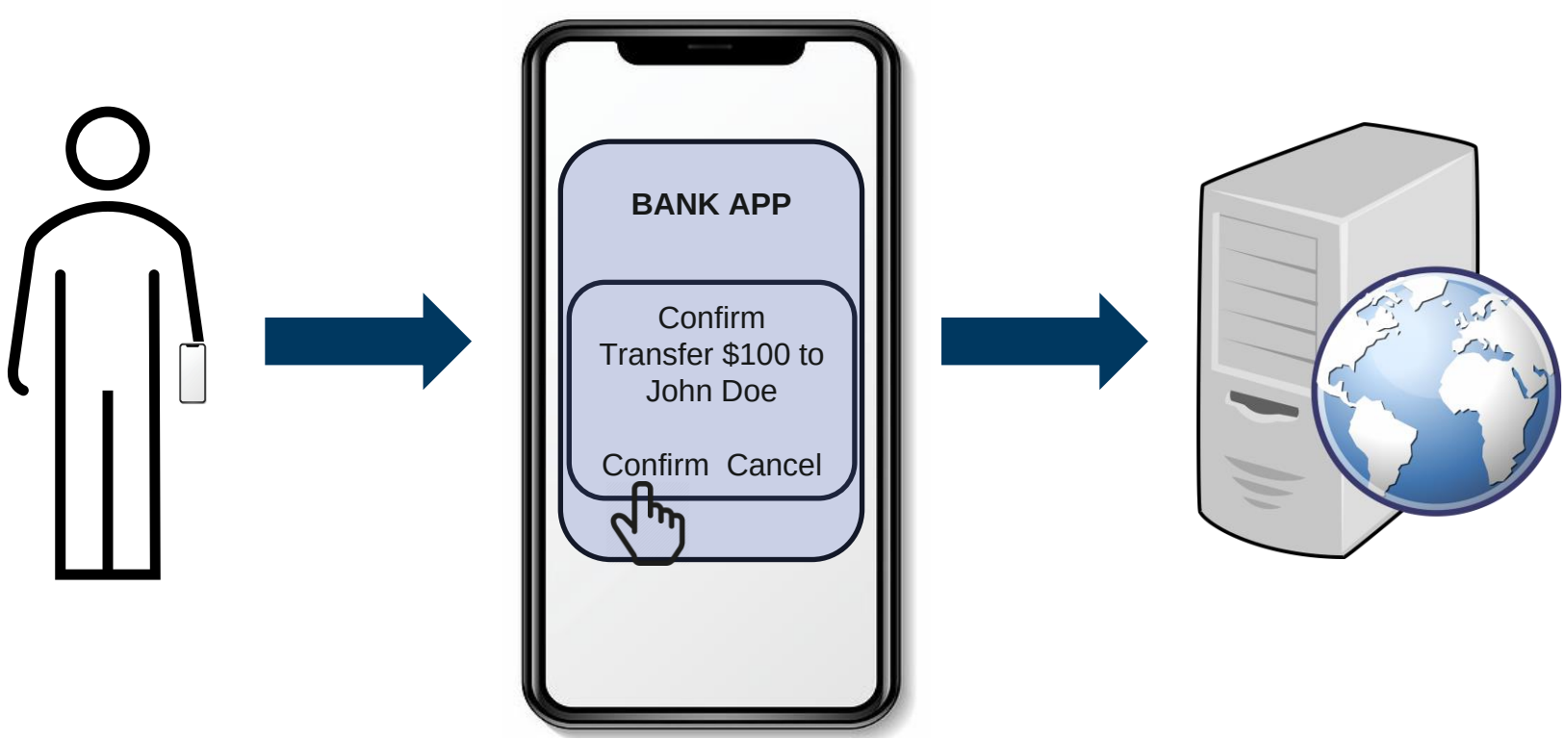
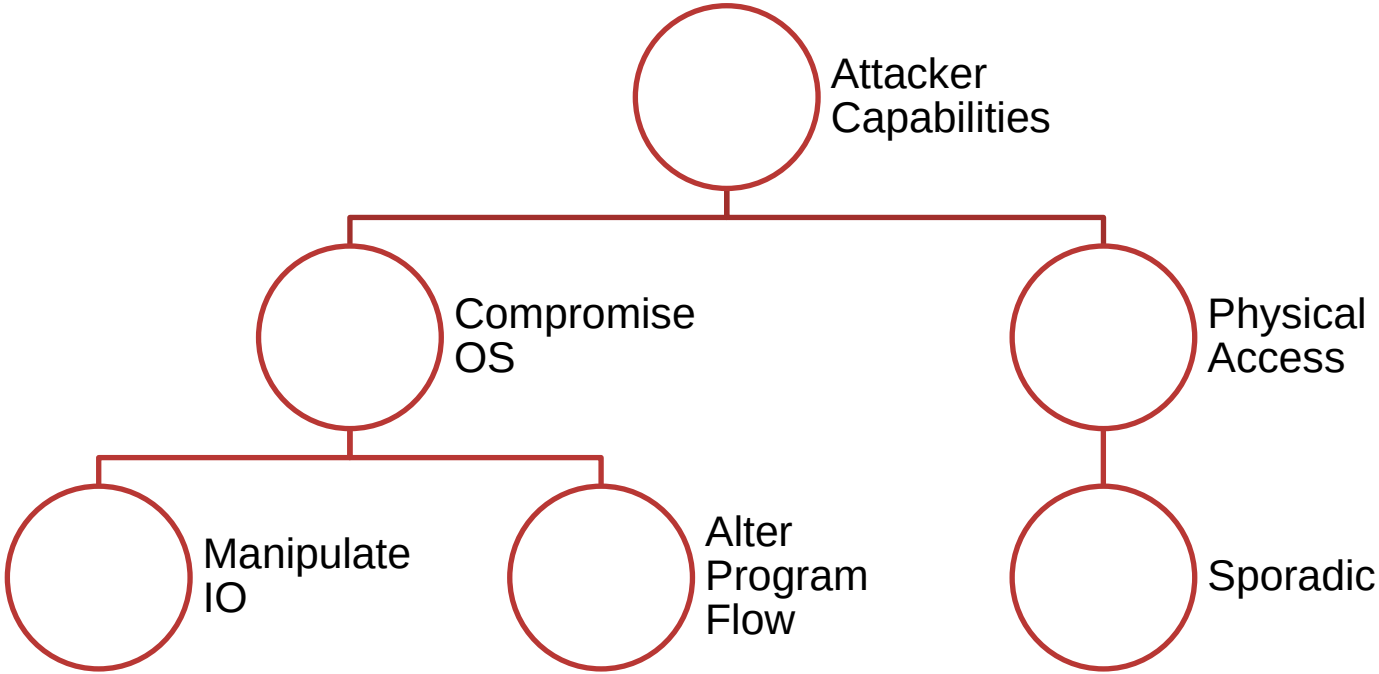
SARA:
Secure Android Remote Authorization

