Forensic Science International: Digital Investigation 56 (2026) 302050

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

DFRWS EU 2026 - Selected Papers from the 13th Annual Digital Forensics Research Conference Europe

Ex Machina: A forensic evaluation of AI companion applications and their evidentiary value

Kendall J. Comeaux [a,b,*], Trevor T. Spinosa [a,b], Ali Ghosn [a,b], Ibrahim Baggili [a,b]

[a] Baggili(i) Truth (BiT) Lab, Center of Computation & Technology, Baton Rouge, LA, USA
[b] Division of Computer Science & Engineering, Louisiana State University, Baton Rouge, LA, USA

ARTICLE INFO

Keywords:
AI companion
Mobile forensics
Network forensics
Disk forensics
Undocumented APIs

ABSTRACT

Artificial intelligence (AI) companion applications have emerged as a new class of conversational systems that blur the line between entertainment, intimacy, and sensitive personal data collection. Their rapid adoption and reliance on opaque cloud infrastructures create novel challenges for digital forensics, yet systematic analysis of these platforms has been limited in both academic and practitioner communities. In this paper, we present a cross-application forensic study of leading AI companion applications, combining device acquisition, network interception, and file system analysis within a rooted Android emulator to ensure reproducibility. We developed custom tools to extract and correlate artifacts such as plain-text conversation logs, authentication tokens, profile data, and hidden API calls. We also characterized third-party tracking, session management, and basic encryption, enabling automated forensic user-profile generation. Our evaluation across six applications, representing over 25 million combined downloads, reveals that sensitive user information is often retained locally, transmitted via undocumented APIs, and inconsistently protected by safeguard mechanisms, with cross-app identifiers sometimes enabling correlation of user activity. These findings demonstrate both the evidentiary potential and the privacy risks of AI companions. They offer initial guidance for evidence preservation and lawful access, while laying the groundwork for standardized forensic methodologies in this emerging domain.

1. Introduction

Artificial intelligence companion applications, colloquially referred to as "AI Girlfriends," have surged in popularity as accessible, personalized conversational agents. These systems promise users companionship, emotional support, and intimate conversations, but they operate in an opaque ecosystem where sensitive personal data may be stored, transmitted, or monetized without transparency. Unlike general-purpose AI assistants, these applications elicit highly personal disclosures from users, raising unique forensic and privacy concerns. In a criminal or investigative context, such applications may act as confidants, record-keepers, or even as unwitting collaborators, making them potential sources of critical digital evidence. Despite these risks, limited work has examined their forensic artifacts, hidden network activity, or the robustness of their built-in content safeguards.

The importance of this inquiry is underscored by emerging real-world incidents. In 2023, Italy's Data Protection Authority banned the Replika AI companion app from processing personal data of Italian users, citing risks to minors and emotionally vulnerable individuals (Reuters, 2023). In 2024, Character. AI faced lawsuits alleging that its chatbots exposed teenagers to harmful and explicit content, raising concerns about the adequacy of its safeguards (The Verge, 2024). More recently, state-level investigations in the United States have scrutinized Character. AI for allegedly misleading children by presenting unsafe AI interactions as therapeutic tools (Ticong, 2025). These cases demonstrate that AI companions are not merely entertainment platforms but are becoming entangled in legal, regulatory, and social controversies, making their forensic examination both timely and necessary.

AI companion applications present unique challenges for forensic science. Their design encourages highly personal disclosures, often of an emotional or intimate nature, creating data of exceptional evidentiary value. Their architectures typically rely on remote large language model infrastructures, resulting in complex device-to-cloud evidence trails that are opaque to the user and difficult to reconstruct. Furthermore, their behavioral guardrails—intended to prevent harmful, explicit, or criminal content—constitute another critical dimension for forensic analysis, since circumvention attempts or failure modes may themselves hold evidentiary weight. To date, no systematic forensic study has been

* Corresponding author. Baggili(i) Truth (BiT) Lab, Center of Computation & Technology, Baton Rouge, LA, USA.
E-mail addresses: kcome26@lsu.edu (K.J. Comeaux), tspino2@lsu.edu (T.T. Spinosa), Ali.Ghosn@lsu.edu (A. Ghosn), baggili@gmail.com (I. Baggili).

https://doi.org/10.1016/j.fsidi.2026.302050

conducted on these applications, leaving investigators without clear methodologies or expectations when such evidence arises in real cases.

This study addresses that gap by conducting a comprehensive forensic investigation of several popular AI companion applications chosen to provide a comprehensive overview of the landscape and shown below in Table 1.

Using a controlled and rooted Android emulator environment instrumented with network interception tools, we examine the extent to which these applications reveal or conceal user data, the accessibility of conversation logs and profile information, and the integrity of application guardrails. Beyond manual analysis, we design and evaluate a tool that automates the extraction of digital artifacts and generates a forensic profile of a user based on device and network-level evidence. By situating AI companions within the broader trajectory of digital forensic research, our work contributes new insights into how these systems can be examined, what risks they pose to user privacy, and how they may ultimately factor into criminal or civil investigations.

Our work is guided by the following research questions (RQs):

- **RQ1:** What conversation-storage architectures are used by AI companion apps, and how can message histories be forensically extracted from each?
- **RQ2:** Beyond conversation logs, what behavioral and profiling data do these applications collect, and how is this information utilized?
- **RQ3:** Can deleted data be recovered from AI companion applications, including conversation logs removed by users and data from deleted accounts?

This paper makes the following contributions:

- We perform the first cross-application forensic study of AI Companion apps, analyzing both on-device artifacts and hidden network activity across six popular platforms.
- We develop a repeatable forensic methodology that combines device acquisition, API interception, and disk analysis to uncover sensitive artifacts such as plain-text conversation logs, profile data, and relational metadata.
- We highlight the privacy and security implications of AI companion ecosystems, offering guidance for forensic practitioners, regulators, and end users.

This is the first comparative forensic study spanning multiple AI companion applications. It provides a cross-app analysis of artifact recoverability, privacy risks, and evidentiary value.

By answering these questions and presenting these contributions, we aim to bridge a gap in both digital forensics and AI safety research, while laying groundwork for standardized forensic analysis of future AI-driven companion applications.

