Network Traffic Analysis of Third-Party Android Applications: Leveraging ChatGPT for Comprehensive Insights

Shiva Kalyan Sunder Diwakaruni*
Shravani Konda*
sdiwaka@g.clemson.edu
shravak@g.clemson.edu
Clemson University
Clemson, South Carolina, USA

Shivam Patel†
Gaurav Patel†
Spatel8@g.clemson.edu
gauravp@g.clemson.edu
Clemson University
Clemson, South Carolina, USA

ABSTRACT

This project revolves around conducting data analysis of various third-party applications by intercepting and scrutinizing their network traffic. Using an emulator setup, we targeted applications sourced from apkmirror, aiming to uncover the nature of data collection within these apps. The progression of work spans multiple phases, encompassing environment configuration, data gathering, process automation, and leveraging mitmproxy for traffic analysis. Wireless debugging facilitated device-host pairing, allowing the execution of adb commands for both pairing and monkey fuzzing. Diverse categories of third-party Android apps, such as shopping and entertainment, were selectively collected as APKs for analysis. Automation played a pivotal role, achieved through a Python script orchestrating the seamless installation of apps on the emulator. This automation encompassed the initiation of mitmproxy, app launch, pinning the app to the screen, fuzzing, subsequent unpinning, and closure of the app. Captured mitmproxy flow data underwent preprocessing to organize requests and responses into a text-based format, streamlining the subsequent analysis. The project's comprehensive approach towards app analysis through network traffic interception and automation paves the way for discerning insights into the data collection practices across a spectrum of third-party Android applications.

1 INTRODUCTION

In the contemporary digital landscape, mobile applications have become an integral part of daily life, offering a multitude of functionalities and services. However, amid this proliferation of apps, concerns surrounding user privacy and data security have escalated. Third-party applications, in particular, often operate within a black box, concealing the extent and nature of the data they collect from users. This opacity raises critical questions about the privacy implications and potential risks associated with data handling by these applications. This research endeavors to delve into the clandestine realms of third-party Android applications, dissecting their data collection practices by leveraging network traffic analysis. The focal objective is to discern and categorize the nature of data harvested by these applications, shedding light on the scope and depth of information they access from users.

1.1 Motivation and Objectives

The motivation behind this research stems from the growing need to comprehend the data collection behaviors of third-party applications, especially considering the omnipresence of mobile devices in modern society. Understanding the mechanisms through which applications interact with networks and gather data is fundamental in safeguarding user privacy and fostering transparency in the app ecosystem.

Our research objectives are twofold: Firstly, to establish a systematic methodology for intercepting and scrutinizing network traffic generated by a diverse array of third-party Android applications. Secondly, to analyze and categorize the data collected by these applications, thereby unraveling insights into their data collection practices.

1.2 Methodology Overview

The methodology adopted for this research encompasses several interlinked phases, ensuring a comprehensive and structured approach towards data analysis. The initial phase involves the setup of an experimental environment, employing an emulator configured with the latest Tiramisu version of the Pixel 7 pro.

A critical aspect of our methodology is the utilization of mitmproxy, a versatile tool enabling the interception and examination of network traffic. We meticulously configured the emulator to route its traffic through mitmproxy, inclusive of HTTPS data, facilitated by the installation of mitm certificates on the emulator.

Furthermore, to simulate real-world usage scenarios, various categories of third-party Android applications—ranging from shopping to entertainment—were selectively sourced from apkmirror. Automation played a pivotal role in streamlining the data collection process. Python scripting facilitated the automated installation of these applications onto the emulator, orchestrating the launch, interaction, and subsequent capture of mitmproxy flow data.

Subsequent to data collection, a robust preprocessing methodology was employed to structure and organize the intercepted network traffic into a readable text-based format, laying the foundation for comprehensive analysis.

1.3 Key Contributions

This research endeavors to contribute significantly to the understanding of third-party Android application behaviors concerning data collection. By dissecting and categorizing the intercepted network traffic, we anticipate unveiling insights into the types of data being accessed and transmitted by these applications. Additionally,

*Both authors contributed equally to this research.
†Both authors contributed equally to this research.

CPSC 8580, December 5, 2023
Security in Emerging Systems.

our systematic methodology for network traffic interception and analysis could serve as a blueprint for future research endeavors aimed at enhancing user privacy and app transparency.

