End-to-end Encryption design in Nextcloud

Contents

- Intro Nextcloud
- What is E2EE
- E2EE requirements
- E2EE technical design
  - Initialization
  - File handling
  - Sharing
- Edge cases & limitations

What is Nextcloud?

- Nextcloud Files

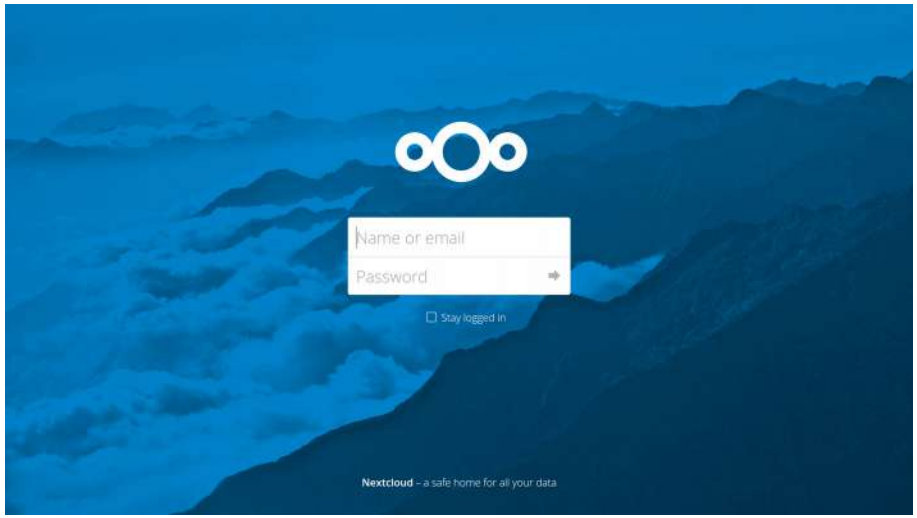
  private, self-hosted cloud keeping your data secure

- Nextcloud Talk

  self-hosted secure video/text chat

- Nextcloud Groupware

  Easy mail/calendar/contact



Features

- Open Source
- Easy to use web UI
- Video/text chat
- Collaborative editing
- Control access rights
- Auditing, workflow
- External storage
- LDAP/ SAML/2FA
- Developer APIs
- Mobile/desktop clients

What is End-to-end Encryption

Fully protects data/communication from user-to-user so no interception in between can capture data, including servers the data passes through.

- Signal, whatapp, ...
- PGP/GPG for mail

End-to-end encryption in Nextcloud

Core goals of our design

- Protect data 100% from the server
  - Keep data safe in case of fully compromised server or malicious administrator
- Be super easy for the end user
  - Complexity is enemy of security. Assumption: user makes mistakes, administrator is competent.

Requirements of E2EE in Nextcloud

- Allow secure sharing and
  - Guarantee confidentiality
    - Only authorized users can have access
  - Guarantee integrity
    - Files can not be tampered with undetected
  - Guarantee authenticity
    - Ownership is always clear
- Use tested, widely used libraries
  - Available on recent versions of iOS, Android, Mac, Windows, Linux, PHP7
- Offer optional data recovery
  - With off-line admin key. Users gets warned when this is enabled.
- Multi-device support
  - Friction-less access for all user devices
- Easy key exchange
  - Sharing should be seamless, secure and not require passwords
- Versioning of protocol
  - Improvements can be made
- Full activity logging possible for auditing

Accepted feature loss

- Only top-folder-level sharing
  - No sharing of individual files or folders in an encrypted folder
- No group sharing
- No public link sharing
- No web access to data
  - No collaborative editing
- No server capabilities like versioning, trash, comments, favorites, server-side search.

Some of these can, in time, be mitigated. Others are inherent to secure End-to-end Encryption where the server has no knowledge of the data.

Example: web interface access requires code from server → which can't be trusted. Would fundamentally break the security model.

Next slides: explain design

- Initialization
  - Create keys, add devices
- File handling
  - Create folder, files, download files etc
- Secure sharing
  - Sharing, unsharing

