Step 1: Extracting the Keys

The Read Path: Memory Scanning

WeChat macOS encrypts local databases using SQLCipher. To bypass this, we use lldb to peek directly into the running process.

Root Access Required

Scanning requires sudo privileges for task_for_pid to attach to the WeChat process (making Docker deployment unviable).

Per-Database Keys

WeChat 4.1.x abandoned the single master key. A scan now yields 24 separate keys (one for each .db file).

Tooling

Skip built-in Mach VM extraction (it fails on 4.1.x). Use wechat-db-decrypt-macos for reliable lldb memory dumps.

Step 2: Automating the UI

The Write Path: macOS Accessibility API

Without a backend API, we write messages by natively automating the macOS GUI using AXUIElement.

The Mechanism

The WeChat-MCP server navigates the macOS UI tree, locates the input field, injects text, and triggers Send.

Permissions

Requires explicit Accessibility permissions granted to your terminal or host app.

Latency vs. Reliability

It takes a few seconds to visually automate the GUI, making it slower than database reads, but it acts outside the sandbox and is incredibly reliable.

Step 3: Decrypt & Serve

Unlocking the Chat History

We pipe the extracted memory keys into chatlog-bot to build a local integration layer.

On-the-Fly Decryption

Uses fsnotify file watching to read the SQLite databases live, outputting clean JSON.

Local Endpoints

Exposes /session, /chatlog, and /contact routes on localhost:5030, plus webhooks for real-time alerts.

Configuration Pitfalls:

1. You must manually prepend db_storage/ to key paths (e.g., db_storage/message/message_0.db) or decryption fails silently.
2. Filter out the __salts__ list from your keys.json to prevent type-mismatch crashes.

The Payoff: AI Orchestration

Full Programmatic Read/Write Pipeline

By bridging the local API and GUI hooks, a closed ecosystem becomes a standard developer platform.

Morning Summaries

Claude hits the local API, pulls the last 200 messages from a hyperactive group chat while you slept, and generates a structured brief.

Tool Calling

Using the WeChat-MCP server, Claude Code can read context and independently draft/send replies natively through the UI.

Automated Triage

Incoming webhooks feed into custom notification pipelines to flag high-priority senders instantly.

Pitfalls: The Upgrade Trap

Beware WeChat Auto-Updates

Tencent updates can break your automation layer. However, encryption keys persist across a process lifetime—missing keys after a restart are just lazy-loaded, not expired.

Extraction Philosophies Compared:

Feature	Memory Scan (chatlog-bot)	Binary Hook (WeFlow)
Method	Regex pattern matching over RAM	Breakpoint injected into KDF entry
Upgrade Survival	High (Pattern-agnostic)	Low (Binary shifts break hooks)
Maintenance	Set it and forget it	Requires developer patches
Best For	Always-on, stable pipelines	Deep, opt-in analytics

Pro Tip: Disable WeChat auto-updates to prevent hook-based tools from spontaneously crashing during a release.