

Client Privacy Guidelines

The privacy of the user is of the utmost importance, because our future clients will wish to use the system to access, share and transmit confidential data. To make these processes secure, the data of our users **must** be **well** protected even if severe server/database breach has happened. To this end it is no question that the client data has to be encrypted. To ensure the users that neither the server operators nor potentially successful attacks of the servers may access their data, full end to end encryption must be carried out between the users.

User authentication

The user's access to their account must be carried out with a [Secure Remote Password protocol \(SRP\)](#). The best research paper (that we know of) on the details of this procedure was published in 2015 and is [publicly available](#).

Short summary and pointers:

- User password may never be transmitted by the client.
- User password may never be stored by the client.
- Server must not reveal the existence of an account to an attacker.
- Server must mitigate pass-the-hash in case of breach.
- Server must mitigate timing attacks on the API endpoints.

Exclusive user data encryption

The encoded password (output of the PPF) is designed to be suitable for cryptographic use, may also be split up and each part can be used for a different purpose. This property will be the corner stone of the secure encryption of the user's data. One part of the encoded password will be called encryption key. This key can be used to encrypt any data that only the user themselves shall access ever. Such encrypted data may be stored by the server without damaging privacy. To make it possible for the user to change their password and retain the access to their data, the encryption key shall be used **only** to encrypt a master key. The master key is generated by the user's client at registration and it's encrypted form will be stored by the server. This process is very similar to how MEGA works according to their [security whitepaper](#).

Short summary and pointers:

- User will need to create a master key that is used solely for their exclusive data.
- The only form of the master key that shall be transmitted, is encrypted by the encryption key.
- All other sensitive data that is exclusive to the user shall be encrypted with the master key before transmission.
- The master key and subsequently decrypted data must be kept securely in the client's session.

- Decrypted client data shall only be stored locally if their security is highly guaranteed or the user accepts the security risk. (note: this is up for discussion)

Peer to peer user data encryption

Users in the system need a reliable way to exchange messages so that only their intended receiver is able to read them. Following a successful registration, our users shall set up asymmetric encryption keys so they will be able to exchange messages/requests securely. The private key must be stored on the server in encrypted form by the user's master key. Having this system also enables the client applications to do some automatic housekeeping duties related to group-encryption setup and update. (see next section)

Short summary and pointers:

- All users must have asymmetric keys on the server to receive and send peer to peer messages.
- Asymmetric keys shall also be used to sign peer to peer messages.

Group data encryption

For performance and storage reasons the encryption used for group related data such as chats and nano requests/responses shall be symmetrical. For a symmetrical encryption all group members must know the key. Initially the key shall be generated by the owner when the group is created. Then upon inviting or accepting users into the group the inviting member shall share the key with the new member using a peer to peer message. The receiver client shall check if the message is from a member and may automatically accept it, encrypting it with the user's master key and storing it for themselves. Sharing previous keys with a new member cannot be regulated since all members know and can decide to share them with anyone anyways. Just as any member will be able to share the unencrypted group data with anyone outside the group they may also share the key of the encryption. This cannot be solved and can hardly be mitigated by the system. All members of a group must all trust each other for their shared privacy to be protected. To mitigate accidental leak of an encryption key the system shall support replacing it with a new one. This will prevent an attacker to have perpetual key for the group's encryption.

Short summary and pointers:

- Groups have symmetrical keys for encryption.
- A key will be stored and thus encrypted by each member separately, by their own master key.
- Members of the group may transmit the unencrypted keys of the group to other members by peer to peer messages exclusively.
- Transmission of group keys may be needed when:
 - a new member has joined
 - a new key has been generated and everyone must receive it
 - a late joiner wishes to see messages posted long ago (they have the current key but need an older one)

- A client may only send a group encryption key automatically (without confirmation from the user) to a member of the group.

Unencrypted server data

The server will only store unencrypted data that is essential for its job to be doable. This data cannot be encrypted from the server by its nature. The privacy of the users will not suffer much though, because these data are accessible to them anyways. For example: this means the server stores the members of a group, but all members of the group can verify the members of the group. There is no way for the server to falsify such data with impunity.

Public group data

Encryption in a group that is publicly or anonymously accessible is not beneficial at best or provides a false sense of security in reality. As such a group that is made public will not encrypt any group data that may be accessible as an anonymous user. If the group is made private again, then from then on all data shall be encrypted as normal.