TrustZone Face Recognition

Shivam Shekhar (ss6960)
…
Ritwik Goel (rg3546)

Estimated Total Hours: 10

Plans (Past Week)

- Dive into ARM TrustZone, OP-TEE, and Mobile Development: Study ARM TrustZone technology and its application in secure environments.

- Explore OP-TEE documentation and understand its APIs. Identify potential face recognition algorithms: Research state-of-the-art face recognition algorithms (e.g., OpenCV, Dlib). Consider factors for the algorithm and keep security in mind. Cannot have a heavy algorithm with too many dependencies. Identify dependencies and critical path items/vulnerabilities.

What Actually Happened

Work completed:

1. We pivoted from OPTEEE to TrustyTEE because of our facial recognition project is specifically targeted at Android devices, Trusty TEE is a better choice due to its integration with the Android ecosystem and support from Google.The reason why we pivoted away from OP-TEE was because Trusty has smooth interaction with the Android operating system, hardware-backed security features, secure IPC mechanisms, dedicated TEE APIs, and broad device compatibility. OPTEE also has limited integration with the Android Ecosystem.
2. We explored the Trusty TEE documentation and we came up with the following Flow Diagram:
   a. User Interaction (Normal World, Android App running on Kotlin)
   b. Capture Facial Data (We are proceeding with BlazeFace model)
   c. Send Data to Trusty TEE (Secure World, We have IPC calls we can make on top of Trusty TEE)
   d. Process and Authenticate Facial Data
   e. Return Authentication Result
   f. Action Based on Authentication

Major challenges+roadblocks:

Deciding between OP-TEE and Trusty TEE. We made a scratch pad for the same. (Link)

Attribution: Trusty TEE integration and IPC Mechanism for the normal world to

interact with secure world

Plans (Next Week)

- Set up QEMU for simulation: Install and configure QEMU to simulate ARM-based environments. Test basic functionalities of the application in the simulated environment.
- Set up Trusty TEE, AppDev Framework, and Android Studio: Install necessary development environments and tools. Ensure compatibility and version consistency between team members.
- Setup a Kotlin android application with a barebone application and initialize integration of facial recognition.

Summary/Overall Progress

Scratch Pad for points against OP-TEE: https://docs.google.com/document/d/1PU3EmJMOInVuJ0ywK-j6c44i7NiDkeLE9FH6yb71it4/edit?usp=sharing

Understood the applications of the Secure API's in Android and got started with QEMU for Secure World Development.

Started the development of the Kotlin App. Decided against Flutter and React Native due to better integration with Android API's. Kotlin apps, being native Android apps, have direct access to platform-specific APIs and resources, allowing for more fine-grained control over resource utilization.