FIGI FINANCIAL INCLUSION GLOBAL INITIATIVE

FIGI Symposium
18 May - 24 June, 2021

New Defenses and Attacks for Smartphones & Implications for DFS

Kevin Butler, Univeristy of Florida

figi.itu.int
#finanicialinclusion

Organized by

Committee on Payments and Market Infrastructures | BIS

THE WORLD BANK IBRD • IDA

ITU

Android Operating System

- Android is the world's most popular operating system with billions of users throughout the world
- Both the operating system and the devices that run it are increasing in security with every subsequent release
- Attacks and attacker capabilities are also increasing
- In this talk:
  - Discussion of new changes in Android 12 and potential implications for DFS
  - Discussion of access control mechanisms in Android
  - Platform-level attacks against Android devices

FIGI FINANCIAL INCLUSION GLOBAL INITIATIVE

Android 12 Changes and DFS

- Biggest new changes in Android 12 that could affect DFS are power management restrictions and changes to location privacy

- Android 12 has new power management restrictions in apps
  - If an app is not being regularly used, its schedule for running will be reduced
- What are the implications on notifications of transactions?

Message Changes in Android 12

- If power management restrictions are active, applications will receive five "high-priority" Firebase Cloud Messaging (FCM) messages
- However, further messages may be delayed if the app is in power saving mode
- *Developers:* Need to consider what this means for users who might not often open their apps
  - Will they notice if transactions affecting their account are occurring?
  - Research and user testing should be done in advance of these changes

Location Privacy in Android 12

- Android 12 has a much greater focus on user privacy than any previous version of the OS
- As part of this, users have a choice of the granularity of location reported to apps regardless of what particular permissions might be set
- If your DFS app relies on locating transactions with **ACCESS_FINE_LOCATION** permission set, users will still be able to restrict the app to coarse location, even if this is set

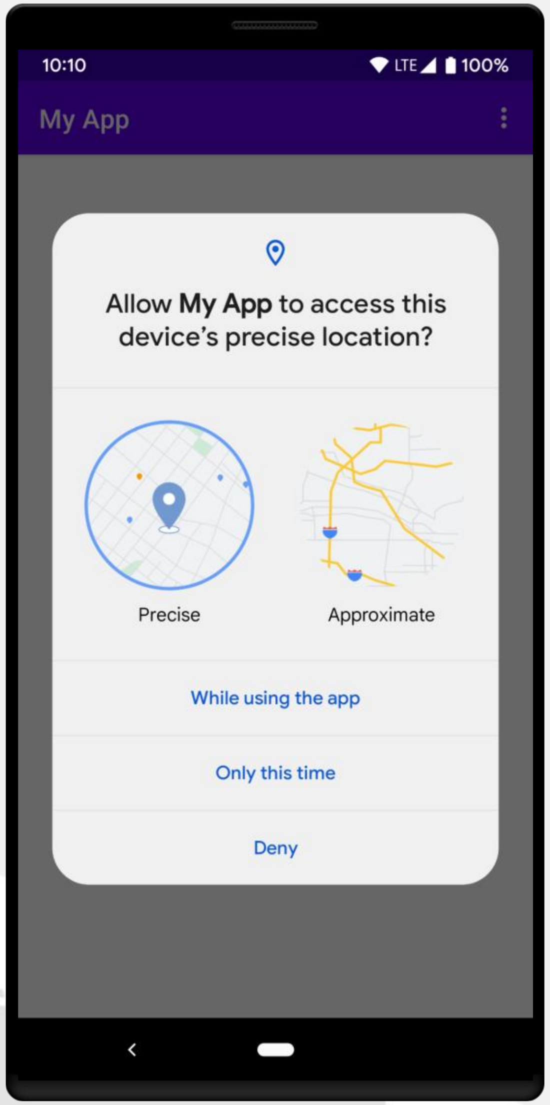


Figure from Android 12 Developer documentation: https://developer.android.com/about/versions/12/approximate-location

Implication of Coarse Location

- From a user privacy perspective, this has great value - deters surveillance activities and fine-grained location tracking (and hence profiling)
- From a DFS perspective, less clear what the drawback might be
  - e.g., if app relies on fine-grained location for determining that a transaction is happening between two registered users in the same location as a fraud-prevention mechanism

- *Developers:* need to think closely about this setting if apps are relying on validating fine-grained location between parties

Android Access Control

- Android has a wide variety of ways to protect access to information
- At the app level: Android permissions set within the Android middleware
- At the platform level: *discretionary* access control (Unix-style permissions) and *mandatory* access control (SELinux mechanisms) within user and kernel space for the underlying Linux kernel
  - Linux capabilities are also supported

FIGI FINANCIAL INCLUSION GLOBAL INITIATIVE

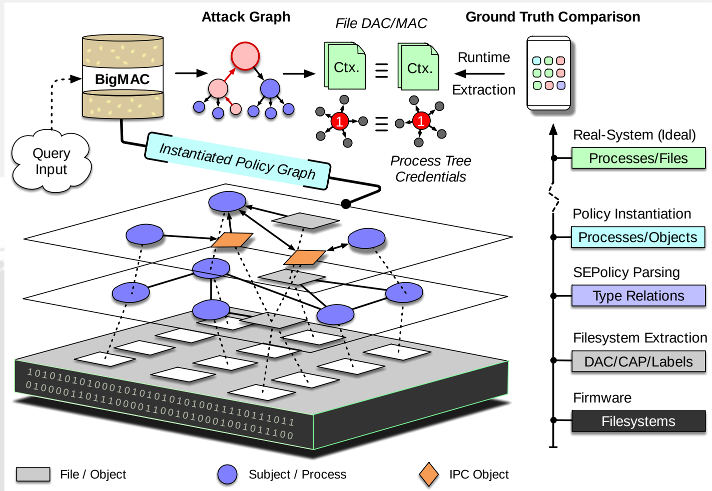
Reasoning About Access Control

- Reasoning about how these various access control mechanisms work together and their implications on programs can be challenging
- There are potential vendor issues with SELinux configuration that can lead to vulnerabilities
  - negation overuse that can allow policy circumvention
  - large policies that are inefficient and difficult to reason about
  - allowing of debugging features in production

