USENIX Security '24 Artifact Appendix: Towards Privacy-Preserving Social-Media SDKs on Android

Haoran Lu, Yichen Liu, Xiaojing Liao, Luyi Xing

Indiana University Bloomington
{haorlu, liuyic, xliao, luyixing}@indiana.edu

A Artifact Appendix

A.1 Abstract

Under the context of XLDH threats, we generalize and define what are privacy-preserving social-media SDKs and their in-app usage, characterize fundamental challenges for combating the XLDH threat and guaranteeing privacy for the design of and practice with social-media SDKs. We also release the entire dataset on our website. We present a practical, clean-slate design and end-to-end systems called PESP to enable privacy-preserving social-media SDKs against the emerging privacy threat. Our thorough evaluation showed its satisfactory effectiveness, performance overhead and practicability for adoption. We employ case studies of demos using PESP to demonstrate its effectiveness. For performance overhead, we provide data and scripts to reproduce the result. Our techniques are expected to significantly elevate privacy assurance and compliance for multiple stakeholders.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

Our artifacts are developed with no inherent security, privacy, or ethical concerns. Nevertheless, as a precaution, we recommend installing and testing our system on a non-production Android phone or using the environment we provide.

A.2.2 How to access

The implementation code of the system is currently available on GitHub at `https://github.com/PESP-privacy-preserving-social-SDK/PESP/tree/4d5b699f40f18ebdb61118e435bd99d0d3d46907`. Please follow the instructions provided in the README.md file.

A.2.3 Hardware dependencies

For the implementation and evaluation of our system, we have comprehensively tested the use case on a real Android device, the Google Pixel 6 (Android version 13, r16), and on an Android emulator, the Google Pixel 3a (Android version 14, ARM64 v8a). The minimum Android device requirement is version 13, r16. For installing Android Studio, we recommend a system with 16GB of memory and a solid-state drive of at least 16GB. Notably, systems with Apple M1/M2 chips are compatible.

A.2.4 Software dependencies

PESP requires Android Studio, and an Android real device (Android version 13, r16) or an Android emulator. Please note that if you install Android Studio by default, you do not have to install Android emulator on your own. We have thoroughly tested the build process on macOS 14 and Ubuntu 22.04.
Dependencies.

- Linux OS/macOS/Windows, preferably `Ubuntu 22.04 LTS, macOS 14`
- Android Studio with emulator, preferably `latest version v2023.2.1, for E1 with Gradle version 7.2, for E2 with minimum Gradle version >= 6.7.1 and <7 and JDK version 1.8`
- Android emulator (do not have real devices), preferably `Pixel or other emulator with Android version` $\geq$ `13, r16, and we recommend using the one installed with Android Studio`
- python 3.11
- numpy $\geq$ v1.24.3
- scipy $\geq$ v1.11.1
- python-dateutil prefer the latest version

A.2.5 Benchmarks

Given the absence of similar works, we do not include specific benchmarks. The baseline of our work is the original Facebook and Twitter SDK, and we provide

the original Facebook SDK for comparison (facebook-android-sdk-main under our project https://github.com/PESP-privacy-preserving-social-SDK/PESP/tree/4d5b699f40f18ebdb61118e435bd99d0d3d46907).
We release the whole Table 1 in our repository under folder use-case-table. It serves to demonstrate the existence of Type I and Type II workflows.

A.3 Set-up

A.3.1 Installation

To install the necessary software dependencies, including Android Studio, download the latest version from https://developer.android.com/studio and refer to the provided documentation for guidance on installation across different operating systems https://developer.android.com/studio/install. Opt for the standard installation process, and follow the on-screen instructions to download all required components. The installation of Android Studio will automatically include an Android emulator. For building the PESP and demo applications, please consult the README.md file on our GitHub page.

Notably, if you want to change Android Studio settings, you can click File > Settings. If you want to change Gradle version when building Android projects, please click File > Project Structure > Project menu and select Gradle version. If it's the first time you build the project, it may take a while. Once it finishes Gradle building, you can click on the "green play button" to install and run the APK on the Android emulator.