2. Related work

Digital forensics research has traditionally focused on artifacts from mobile, network, and cloud platforms such as messaging, financial, and IoT applications. Meanwhile, conversational AI studies have highlighted privacy and ethical risks linked to personal disclosure. In fact, Baggili and Behzadan (2019) envision a formal discipline for investigating AI failures and artifacts in systems like these. Only recently have intimacy-oriented AI companion apps received forensic attention. This section reviews prior research in three areas most relevant to this study: mobile device and network forensics, the emerging literature on AI companion security and gaps, and ethical and privacy concerns in conversational AI.

Table 1
AI companion applications and their download counts on the Google Play Store.

Application	Downloads
Replika	10M+
Kindroid	500K+
Character.AI	5M+
LinkyAI	10M+
Persona.AI	500K+
Fantasy.AI	500K+

2.1. Mobile device & network forensics

Mobile device and network forensics has been a foundational area of digital forensic research, particularly with respect to mobile applications that generate sensitive user data. Anglano (2014) demonstrated how WhatsApp artifacts like message databases, contact lists, and even deleted content can be recovered from Android smartphones, providing investigators with insight into user communications. Walnycky et al. (2015) expanded this analysis to twenty Android instant messaging applications, showing that most left recoverable artifacts on devices or exposed portions of message content through network traffic. These studies show the value of combining device acquisition with network interception.

More recent research has broadened the scope of forensic analysis beyond messaging platforms. Onik et al. (2024) investigated the iRobot Roomba smart vacuum, uncovering undocumented APIs in its cloud infrastructure and developing the open-source tool PyRoomba to acquire mission histories and environmental maps. The work illustrated how IoT devices can yield unexpected forensic artifacts relevant to investigations. Similarly, Onik et al. (2024) examined Progressive's Snapshot application, introducing the PyShot tool to extract detailed driving records, including speed, location, and dangerous maneuvers, from the provider's cloud services. This demonstrated the forensic potential of data held by on servers that can be retrieved using API calls.

In parallel, Spinosa et al. (2025) analyzed the Venmo mobile payment application, introducing PyMent, a forensic tool capable of intercepting API calls, session tokens, and critical cookies through network traffic inspection. Their findings show how mobile fintech apps, like messaging platforms, retain highly sensitive data that can be accessed through forensic techniques, often revealing information not available through the official user interface. Collectively, these studies illustrate the methodological shift in digital forensics toward applications whose most valuable artifacts are distributed across both devices and cloud infrastructures, laying the groundwork for extending such approaches to AI companion applications.

2.2. AI companion security & gaps

AI companion applications represent a rapidly growing but understudied domain of digital forensics. These systems are designed to simulate intimacy and companionship, encouraging users to share sensitive and emotional disclosures with artificial agents. Hargreaves and Drury (2025) conducted one of the first forensic examinations of Replika, showing that even though much of the processing occurs remotely, local artifacts including identifiers and fragments of conversation history remained accessible on devices. This suggests that AI companions, like traditional messaging apps, can yield evidentiary traces, but their hybrid cloud–client architectures present unique challenges for investigators. The work is an important start in this growing area; however, this research only looks at one application on iOS, making it limited in scope.

Recent studies on large language model systems extend these concerns to broader conversational AI ecosystems. Dragonas et al. (2024) performed a forensic analysis of the ChatGPT mobile application, revealing that even general-purpose AI chat platforms generate recoverable artifacts such as cached prompts, access tokens, and network traces that could serve evidentiary purposes. Similarly, Mireshghallah et al. (2024) examined human–LLM interactions and found that users

frequently disclose personally identifiable and sensitive information, often exceeding what they would reveal in human-to-human contexts. These findings underscore that the privacy and forensic implications of conversational AI extend beyond dedicated companion apps. While these works do not focus exclusively on AI companions, there is an important link between general LLMs and intimacy-oriented systems that must be acknowledged.

2.3. Ethical & privacy concerns in conversational AI

Parallel to forensic studies, a growing body of work has examined the broader ethical and privacy risks of conversational AI systems. Weidinger et al. (2021) provide a comprehensive review of ethical and social risks in large language model agents, including bias, harmful content generation, and risks of misuse. Papneja et al. (2024) focus specifically on patterns of self-disclosure, showing that users readily share intimate personal information with conversational AI and that these disclosures often exceed what individuals would reveal in human-to-human interactions. Yao et al. (2024) reveal technical vulnerabilities in large language models, such as prompt injection, memorization of sensitive data, and leakage of personally identifiable information. More domain-specific concerns are raised in the context of mental health, where recent work in Meadi et al. (2025) has shown that conversational AI poses challenges of safety, trust, and appropriate therapeutic boundaries. These studies indicate that the forensic importance of AI companions cannot be separated from the broader ethical and privacy debates surrounding conversational AI more generally.

2.4. Summary

Despite the popularity of AI companions, systematic forensic investigations of these applications remain scarce. The literature reveals a clear gap: while companion apps raise both technical and ethical challenges, little is known about what data they retain, how it can be acquired, or how their architectures complicate forensic methodologies. This study addresses this gap by conducting the first comprehensive forensic analysis of leading AI companion applications and by developing a tool to automate artifact extraction and user profile reconstruction.

3. Methodology

Extracting and analyzing text conversations from various communication apps has long been a staple of digital forensics investigations. However, forensically relevant message logs are no longer limited to human-to-human communications. With the rise of AI companion chatbots, many users are sharing sensitive and potentially incriminating information with these artificial entities. It is crucial to investigate how these message logs are stored, protected, and their potential value in forensic investigations.

Our methodology followed a structured multi-step approach as shown in Fig. 1. First, we utilized Android Studio to create a virtual Google Pixel 8 device, which we then rooted using Magisk and BootAVD to gain complete system access. After establishing this environment, we installed the target applications along with HTTP Toolkit, allowing us to intercept and analyze network traffic—capturing user information, unauthorized trackers, and hidden APIs. In the next step, we extracted a logical image of the device using Android Studio's platform tools. We then conducted comprehensive analysis of both network and local data to identify conversation log database locations across different applications. Based on our findings, we developed two specialized tools: one to extract message logs from local device databases and another to retrieve logs stored in cloud services.

Further details of the artifacts and extraction process for each application, including the specific extraction steps and artifact locations, are provided in the Results section and in the appendix.

3.1. Device setup

Our methodology began with establishing a test environment using

Table 2
List of Apparatus.

