Delta Chat and Iroh

Integrating Iroh into an Application

number 0

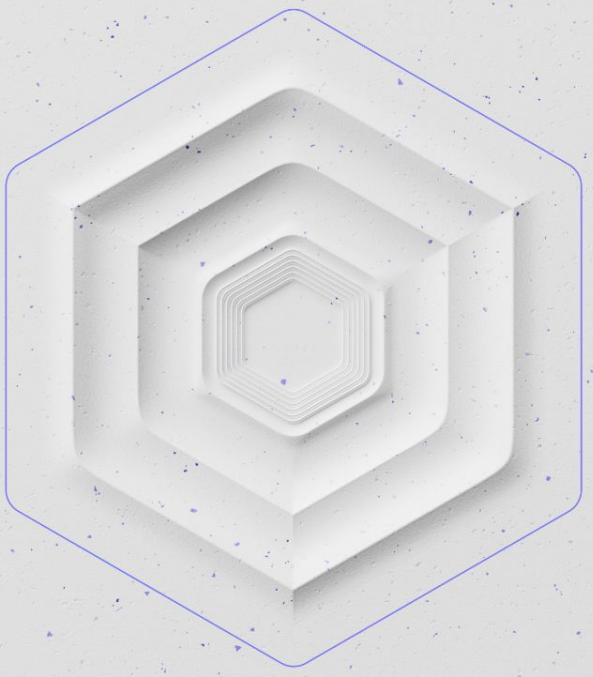
https://mastodon.social/@flub

Iroh

IPFS reimagined

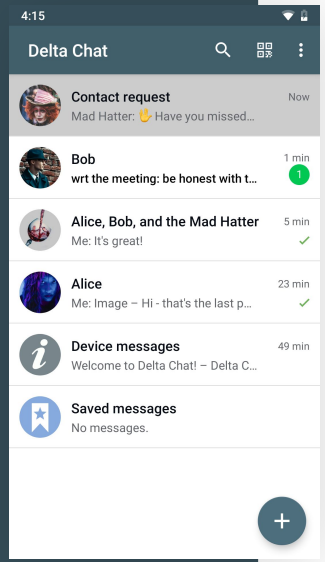
First prototype:

- Transfers data, verified
- Content addressable
- P2p
- Authenticated



Delta Chat

- Messenger with no infrastructure
- Chat over Email
  - Chat with everyone
- Remarkably censorship resistant
- Opportunistic encryption
- Multi-platform
  - Android
  - iOS
  - Desktop



GNU/Linux

Download On FLATHUB | Get .deb for Debian/Ubuntu | Get .rpm for Fedora | Get Universal AppImage | Get Universal tar.gz | Build from AUR

Source Code

Flatpak manual install: `flatpak install flathub chat.delta.desktop`

Arch manual install: `yay -S deltachat-desktop-git`

Nix manual install: `nix-env -f "<nixpkgs>" -iA deltachat-desktop`

Other platforms

Android

GET IT ON Google Play | GET IT ON F-Droid | GET IT ON Paskoocheh | DISPONIBLE EN Apklis | available at amazon appstore | Download APK