Abdullah Imran, Habiba Farrukh, Muhammad Ibrahim,
Z. Berkay Celik, Antonio Bianchi
Purdue University

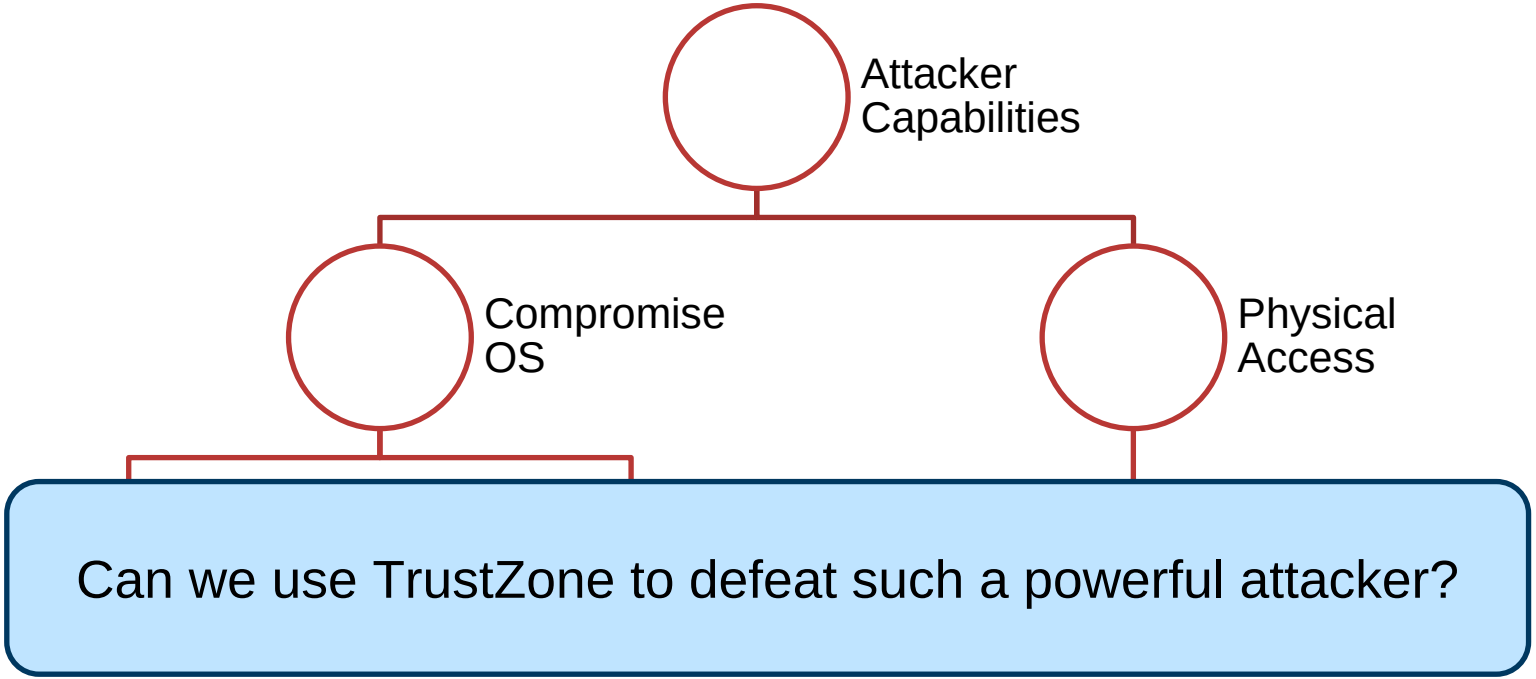
Mobile Devices in Authorization Schemes



Threat Model



Threat Model

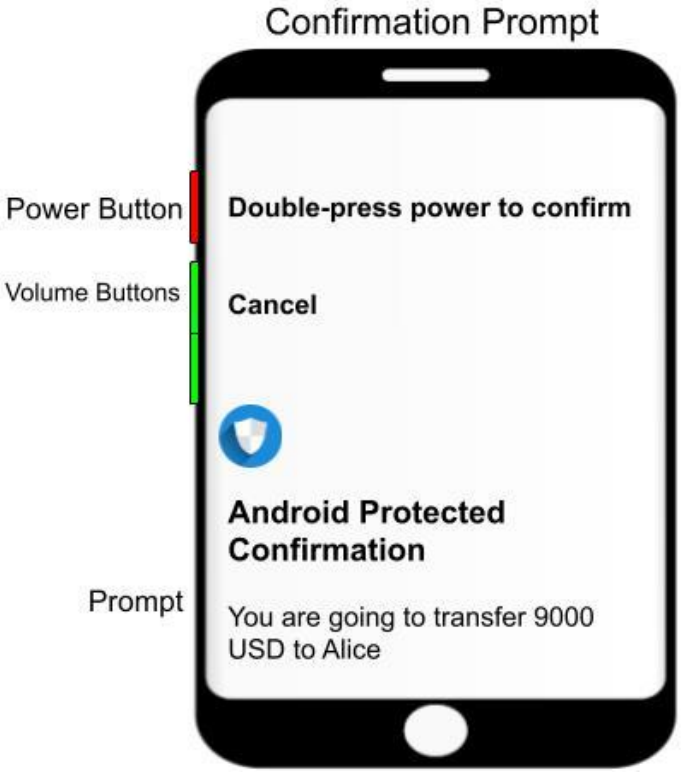


Existing APIs in Android

- Key Storage in TrustZone
- Key Attestation in TrustZone
- TrustZone controlled Secure UI
- Biometric Prompt

Existing APIs in Android

- Key Storage in TrustZone
- Key Attestation in TrustZone
- TrustZone controlled Secure UI
- Biometric Prompt



Existing APIs in Android

- Key Storage in TrustZone
- Key Attestation in TrustZone
- TrustZone controlled Secure UI
- Biometric Prompt

Market Analysis

112,886 Apps (Google Play Store)