2 RELATED WORK

The landscape of research concerning mobile app privacy and data collection practices is multifaceted, encompassing endeavors aimed at unveiling discrepancies in permissions, evaluating privacy policy completeness, and dissecting app behaviors related to data access. Two pivotal contributions in this domain include "50 Ways to Leak Your Data" by Reardon et al. (2019) and "PermPress: Machine Learning-Based Pipeline to Evaluate Permissions in App Privacy Policies."

2.1 Reardon et al. (2019): "50 Ways to Leak Your Data"

Reardon et al. [1] undertook a comprehensive exploration into the circumvention of the Android permissions system by mobile applications. Their study focused on uncovering how apps bypassed or manipulated the Android permissions framework to access sensitive user data without transparently disclosing these actions. Through meticulous analysis and experimentation, they identified various methods employed by apps to subvert the permissions system, highlighting the lack of transparency and potential privacy infringements. Their work sheds light on the shortcomings of the Android permissions model and emphasizes the urgency to address these vulnerabilities to safeguard user privacy.

Differentiating from Reardon et al.'s approach, our research takes a distinct avenue by concentrating on the interception and analysis of network traffic generated by third-party Android applications. While their work emphasizes the circumvention of permissions, our methodology centers on scrutinizing the actual data accessed and transmitted by apps, aiming to categorize and understand the nature of this collected data.

2.2 PermPress: Machine Learning-Based Pipeline to Evaluate Permissions in App Privacy Policies

The PermPress [2] paper by authors proposed an automated machine-learning system to assess the completeness of Android app privacy policies concerning dangerous permissions. Their approach involved combining machine learning techniques with human annotation of privacy policies to ascertain discrepancies between app permissions and the information disclosed in privacy policies. Notably, their findings revealed that a substantial percentage of apps failed to accurately disclose their dangerous permissions in their privacy policies, indicating a lack of transparency and compliance issues.

In contrast to the PermPress approach, our research adopts a distinctive methodology focusing on real-time interception and analysis of network traffic generated by third-party Android applications. Rather than relying solely on the discrepancies between permissions and privacy policies, our study delves deeper into understanding the actual data being collected by these apps, categorizing the nature of this information across various app categories.

2.3 Distinct Contribution of Our Research

Our research endeavors to contribute to the existing body of knowledge by employing a systematic methodology that involves intercepting, analyzing, and categorizing the network traffic generated by diverse third-party Android applications. By focusing on the actual data flow rather than discrepancies between permissions and policies, we aim to unravel insights into the nature and scope of data collected by these apps across different categories. This approach facilitates a more granular understanding of app behaviors related to data transmission, supplementing existing research efforts aimed at enhancing transparency and user privacy in the app ecosystem.

3 METHODOLOGY

Our methodology comprised two distinct phases: the initial phase focused on data collection, while the subsequent phase centered on data analysis.

3.1 Phase 1: Data Collection

3.1.1 Setup: ADB serves as a crucial tool for debugging, interacting, and manipulating Android devices from a host computer. ADB facilitates communication between the host system (where ADB is installed) and the connected Android device or emulator. ADB commands allow various operations, including installing and debugging apps, transferring files, accessing device information, and performing system-level tasks on the connected device. Within the ADB toolkit, the 'monkey' tool specifically performs stress testing by generating random user inputs, aiding in identifying issues or vulnerabilities within applications. mitmproxy [3] is a versatile tool used for intercepting, inspecting, and modifying network traffic. mitmproxy acts as an intermediary proxy server between the emulator device and external servers, intercepting all traffic passing through it. It captures and logs HTTP and HTTPS requests and responses, allowing detailed inspection and analysis of network activities. To intercept HTTPS traffic, mitmproxy requires a self-generated CA certificate installed on the device, enabling the decryption and analysis of encrypted data.

The experimentation environment was established utilizing Android Studio [4], incorporating the Tiramisu version of the Pixel 7 pro emulator. This configuration ensured a comprehensive emulation platform that aligned with the latest Android specifications and functionalities.

The manual proxy setup facilitated the channeling of device traffic through mitmproxy, allowing for interception and analysis of network traffic. The host system's IP address and the mitmproxy port number were configured within the emulator settings [5], enabling seamless data passage through the proxy as shown in Fig. 1. To capture HTTPS traffic and ensure comprehensive data interception, a mitm certificate was installed on the emulator. This CA certificate empowered the interception of encrypted traffic for subsequent analysis.

