Shipping End-to-End Encryption to Billions

Jon Millican

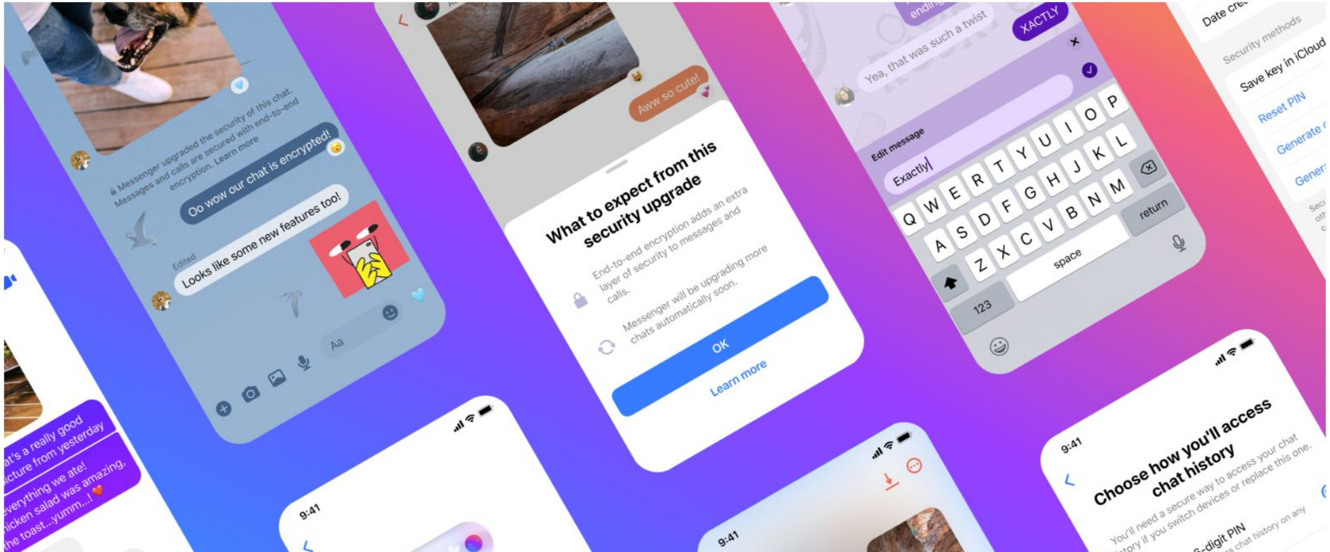
March 2024

Messenger

Launching Default End-to-End Encryption on Messenger

December 6, 2023

By Loredana Crisan, Head of Messenger



Messenger upgraded the security of this chat. Messages and calls are secured with end-to-end encryption. Learn more

Oo wow our chat is encrypted!

Edited

Looks like some new features too!

Aww so cute!

What to expect from this security upgrade

End-to-end encryption adds an extra layer of security to messages and calls.

Messenger will be upgrading more chats automatically soon.

Yea, that was such a twist

XACTLY

Edit message

Exactly

Choose how you'll access chat history

You'll need a secure way to access your chat history if you switch devices or replace this one.

Agenda

1 **The Messenger product**

2 End-to-end encryption vs “Open” messaging

3 Storage

4 Protocol challenges

5 Features

6 Summary

SMS

Single device identities.

Minimal cloud augmentation.

Webmail

Access everywhere anywhere.

No specific home device.

Cloud Rendering

Messenger Product

Facebook account-linked

Accessed via cloud-based account.

👤 Addressed by name/photo

Global identifiers are internal implementation detail.

📱 💻 Multi-device native

No defined home device. Sometimes no de facto home device!

👩‍💻 Web heavy

Web remains an important surface for many people.

🌟 Feature-rich

Vast number of features, each with their own semantics.

👥 Graph-integrated

Heavily used for sharing Facebook content.

