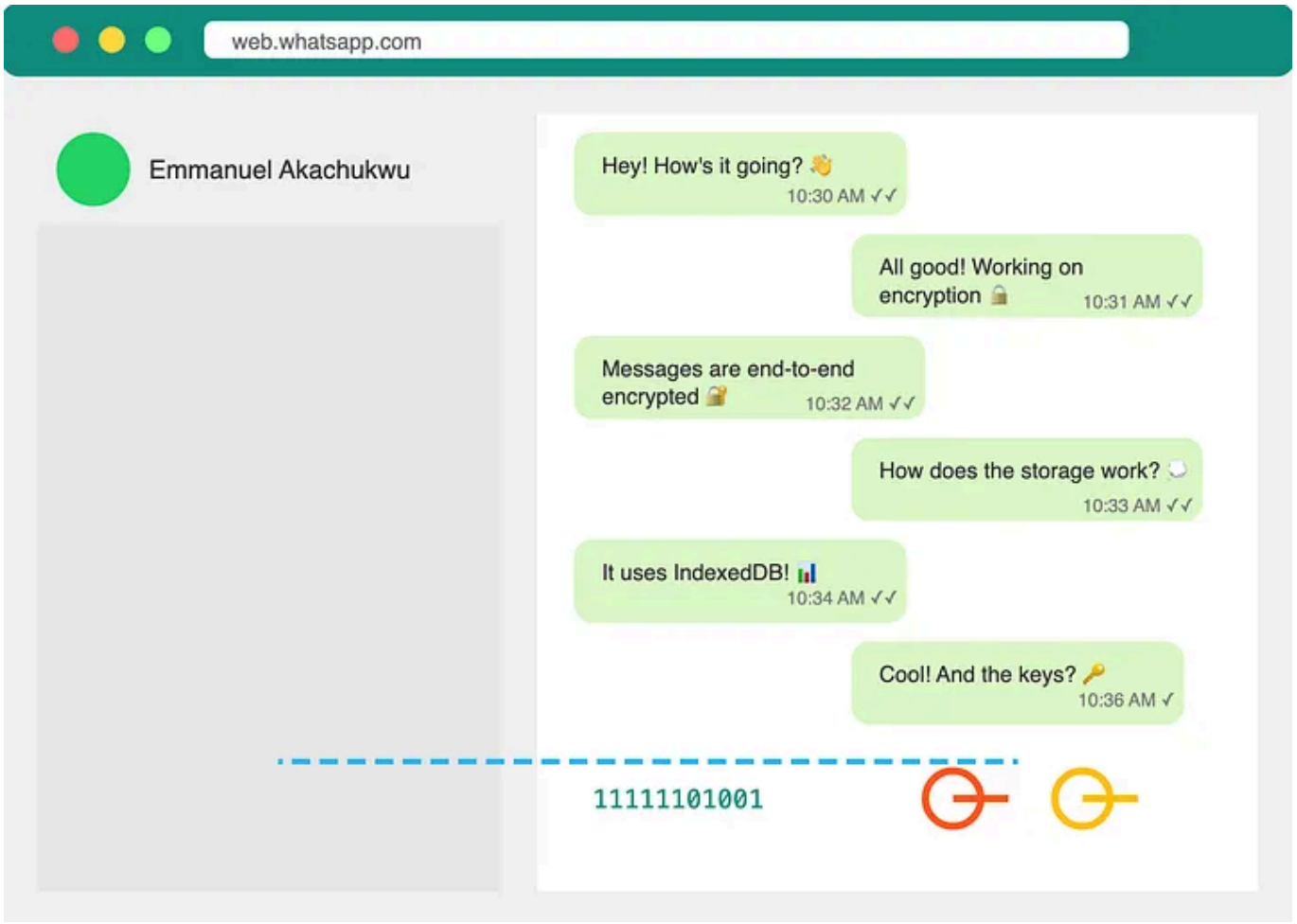
WhatsApp Web's Data Journey: Storage Locations and End-to-End Encryption

Emmanuel Akachukwu Follow 6 min read · Jan 22, 2025



While trying to figure out the architecture and design of WhatsApp some weeks back, I stormed into this article by cometchat which gives a good enough overview of how WhatsApp works. Note that in the conclusion part of the article, it was clearly stated that the exact specifications of WhatsApp's architecture and design were unknown by the writer so the article is based on the writer's research and ideas of what WhatsApp might be doing and not from any supposed WhatsApp ex-employee.

