ACM CCS26 - DOCTORAL SYMPOSIUM

Systematic Security and Privacy Auditing of the Android Ecosystem

via Automated System-Level Re-Hosting

Thomas Sutter

Supervision: Timo Kehrer, Marc Rennhard, Bernhard Tellenbach
University of Bern | ZHAW | CYD Campus | Armasuisse W+T









Whoami

PhD Student Computer Science – Software Engineering Group, University of Bern
Research Associate - Information Security Group, Zürich University of Applied Sciences



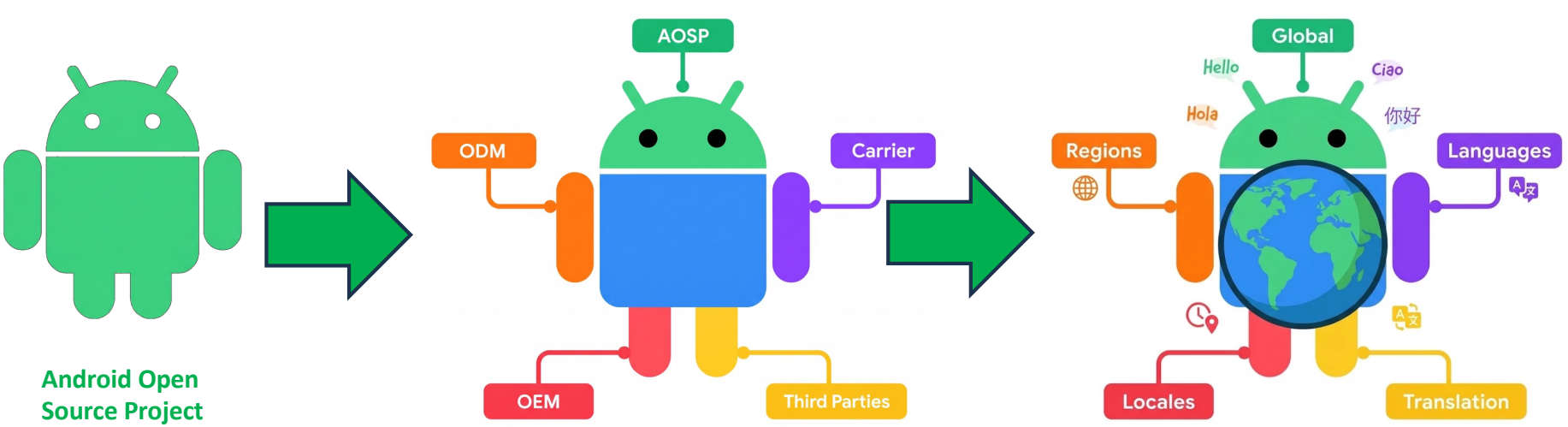
Thomas Sutter

- I work at the intersection of Software Engineering and Security
- Focus on Mobile Security
- Re-Hosting & Software Testing



The Problem

Android's Software Product Line



Android Open Source Project

Fragmentation makes the security analysis often challenging

The Fragmentation of the Supply Chain

Proprietary modifications create massive visibility gaps.

- **Multi-Tier Injections** OEMs and carriers embed closed-source applications into read-only partitions.
- **Privileged Execution** Apps inherit dangerous permissions directly from the factory image.
- **Undeletable Code** Users cannot remove or easily monitor these embedded vendor components.

~27–1,000

Pre-installed applications per device.

The vast majority are unavailable on public app stores.

The Auditing Bottleneck

Current dynamic analysis methodologies fail at ecosystem scale.



Literature Review Gap

Systematic review of publications reveals dynamic security analysis on pre-installed apps is heavily ignored.



Static Analysis

Static analysis suffers high false-positive rates and fundamentally misses complex runtime behaviors.