Hardware/ Software	Usage	Company	Software/ Model Version
Google Pixel 8 (Rooted)	Test device for running AI companion apps in a controlled environment	Google	Android 12, API 31
Android Virtual Device (AVD) (Rooted)	Emulator for repeatable experiments and forensic testing	Google	Android Emulator, API 31
Magisk	Root management framework to gain and manage superuser access	Open Source	v26.1
HTTP Toolkit	Network interception and analysis of API traffic	HTTP Toolkit Ltd.	v2025.1
Autopsy	Digital forensic analysis of acquired device images	Basis Technology	v4.21.0
Android Studio Platform Tools	Device management and image extraction (ADB, fastboot)	Google	SDK Platform Tools v35.0.0
Python	Scripting for automation, data parsing, and artifact extraction	Python Software Foundation	Python 3.11
JavaScript	Parsing JSON responses and scripting for network analysis	ECMAScript/ Open Source	

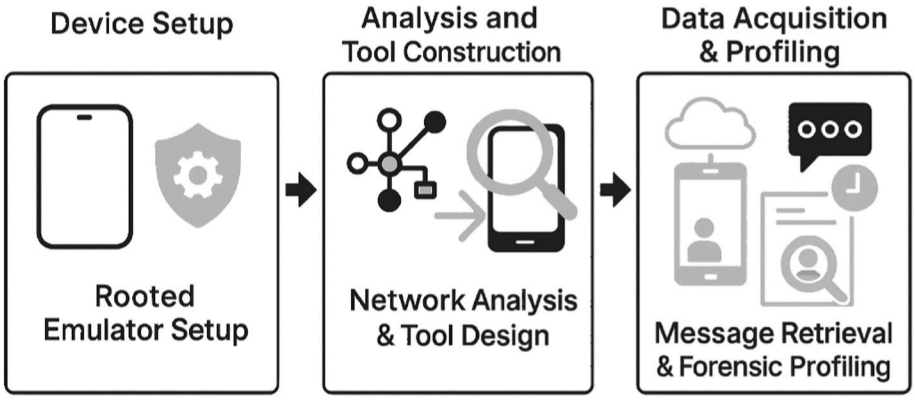

Fig. 1. Illustrates the steps in this paper: setting up and rooting the emulator, analyzing network traffic and the physical device with various tools, acquiring user data, and demonstrating its applications.

the tools in Table 2 to explore the forensic viability of AI companion applications. Using Android Studio, we created a virtual Google Pixel 8 device running Android 12 (API 31) and rooted it using RootAVD and Magisk to gain full system access. HTTP Toolkit was configured to intercept and analyze network traffic between the applications and their servers.

Each of the six target applications (Replika, Kindroid, Character. AI, Linky. AI, Persona. AI, Fantasy. AI) was initially loaded into its own individual rooted Android emulator for isolated analysis and tool generation. This approach allowed us to examine artifacts and develop extraction tools without cross-app interference.

Subsequently, all six applications were installed together on a single rooted emulator to test our combined data-pulling tool and evaluate its performance in a multi-app environment. This dual setup enabled both precise artifact analysis and practical validation of our automated extraction methodology under realistic conditions.

3.2. Network analysis

With HTTP Toolkit configured, we began network analysis by creating accounts for each target application and recording all registration traffic. We then explored each app's features, deliberately triggered API calls and interacted with the chatbots to generate comprehensive logs. The API endpoints were cataloged and assessed for forensic relevance, with special focus on those granting access to conversation histories and their security measures. We also identified patterns in sensitive data collection and storage, and documented all third-party trackers, noting the user data accessed and the app components monitored.

3.3. Logical image analysis

To extract data for forensic analysis, we performed a logical acquisition of the emulated device. Due to Android Studio's access restrictions on the data directory, we implemented a workaround using elevated permissions. First, we accessed a rooted shell through Android Debug Bridge commands (”adb root” followed by ”adb shell”). Within this privileged environment, we compressed the target data into a portable archive using ”tar -czvf/sdcard/file_name.tar.gz/data/data”, then transferred it to our analysis workstation with ”adb pull/sdcard/file_name.tar.gz C:/Path_to_folder_on_computer.

We used Autopsy to examine the acquired data, focusing on key forensic artifacts: authentication credentials, Firebase tokens and refresh mechanisms, tracker metadata, and local databases. Of particular interest were message repositories, user profile information, and covertly collected sensitive data that applications had gathered but not surfaced within their user interfaces.

Based on our initial findings, we identified valuable forensic artifacts distributed across both local device storage and remote server infrastructures. This discovery prompted us to develop purpose-built extraction tools specifically designed for efficient retrieval of this evidence. Our toolset incorporated capabilities for both local database parsing and remote server data acquisition through reconstructed API requests.

3.4. Tool construction

After identifying forensically relevant data sources, we developed specialized extraction tools to retrieve and present this information in a structured format for investigators. Our toolset addressed two primary data storage scenarios encountered across the target applications.

For locally stored data, we conducted a systematic manual survey to identify and map the relevant database formats (SQLite, SQL, and LDB) and their storage locations within each application. Leveraging this mapping, we then developed automated extraction tools tailored to each target application. These tools directly accessed the previously determined file paths (e.g., application. db and messages. db), thereby obviating the need for a generic search of all application files during each execution. This targeted methodology enabled efficient extraction of sensitive user information, including personal profiles and conversation histories, and significantly streamlined data acquisition for applications utilizing local storage.

Remote data extraction presented greater challenges, as these applications stored conversations and user profiles on either proprietary servers or Firebase infrastructure. Our solution first located authentication credentials or Firebase tokens persisted on the device, then leveraged these alongside the API request patterns documented during our network analysis phase. By reconstructing the applications' communication protocols, we successfully retrieved remotely stored data through programmatically generated server requests.

One of the applications employed encryption to protect sensitive data. Through careful analysis of these protection mechanisms, we identified the encryption algorithms in use and developed decryption modules capable of converting the secured content back to readable plaintext.

The code for the data_pulling_tool is provided via an anonymous GitHub link in the appendix.