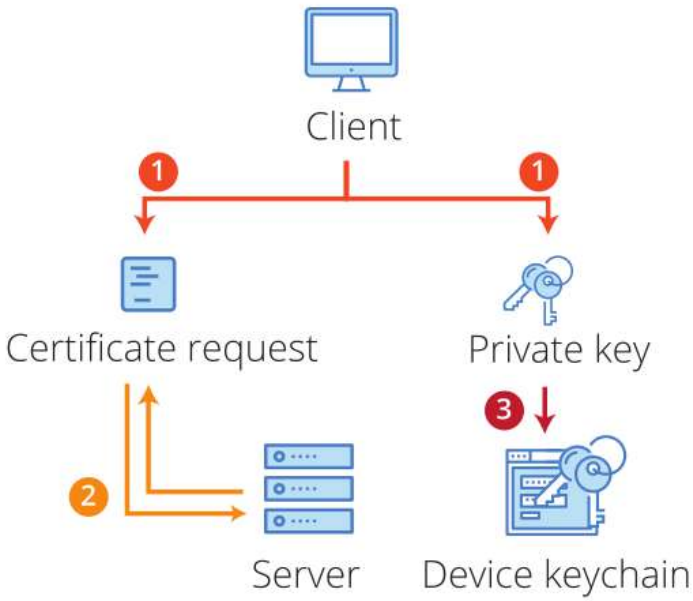
Creating a secure identity

- Keys:
  - Generating
  - Signing
  - Encrypting
  - Syncing
- Adding new device

Initialization – step 1

1

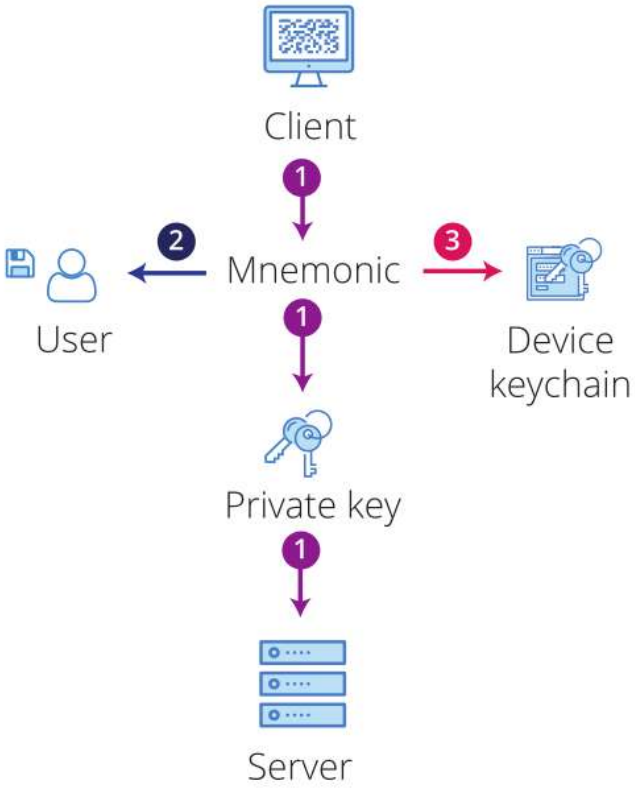
1. Client generates a new X.509 certificate request and private key.
2. Certificate gets signed by server.
3. Private key is stored in keychain of device.



Initialization – step 2

2

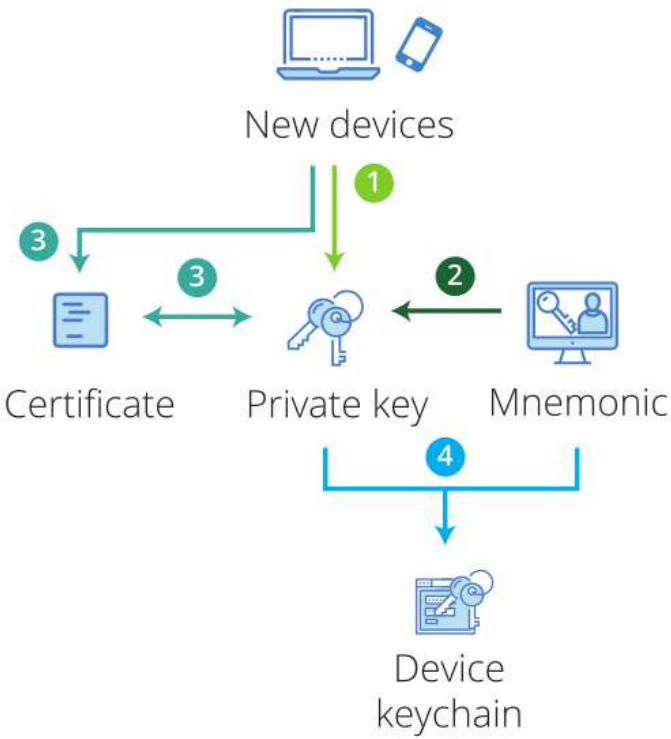
1. Client encrypts private key with 12 word mnemonic and uploads it to the server.
2. Mnemonic is displayed to users and they are asked to store it.
3. Mnemonic is stored in keychain of device.



Initialization – step 3

3

1. New devices download the encrypted private key.
2. Private key gets decrypted with the 12 word mnemonic from the user.
3. New devices check if private key belongs to certificate.
4. Private key and mnemonic are stored in the keychain of the device.