Dynamic Analysis

- Usually done on a small set of devices or crowed-sourced datasets.
- Emulation based-on x86



Hardware Constraints

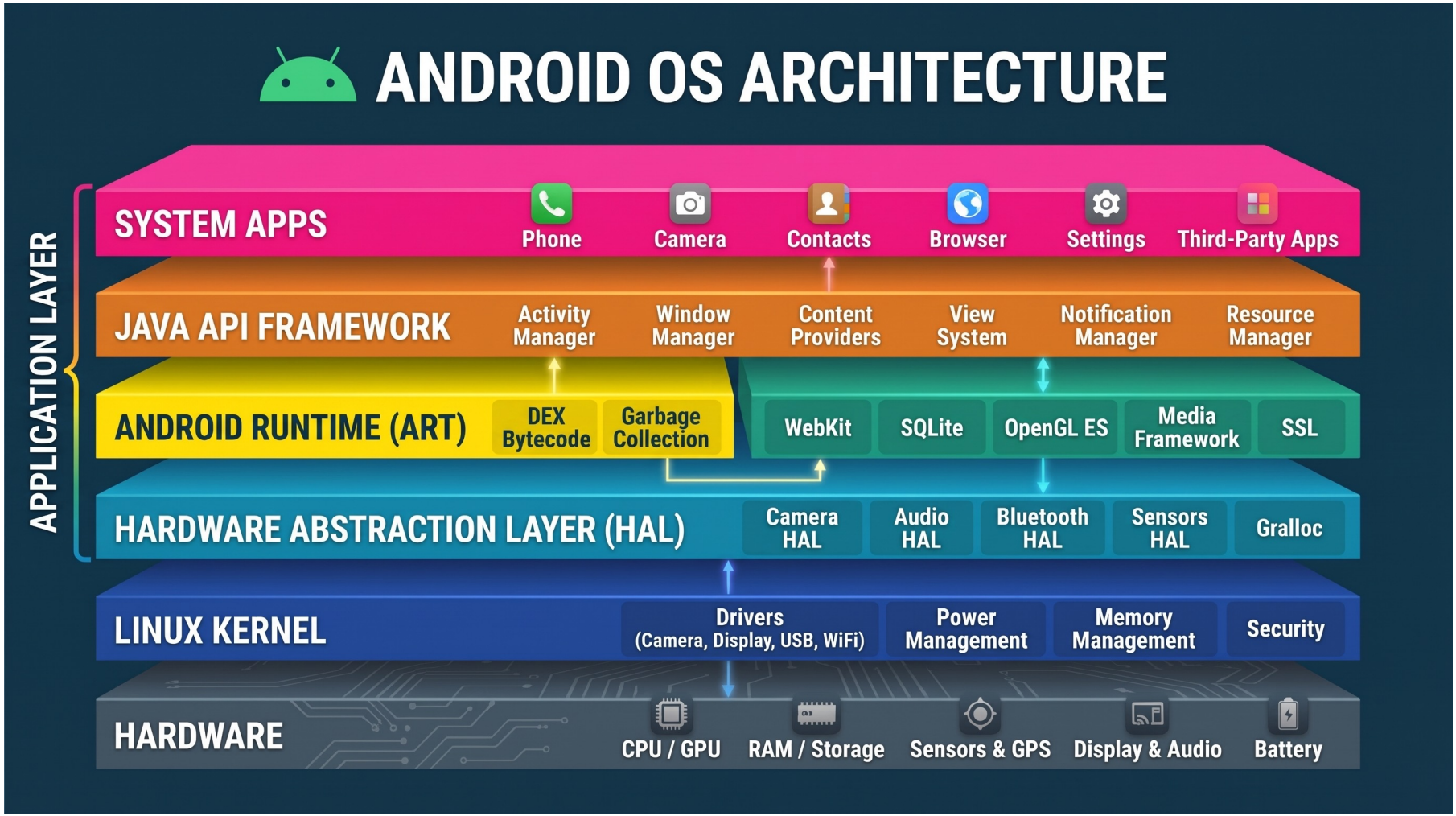
Most studies use only a small set of devices for testing.

Devices are often locked by default.

Reproducing results is often not possible.

