Profile Coverage: Using Android Compilation Profiles to Evaluate Dynamic Testing

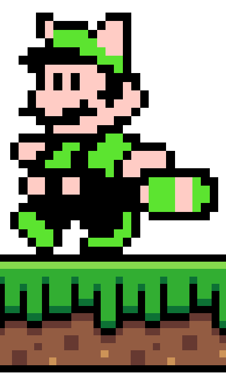
Jakob Bleier
jakob.bleier@seclab.wien

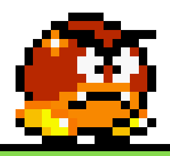
Felix Kehrer
felix.kehrer@seclab.wien

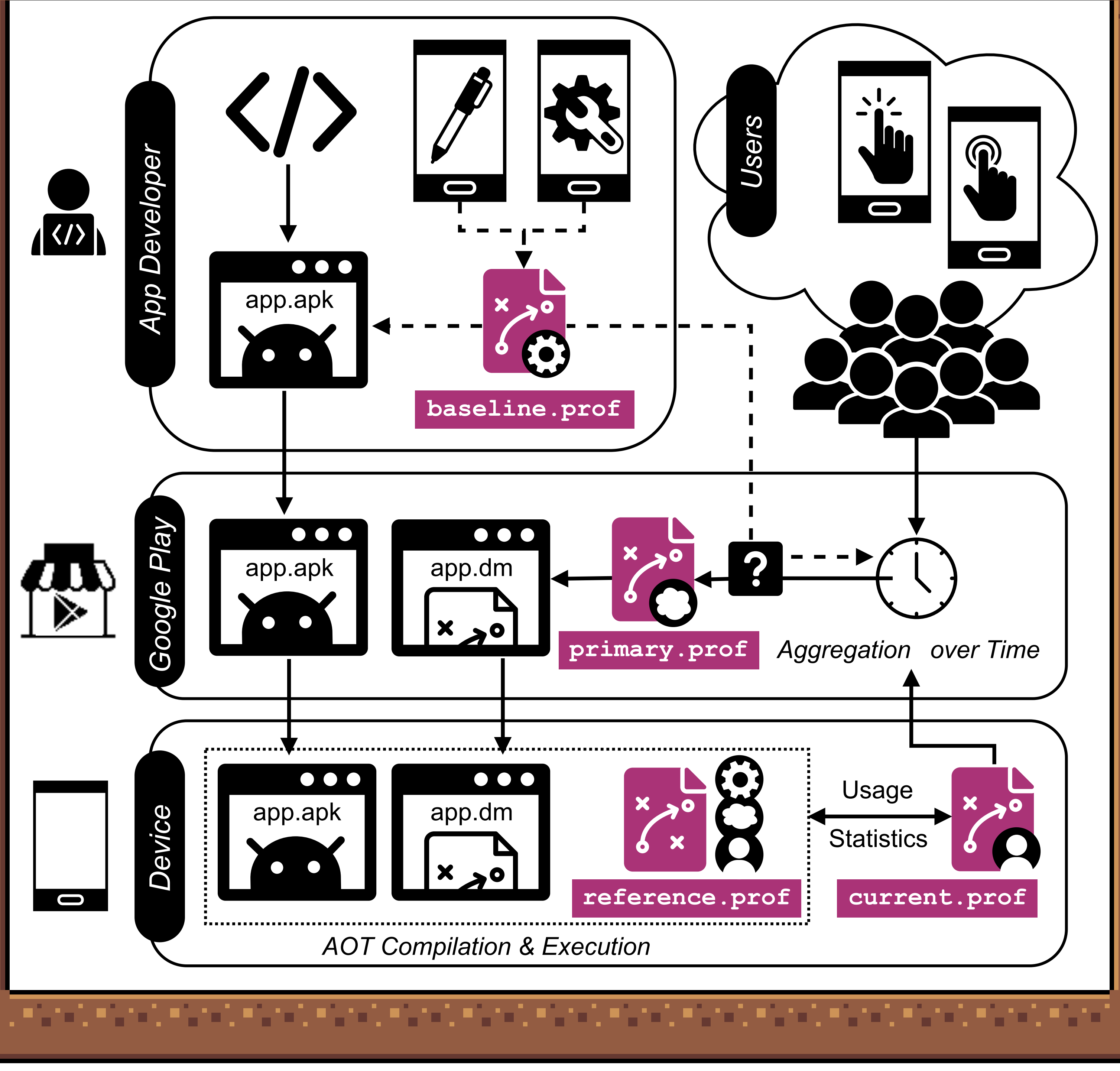
Jürgen Cito
juergen.cito@tuwien.ac.at

Martina Lindorfer
martina@seclab.wien

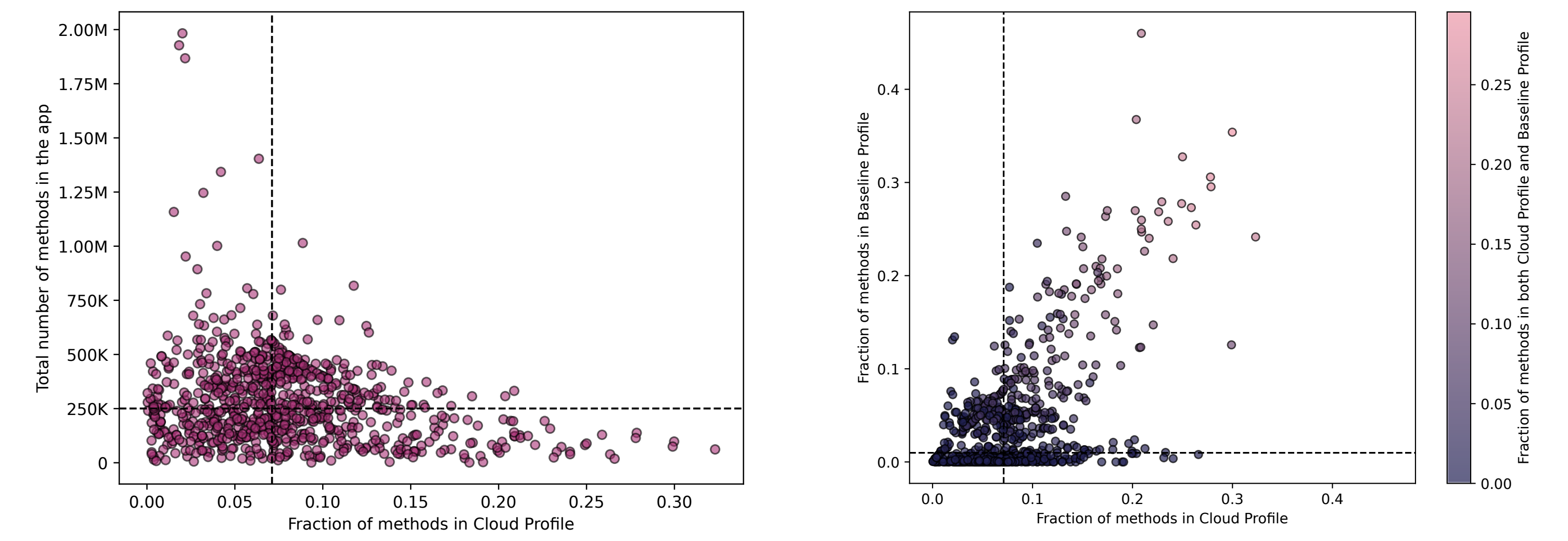
Cloud Profiles



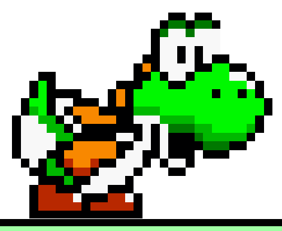




Cloud Profiles (saved in `primary.prof`) contain ***aggregated usage information*** from Play Store users. As one of the compliation profiles on Android, their ***list of "hot" methods*** informs the compiler which parts of an app are ***compiled ahead-of-time***, e.g., during installation. They are available for virtually ***all popular apps*** and often do not match ***developer-provided Baseline Profiles***.