Android Protected Confirmation

- 0 Apps using

Key Attestation

- 5 Apps using
- All local use cases

API Limitations

Biometric Prompt

- Fake Prompt
- Everlasting Biometric

Android Protected Confirmation

- Illegitimate User
- Overwriting Confirmation

API Limitations

Biometric Prompt

- Fake Prompt
- Everlasting Biometric

Android Protected Confirmation

- Illegitimate User
- Overwriting Confirmation

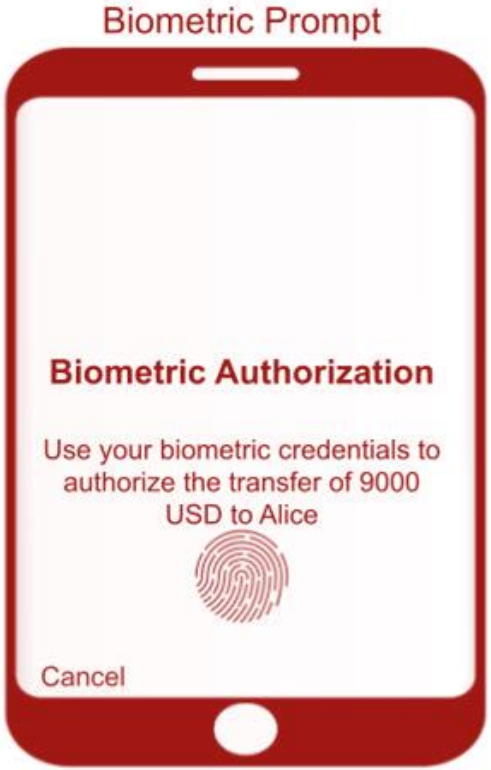
API Limitations

Biometric Prompt

- Fake Prompt
- Everlasting Biometric

Android Protected Confirmation

- Illegitimate User
- Overwriting Confirmation



API Limitations

Biometric Prompt

- Fake Prompt
- Everlasting Biometric

Android Protected Confirmation

- Illegitimate User
- Overwriting Confirmation

API Limitations

Biometric Prompt

- Fake Prompt
- Everlasting Biometric

Android Protected Confirmation

- Illegitimate User