Agenda

1 The Messenger product

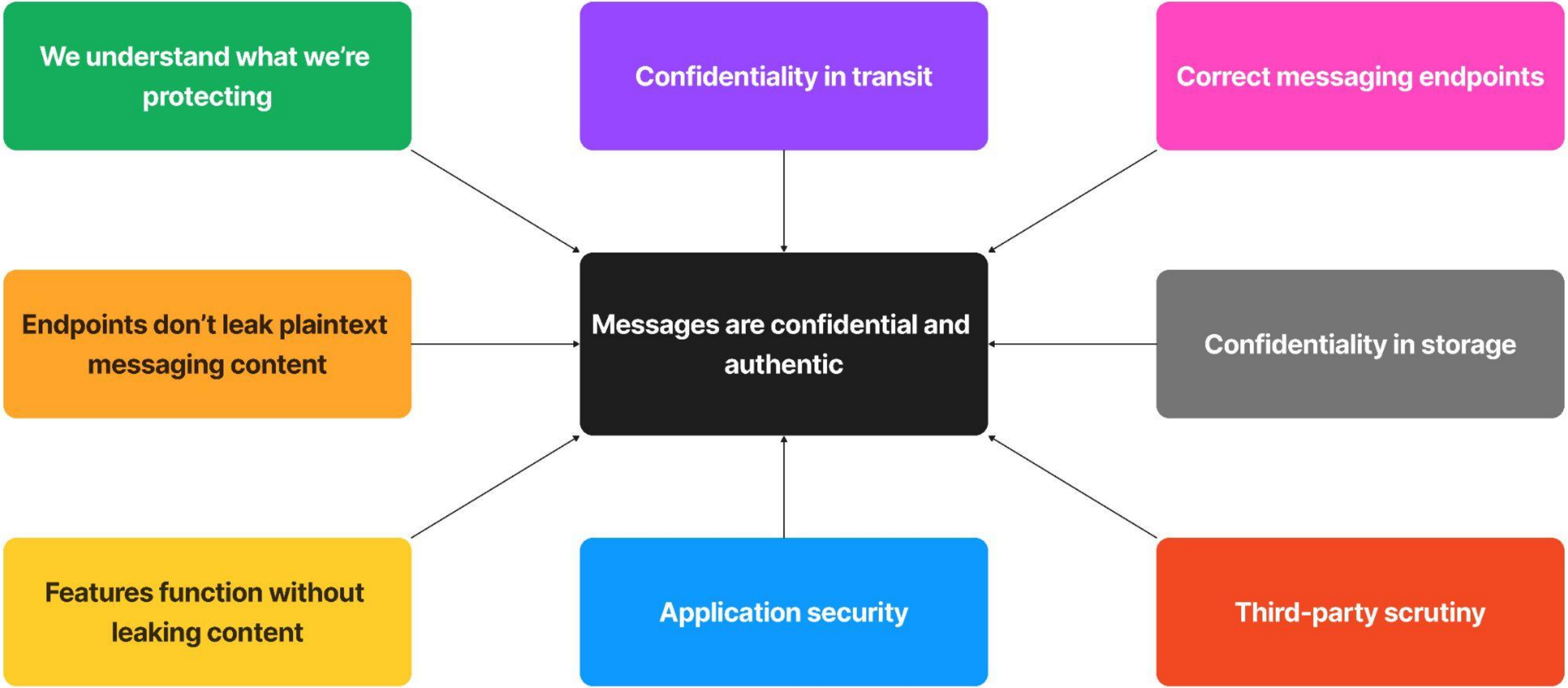
2 **End-to-end encryption vs “Open” messaging**

3 Storage

4 Protocol challenges

5 Features

6 Summary



We understand what we're protecting

Confidentiality in transit

Correct messaging endpoints

Endpoints don't leak plaintext messaging content

Messages are confidential and authentic

Confidentiality in storage

Features function without leaking content

Application security

Third-party scrutiny

It’s (mostly) not about cryptography!

🔒 Few outside of this room cares about the crypto!

People care about protecting data.

✉️ E2EE transmission wasn’t our hardest problem to solve

Signal Protocol, MLS, etc already exist

It’s all about the client device!