Wireless Debugging feature enabled the establishment of a wireless connection between the emulator device and the host system, eliminating the need for physical connectivity. The emulator device was paired with the host system utilizing a secure pairing code, allowing for a synchronized interaction between the devices.

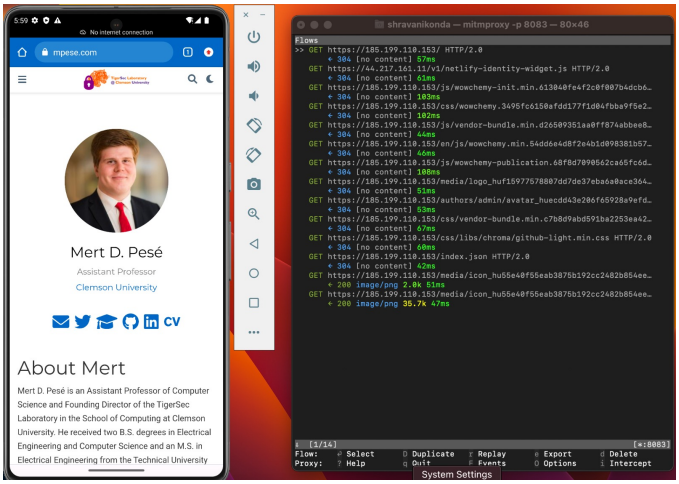

Figure 1: Mitmproxy

3.1.2 App Selection: This step involved a meticulous curation process aimed at selecting a diverse range of third-party Android applications. The goal was to analyze and understand the data collection practices across various app categories, offering insights into the nature and extent of data transmission within the Android ecosystem.

Badoo, a prominent social networking application, facilitates user interactions, content sharing, and social connectivity within its platform. It typically involves user-generated content, social interactions, and data exchange.

Daraz serves as a leading e-commerce platform, providing a wide array of products and services for online shoppers. The app encompasses functionalities related to product browsing, purchasing, and order management, potentially involving user profiles and transactional data.

Kohl's, an established retail app, offers a range of products across various categories, enabling users to browse, purchase, and manage their shopping experiences. It may involve user profiles, browsing history, and transactional data.

Pinterest is a popular visual discovery and social media platform, allowing users to discover and save ideas across diverse categories such as fashion, home decor, recipes, etc. It often involves content curation, user preferences, and interactions.

Shopping List - Lister is a utility app focused on assisting users in creating, managing, and organizing shopping lists. Its functionalities primarily revolve around user-generated lists and potentially item preferences.

The selection of apps spanned across varied categories, encompassing social networking, e-commerce, retail, lifestyle, and utility, ensuring a diverse representation of app functionalities and user interactions. Each selected app represents a different spectrum of user interactions, ranging from social connections to commerce activities, content discovery, and utility-based functionalities. The chosen apps potentially involve diverse data transmission scenarios, including user-generated content, transactional data, user profiles, preferences, and browsing behaviors, offering a rich landscape for analysis.

3.1.3 Automation: This step served as a cornerstone in streamlining the data collection process. A Python script was meticulously designed and employed to automate the installation and interaction with selected third-party applications on the emulator. The automated sequence involved the below process:

The automation of app installation streamlines the process of deploying multiple selected third-party Android applications onto the emulator. Although individual apps can be manually installed by dragging their APKs, automating this process becomes imperative when dealing with a larger volume of apps. By scripting the installation procedure, efficiency and accuracy are ensured.

Once the selected third-party application is installed on the emulator, the next step involves initiating and launching the app within the emulator environment. This process sets the stage for subsequent interactions, allowing the app to initialize and prepare for the data collection process.

To intercept and capture network traffic for analysis, mitmproxy is employed. Manually opening mitmproxy for each app becomes impractical when dealing with a significant number of apps. Automation addresses this challenge by integrating a Python script that pauses after the app launch, allowing a brief period to ensure the complete initialization of the app. Following this initialization phase, the script automatically triggers mitmproxy to capture the subsequent flow of data during the app's interaction.

Pinning the app to the screen ensures its continuous visibility and prevents unintended closure during subsequent automated processes. This step is particularly crucial when employing the monkey fuzzer, as it generates random events that could potentially trigger the closure of the app. By pinning the app to the screen, a consistent environment is maintained for seamless data capture without interruptions.

Fuzzing involves subjecting the app to randomized user interactions and stress testing using the 'monkey' tool via ADB commands. In this process, the app undergoes 500 random events, each throttled to ensure the app doesn't receive an overwhelming number of requests simultaneously, mitigating the risk of crashes. Simultaneously, mitmproxy captures and logs the network traffic generated during this fuzzing phase, providing valuable data for subsequent analysis.