Source Code

iPhone/iPad

Download on the App Store

Source Code

macOS

Download on the Mac App Store | Download DeltaChat.dmg | Install with Homebrew

Source Code

Homebrew manual install: `brew install --cask deltachat`

Windows

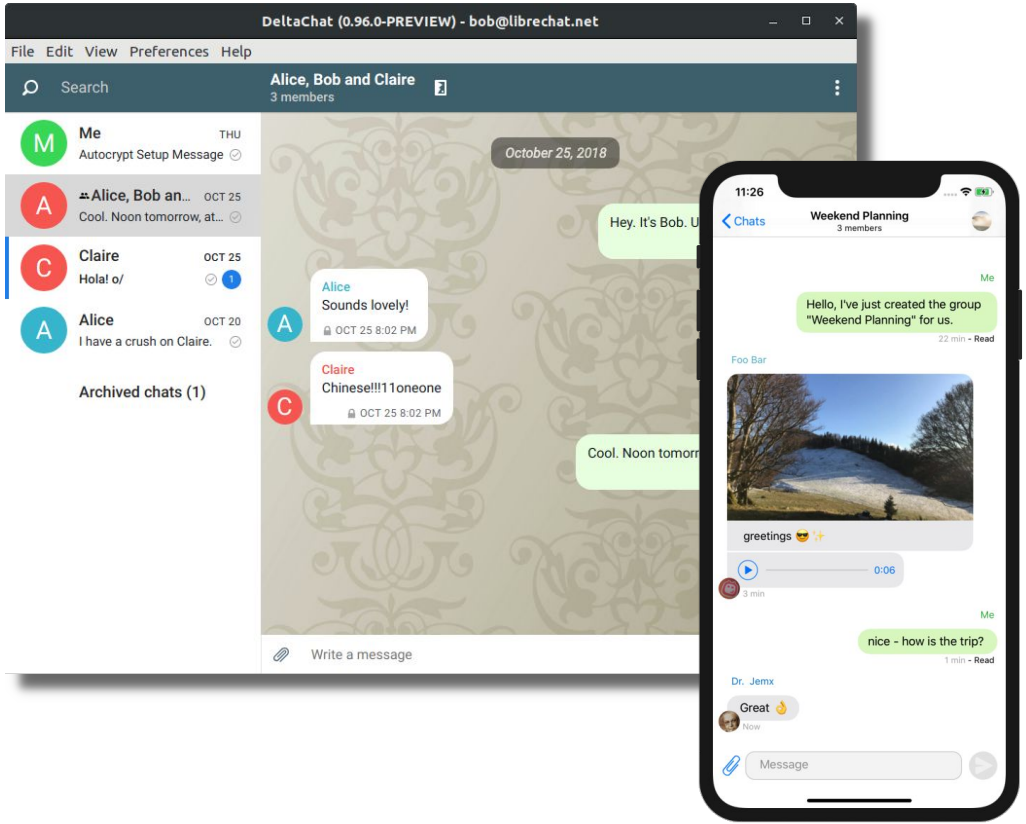
Get it from Microsoft | Download Setup.exe

Multi-Device

- Messages stored on IMAP server
- Multiple clients sync unread state.

Yet… remember encryption

- Private key needs syncing
- => Export-Import backups



Transferring Backups

- Has worked for many years
- By hand this is painful!
- Let's add iroh!

Initial Restrictions

- Same local network
- Out-of-band communication using QR codes
  - An established pattern in Delta Chat
- One direction:
  - The provider shows a QR code
  - QR code contains hash of backup
  - QR code contains authentication

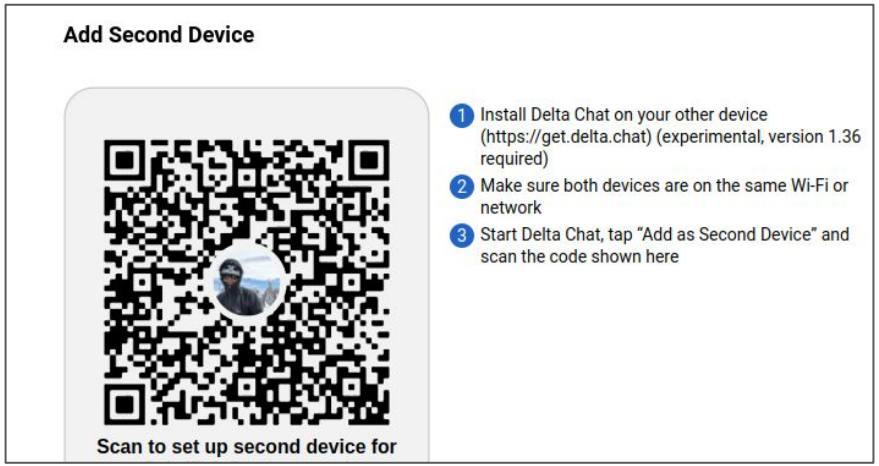
Delta Chat and Iroh:
The perfect match

- Both projects written in Rust
  - Delta Chat has a rust “core” for all platforms + 3 apps with bindings to the core
- Delta Chat already ships Rust code to the platforms
  - C FFI bindings use by Android and iOS
  - Desktop uses a mixture of FFI + JSON-RPC
- => This is easy!

The Plan

- Iroh transfers Collections
- Delta Chat Provider does:
  - Export database
  - Add export to collection
  - Add all attachments/blobs to collection
  - Start provider/server
  - Create QR code
- Delta Chat Receiver does:
  - Gets QR code
  - Connect to provider
  - Requests Hash of collection
  - Receives all files
  - Imports database
  - Starts system

- Iroh impl prepared
- Delta Chat core PR prepared
  - Including tests & test demo tool
  - Reviewed
- Time for the apps to use it!
- What could possibly go wrong?