- Overwriting Confirmation

Goals

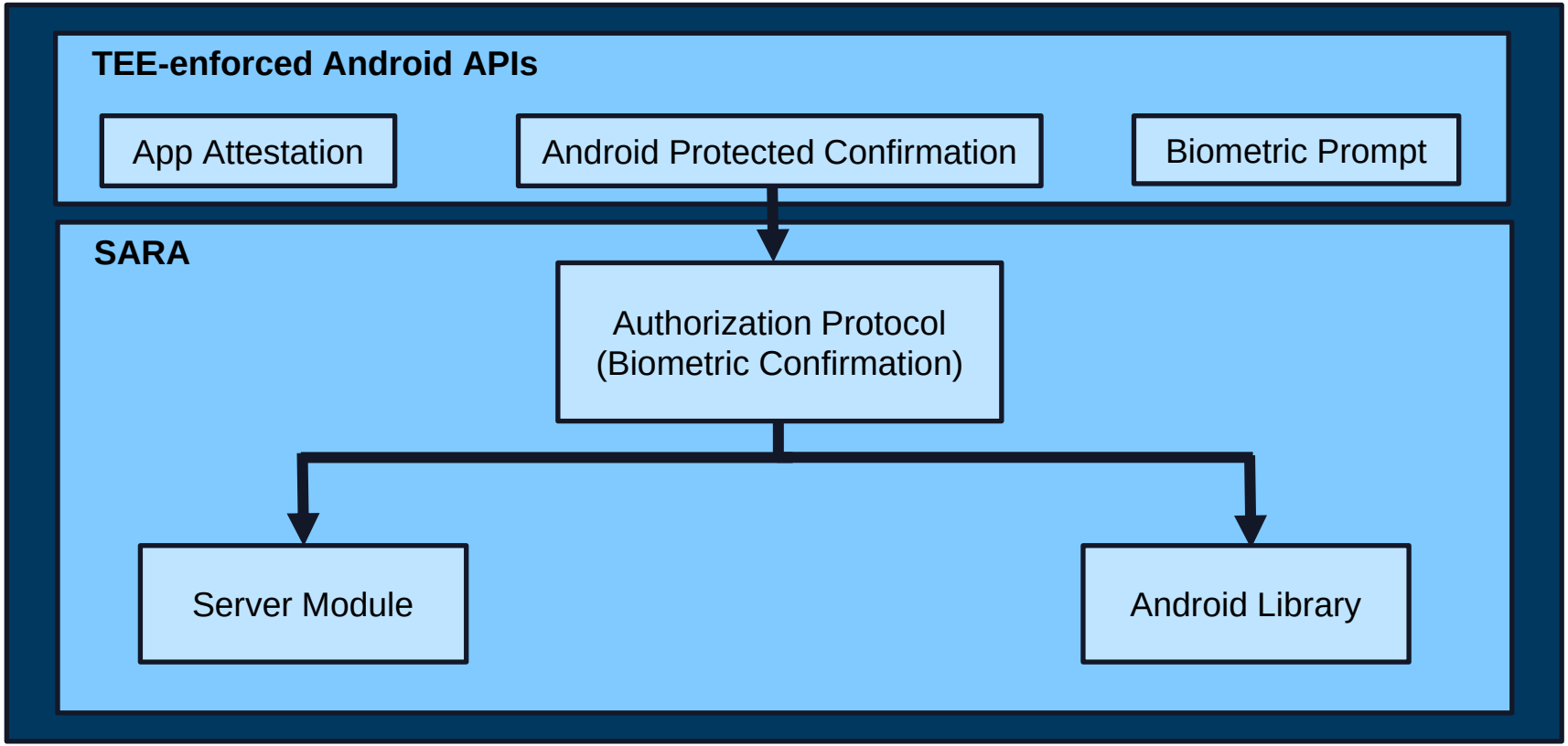
Usability Goals

- **Easy to use for developers**
- **Use existing Android APIs**

Security Goals

- **OS Compromise**
  - TEE Usage
  - Key Attestation
  - User Awareness
  - Server Verification
- **Physical Attacks**
  - User Awareness
  - Physical Authorization

SARA's Architecture



SARA's Process The User's Experience

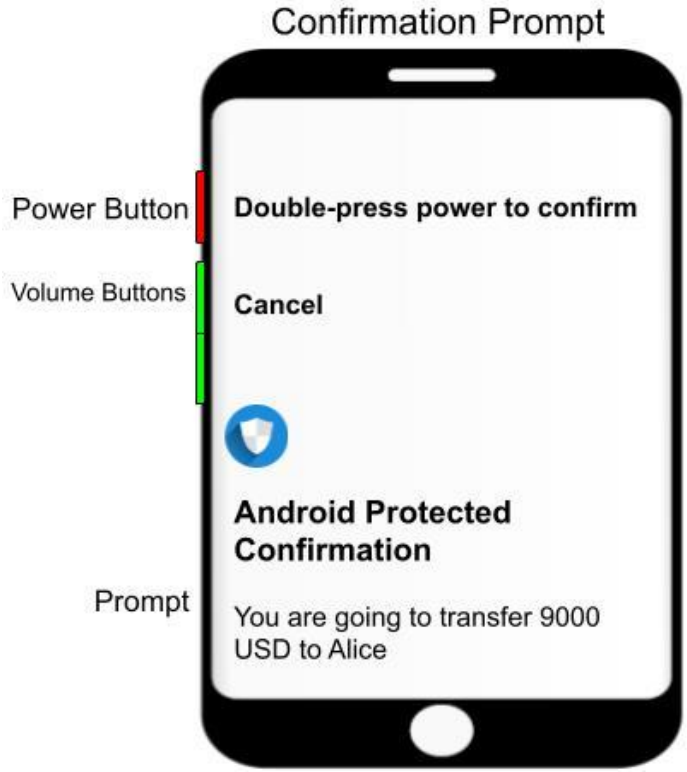
- Enable authorization -> One time only process
- Keypair generation and Attestation
- Authorize Action
- Biometric Prompt Displayed
- User provides biometric input (i.e., fingerprint)
- Prompt gets signed upon user's valid biometric input
- Confirmation Prompt Displayed
- User sees Android Protected Confirmation Prompt
- User presses hardware button to accept prompt
- Second signature takes place and sent to server for verification