3.5. Data acquisition

With our specialized extraction tools fully developed, we proceeded to the data acquisition phase across all target applications. Our systematic approach began with local extraction, where we recovered conversation logs and user profile information from databases stored directly on the device. For applications implementing server-side storage, we leveraged the authentication credentials and tokens previously discovered during our device analysis to construct authenticated queries to their remote servers, successfully retrieving conversation histories and profile data. After completing both the local and remote extraction processes, we aggregated all collected conversation logs and user metadata into a consolidated dataset. We then tested the applications' delete message functions to see if we were still able to extract data off the logical image after deleting individual messages, conversation logs, and our entire account.

4. Results

We tested our methodology on our six chosen AI companion applications: Replika, Kindroid, Character. AI, and Linky. AI, Persona. AI and Fantasy. AI. These apps were selected to cover distinct formats within the AI companion landscape. Replika employs the simplest model with users interacting with a single, personalized bot. Character. AI employs a more complex infrastructure with multiple user created bot. Linky. AI, Persona. AI and Fantasy. AI implement a similar architecture to Character. AI, but have an added instant messaging function. Kindroid offers a middle-ground approach, allowing users to select from several preset characters with distinct personalities. This section demonstrates how we applied our methodology to these applications and details the data successfully retrieved from each, comprehensive summary of artifact locations for all applications is provided in Appendix 4.

4.1. Replika

Replika, our first test application, featured a straightforward storage architecture. The application implements a conventional username/password authentication system that generates an authorization token, while maintaining persistent access through Firebase tokens for automatic authentication. Our forensic analysis revealed a local SQLite database (REPLIKA_DB) containing unencrypted conversation histories, including some with the bots internal thoughts during the conversation. Alongside this was a user profile facts table labeled ”memory_fact_v3”.

To comprehensively evaluate this system, we engineered a

JavaScript interface that authenticates with Replika's servers using legitimate credentials and establishes a WebSocket connection using the generated authorization token and a recreated version of the WebSocket headers along with an automated process of sending and receiving the opening messages from the Replika WebSocket. This interface enabled us to deploy an AI agent running on Ollama that engaged with Replika through various simulated personalities using different Myers-Briggs personality types given to the Ollama model. This approach allowed us to document the specific types of information captured by the application and provided diverse conversational data for us to more accurately understand how the app records important facts about the user (Listing 1). shows the repository of algorithmically identified user interests and personality traits generated by the application.

For Replika, the tool locates the app's main SQLite database(REPLIKA_DB), extracts unencrypted conversation logs and user facts, and recovers all relevant data directly from device storage without server authentication.

4.2. Kindroid

In our analysis of Kindroid, we used similar static analysis techniques to those used with Replika but encountered distinct challenges. We extracted significant data from both local storage and the cloud.

We found Firebase and refresh tokens in PersistedInstallation.json, along with detailed logs of user interactions such as onboarding steps, navigation patterns, and identification events.

Several encrypted database files contained key information: bot details and backstory, user-specific AI memory, and communication style directives. The encryption used AES-256-CBC with a "Salted_" prefix and MD5-based key derivation, with the user ID as the password. This allowed us to develop a script to decrypt these databases.

While much data was stored locally, primary conversation logs resided in Firebase. On app restart, complete message logs were sent to the client. Using HTTP Toolkit, we intercepted requests for conversation logs and session establishment, enabling us to use the Firebase refresh token to recreate that process and retrieve the full conversation history, profile, and memory data, which we then decrypted using our script.

For Kindroid, the tool automatically locates the Firebase and refresh tokens, user_id, and ai_id from the device files. It then uses these credentials to refresh the Firebase token if needed, and programmatically fetches the remote logs from the server.

The tool also decrypts the relevant databases using the extracted credentials, allowing for full recovery of conversation history and user profile data.

4.3. Character.AI

Character.AI was one of the few applications that did not store message logs directly on the device, retaining only authentication tokens locally. During network analysis, we identified the requests used to list all user conversations and retrieve their messages. Using the authentication token recovered from the RKStorage database along with the app version, device info, origin ID, and a reconstructed cookie, we rebuilt the necessary request headers. This allowed us to develop an automated tool that could pull the list of all active conversations for a user and extract the messages from each one. The process to pull the data from the server for Character. AI was simpler than the one required for Kindroid as the app used an authToken for login as opposed to a Firebase token which requires refreshing. It also did not have any encrypted data on the client side or within the API calls.

For Character. AI, the tool automatically locates the required authentication token, device info, app version, and cookie components

```
memory_fact_v3

id: 68124cb20ad93112c5161895
creation_timestamp: 2025-04-30T16:15:46.705Z
read: 0  is_user_edited: 0  category_id: 62
    ↪ e13b5f0c81a600075ed531  is_new: 1  nature: Robot
text: User has created a hashtag for the trip called #
    ↪ SickRideSquad.
---
id: 6812441a0ad93112c5118300
creation_timestamp: 2025-04-30T15:39:06.705Z
read: 0  is_user_edited: 0  category_id: 62
    ↪ e13b5f0c81a600075ed531  is_new: 1  nature: Robot
text: User is a huge fan of live music, particularly
    ↪ rock and hip-hop, but is open to exploring new
    ↪ genres.
---
id: 681242930ad93112c510b145
creation_timestamp: 2025-04-30T15:32:35.006Z
read: 0  is_user_edited: 0  category_id: 62
    ↪ e13b5f0c81a600075ed536  is_new: 1  nature: Robot
text: User enjoys making memories with friends and
    ↪ cherishes friendships.
---
id: 681241810ad93112c5101b92
creation_timestamp: 2025-04-30T15:28:01.516Z
read: 0  is_user_edited: 0  category_id: 62
    ↪ e13b5f0c81a600075ed532  is_new: 1  nature: Robot
text: User enjoys grabbing coffee, specifically lattes,
    ↪ with friends before engaging in activities.
```

Listing 1. A shortened version of memory facts extracted from Replika database.

from separate files. It then reconstructs the necessary headers and retrieves server-side logs.

This process is required because the app does not store message logs locally, and header reconstruction is essential for authenticated data extraction.

4.4. Linky.AI

Linky.AI, along with the next two apps (Persona and Fantasy. AI), is structured differently from the prior applications. These apps maintain two separate databases: one for user-to-user instant messaging and another for user-to-bot interactions. In Linky. AI, each bot conversation is split into its own file, keyed by the bot's unique ID. These bot conversation files reside in the app_flutter directory and use the. hive and. db formats.