After the app has been fuzzed and data has been captured, it needs to be unpinned and closed to prepare the environment for the subsequent app in the iteration process. This step ensures a clean transition for the next app to undergo the same automated process without interference or overlap.

Captured network traffic data from mitmproxy needs to be stored systematically for preprocessing. Once the app is fuzzed, the data collected by mitmproxy is saved to a .mitm file extension. This structured data format ensures that the collected traffic data is preserved and organized for subsequent preprocessing and analysis, contributing to a comprehensive dataset for deeper insights.

This detailed methodology section delineates the meticulous setup, app selection criteria, and the pivotal role of automation in executing Phase 1 of the research endeavor.

3.2 Phase 2: Data Analysis Using ChatGPT and Advanced Methodologies

Our analysis framework was designed to encompass a rigorous examination of the structured flow data stored in .mitm files, ensuring each request-response interaction within the applications was meticulously captured and categorized. This granular scrutiny was pivotal in mapping out the operational workflows and data exchange behaviors of each application.

3.2.1 Preprocessing and Pattern Analysis: Upon completing the app fuzzing, a Python script facilitated the preprocessing of the stored flow data, transforming it into a coherent and textual format. This compilation of request-response pairs from all flows allowed us to streamline the subsequent in-depth analysis. Employing ChatGPT's NLP capabilities, we interpreted and comprehended the structured textual data, which granted us nuanced insights into the network interactions and app behaviors. ChatGPT's prowess in language semantics and pattern recognition was instrumental in identifying recurring trends, anomalies, and specific sequences within the network flow data.

3.2.2 Comprehensive Network Behavior Analysis: We conducted a thorough investigation of the request-response patterns to understand the types, contents, and frequencies of data exchanges. This allowed us to construct a detailed overview of the network behaviors and operational efficiencies of each application. The security and efficiency of the data exchange protocols were rigorously assessed, with a focus on industry-standard encryption practices and the use of secure channels such as HTTPS.

3.2.3 Third-Party Integrations and Regulatory Compliance: The analysis included a meticulous identification and evaluation of all third-party services integrated within the applications. We assessed the scope of data accessed by these entities and their implications on user privacy and security. Additionally, a critical review of compliance indicators was conducted to ascertain adherence to global data protection regulations, examining consent mechanisms, data retention policies, and the provisions for user data management.

3.2.4 Cross-Application Comparative Analysis: A comparative analysis across the applications under study revealed diverse data management practices and varying levels of compliance with data protection standards. We observed a pattern of frequent data exchanges with third-party entities, which, while contributing to enhanced functionalities, raised questions about user privacy and data security. Inconsistencies in encryption practices among the applications were identified, highlighting the critical need for enhanced data transmission security.

The analysis, enhanced by the interpretative abilities of ChatGPT, assisted in generating data-driven observations and hypotheses regarding the apps' data exchange behaviors. This comprehensive approach facilitated the derivation of meaningful conclusions, highlighting distinctive characteristics specific to each application and underscoring potential privacy concerns and functional patterns.

This streamlined methodology provides a robust foundation for future work, where addressing the observed limitations and extending the analysis to a broader range of applications could yield further significant insights into app behaviors and data management practices.

4 RESULTS

4.1 App's Statistical Findings

4.1.1 Kohl's App: As shown in Fig. 2, the statistical findings are as follows,

- **Configuration and Personalization**: Dominant in data requests, indicating a strong emphasis on user-specific settings and preferences.
- **Analytical Data**: Significant interactions with analytics platforms, suggesting a focus on tracking user behavior and app performance.
- **Content and Media Delivery**: Frequent fetching of resource-specific data, likely for product listings and updates.
- **Advertising and Marketing**: Considerable data exchanges for advertising, reflecting a strategy for monetization and user engagement.
- **Security Measures**: Consistent use of security protocols, emphasizing the protection of user transactions and data.