SARA's Process The User's Experience

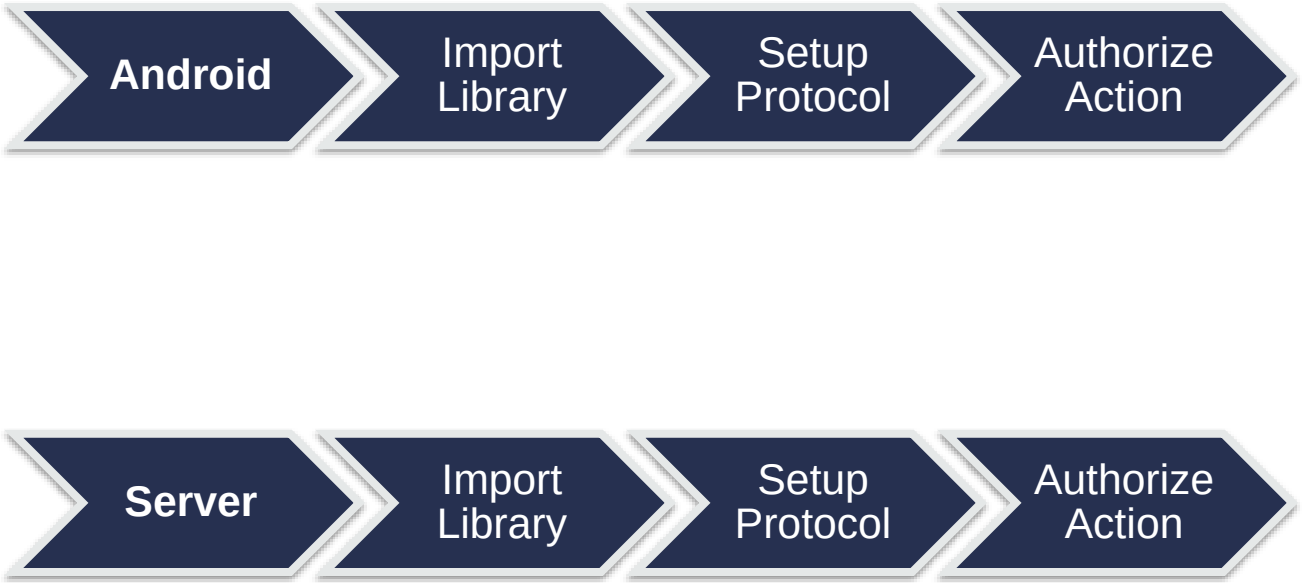
- Enable authorization -> One time only process
- Keypair generation and Attestation
- Authorize Action
- Biometric Prompt Displayed
- User provides biometric input (i.e., fingerprint)
- Prompt gets signed upon user's valid biometric input
- Confirmation Prompt Displayed
- User sees Android Protected Confirmation Prompt
- User presses hardware button to accept prompt
- Second signature takes place and sent to server for verification

SARA's Process The User's Experience

- Enable authorization -> One time only process
- Keypair generation and Attestation
- Authorize Action
- Biometric Prompt Displayed
- User provides biometric input (i.e., fingerprint)
- Prompt gets signed upon user's valid biometric input
- Confirmation Prompt Displayed
- User sees Android Protected Confirmation Prompt
- User presses hardware button to accept prompt
- Second signature takes place and sent to server for verification



SARA's Process The Developer's Experience



SARA's Security Evaluation

ProVerif Model

- Model SARA's authorization protocol in ProVerif's cryptographic protocol verifier.

Verify that SARA's protocol satisfies the following security goals for any action undertaken by a server:

- The legitimate user sees the action the server performs
- The legitimate user physically authorizes the action the server perform
- Server has a guarantee the that the action has been authorized by the legitimate user

Attacks on Incomplete Protocols

- Modeled alternate protocols in ProVerif to show the possible attacks on them due to their limitations

User Study

Comparison between Native API and SARA's API

Two identical tasks divided into 3 subtasks each

Answers to two questions:

- Does using SARA make it easier for developers to use Android's TEE-enforced APIs?
- How long does it take for a developer to learn how to use SARA?

