \text{0} & \text{0} & \text{0} & \text{1} & \text{0} \end{matrix} ?$

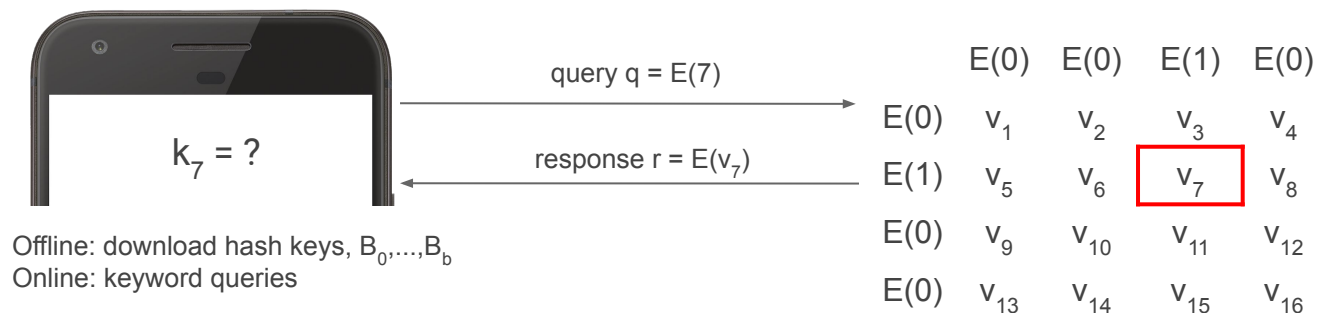
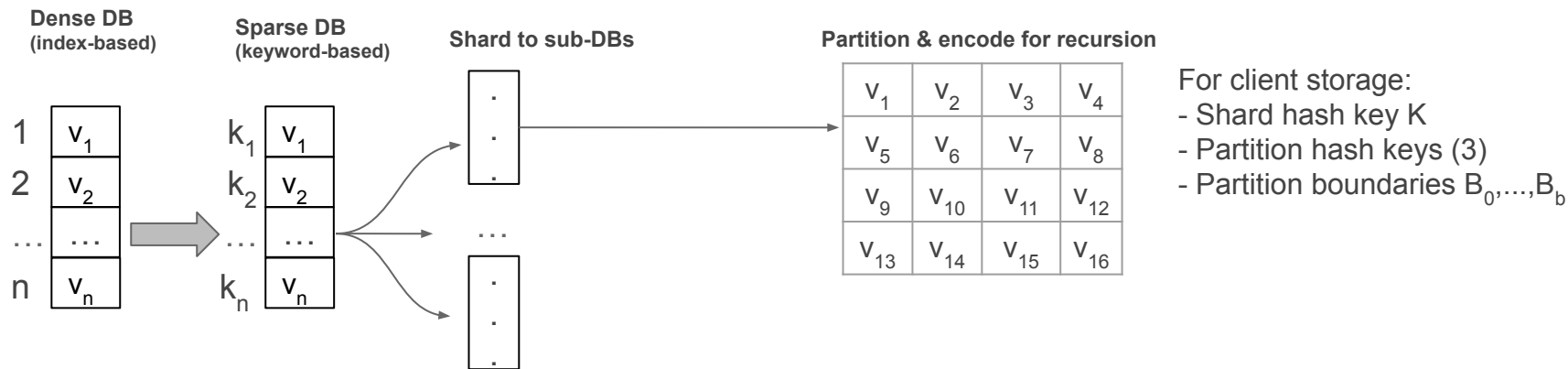


Keyword PIR framework

- Framework transforms standard lattice-based PIR schemes into keyword PIR
 - User has a query PN/UserID, not the index of the DB record
- Encodes a sparse DB as linear combination of its records
 - DB size reduction using small additional client storage
 - Compatible with recursion
 - Compatible with compression and oblivious expansion
 - Ensures minimal noise growth for fully homomorphic encryption
- Performance
 - 2x reduction in response size
 - 2x reduction in response overhead for batch PIR



Keyword PIR framework



Cost estimates

Assumptions:

- 10BN records
- Size 1.28 TB
- 10k shards -> 1M records each

Parameter	Cost estimate
PIR Public Key Size Per Device (storage required)	14 MB
Upload Bandwidth Per Query	14 KB
Download Bandwidth Per Query	21 KB
Client Time Per Query	0.1s
Server Time Per Query (Single Thread)	0.8 - 1s

Questions

Cross-service identity spoofing

- Alice messages Bob at Bob's preferred service (bob@Threema)
- Eve messages Alice impersonating Bob using bob@FooService
- Alice needs some indicator or UI to know that bob@Threema isn't bob@FooService and that when bob@FooService messages, it should not be assumed that bob@FooService is bob@Threema.

Options for solving this are:

- Storing the supported services for a contact in Contacts and if a recipient receives a message from an unknown sender, to treat it as spam or otherwise untrusted from the start.
- Requiring the fully qualified username for services that rely on usernames only - e.g. bob@threema.com vs bob