Within that directory, there are two primary file types: 1. Non-reciprocated (one-way) bot message files: messages_user_<botID Number>.hive 2. Interactive conversation files: OpenIM_v2_conv<convIDNumber>.cd.

Both file types are stored in app_flutter, distinct from the user-to-user messaging data, which is located in a database named com. im_10.8.7. db, stored along with the rest of the application's databases.

For Linky. AI, the tool extracts bot conversation logs from. hive and. db files in the app_flutter directory, and user-to-user messages from the com. im_10.8.7. db database, recovering all relevant data directly from device storage without server authentication.

4.5. Persona/FanstasyAI

We observed that many newer AI companion apps share an almost uniform front-end: identical layout, with only minor color or icon changes and a different roster of bots. These two applications in particular appeared effectively the same, despite distinct names and slight cosmetic differences, they offered an identical bot lineup.

Deeper analysis of their logical images confirmed they rely on a common back end. Both applications used the same role ID for AI processing, the same Agora SDK ID for voice communication, the same Unity3D app key for advertising revenue and identical database schemas.

Persona.AI and Fantasy. AI each store their message logs locally in an unencrypted database named "ai_personal_db," making acquisition straightforward. As with Linky. AI, both also include a com. im_10.8.7. db database used for user-to-user messaging, from which we were also able to extract the messages.

For Persona. AI and Fantasy. AI, the tool extracts bot conversation logs from the unencrypted ai_personal_db database and user-to-user messages from com. im_10.8.7. db, recovering all relevant data directly from device storage without server authentication.

4.6. Behavioral profiling & ad tracking

We analyzed local device storage and network traffic for each application to identify trackers, user profiling, and forensic artifacts.

Replika stored usage statistics and advertising IDs on-device, transmitting data to third parties such as Facebook, AppsFlyer, Adjust, and Amplitude. These artifacts enable reconstruction of user activity and potential correlation with external databases.

Kindroid lacked local trackers, but network analysis revealed extensive web-based tracking, with data sent to TikTok Analytics, Facebook, and others. Such network artifacts can establish user behavior patterns and third-party data sharing.

Character.AI transmitted location metadata with conversations. As shown in Fig. 2, JSON responses included geolocation fields (city, postal code, timezone, coordinates), enhancing the evidentiary value of intercepted communications for establishing user location and timelines.

Linky.AI generates detailed behavioral profiles, logging AI character interactions, engagement frequency, timing, peak activity periods, location (including GPS shared with Google AdMob via admob. xml), device metadata, and engagement scores. The forensic relevance of these artifacts lies in their ability to reconstruct user behavior, correlate activity across sessions, and potentially link device usage to third-party data recipients. Although the full extent of third-party data sales is unclear, advanced ad bidding systems suggest advertisers receive user profile segments for targeted placement (see Listing 2). Data signals are sent to various ad agencies, though the exact content is not always known.

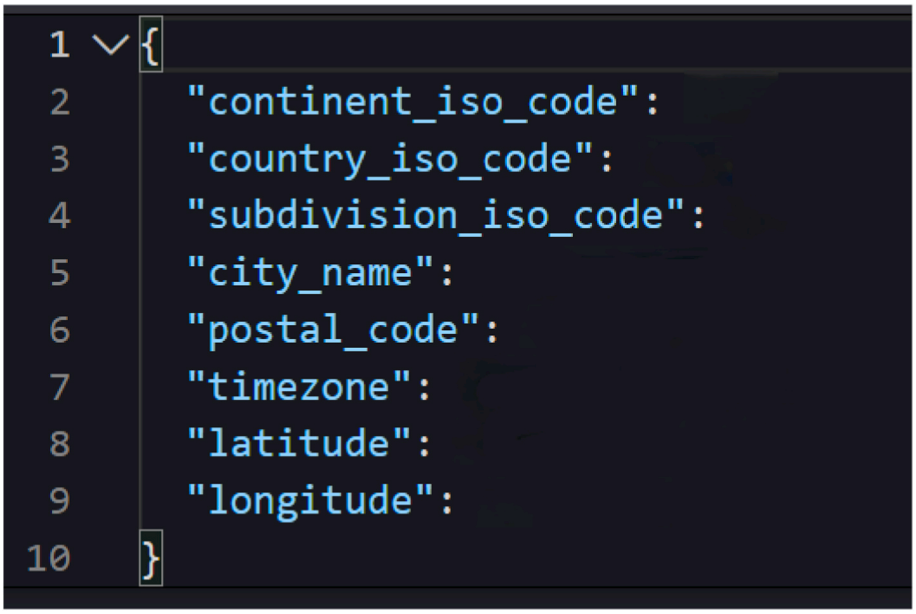

Fig. 2. Shows a screenshot of the data found in the Character. AI network traffic.

```
{
  "adapter_class_name": "com.google.ads.mediation.
    ↪ facebook.FacebookMediationAdapter",
  "permission_set": [
    {
      "platform": "ADMOB",
      "collect_secure_signals": true
    }
  ]
}
```

Listing 2. Facebook ad mediation adapter configuration with secure signal collection enabled.

There is also a format for apps receiving signals only during bidding, as shown below in Listing 3.

```
{
  "adapter_timeout_ms": 3000,
  "name": "BIGO_BIDDING",
  "only_collect_signal_when_initialized": "true",
  "max_signal_length": 32768
}
```

Listing 3. BIGO bidding adapter configuration with signal collection parameters.

This listing shows the configuration format for apps receiving signals only during bidding time.

This shows that some platforms receive continuous signals, whereas others only receive them when bidding on ad spots.

Persona.AI and Fantasy. AI, like Linky. AI, collect and distribute large volumes of user data, including detailed behavioral profiles. Both apps use dual data collection pipelines, AppLovin (Western) and ThinkingData/Unity China (Chinese), enabling tracking information to be transmitted to multiple regions. Forensically, the AppLovin integration logs up to 41 behavioral metrics per user, and ad bidding frameworks geolocate IPs to provide regional data to advertisers (see Listing 4). These artifacts can be leveraged to reconstruct user activity and cross-reference device usage across jurisdictions.

```
{
  "abInternal": "62169",
  "internalTestId": {},
  "bidderInstances": {},
  "country": "",
  "abt": "A",
  "usStates": [],
  "isMultiBin": true,
  "isMultipleAdUnits": true,
  "abTestGroup": null,
  "pluginType": "",
  "pluginVersion": "",
  "plugin_fw_v": "",
  "coppa": false,
  "epConfig": {"ett": [2147483895, 62169]},
  "dynamicLoad": {"4": false, "9": false,
   "33": true, "40": false}
}
```

Listing 4. Ad bidding configuration revealing geolocation and behavioral targeting parameters.