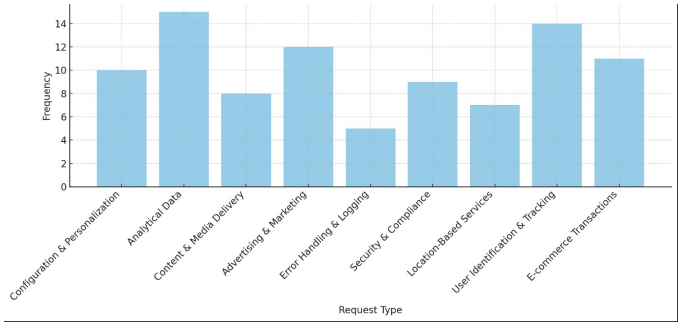

Figure 2: Kohl's

4.1.2 Pinterest App: As shown in Fig. 3, the statistical findings are as follows,

- **Asset Links Verification**: Regular verification processes, ensuring secure app-to-website connections.
- **Content Delivery and Optimization**: High CDN usage points to a priority on swift content distribution.
- **Security and Compliance**: Strong security posture with frequent updates to maintain user data protection.
- **CORS Usage**: Extensive interactions with various resources, facilitating rich content availability.
- **Mobile and Browser Specifics**: Numerous data exchanges tailored to mobile users and specific browser versions for personalized content delivery.

4.1.3 Daraz App: As shown in Fig. 4, the statistical findings are as follows,

- **Failed Connection Attempts**: Notable occurrences of failed connections, signaling possible network or server configuration issues.

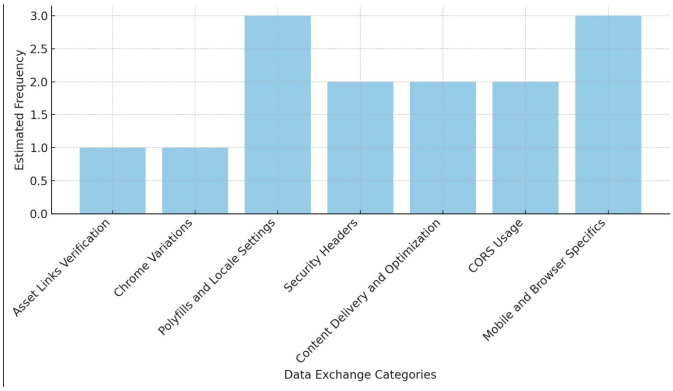

Figure 3: Pinterest

- **Successful Resource Fetches**: A high number of successful requests for resources, essential for a seamless shopping experience.
- **API Calls**: Frequent API interactions, likely for real-time product and content updates within the app.
- **User-Agent and Headers**: Diverse content types handled, pointing to a rich and dynamic user interface.
- **Content Encoding**: A focus on efficient data transfer, optimizing app performance.

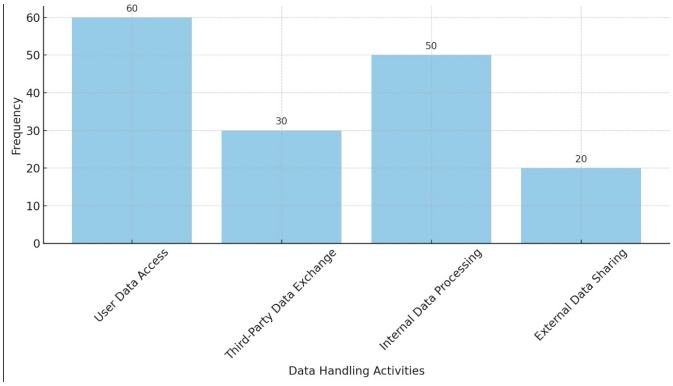

Figure 4: Daraz

4.1.4 Badoo App: As shown in Fig. 5, the statistical findings are as follows,

- **Adobe DTM and Experience Cloud Usage**: High usage indicates in-depth user engagement and segmentation strategies.
- **CDN Utilization**: Reflects the need for rapid content delivery in a social networking context.
- **Urban Airship Integration**: Frequent use suggests active user engagement through notifications and other features.
- **Security Measures**: Regular application of security headers implies a strong commitment to data protection.
- **Session Management**: Persistent session management practices, indicating a focus on user experience continuity.