📡 **Bytes transmitted == bytes received**

Features must be architected in a client-centric manner.

📱 **Recipient devices known in advance**

“End-to-end” implies you know the ends.

💾 **Storage managed by endpoints**

Devices become source of truth for message history.

The server can’t always help you out!

🖥️ Server can’t ensure compatibility

No transcoding, format sanitisation, etc.

🔧 Any server augmented features are difficult

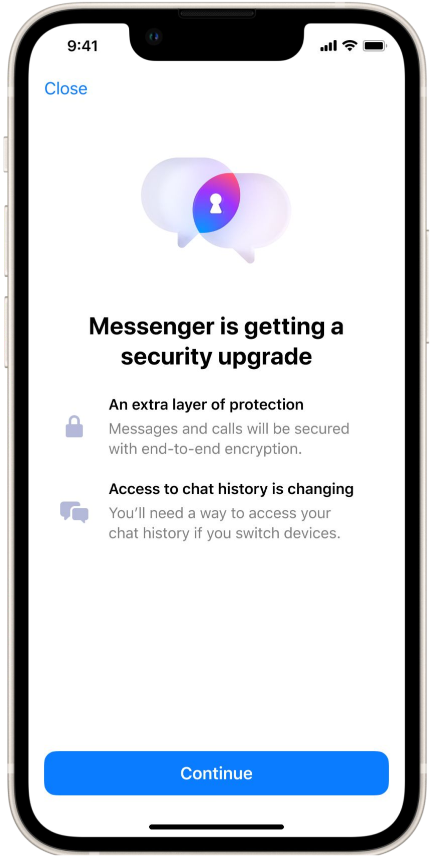
Can’t reveal data that leaks message content.

🔀 Can’t shim for clients when version changes happen

Data formatting, protocol versioning, etc

Agenda

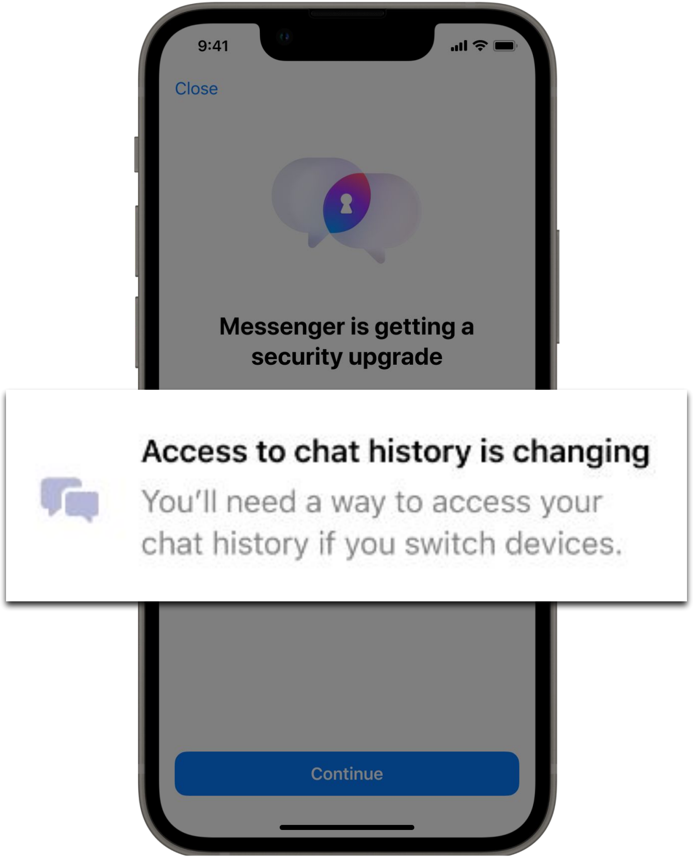
1. The Messenger product
2. End-to-end encryption vs “Open” messaging
3. **Storage**
4. Protocol challenges
5. Features
6. Summary



Messenger is getting a security upgrade

An extra layer of protection
Messages and calls will be secured with end-to-end encryption.