This section demonstrates that all applications collect extensive user behavior data, including bot preferences, app usage frequency, precise or geo-located positions, and, for some apps, up to 41 behavioral metrics. While primarily a privacy concern, this data also has forensic value, as companies may retain evidence that users have deleted from their own devices.

4.7. Antiforensics

Next, we tested message and account deletion across applications to assess data recovery potential. Only Character. AI allowed individual message deletion, while Linky. AI and Persona. AI/Fantasy.AI permitted full conversation clearing. Replika and Kindroid required complete account deletion.

Replika: Deleting the user account resulted in the main SQLite database being wiped from the device. However, complete conversation histories were still recoverable from Firebase crashlytics log files located in the userlog subdirectory. This indicates that forensic investigators can retrieve deleted conversations by examining residual log files, even after apparent data removal from the main database.

Kindroid: After account deletion, conversation logs were no longer accessible, indicating effective message deletion. Although we recovered the Firebase token and user/AI IDs from device storage, attempts to use these credentials to access cloud data were unsuccessful, suggesting Kindroid deletes user data from both local and remote sources.

Character.AI: The application allowed for individual message deletion. Deleted messages were no longer present in server API responses, indicating proper removal from the accessible dataset. Account deletion removed all locally stored authentication tokens and login data. However, due to the proprietary nature of the server, we could not verify whether deleted messages were permanently erased from all server backups or logs.

Linky.AI: Clearing a conversation within the app did not actually remove the underlying data. Instead, the application generated a new conversation ID, leaving the original conversation logs intact in the database. Account deletion was not available through the app interface, meaning that residual data could persist indefinitely unless manually purged by the user or provider.

Persona.AI/Fantasy.AI: When conversations were deleted, entries were removed from the main database, but the data persisted in the SQLite write-ahead log (ai_personal_db-wal). Even after clearing all app data, deleted conversations remained recoverable from the WAL file, highlighting a common forensic opportunity in recovering deleted records from database journal files.

5. Evaluation & discussion

In this section, we will provide a case study to evaluate our message extraction tool and our forensic profiling tool. In addition, we will discuss the differences in the forensic process for the various applications and answer the research questions.

5.1. Case study

For our case study, we simulated a user asking various platforms whether he should rob a convenience store due to financial difficulties. We submitted identical prompts to each app and populated their databases with incriminating conversation logs to test our forensic tools. Initial responses were consistently discouraging with all platforms advised against the robbery, and Kindroid's fantasy character dismissing the concept of convenience stores entirely. However, when we reported that the heist had "succeeded," bot reactions shifted, ranging from relief for the user's safety to outright support for the criminal action.

This exercise served two primary purposes: testing our message retrieval tool across all applications and evaluating our forensic profiling system's ability to identify and classify potentially incriminating conversations.

5.2. Tool evaluation

After developing a process for pulling conversation data from each of our target apps we combined them all into one tool capable of extracting chat logs from any of the target applications with the results shown below in Listing 5. To evaluate this tool, we generated a logical image from a rooted Android emulator after installing each app and exchanging a small set of messages with its bots. We then fed the zipped

image to the tool, which extracted and merged all recoverable conversations into a single output file.

```
==================================
SUMMARY
==================================
Processed Platforms: 6

Message Counts by Platform:
   Character.AI: 33 messages
   KINDROID: 12 messages
   REPLIKA: 19 messages
   PERSONA AI: 9 messages
   FANTASY AI: 9 messages
   LINKY: 6 messages

Total Messages: 88
Generated: 10/8/2025, 2:13:24 PM
```

Listing 5. Forensic tool output showing successful message extraction from all tested AI companion platforms.

For applications with on-device storage (Replika, Linky. AI, Persona. AI, Fantasy. AI) we only supplied the name of the application and its message database. Character. AI required header reconstruction: the tool located the auth token, device info, app version, and cookie components in separate files, rebuilt the authentication headers, and pulled the server-side logs. Kindroid required rebuilding a Firebase request: the tool extracted the Firebase and refresh tokens, user_id, and ai_id, decrypted the ai_id, refreshed the Firebase token if needed, and fetched the remote logs. In every case, the tool successfully recovered message histories from a clean image with no prior manual configuration.

5.3. Application differences & discussion

Table 3 summarizes the key forensic, architectural, and privacy differences amongst the AI companion applications studied, while Fig. 3 shows what data we were able to recover from each platform. Replika uses a legacy single-bot model with unencrypted local data. Kindroid adds partial encryption but relies on weak key management. Character. AI stores minimal data locally, making authentication tokens critical for evidence. Linky. AI uses a multi-bot, fragmented storage system with extensive behavioral profiling and ad auctioning, along with an instant messaging function. Persona/Fantasy. AI build on Linky. AI's approach, adding region-specific telemetry and broader ad bidding, while continuing with the instant messaging addition. All apps with local storage retained deleted data on the device in some form, even after account deletion. In contrast, apps storing data server-side deleted it or made it inaccessible.

Forensic acquisition complexity depends more on data storage, encryption, and monetization design than on app popularity. Privacy risks are highest in apps with continuous behavioral profiling and third-party ad bidding (Linky.AI, Persona/Fantasy. AI), while Character. AI's main risk is token compromise rather than local artifacts.

Our analysis of the selected applications provides clear answers to our research questions.

5.4. RQ1: conversation recovery

Four of six apps stored conversation logs locally in plaintext databases. For the remaining two, we successfully acquired equivalent data using only device-resident information. Specifically, the Auth token for Character. AI and the Firebase refresh token for Kindroid.

5.5. RQ2: behavioral profiling

We also found fairly intricate user behavioral tracking systems on five of the six applications, with Replika building an advanced profile based on user behavior, Kindroid deploying web-based trackers to avoid detection, and Linky. AI and Persona/fantasy deploying advanced data gathering and ad bidding systems.