Mobile devices have limited hardware ressources (e.g., RAM)

The Idea


ANDROID OS ARCHITECTURE
APPLICATION LAYER
SYSTEM APPS
Phone
Camera
Contacts
Browser
Settings
Third-Party Apps
JAVA API FRAMEWORK
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Resource Manager
ANDROID RUNTIME (ART)
DEX Bytecode
Garbage Collection
WebKit
SQLite
OpenGL ES
Media Framework
SSL
HARDWARE ABSTRACTION LAYER (HAL)
Camera HAL
Audio HAL
Bluetooth HAL
Sensors HAL
Gralloc
LINUX KERNEL
Drivers (Camera, Display, USB, WiFi)
Power Management
Memory Management
Security
HARDWARE
CPU / GPU
RAM / Storage
Sensors & GPS
Display & Audio
Battery

PhD Vision

What I want to achieve with my research:



Objectives

- **A Scalable Re-Hosting Framework**
  - Focus on the Application Layer
  - Focus on Android mobile phones (ARM)
  - Support for major vendors
  - Cloud-based virtual instances
  - Robust and applicable for security use-cases
- **Systematic Ecosystem Insights**
  - Empirical insights into what pre-installed apps communicate over HTTP/HTTPS.
  - Accurate real-world measurements.

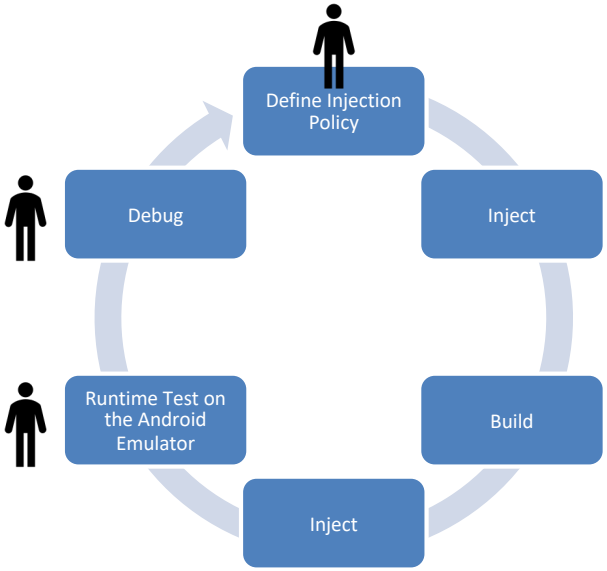


Physical Device (ARM) → ARM to ARM → **Virtual Device (ARM)**

Re-Hosting in a Nutshell

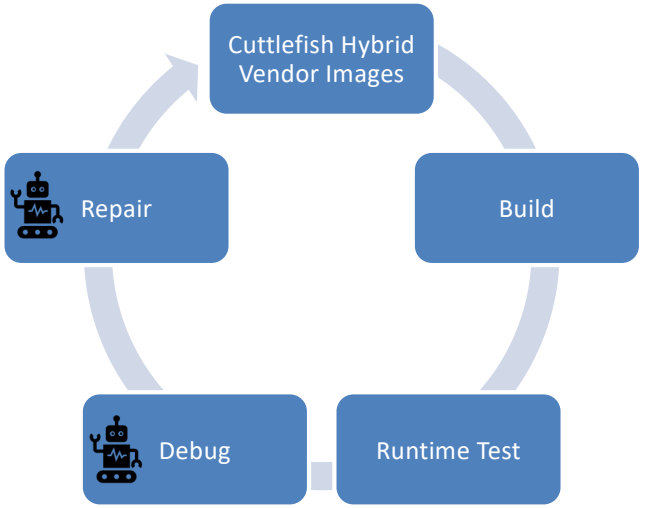
Re-Hoster Prototype

Policy-based ReHoster (v1)
(structual injection)



Removing the human

Repair-based ReHoster (v2)
(semantic injection)