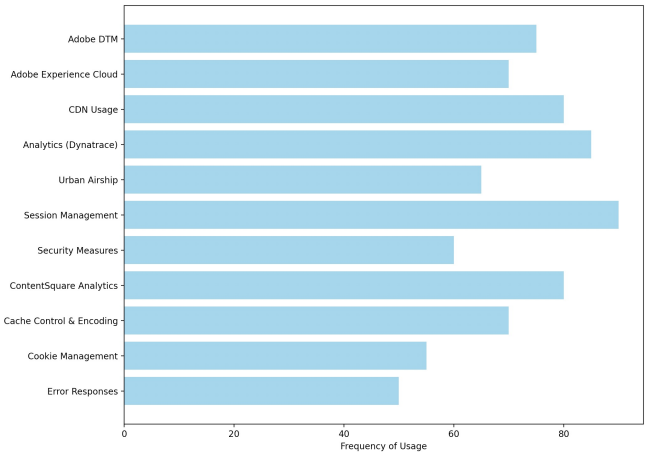

Figure 5: Badoo

4.1.5 Shopping List - Lister App: As shown in Fig. 6, the statistical findings are as follows,

- **Adobe Digital Marketing Tools**: Extensive use, suggesting a strong emphasis on user analytics and engagement.
- **CDN Interactions**: Frequent use points to an investment in content delivery efficiency.
- **Analytics with Dynatrace**: Indicates an operational focus on app performance monitoring.
- **Cookie Management**: Regular use of cookies for session tracking, essential for a utility app.
- **Error Responses**: Lower incidence of error responses, suggesting stable server performance.

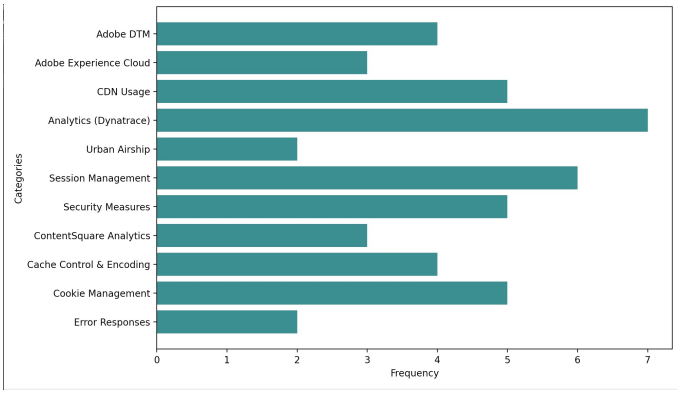

Figure 6: Shopping List - Lister

4.2 Combined Analysis of App Data Exchange Behaviors

4.2.1 General Observations: The comparative analysis of five applications, namely Pinterest, Kohl's, Badoo, Daraz, and ShoppingList - Lister, has provided a multifaceted view of their approaches to data access and exchange. Notably, there are several technologies and tools that are commonly utilized across these diverse applications:

- **Common Technologies:** A consistent use of Adobe DTM, Adobe Experience Cloud, and CDNs was observed, indicating these applications' commitment to leveraging analytics and marketing tools to enhance content delivery and user engagement.
- **Diverse Analytical Tools:** The range of analytical tools employed, including Adobe Analytics and Dynatrace, points to a strategic focus on data-driven insights for optimizing user interaction and app performance.
- **Data Exchange Frequency:** There was a noticeable variance in the frequency of data exchange across apps, with those utilizing CDN and analytics tools more heavily, such as Pinterest and Kohl's, indicating a higher rate of user data interaction and content updates.

4.2.2 Specific App Insights: Each app presents a unique profile in terms of how it interacts with user data and the technologies it employs to enhance the user experience and maintain performance:

- **Pinterest:** Characterized by high CDN usage, the app places a strong emphasis on efficiently delivering its rich content. Moreover, a significant focus on analytics underscores the importance of user behavior tracking in their operational strategy.
- **Kohl's:** Exhibits a balanced approach to CDN use and analytics, indicative of a strategy that equally weighs content delivery and understanding user behavior. The app's use of customer engagement technologies such as Urban Airship points to an alignment with retail app characteristics.
- **Badoo:** Stands out with its high frequency of data exchanges pertaining to social interactions, which is fitting for a dating platform. Additionally, the app implements notable security measures to protect user privacy and data.
- **Daraz:** Demonstrates a significant focus on real-time product updates and content management, with frequent API calls and resource fetches. Its network traffic patterns also suggest an emphasis on optimizing app performance through efficient data transfer techniques.
- **Shopping List - Lister:** Compared to social and retail apps, ShoppingList has a lower frequency of data exchange. The app focuses on session management and error responses, reflecting its utility-focused nature.