Did anyone mention platforms?

- New dependencies, new breakage
- Iroh uses QIUC via the quinn crate
  - ECN not supported on some older linux (hi ancient android)
  - sendmmsg not available
  - quinn merged PRs swiftly!

Who needs connectivity anyway?

- “Discovery is not our problem, it’s the local network”
- …
- What is my local network?

Solution:

- Provider binds to 0.0.0.0
- Puts all local IP addresses in QR code
- getifaddr(3) exists, easy!
- Platforms, permissions, versions
  - How about using a netlink socket?
    - On some versions, sure
  - How about dlopen libc.so?
    - On some versions, sure
  - Running tests on the phones still didn’t surface all permission issues.

Users cancel operations

- We (developers) think this is rare…
- User:
  - Start
  - No, did I press that button? Cancel!
  - Oh let's start anyway.
  - Wait, didn't see that. Cancel!
  - Ok, let's start this thing anyway
- Me:
  - But… this is the middle of a database transaction?
  - I'm just writing this huge file!

Users like progress

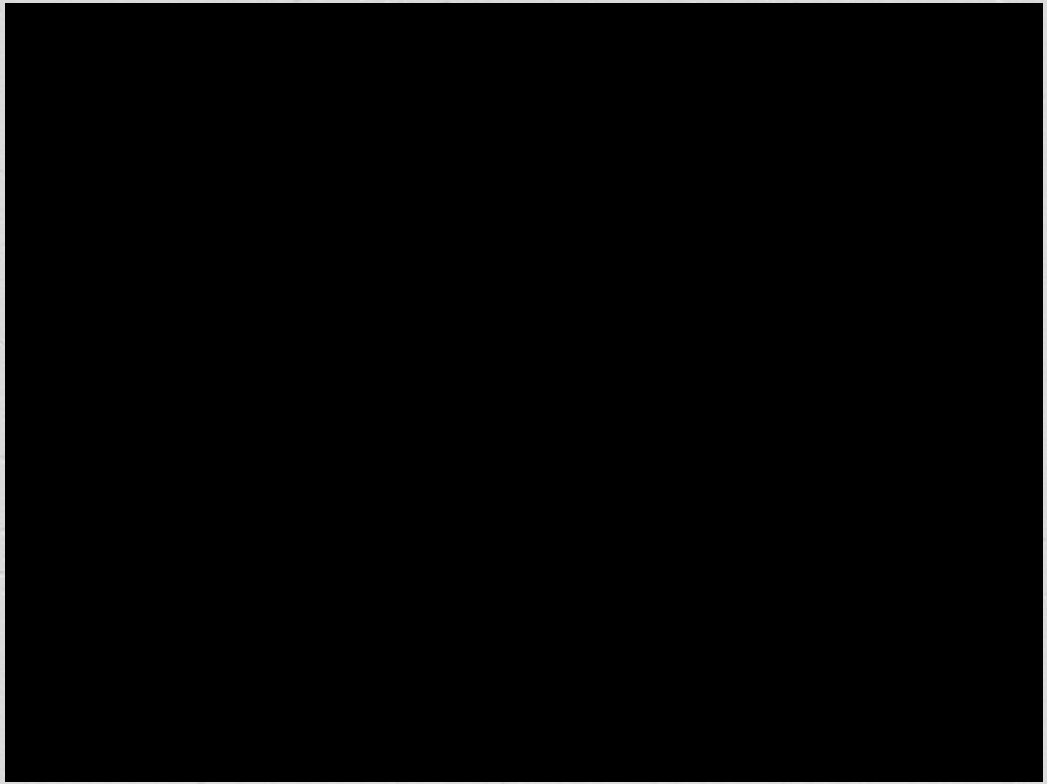
- Iroh has very smooth **receiver** side progress
- **Sender** side progress is a bit rougher unfortunately
  - We should fix this
  - Users really are bothered

The Result

- 3 iroh releases
- 2 default-net releases
    - (apologies to external maintainer!)

Majority of issues are with integration code. Yet:

- This was written by me
- I knew both code bases well



The Future

- Connectivity across networks
  - NAT traversal & hole punching
- Bi-directional
  - Users don’t like a single scan including all secrets
- Who needs content addressing anyway?
  - Repeated point of friction
  - “Give me the current export”
  - IPNS?
  - “Just give me a stream”