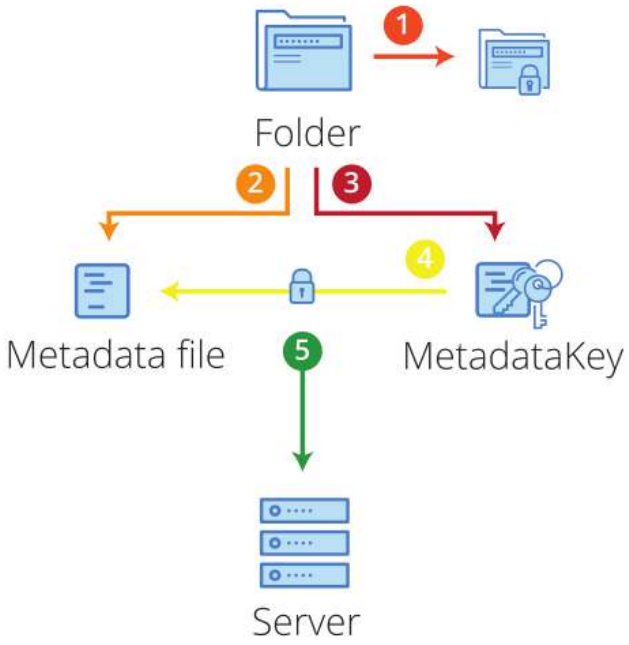
File handling

- Create E2EE folder
- Upload to server
- Add files
- Download on other device

File Handling – Create folder

1

1. Create folder and mark to server as encrypted.
2. Generate metadata file.
3. Generate metadataKey, encrypted to all public keys that have access to the folder.
4. Use metadataKey to encrypt all values in metadata file.
5. Store encrypted metadata on server.



File Handling – Add file

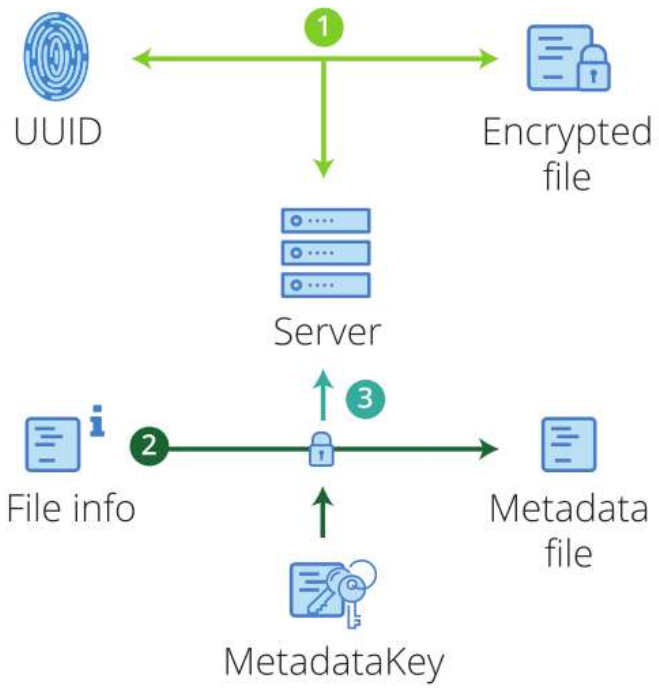
2

1. Generate new 128bit encryption key for file.
2. Encrypt it using AES/GCM/NoPadding.

File Handling – Upload to server

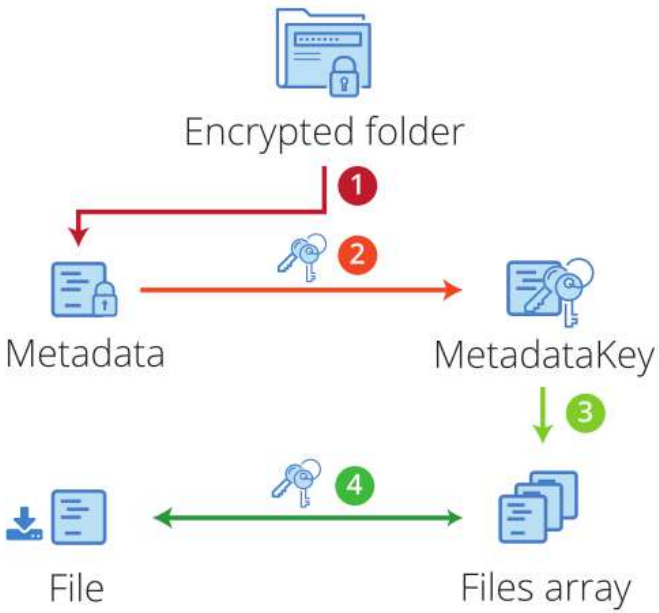
3

1. Generate random identifier (UUID) and upload encrypted file, using random identifier as file ID.
2. Add new file info to files array in metadata file, encrypted with metadataKey.
3. Update metadata on server.