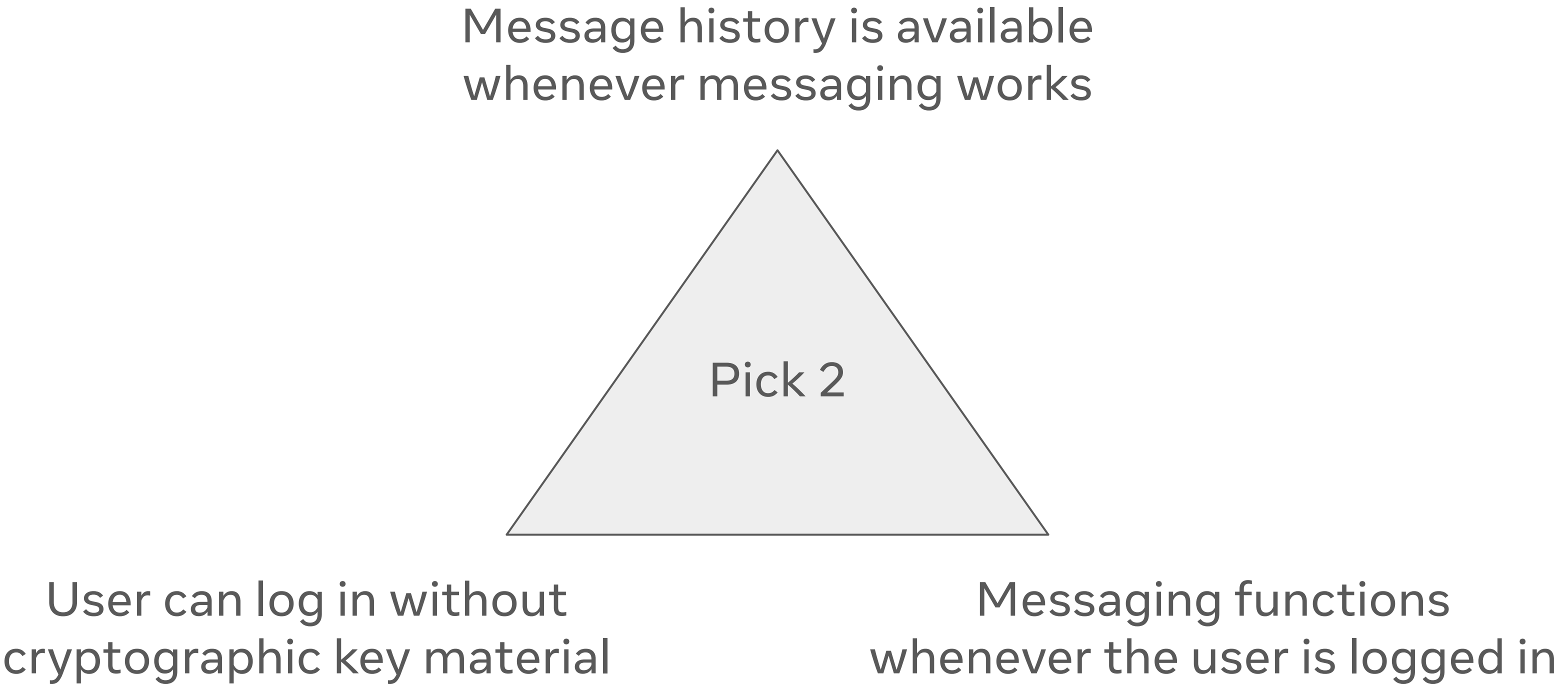
Access to chat history is changing
You'll need a way to access your chat history if you switch devices.



Messenger is getting a security upgrade

Access to chat history is changing

You'll need a way to access your chat history if you switch devices.



