MIMIMI: Details

Raphael Robert and Konrad Kohbrok

Connection Keys

- Alice connects with Bob
- They exchange `connection_keys`

Pseudonymous Credentials

- Alice wants to publish a KeyPackage
- She creates a `PseudonymounsCredentialTBS`
- She signs the `PseudonymousCredentialTBS` and creates an `IdentityLinkTBE`

```
struct {
      IdentifierUri client_pseudonym;
      IdentifierUri user_pseudonym;
      opaque signature_public_key;
} PseudonymousCredentialTBS

struct {
      opaque pseudonymous_credential_signature<V>;
      Credential client_credential;
} IdentityLinkTBE
```

Pseudonymous Credentials cont’d

- Alice samples an `identity_link_key` from her `connection_key`
- Alice encrypts the `IdentityLinkTBE` and creates a `PseudonymousCredential`

```
identity_link_key = HKDF(connection_key,
      PseudonymousCredentialTBS, …)
```

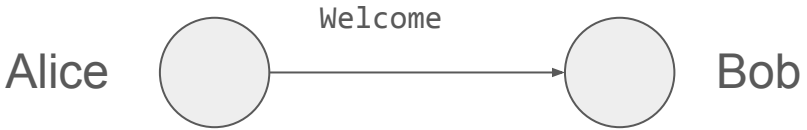
```
struct {
  IdentifierUri client_pseudonym;
  IdentifierUri user_pseudonym;
  opaque signature_public_key;
  opaque identity_link_ciphertext<V>;
} PseudonymousCredential;
```

Groups/Rooms

- When creating a group, Alice samples an `identity_link_wrapper_key`
- The key is static (for now)
- Alice encrypts the `identity_link_key` for her leaf under the `identity_link_wrapper_key`
- Alice includes the resulting ciphertext in the group’s AppSync state

Alice adds Bob

- Alice gets a KeyPackage for Bob
- Alice adds Bob to the group
- Alice includes an AppSync proposal with Bob's wrapped `identity_link_key` in the Commit
- Alices includes the group's `identity_link_wrapper_key` in the Welcome via GroupInfo Extension



- Bob decrypts (or derives) Alice's `identity_link_key` for this leaf
- Bob decrypts the `identity_link_ciphertext` in Alice's `PseudonymousCredential`
- Bob verifies the signature on Alice's `PseudonymousCredentialTBS`

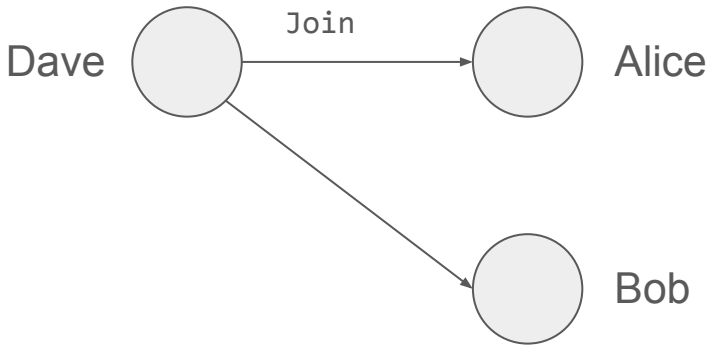
Bob adds Charly

- Charly is connected with Bob (but not necessarily with Alice)
- Bob fetches Charly's KeyPackage
- Bob derives the matching `identity_link_key`
- Bob encrypts the `identity_link_key` under the group's `identity_link_wrapper_key`
- Bob includes an AppSync proposal in the Commit to adds the ciphertext to the group's AppSync state

- Alice decrypts Charly's `identity_link_key`
- Alice verifies Charly's `PseudonymousCredential`

Dave joins the group

- Dave acquires the group's `identity_link_wrapper_key` OOB (e.g. via a join link)
- Dave creates an External Commit with an AppSync proposal[1] containing his own (wrapped) `identity_link_key`
- Alice and Bob decrypt Dave's `identity_link_key` and verify his `PseudonymousCredential`



1: Requires AppSync proposals to be legal in External Commits

Summary

- Exposes pseudonyms to the Hub
- Hides real identities by adding an encrypted credential section
- Keys for de-pseudonymization are selectively disclosed per-group via key-wrapping
- Requires the notion of a “connection” and exchange of key material during connection establishment
- Requires OOB distribution of key material for joins
- For user-level pseudonymity, just don’t include an `identity_link_key`