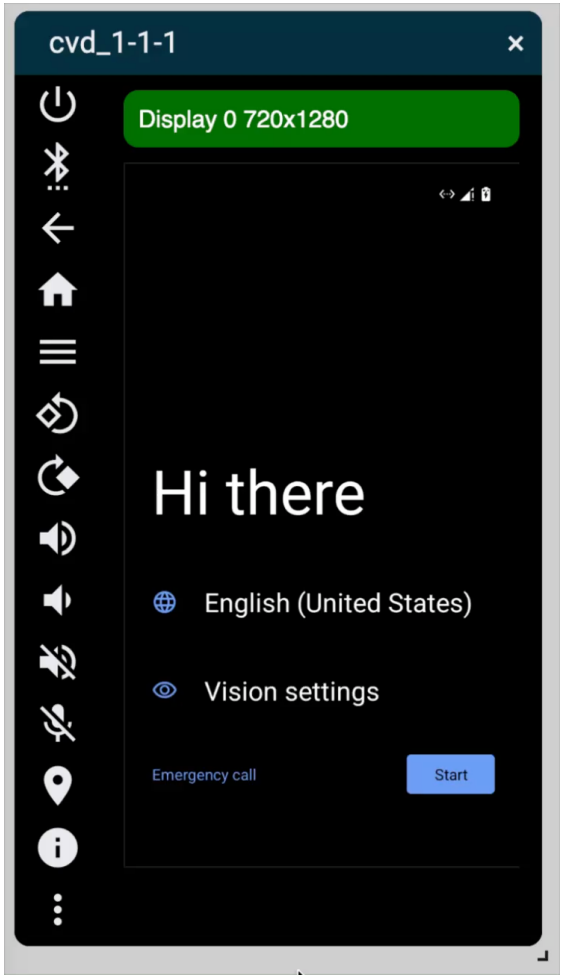
- Works to some extend but iterations are slow and error prone
- Generic policy are unlikely to work

- First tests show significant improvements
- **Quality assurance** is more challenging
  - **<u>Deterministic changes possible?</u>**

Demo

- Re-Hosting
  - Application Layer
    - System.img
    - Product.img
    - Odm.img
    - Vendor.img (not yet) -> Hard Part

- Example Code Repairs:
  - ASUS: Custom commands in init.rc
    - Dmount
  - Nothing:
    - Signing NFC app with platform key
      - Default is the nfc key
  - Samsung:
    - HAL for Batery
      - Stubbed peripherals



Fairphone example

Results so far

Vendor	Result	Image
ASUS	Succeeded	chd-img-e24c6b53-a1.zip
Nothing	Succeeded	chd-img-1a36164d-a1.zip
Fairphone	Succeeded	chd-img-26e20194-a1.zip
Xiaomi	Succeeded	chd-img-a6b359fe-a1.zip
OnePlus	Succeeded	chd-img-52384939-a1.zip
OPPO	Succeeded	chd-img-a279e178-a1.zip
Samsung	Not rehostable this way	none
Google, Motorola, realme	Skipped (Android 16)	none

PhD Mission

Migrating proprietary vendor components into virtualized environments to allow automated security testing

1

Dataset

Collected a dataset with around ~30TB of Android mobile firmware: https://vault.cloudlab.zhaw.ch

2

Extraction

Handling all the custom formats to get the actual binaries and meta-data: https://firmwaredroid.github.io/

3

Structural Re-Hosting

Deploy generated images to scalable, fully controlled virtual device instances.

4

Runtime Testing

Execute the pre-installed apps. Monitor network traffic and code coverage.

5

Repair-Based Re-Hosting

Advance our re-hosting method with semantic merging and code repair.

6

Runtime Testing v2

Testing some typical security use-cases.

What's next…

Device Extension

- TV Boxes:
  - https://firmwaredroid.github.io/case-studies/android-tv-boxes/
- Android Automotive
- Android IoT Devices (e.g., Drones)
- …

Peripherals Simulation

Automated simulation or emulation of expected peripherals:

- Networking (modems, wifi, bluetooth, …)
- Sensors (biometrics, GPS, thermal, …)
- …

Security Use-Cases

- Network Monitoring (Privacy)
- Reproducing Vulnerabilities
- Permission / Authorization
- Attestation
- Fuzzing
- Compliance Testing
- Malware Testing

Your idea?

Open for your suggestions

Thank You

Questions & Discussion

Thomas Sutter
thomas.sutter@unibe.ch
https://7homassutter.github.io/