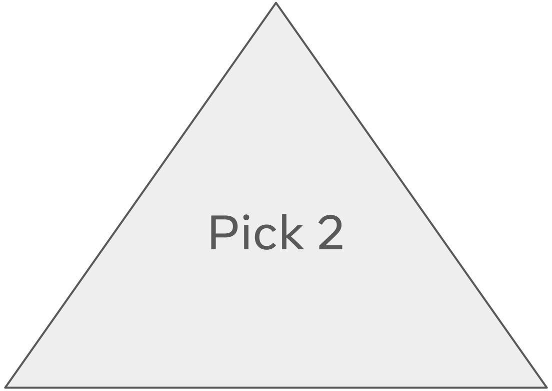
Message history is available whenever messaging works

Pick 2

User can log in without cryptographic key material

Messaging functions whenever the user is logged in

~~Message history is available whenever messaging works~~



Pick 2

User can log in without cryptographic key material

Messaging functions whenever the user is logged in

Storage Scenarios

📲 Changing “primary” device

Must support upgrade / transfer

😢 Lost devices

Must function as a true backup.

🗑️ Low-storage devices

Offload data to the server.

🔀 Platform switching

Support users who roam across platforms..

🕸️ Web support

Can’t rely on mobile-only infrastructure.

📱💻 Multi-device

Seamlessly shared across devices

Storage Privacy

🔒 **Inaccessible to Meta**

Important E2EE goal.

👍 **Under user’s control**

We can’t override their settings.

User friction is rough!

🚫 Users don’t want to be interrupted when opening Messenger

Frequently click away; often don’t read; sometimes permanently churn!

🤔 Purpose is hard to grok

Why does this matter? Is it authentication?

📊 No quick tests of long-term performance

It will take months or years to understand some edge cases

Minimising impact

⚖️ **Provide best option for each user**

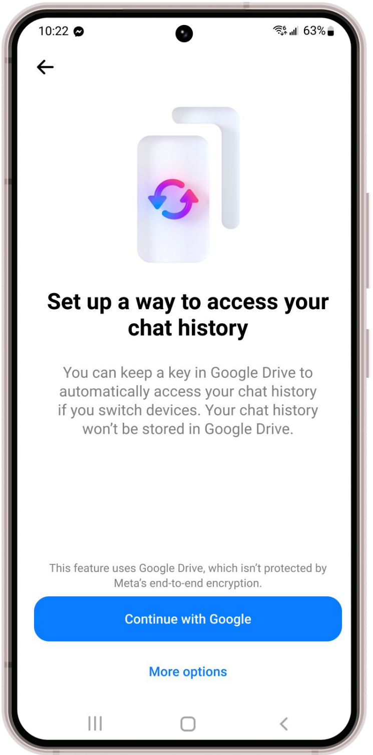
Different methods work best for different usage patterns

🔁 **Give them multiple opportunities**

Choice is forced; but allow dismissal initially

🫵 **Focus on immediate impact to them**

Security benefits don’t always resonate

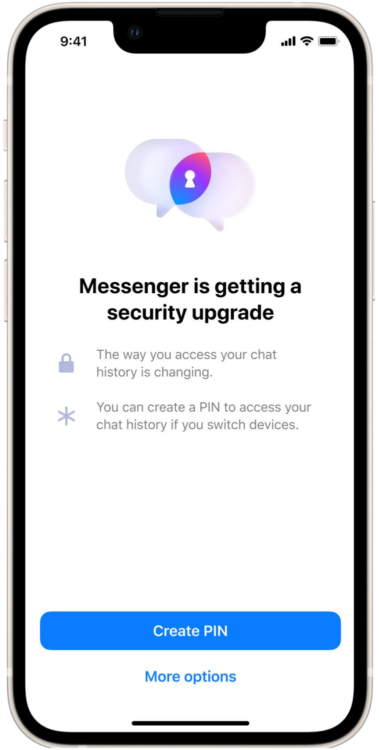


Set up a way to access your chat history

You can keep a key in Google Drive to automatically access your chat history if you switch devices. Your chat history won't be stored in Google Drive.

This feature uses Google Drive, which isn't protected by Meta's end-to-end encryption.