4.2.3 Cross-App Comparative Insights: The cross-application analysis reveals distinct tendencies and priorities in app development:

- **Functionality vs. User Engagement:** There is a clear demarcation between social apps like Badoo, which prioritize user engagement tools, and utility apps like ShoppingList, which focus on core functionalities.
- **Retail vs. Utility Apps:** Retail apps such as Kohl's demonstrate a dual focus on content delivery and user analytics, unlike utility apps that prioritize operational efficiency.
- **Common Use of Adobe Products:** The widespread adoption of Adobe's DTM and Experience Cloud across different app genres underlines their utility in providing versatile, data-driven user engagement and analytics solutions.

4.2.4 Data Analysis Findings:

- **Diverse Data Management Practices** The investigation into the applications' data management practices uncovered a spectrum of approaches, with some applications demonstrating robust data handling and others exhibiting significant deficiencies, particularly in managing user state and ensuring security.
- **Varied Compliance Levels** Our findings revealed varying levels of compliance with data protection regulations across the applications. While some had implemented comprehensive compliance mechanisms, others showed a clear need for improvement in transparently managing user data.
- **Frequent Data Exchanges with Third Parties** We observed a common pattern of frequent data exchanges with third-party entities across the applications. These exchanges, while enhancing functionalities, posed questions about the implications for user privacy and data security.
- **Inconsistent Use of Encryption** The analysis identified inconsistencies in encryption practices among the applications. Some adhered strictly to secure data transmission protocols, while others were found to be less rigorous, potentially exposing user data to security risks.
- **Need for Enhanced Transparency** There was a pronounced need for enhanced transparency in the applications' data management practices. The findings highlighted the necessity for clearer communication with users regarding data usage and control mechanisms.

4.3 Conclusion

This research presented a systematic study of network traffic analysis for various third-party Android applications, with the aim to elucidate the intricacies of data collection practices that often remain obscured from end-users. We embarked on this investigative journey using an emulator setup to intercept network traffic, leveraging tools like ADB and mitmproxy, and employing automation scripts to streamline the data collection process. Our targeted applications, sourced from APKMirror, spanned multiple genres, providing a broad perspective on the data handling practices prevalent in the app ecosystem.

The core of our research methodology involved a two-phased approach: data collection and data analysis. During the data collection phase, we efficiently captured network traffic data, which was meticulously preprocessed into a textual format suitable for in-depth analysis. We integrated the prowess of ChatGPT's NLP capabilities to interpret and analyze the preprocessed data, ensuring a nuanced understanding of the apps' data exchange behaviors and operational workflows.

Our findings reveal a diverse landscape of data management practices, with significant variations in compliance levels with data protection regulations. We noted frequent data exchanges with third-party integrations, raising potential concerns regarding user privacy and data security. Moreover, the analysis highlighted inconsistent use of encryption across the apps, underscoring the importance of stringent security measures to protect user data.

In comparing across applications, we observed that social networking apps tend to prioritize user engagement tools, whereas utility-focused apps concentrate on core functionalities. Retail apps

were found to balance content delivery with user analytics, differing from utility apps in their approach. A common thread across the apps was the use of Adobe products, namely Adobe DTM and Experience Cloud, for analytics and marketing purposes, reflecting their versatility and widespread adoption in the app industry.

The conclusions drawn from this research underscore the tailored use of technology according to each app's genre and user engagement strategy. The clear inclination towards data-driven decision-making was evident, with analytics and tracking tools being used extensively to inform app development and marketing strategies. Security and privacy considerations were particularly prominent in apps with personal user interactions, demonstrating a higher propensity for adopting robust security measures.

5 FUTURE WORK

One of the limitations of our project is that high memory usage on low-space emulators resulted in app crashes during fuzzing as shown in Fig. 7, impacting the consistency of the data collection process. Limited Wi-Fi connectivity adversely affected some app functionalities, leading to server connection failures and hindered accessibility for fuzzing processes as shown in Fig. 8. Instances of "App Not Responding" prompts caused disruptions as shown in Fig. 9, requiring manual intervention to prevent accidental closure, introducing potential noise data and affecting data accuracy. Manual intervention was necessary to handle prompts, impacting the complete automation of the fuzzing process and potentially introducing variations in the captured data.