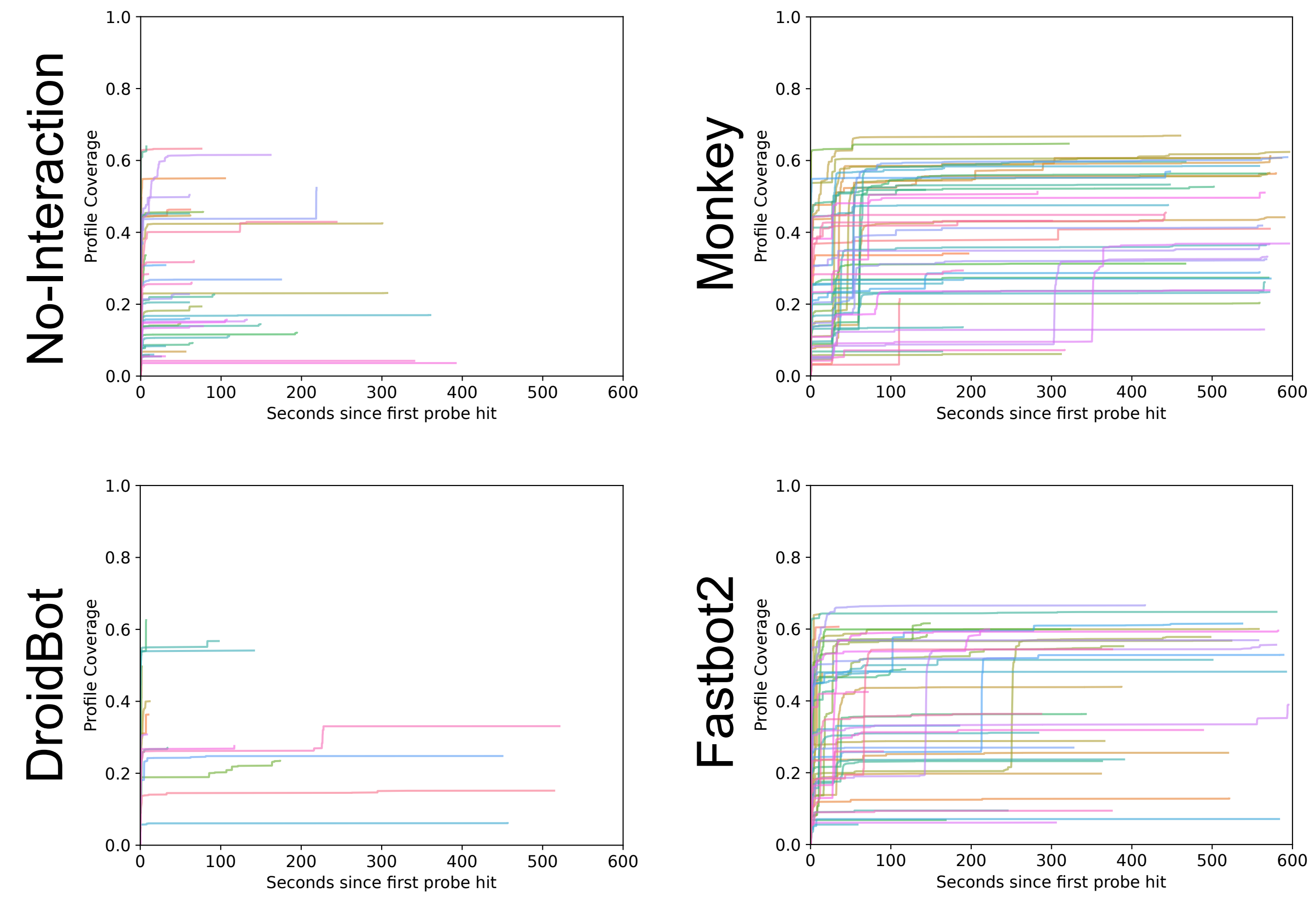


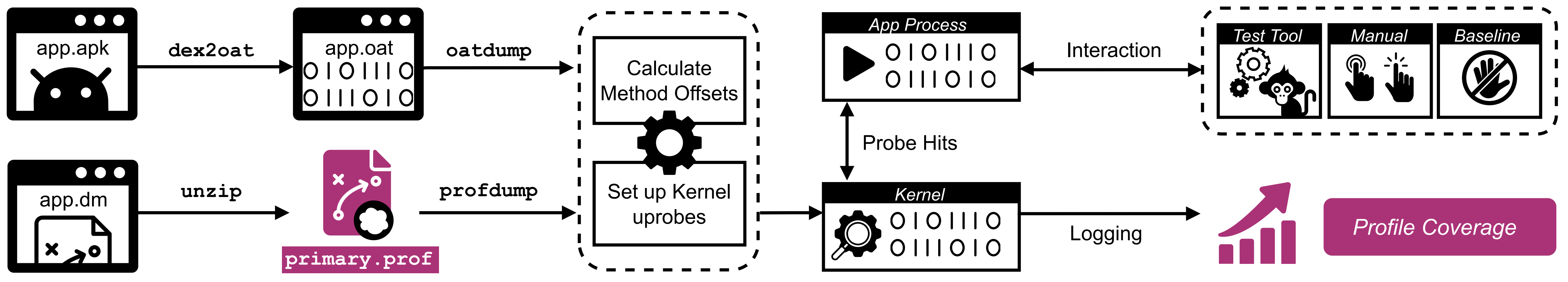
Profile Coverage



Profile Coverage is the ratio of a (cloud) profile's number of executed methods to the profile's overall number of methods. It quantifies how well dynamic or manual testing ***covers aggregated usage*** data. A low score indicates that core functions commonly used were not executed during testing.

We compared a ***No-Interaction baseline***, ***Monkey***, ***DroidBot***, and ***Fastbot2*** using our novel uprobe-based method tracer. Surprisingly, Monkey's ***random interaction slightly outperformed*** Fastbot2's model-based approach, even when compared to traditional code coverage using ***acvtool***. No tool achieved complete profile coverage, which motivates future work toward ***enhancing coverage and reducing blind spots***.





Our tracer utilizes Android's AOT compilation and Linux uprobes for tracing methods ***without modifying apps or the system***, and works on emulators and hardware devices.

Code available at github.com/SecPriv/android-profile-tracing#

TECHNISCHE UNIVERSITÄT WIEN Vienna | Austria
WWTF
FWF Österreichischer Wissenschaftsfonds
SBA Research