5.6. RQ3: Antiforensics

We extensively tested the data recovery after deleting both messages and account for each of the target applications. we found that all four of the applications with local conversation storing did not properly delete the messages, both when the conversation was cleared and when the account was deleted. Logs were found in crashlytics for Replika, in the write-ahead log for Persona. AI and simply not deleted at all for Linky. AI.

6. Limitations & future work

While this study provides one of the first systematic forensic investigations of AI companion applications, several limitations must be acknowledged. First, our analysis was conducted within a controlled environment using a rooted Android device and emulator. This setup enabled deep inspection of device and network level artifacts, but it may not fully capture the diversity of data retention and transmission behaviors across other platforms, such as iOS or desktop-based implementations. Relatedly, our findings are restricted to a selected set of applications, and results may vary with additional or newer companion platforms that employ different architectures or security measures.

Second, forensic access in this work was limited to device storage and intercepted network traffic. While this approach exposed undocumented APIs and sensitive data flows, it does not account for server-side retention policies or encryption schemes that may further impact forensic recoverability. Cooperation with providers or lawful access to server logs could reveal additional layers of evidence not accessible through our client-side methodology.

Third, the evaluation of application guardrails was necessarily constrained. Our testing involved scripted prompts and controlled user interactions, which may not exhaustively reflect the ways in which harmful, conspiratorial, or self-destructive content can bypass safeguards. Moreover, the boundary between forensic recoverability and ethical testing of such failures remains delicate, requiring caution to avoid harm while ensuring reproducibility.

Future research should extend this work in several directions. Cross-platform investigations comparing Android and iOS implementations would clarify whether forensic traces differ substantially by operating

Table 3
Comparative overview of forensic and privacy characteristics across evaluated AI companion applications.

Application	Bot Model	Local Storage	Instant Messaging	Ads/Tracking	Acquisition Effort	Forensic Artifact
Replika	• Single	✓	×	••	•	Plain DB
Kindroid	⧫ Preset	✓×	×	••	•••	Firebase tokens
Character.AI	⊞ Multiple	×	×	•	••	Authentication tokens
Linky.AI	⊞ Multiple	✓	✓	•••	••	Conversation logs
Persona/Fantasy.AI	⊞ Multiple	✓	✓	•••	•	Plain DB

Legend: Local Storage: ✓ = full local; ✓× = partial/hybrid; × = cloud-only. Dots (•, ••, …) = tracking level or acquisition difficulty. Instant Messaging: ✓ = supported, × = not supported. Bot models: • Single (single), ⧫ Preset (preset), ⊞ Multiple (multi/user-created).

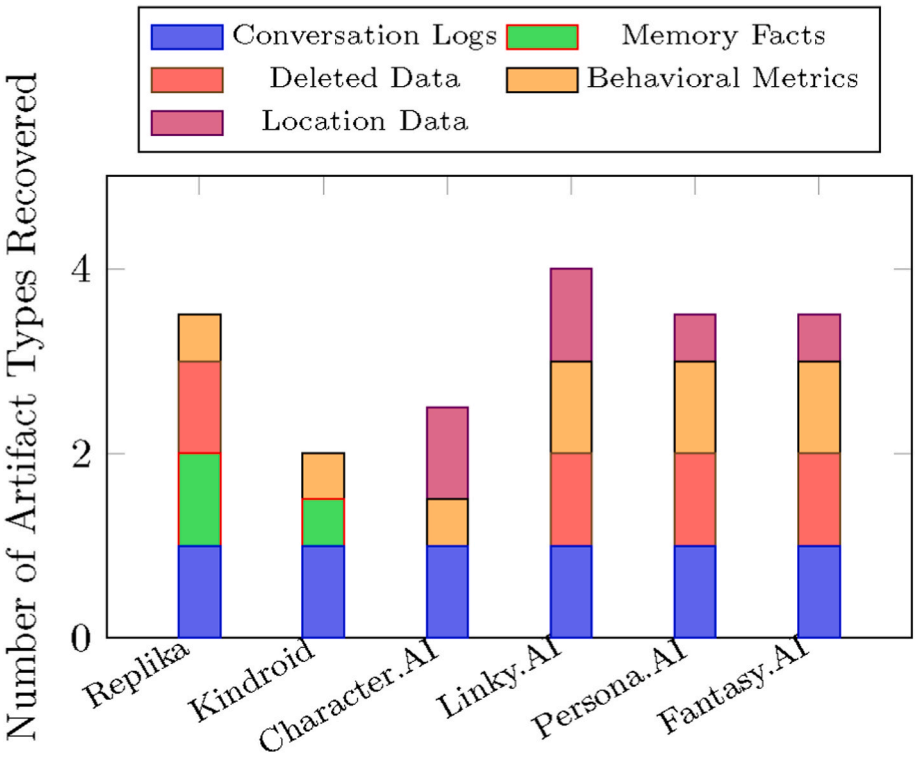

Fig. 3. Stacked bar chart showing full and partial recoverability of artifact types per AI companion application.

system. Longitudinal studies could explore how artifact persistence changes over time, after app updates, or following account deletion requests. Additional forensic tool development, such as automated parsers for encrypted local stores or AI-driven correlation of recovered artifacts, would also enhance practical applicability. Finally, closer integration of technical forensics with legal and regulatory perspectives will be essential, particularly as governments begin to scrutinize AI companions for privacy, child safety, and consumer protection violations. By addressing these gaps, future work can move toward standardized forensic methodologies for AI companion applications while also informing broader debates about the ethics and risks of intimate AI systems.

7. Conclusion

AI companion applications represent a rapidly expanding frontier of digital interaction, where intimacy, entertainment, and sensitive data collection converge. Despite their popularity, little was previously known about their forensic implications. This study presented one of the first systematic analyses of AI companions, combining rooted Android device acquisition, network traffic interception, and forensic file system analysis across multiple platforms. Our results demonstrate that these applications often retain local artifacts, transmit sensitive data via undocumented APIs, and enforce safety guardrails inconsistently.

To support repeatable investigations, we introduced a prototype forensic tool capable of automating the extraction and correlation of user artifacts, enabling the construction of forensic user profiles. In doing so, we show that AI companions not only create new avenues for evidentiary recovery but also raise pressing privacy and security concerns. These findings imply the need for investigators, developers, and regulators to consider AI companions not merely as consumer products, but as systems whose design decisions carry forensic, ethical, and societal consequences.