FIGI FINANCIAL INCLUSION GLOBAL INITIATIVE

DFS Developer Perspective

- Developers should be aware of the underlying platforms that they are developing for and to be aware of any vulnerabilities raised due to access control issues
- Currently an area of research in the academic community
  - e.g., our research on **BigMAC**, a system that allows querying of policies based on the actual firmware of deployed devices that incorporates mandatory and discretionary access controls and Linux capabilities
  - findings; some weaknesses in vendor SELinux policies (disclosed to vendors)
  - more information: https://www.usenix.org/conference/usenixsecurity20/presentation/hernandez



figi.itu.int
#financialinclusion

FIGI FINANCIAL INCLUSION GLOBAL INITIATIVE

Trusted Execution Environments

- Trusted execution environments (TEEs) can be very beneficial for developers to use, and are recommended as best practice for storing sensitives information in the Digital Financial Services security assurance framework
  - Best practices involves using services such as the Android Keystore for cryptographic keys and the keychain API
- TEEs run in their own enviornment isolated from other activities on the mobile device
  - e.g., on ARM processors, the Android OS runs in the "normal world" and the TEE runs in the "secure world" configured by hardware through TrustZone
  - separate OS used for TEEs (e.g., Trusty TEE for AOSP, Qualcomm QSEE, Trustonic Kinibi)

Threats Against Trustlets

- Implementation of secure world code has to be done very carefully
- The limited operating environment of trusted OSes means less support for software security mechanisms (e.g., no ASLR)
- Trusted applications (TAs, or Trustlets) are designed to run in the TEE, but flaws in any of them can put the entire TEE at risk

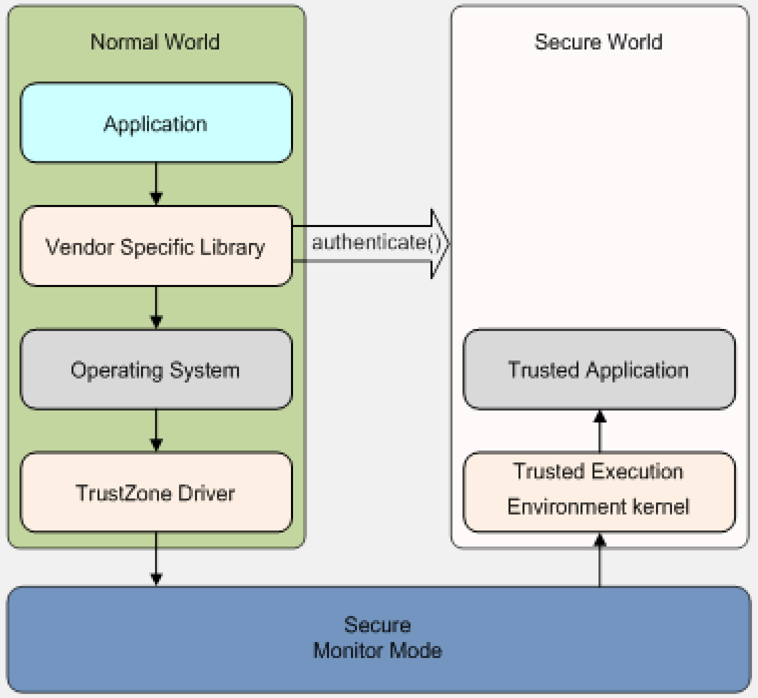


Figure from ARM Developer documentation: https://developer.arm.com/documentation/100935/0100/Interaction-of-Normal-and-Secure-worlds-?lang=en

Testing Trustlet Security

- **PARTEMU** work from Samsung Research (https://www.usenix.org/conference/usenixsecurity20/presentation/harrison) focused on fuzz testing TAs on emulated firmware platforms from 12 different smartphone vendors
- discovered 48 previously-unknown vulnerabilities, some of which were exploitable (e.g., privilege escalation, arbitrary code execution)
- Often caused by developer mistakes
  - assumptions about coding practices that differ in the secure world
  - not validating information received from the normal world)

FIGI FINANCIAL INCLUSION GLOBAL INITIATIVE

Speculative Execution

- Meltdown and Spectre attacks are possible against mobile processors
- Branch prediction pre-computation can be a source for creating "side channels" that lead to understanding of program state and stealing data
- Mitigations exist in Android but a particular vulnerability (Straight-Line Speculation) exists for ARM processors
  - Challenging to mount but dangerous if they are successful
  - Mitigations have been added to compilers by ARM
- Speculative execution attacks are an area of great current academic interest

Advice to DFS Developers

- Many of these new attack vectors are experimental and have not been widely deployed, but concerns are being addressed by industry and academic
- Importantly: **keep adhering to best practices!**
  - Use Keystore and trusted hardware environments when they exist on platforms
  - Exercise least privilege and make use of access control mechanisms
- But: security is a moving target
- *Keep aware* of what is happening in the community and ensure that your applications are responsive to changes
- **Always be proactive!**

Thank You

- Questions?
  - Email: butler@ufl.edu
  - Twitter: @kevinrbbutler