File Handling – Add 2nd device

1. Download metadata of encrypted folder.
2. Use private key to decrypt metadataKey.
3. Use metadataKey to decrypt to files array.
4. Download the files and decrypt them using 128bit AES/GCM/NoPadding using keys from the files array.

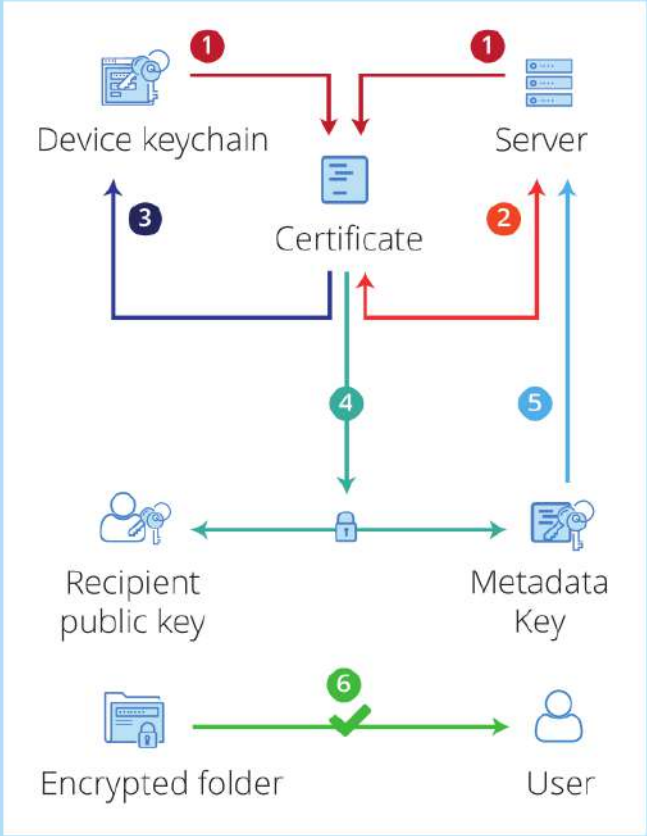


Sharing and unsharing

- Sharing
- Unsharing

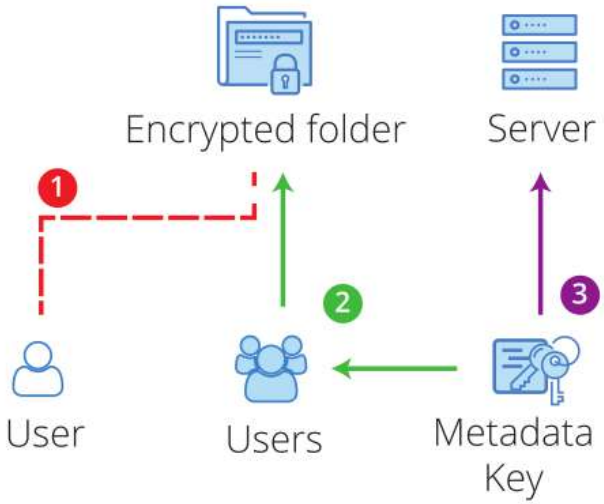
Sharing

1. Check for certificate of specified user ID or download it from the server (Trust On First Use / TOFU).
2. Verify if the certificate is issued by the server.
3. Store user certificate locally.
4. Re-encrypt metadataKey to recipient public key.
5. Upload updated metadata to server.
6. Share folder with user through OCS sharing API.



Unsharing

1. Unshare encrypted folder with user through OCS sharing API.
2. Generate new metadataKey and encrypt to everyone who now has access.
3. Upload metadata to server.



Edge case: complete key loss

Options available in case the user lost the key.

Recall: design assumes user is weakest link. So:

- User does not choose a password but is given one
- User is asked to store password but assumption is user won't

- Any user device can recover mnemonic to decrypt key
  - Lost phone? Add new phone, using laptop to show key
- Optional recovery key
  - When recovery key is enabled, private/public key pair is generated. Users will encrypt all data against public key. Private key protected with mnemonic, shown once to server admin for secure, off-line storage.
  - All devices lost? Admin can use recovery key to recover user data. NOT USER KEY or IDENTITY, they are lost.
  - Enterprise use case: employees which have left the company.
- If CSR/HSM: new user key and identity can be created.
  - A hardware security module can securely generate a new user identity.

More information

- nextcloud.com/endtoend
  - Contains link to detailed design whitepaper
- github.com/nextcloud
  - /ios
  - /android
  - /client
  - end_to_end_encryption
  - end_to_end_encryption_rfc

A safe home for all your data

Nextcloud GmbH
Kronenstr. 22A
70173 Stuttgart

Germany

+49.711.896656-0
hello@nextcloud.com

nextcloud.com