> *While we don't know the exact specifications of WhatsApp's technical architecture and system design, we can get a good idea based on the technologies that WhatsApp employs*

With that in mind, it is important to note that this article is based on the same.

Storage

As you probably noticed, unlike other messaging services like Telegram, WhatsApp stores messages offline — meaning they only save your messages temporarily on their servers at least until all recipients get the message after which they can safely delete the messages. This is the reason for the daily 2 am backups by the WhatsApp application. The backups are saved to the device's storage so the user has a local copy of their chats. The backup can be further enhanced by integrating it with cloud services like Google Drive, allowing it to upload the backup online. This makes it easier to restore your chats during a new WhatsApp setup, and you likely lose fewer messages.

Considering the above, the question that would pop up usually would be — how does WhatsApp save these messages within itself? well, going through the aforementioned cometchat article, the answer is SQLite database. SQLite is a lightweight database designed to be embedded into applications. So it is no surprise that WhatsApp bundles this together with their native applications like iOS, Android and desktop applications. Well, there is a catch here, which is perhaps how the thought of writing this article was

birthed. WhatsApp also runs on web browsers via https://web.whatsapp.com. Given that it is nearly impossible to bundle software like SQLite into a web browser frontend, it raises the question of how messages are stored there, let alone encrypted.

I decided to conduct this investigation myself, as I could not find any satisfactory article or documentation addressing my burning question. Let's examine some available client-side storage mechanisms in web browsers:

- Local Storage
- Session Storage
- Indexed DB
- Cookies
- Shared Storage

While I am not entirely familiar with session and shared storage, there was a higher probability that my answer was between local storage and indexed DB. Since I wasn't very familiar with IndexedDB and had more experience with local storage, I decided to check local storage first, followed by IndexedDB. Another good thing that got me hyped up about this was that I was hoping to see how WhatsApp structured their data.. hopefully. This would not have been feasible on a native application, so, yeah, the web browser to the rescue.

First, I had to link WhatsApp Web to my device to get it up and running and also allow it some time to synchronize with my device, hoping to get more up-to-date data to inspect. Inspecting the local storage in the developer tools showed nothing useful — just some basic individual data, which makes sense since local storage is just a key-value storage mechanism and wouldn't be used to store large amounts of data. Next, I checked the IndexedDB, which, upon expanding, contained a good number of databases with sufficient data

that answered the question. So, the answer to our question: yeah, IndexedDB.

Encryption

But I didn't stop there; I wanted to see if I could figure out how WhatsApp implements its end-to-end (E2E) encryption. End-to-end encryption means that, from the sender to the recipient, no one else should be able to read the messages, as they are encrypted. This implies that encryption and decryption occur entirely on the front end. So, how does WhatsApp achieve this on its web browser interface?

WhatsApp uses an encryption service known as Signal Protocol for its E2E encryption. I used the typescript implementation of the Signal Protocol — libsignal-protocol-typescript and its demo to learn how it works on the front end.

Initializing

During initialization, Signal first creates an identity for the user, including a registration ID that it stores. On the WhatsApp web indexed DB, most signal-related data as far as I can tell are kept in the `signal-storage` database. The registration key can be found in `signal-storage.signal-meta-store.signal_reg_id` and is a four-digit figure in my case. Next, an identity key pair is generated. The pair includes a private and public key which are used to generate a signed pre-key that looks something as shown below:

```
{
```

```
        "keyId": 477,
        "keyPair": {
            "pubKey": {...},
            "privKey": {...}
        },
        "signature": {...}
    }
```

A pre-key or pre-shared key (PSK) according to Wikipedia;

> *In cryptography, a pre-shared key (PSK) is a shared secret which was previously shared between the two parties using some secure channel before it needs to be used.*

Inspecting the WhatsApp web's `signal-storage.signed-prekey-store` reveals similar data as shown above which can be safely assumed to be the user's pre-key. Pre-keys for your contacts can be found in `signal-storage.prekey-store` which I think are just imported from your main device. A public signed pre-key is also generated from the public key and signed with the private key. A user's public key bundle is sent to WhatsApp servers where it is saved so that other users can have access to it while your private key stays safe on your device and is never shared with anyone or device. The public key bundle consists of the following;

- Identity key: the long-term public key used for identifying a user.
- One-time pre-key: a temporary public key used when creating chat sessions. The key can only be used once. This enhances security by ensuring that each session is unique, thereby enabling forward and backward secrecy — meaning past or future messages remain secure even in the event of a compromise.
- Signed pre-key: the public signed pre-key which was signed using the user's private key.
- Registration ID: the ID generated during the user's registration.