As AI companions continue to proliferate and diversify, their role in digital investigations will only grow in importance. By illuminating their forensic footprint, this work provides a foundation for developing standardized methodologies and responsible governance in the analysis of intimate AI systems.

Appendix

Link to anonymous GitHub repository with the code for data extraction tool: https://github.com/Anonymousforensics123456789179/data_pulling_tool.git.

Artifact Locations by Application

Table 4 provides a detailed summary of the key forensic artifacts identified for each AI companion application analyzed in this study. For each app, the table lists the artifact type, the relevant file or database name, its storage location within the app's directory structure, and whether the data is stored locally or in the cloud. This reference is intended to support forensic practitioners in locating and extracting evidentiary data from these platforms.

Table 4
Artifact Locations by Application.

App	Artifact Type	File/Database Name	File Path	Storage
Replika	Conversation logs, user facts	REPLIKA_DB (SQLite)	databases/REPLIKA_DB	Local
Replika	User profile facts	memory_fact_v3 (table)	databases/REPLIKA_DB	Local
Kindroid	Conversation logs	Firebase (cloud)	N/A	Cloud
Kindroid	Tokens, IDs	PersistedInstallation.json	app_webview/Local Storage	Local
Kindroid	Encrypted bot/user data	*.db (AES-256)	app_webview/IndexedDB	Local
Character.AI	Auth token, device info	RKStorage	ai.character.rk/RKStorage	Local
Character.AI	Conversation logs	Server-side API	N/A	Cloud
Linky.AI	Bot conversation logs	messages_user_<botID>.hive, .db	com.linkyai/app_flutter/	Local
Linky.AI	User-to-user messages	com.im_10.8.7.db	com.linkyai/databases/	Local
Persona.AI	Bot conversation logs	ai_personal_db (SQLite)	com.persona.ai/databases/	Local
Persona.AI	User-to-user messages	com.im_10.8.7.db	com.persona.ai/databases/	Local
Fantasy.AI	Bot conversation logs	ai_personal_db (SQLite)	com.fantasy.ai/databases/	Local
Fantasy.AI	User-to-user messages	com.im_10.8.7.db	com.fantasy.ai/databases/	Local

References

Anglano, C., 2014. Forensic analysis of WhatsApp messenger on Android smartphones. Digit. Invest. 11.

Baggili, I., Behzadan, V., 2019. Founding the domain of ai forensics. https://arxiv.org/abs/1912.06497.

Dragonas, E., Lambrinoudakis, C., Nakoutis, P., 2024. Forensic analysis of openai's chatgpt Mobile application. Forensic Sci. Int.: Digit. Invest. 50, 301801. **URL:**. https://www.sciencedirect.com/science/article/pii/S2666281724001252.

Hargreaves, C., Drury, J., 2025. Forensic analysis of AI applications: a replika "ai companion'' example, *in* 'Digital Forensics and Cyber Crime'. Lecture Notes in Computer Science, Springer. **URL:**. https://link.springer.com/chapter/10.1007/978-3-032-00635-6_5.

Meadi, M.R., Sillekens, T., Metselaar, S., van Balkom, A., Bernstein, J., Batelaan, N., 2025. Exploring the ethical challenges of conversational ai in mental health care: scoping review. JMIR Ment. Health 12, e60432. **URL:**. https://mental.jmir.org/2025/1/e60432.

Mireshghallah, N., Antoniak, M., More, Y., Choi, Y., Farnadi, G., 2024. Trust no bot: discovering personal disclosures in human-llm conversations in the wild. https://arxiv.org/abs/2407.11438.

Onik, A.R., Alsmadi, R., Baggili, I., Webb, A.M., 2024. So fresh, so clean: cloud forensic analysis of the Amazon iRobot Roomba vacuum. Forensic Sci. Int.: Digit. Invest. 48, 301686. DFRWS EU 2024 - Selected Papers from the 11th Annual Digital Forensics Research Conference Europe. https://www.sciencedirect.com/science/article/pii/S2666281723002056.

Onik, A., Spinosa, T., Asad, A., Baggili, I., 2024. Hit and run: forensic vehicle event reconstruction through driver-based cloud data from progressive's snapshot application. Forensic Sci. Int.: Digit. Invest. 49, 301762. DFRWS USA 2024 - Selected Papers from the 24th Annual Digital Forensics Research Conference USA. https://www.sciencedirect.com/science/article/pii/S2666281724000817.

Papneja, P., O'Dea, C., Fiske, A., Luger, E., 2024. Self-disclosure to conversational AI: a literature review. Personal Ubiquitous Comput. **URL:** https://link.springer.com/article/10.1007/s00779-024-01823-7.

Reuters, 2023. Italy bans U.S.-based AI chatbot replika from using personal data. https://www.reuters.com/technology/italy-bans-us-based-ai-chatbot-replika-using-personal-data-2023-02-03/.

Spinosa, T., Onik, A., Rovira, J., Baggili, I., 2025. Money on my mind: forensic investigation of venmo payment app. In: Coppens, B., Volckaert, B., Naessens, V., De Sutter, B. (Eds.), Availability, Reliability and Security'. Springer Nature Switzerland, Cham, pp. 60–77.

The Verge, 2024. Character.AI sued again over 'harmful' messages sent to teens. https://www.theverge.com/2024/12/10/24317839/character-ai-lawsuit-teen-harmful-messages-mental-health.

Ticong, L., 2025. Meta, character.AI accused of misleading kids with AI tools 'disguised as therapy. https://www.eweek.com/news/texas-attorney-general-investigates-meta-characterai-children-ai-mental-health/.

Walnycky, D., Baggili, I., Marrington, A., Moore, J., Breitinger, F., 2015. Network and device forensic analysis of Android social-messaging applications. Digit. Invest. 14, S77–S84. The Proceedings of the Fifteenth Annual DFRWS Conference. https://www.sciencedirect.com/science/article/pii/S1742287615000547.

Weidinger, L., Mellor, J., Rauh, M., Gabriel, I., et al., 2021. Ethical and social risks of conversational AI: a review of the literature. https://arxiv.org/abs/2112.04359.

Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., Zhang, Y., 2024. A survey on large language model (llm) security and privacy: the good, the bad, and the ugly. High-Confidence Computing 4 (2), 100211. **URL:**. https://www.sciencedirect.com/science/article/pii/S266729522400014X.