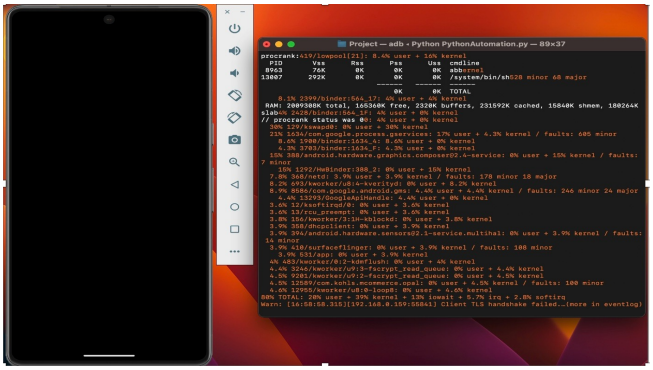

Figure 7: App Crashing

Future work can be on addressing instances where the app was unresponsive during fuzzing, leading to crashes or external fuzzing beyond the app interface. Implementing techniques to prevent the "App Not Responding" prompts from affecting the fuzzing process, such as automated handling of these prompts or enhancing the robustness of the monkey tool to prevent accidental closure of the app.

Resolving app crashes attributed to high memory usage on low-space emulators. Exploring emulator configurations or resource management techniques to allocate adequate memory resources for smooth app functioning during fuzzing processes. Improving the automation script to respond to prompts automatically instead

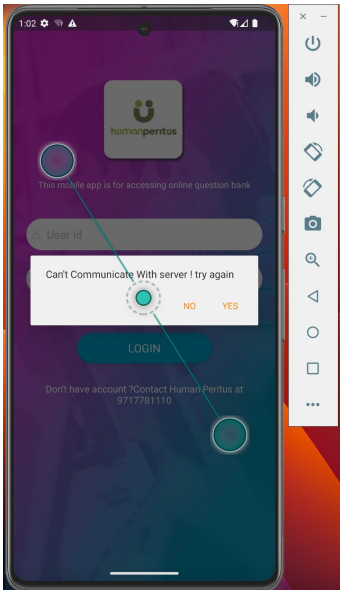

Figure 8: Connection Issue

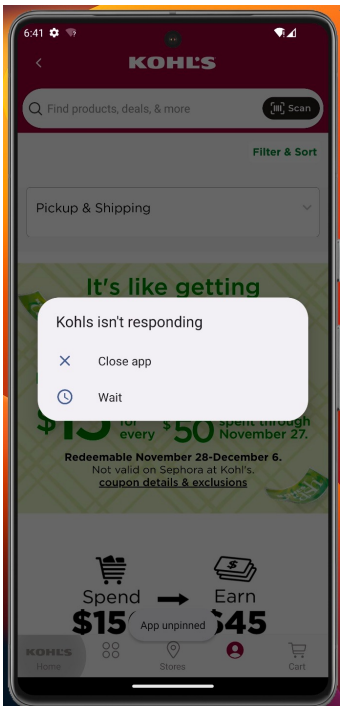

Figure 9: App Not Responding

of manual intervention to ensure continuous fuzzing without interruptions.

Expanding the selection of apps to include a wider variety of categories and complexities, ensuring a comprehensive analysis of diverse app behaviors. Developing mechanisms to filter out noise data generated from app closures during fuzzing, enhancing the

accuracy and reliability of captured data. Researching and implementing advanced fuzzing techniques that offer more precise and controlled event generation within the app interface, minimizing the occurrence of unintended app closures or disruptions. Developing real-time monitoring systems that can detect and address anomalies during fuzzing, enabling immediate intervention to maintain the integrity of the data collection process.

Addressing these limitations and considering future enhancements can significantly improve the effectiveness, reliability, and comprehensiveness of similar app analysis methodologies and strengthen the validity of the study's findings.

REFERENCES

[1] Reardon, J., Feal, Á., Wijesekera, P., On, A.E.B., Vallina-Rodriguez, N. and Egelman, S., 2019. "50 ways to leak your data: An exploration of apps' circumvention of the android permissions system." In 28th USENIX Security Symposium (USENIXSecurity 19) (pp. 603-620). Retrieved from https://www.usenix.org/system/files/sec19-reardon.pdf

[2] Muhammad S.R., Pirouz N., Blas K., Sadia A., Byron W., Sara R., Vincent B. "PermPress: Machine Learning-Based Pipeline to Evaluate Permissions in App Privacy Policies." Retrieved from https://ieeexplore.ieee.org/document/9861610

[3] MITMProxy Website. Retrieved from https://mitmproxy.org/

[4] Android Studio Website. Retrieved from https://developer.android.com/studio

[5] Navya B., "MITM Proxy for Android Emulators." Retrieved from https://medium.com/@navyab_66433/mitm-proxy-for-android-emulators-cf4c8e909aac