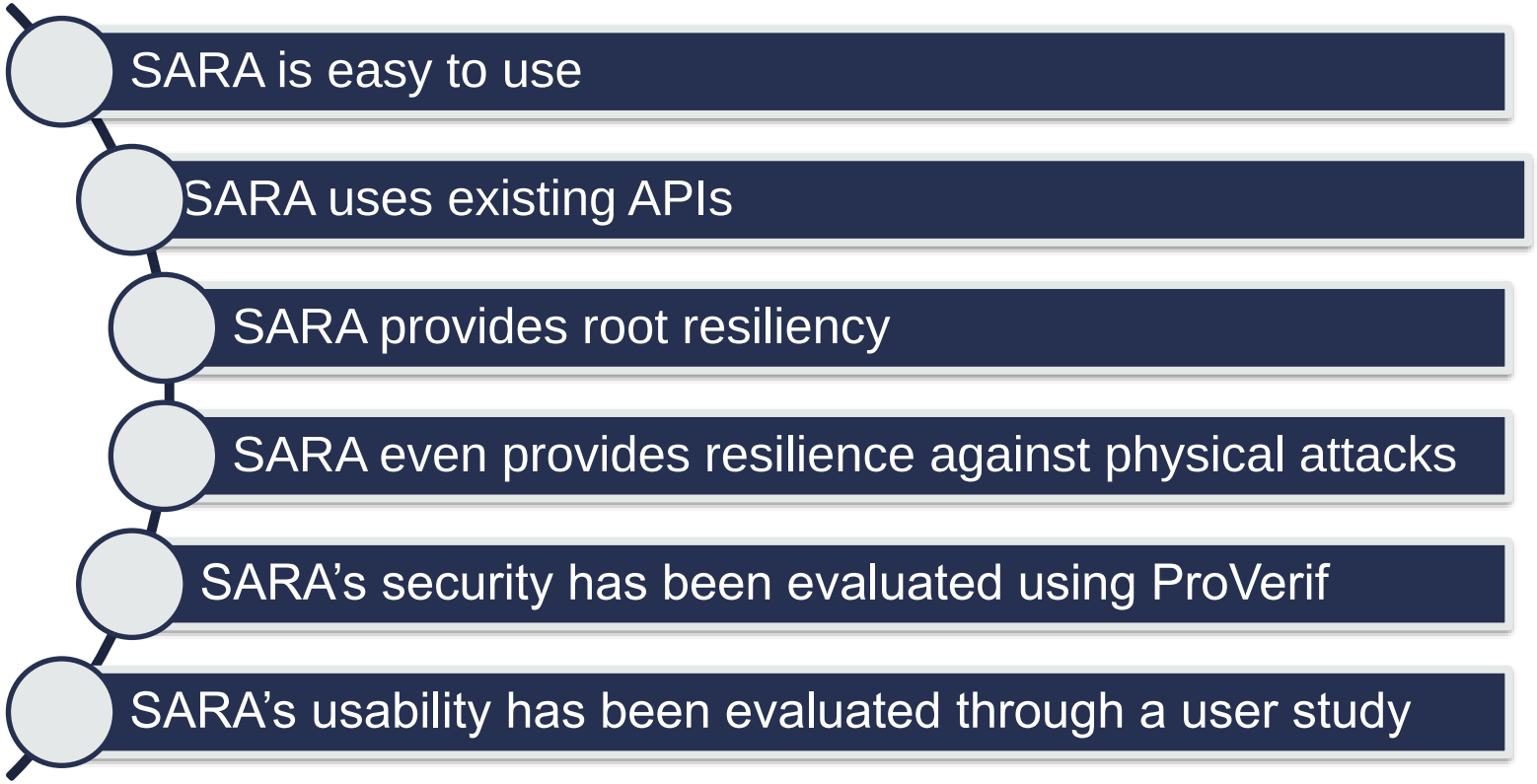
Completion Results

Completed after 105 minutes	Native Task	Library Task
Subtask-1: Successfully created a keypair(s) with the requisite properties	0/14	14/14
Subtask-2: Successfully created the confirmation and biometric prompts	0/14	14/14
Subtask-3: Successfully attested the keypair(s) on the server	0/14	14/14

Evaluation Survey Summary

	SARA	Native API
Positive Experience	93%	0%
Preference of Usage	100%	0%
SUS Score	95.18	11.61

Conclusion


SARA is easy to use
SARA uses existing APIs
SARA provides root resiliency
SARA even provides resilience against physical attacks
SARA's security has been evaluated using ProVerif
SARA's usability has been evaluated through a user study

THANK YOU!!!

Any Questions?

https://github.com/purseclab/SARA-Secure-Android-Remote-Authorization

imran8@purdue.edu