A.3.2 Basic Test

After installing Android Studio, you have the option to test the Android emulator, with Google Pixel 3a set as the default. Locate the plug icon next to "Running Devices" and click it. Then select the first virtual device, Pixel3a, from the list. The virtual device should power on within a few minutes.

To check the versions of numpy and scipy installed on your system, you can use the following commands in your terminal:

```
python3 -c "import numpy; print(numpy.version.version)";
python3 -c "import scipy; print(scipy.version.version)"
```

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): *The example application illustrates the effectiveness of PESP. To replicate the case study and security analysis, which involves displaying a Facebook user profile under PESP as described in Section 5.2, you can follow and verify Experiment (E1). The results are presented in Figure 3 in the Appendix of the paper.*

(C2): *The performance overhead of PESP is practical compared to baseline. This is supported by experiments (E2, E3) described in section 5.4.1 whose results are illustrated in Tables 2.*

A.4.2 Other Claims

In our repository, we provide four materials.

- paperImplementation: An Android studio Project containing the client_app(developers' app in the paper) and an sdk_app(social medial SDK in the paper).
- facebook-android-sdk-main: An Android studio Project containing the original implementation of FB SDK and Twitther SDK from the official Facebook and Twitter code release, used as a baseline with logging enabled for evaluation purposes.
- evaluation-table-data: Original logging and metric computation code in Python used to fill the tables, including Table 2 Performance overhead (single social SDK per app) and Table 3 Performance overhead (multiple social SDKs per app).
- use-case-table: We provide the detailed table mentioned in Table 1 in the paper.

In paperImplementation, we provide commands to execute the prototype implementation. For facebook-android-sdk-main, we use it as the baseline, and we provide commands to run the sdk prototypes. For the Table 2 and Table 3 mentioned in paper, we provide scripts and logs to reproduce the results. More specifically, we provide detailed steps to manually collect logs and reproduce Table 2 (see evaluation-table-data/table2/table2.md), but we will not provide instructions for Table 3. In use-case-table, we provide detailed table mentioned in Table 1. The results are analyzed by hand by two authors, and we will not include this part for AE.

A.4.3 Experiments

(E1): *Case study and security analysis of displaying Facebook user profile under PESP [5 human-minutes + 10 compute-minutes]: build and run the case study example app on an Android emulator or real Android device.*
How to: *Build and run our example app using PESP.*
Preparation and execution: *Please follow the instructions in README.md within the code package to build sdk_app and client_app. Note that building an Android project in Android Studio for the first time may take longer.*
Results: *After building the PESP sdk and example app, on the Android emulator or the real device, you can*

click "FB PROFILE" to recreate Figure 3 in our paper. You can also click button "FB DEMO" and "TWITTER DEMO", these are for section 5.1 and section 5.3, and we will collect the time in logs to calculate the performance overhead in Table 2.

(E2): *Base case of using original Facebook SDK without PESP [5 human-minutes + 10 compute-minutes]: build and run the samples in an Android emulator or real Android device. The aim is to compare the performance overhead of using original Facebook SDK and PESP.*

How to: *Build original Facebook SDK and run samples.*

Preparation and execution: *Please follow the instructions in README.md on Github for building and running Facebook SDK samples, such as FBLoginSample. Be aware that the required Gradle version for these samples differs from that for PESP; the minimum Gradle version is v6.7.1, and JDK v1.8 is recommended. Additionally, note that building an Android project in Android Studio for the first time may take longer.*

Results: *After building the original Facebook SDK and the sample app, on the Android emulator or the real device, you can use basic functions like login or display user profile.*

(E3): *Performance overhead [5 human-minutes + 3 compute-minutes]: Reproduce results in Table 2.*

How to: *Please follow the python commands in README.md and evaluation-table-data/table2/table2.md on Github.*

Results: *The results will display directly after running each command.*

A.5 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at `https://secartifacts.github.io/usenixsec2024/`.