Google Drive

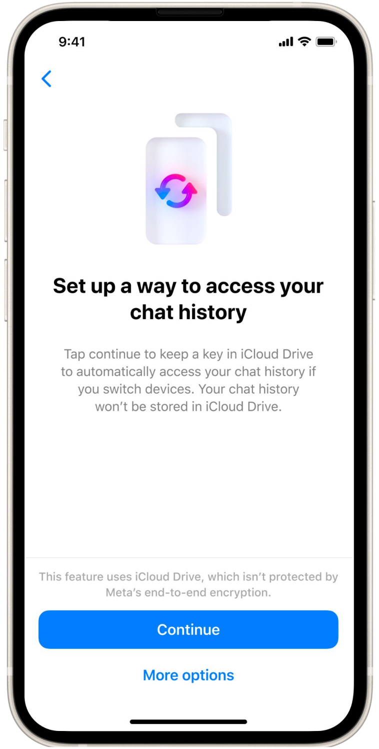


Messenger is getting a security upgrade

The way you access your chat history is changing.

You can create a PIN to access your chat history if you switch devices.

PIN



Set up a way to access your chat history

Tap continue to keep a key in iCloud Drive to automatically access your chat history if you switch devices. Your chat history won't be stored in iCloud Drive.

This feature uses iCloud Drive, which isn't protected by Meta's end-to-end encryption.

iCloud Drive

Agenda

1 The Messenger product

2 End-to-end encryption vs “Open” messaging

3 Storage

4 **Protocol challenges**

5 Features

6 Summary

New storage protocol

✏️ Storage required new protocol design

Ratcheted encryption not suitable for long-term storage

🤫 No forward secrecy

Explicit anti-goal of backups-like system

🤫 Endpoints can be virtual devices

Enable data recovery without a physical device

🚫 Device revocation

Key rotation is on its way!

The Labyrinth Encrypted Message Storage Protocol

Meta

December 6, 2023
Version 1

Metadata was operationally necessary

🫣 We tried to de-identify storage!

Unlinked mailboxes, PRFs for thread IDs, OPE for timestamps

🐞 De-identification made debugging infeasible

Employees reported problems, but no way to dig in

🧐 Re-identified storage to achieve product readiness

Closer to original well-understood architecture

Edge cases remain!

🔐 Messages encrypted to known devices

Signal Protocol endpoints are physical devices

🔃 Storage only populated on message receipt

Must decrypt before storing

🗑️ Message loss if devices go permanently offline

Nothing to decrypt and store

Performance explorations

⌛ Offline devices take a while to catch up

Signal Protocol assumes mostly in-order delivery

⇉ Sometimes use Labyrinth over Signal

Faster to populate multi-device inbox from secure storage than transport.

👥 Groups will be harder to scale

Some per-device costs scale linearly with devices

Agenda

1 The Messenger product

2 End-to-end encryption vs “Open” messaging

3 Storage

4 Protocol challenges

5 **Features**

6 Summary

Client-side limitations

⚙️ **Some features cannot function purely client-side**

Key data lives on the server

⚖️ **Prioritising functionality alongside privacy**

Tough trade offs required in places

Example:
Sticker Search

😴 Sticker library only useful with generic search queries

Not interesting on their own

📚 Large sticker library

Can’t store fully client-side

💿 Hosted by Meta

Nobody else to query

De-identification Technology

📰 Generic values are not always sensitive

Primarily aim to protect when user-linked.

🤫 Oblivious HTTP

Hide IP addresses, which can be personally identifiable.

😷 Anonymous Credentials

Authenticate access; rather than users.

First-party previews

🖼️ **Users value in-thread previews**

Especially important for a social network chat function.

🧐 **Content IDs already known to Meta**

These aren’t new information.

🌐 **Shared content skews public**

Can be loaded without knowing who’s accessing it

Agenda

1. The Messenger product
2. End-to-end encryption vs “Open” messaging
3. Storage
4. Protocol challenges
5. Features
6. **Summary**

Summary

☁️ **E2EE is complex for cloud-style services**

Webmail vs SMS is a huge difference in messaging

🗝️ **Key management remains hard for users**

Not a simple transition, and we’re still learning

🛳️ **We’re getting there!**

5 years in the making, and we’re shipping!

Thanks!