Creating the session

To start messaging, a session would have to be started with the recipient. This is done by creating a signal session using the sender and recipient's key bundles. The recipient's key bundle is fetched using their Signal Protocol address which in WhatsApp's case, as seen in `signal-storage.session-store` is usually the user's phone number. The recipient's key bundle is fetched and then processed by the signal session to create a chat session. The session fails to create if the fetched recipient's identity key differs from a previously seen key for their address. Instances where an identity key mismatch might occur include:

- Re-installation of the WhatsApp application
- Linking or unlinking of a device
- Change of phone number
- Fresh registration

It is safe to assume that a change in identity key serves as WhatsApp's cue to notify the sender of a change in the recipient's security code. The security code is a feature in WhatsApp that helps users verify the authenticity of the received public keys. It is a 60-digit number found in the user's contact information under the encryption section.

Both users involved in a session validate each other's signed pre-keys using their respective identity keys. This helps prevent impersonation. A key exchange occurs at this point, allowing both users to authenticate each other while messaging.

Sending a message

With the session ready, the only thing left to do is to send encrypted messages by creating a cipher using the sender's key bundle and the recipient's address. A cipher is an algorithm or a series of steps used to

encrypt or decrypt data. This cipher encrypts any outgoing message from the sender to the recipient.

Receiving and decrypting the message

To decrypt the encrypted message at the other end, the recipient's key bundle is fetched and used along with the sender's key bundle, which is retrieved using their address, to generate a shared cipher key. This cipher key is used to decrypt the encrypted message, which is then displayed to the recipient in plain text. With this, the cycle is complete.

. . .

Conclusion

We discussed the storage mechanisms used by WhatsApp in its native applications and web version. Additionally, we explored how WhatsApp's end-to-end encryption works, from initialization to decrypting a received message, using the web version's storage as a visual aid to understand the structure of some of the stored keys.

It is important to recall the first paragraph and note that the entire effort put into this article is purely investigative and research-based, and some parts of it may be inaccurate or missing vital information. With that said, any additions or corrections are welcome and can be shared here as comments.

WhatsApp | Encryption | Whatsapp Storage

Written by Emmanuel Akachukwu

23 followers · 19 following

A gem of a software engineer

Responses (1)

Write a response

What are your thoughts?

Tolulope Makinde
Jan 23, 2025

I had no idea that WhatsApp utilizes indexedDB for data storage.

It's great to understand how this works.

One thing I was confused about before was the difference between a signed pre-key and a signature. I found that the former is used to establish a... more

1 reply Reply

More from Emmanuel Akachukwu

In Dev Genius by Emmanuel Akachukwu

Emmanuel Akachukwu

How to Customize Your Shell: Configuration Files and History

Throughout this guide, we've explored various ways to customize the shell environment,...

Mar 24, 2025

SQL CTEs Explained: Master Common Table Expressions for...

Having worked on an ETL related project that had a lot to do with writing SQL queries, man...

Oct 27, 2025 1

In Dev Genius by Emmanuel Akachukwu

How to Customize Your Shell: Search Path Variables

In the first chapter, we explored the basics of shell customization, focusing on variables an...

Mar 22, 2025

In Dev Genius by Emmanuel Akachukwu

How to use multiple git configs on a PC

Photo by Praveen Thirumurugan on Unsplash

Sep 17, 2023 1

See all from Emmanuel Akachukwu

Recommended from Medium

Himanshu Singour

How I Learned System Design

– The honest journey from total confusion to clarity

Aug 7, 2025 231

In Generative AI by Adham Khaled

Stanford Just Killed Prompt Engineering With 8 Words (And I...

ChatGPT keeps giving you the same boring response? This new technique unlocks 2×...

Oct 19, 2025 594

Joe Njenga

I Tried New Claude Code Ollama Workflow (It's Wild & Free)

Claude Code now works with Ollama, which takes the game to the next level for...

Jan 19 19

In Write A Catalyst by Dr. Patricia Schmidt

As a Neuroscientist, I Quit These 5 Morning Habits That Destroy You...

Most people do #1 within 10 minutes of waking (and it sabotages your entire day)

Jan 14 346

In Data Science Coll... by Marina Wyss - Gratitude...

AI Agents: Complete Course

From beginner to intermediate to production.

Dec 6, 2025 97

Will Lockett

The AI Bubble Is About To Burst, But The Next Bubble Is Already...

Techbros are preparing their latest bandwagon.

Sep 14, 2025 916

See more recommendations