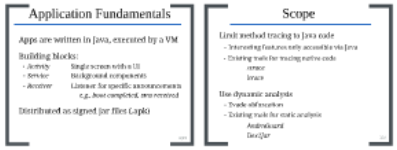
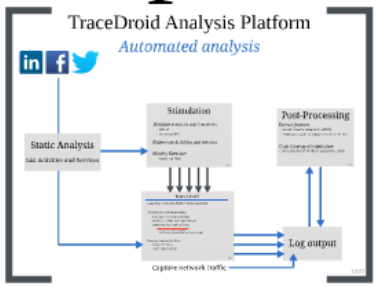
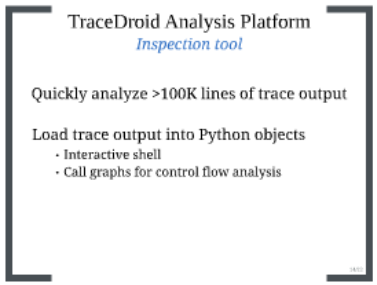
http://tracedroid.few.vu.nl

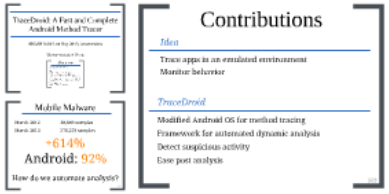
Introduction

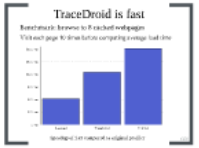


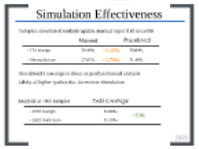
Implementation

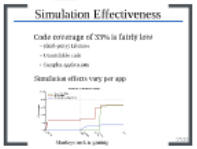












Evaluation

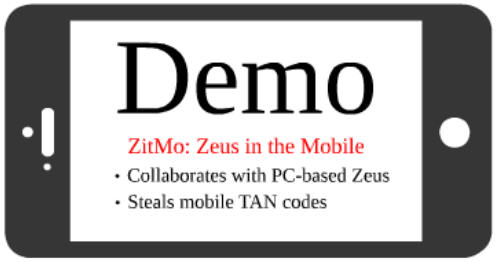
Conclusions

http://tracedroid.few.vu.nl/

Submit your .apk for automated analysis
.tar.gz output containing:
- method traces
- network dump
- call graph

Contact me if you would like to analyze a batch

No source or inspect tool available yet



TraceDroid: A Fast and Complete Android Method Tracer

OWASP BeNeLux Day 2013, Amsterdam

Victor van der Veen

About me

Security Consultant at ITQ

Past

- MSc. in Computer Science, VU University
- Capture the Flag 'hacking' competitions
- Memory Errors: The Past, The Present and the Future (RAID 2012)
- (partial) Implementation of a trustworthy voting machine
- Worked on Andrubis with the iSecLab team in Vienna

Twitter: @vvdveen
E-Mail: vvanderveen@itq.nl

About me

Security Consultant at ITQ

Past

- MSc. in Computer Science, VU University
- Capture the Flag 'hacking' competitions
- Memory Errors: The Past, The Present and the Future (RAID 2012)
- (partial) Implementation of a trustworthy voting machine
- Worked on Andrubis with the iSecLab team in Vienna

Twitter: @vvdveen

E-Mail: vvanderveen@itq.nl

Mobile Malware

March 2012	38,689 samples
March 2013	276,259 samples

+614%

Android: 92%

How do we automate analysis?

Contributions

Idea

Trace apps in an emulated environment

Monitor behavior

TraceDroid

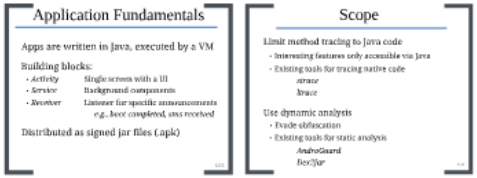
Modified Android OS for method tracing

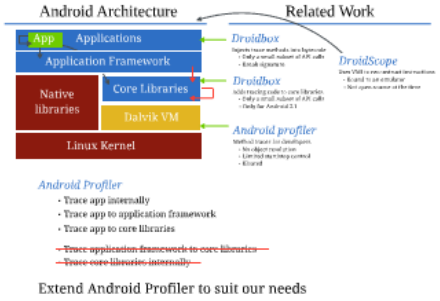
Framework for automated dynamic analysis

Detect suspicious activity

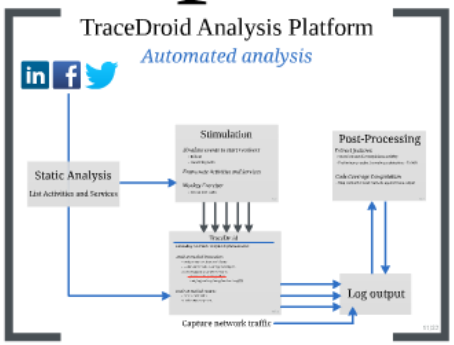
Ease post analysis

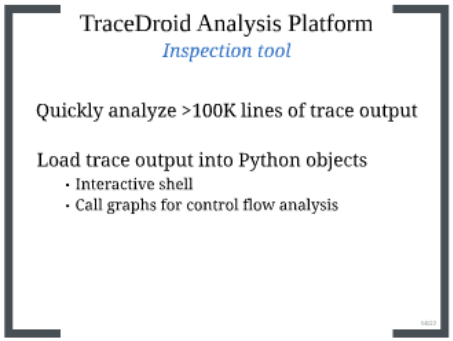
Introduction

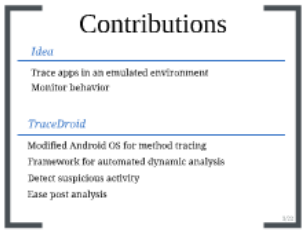


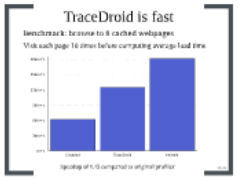


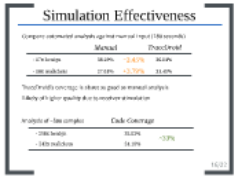
Implementation

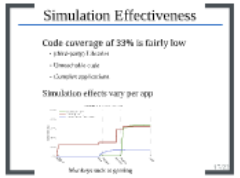






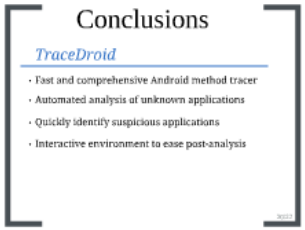






Evaluation

Conclusions



http://tracedroid.few.vu.nl/

Submit your .apk for automated analysis
.tar.gz output containing:

- method traces
- network dump
- call graph

Contact me if you would like to analyze a batch

No source or inspect tool available yet

Demo

ZitMo: Zeus in the Mobile

- Collaborates with PC-based Zeus
- Steals mobile TAN codes

Scope

Limit method tracing to Java code

- Interesting features only accessible via Java
- Existing tools for tracing native code

  strace

  ltrace

Use dynamic analysis

- Evade obfuscation
- Existing tools for static analysis

  AndroGuard

  Dex2Jar

Application Fundamentals

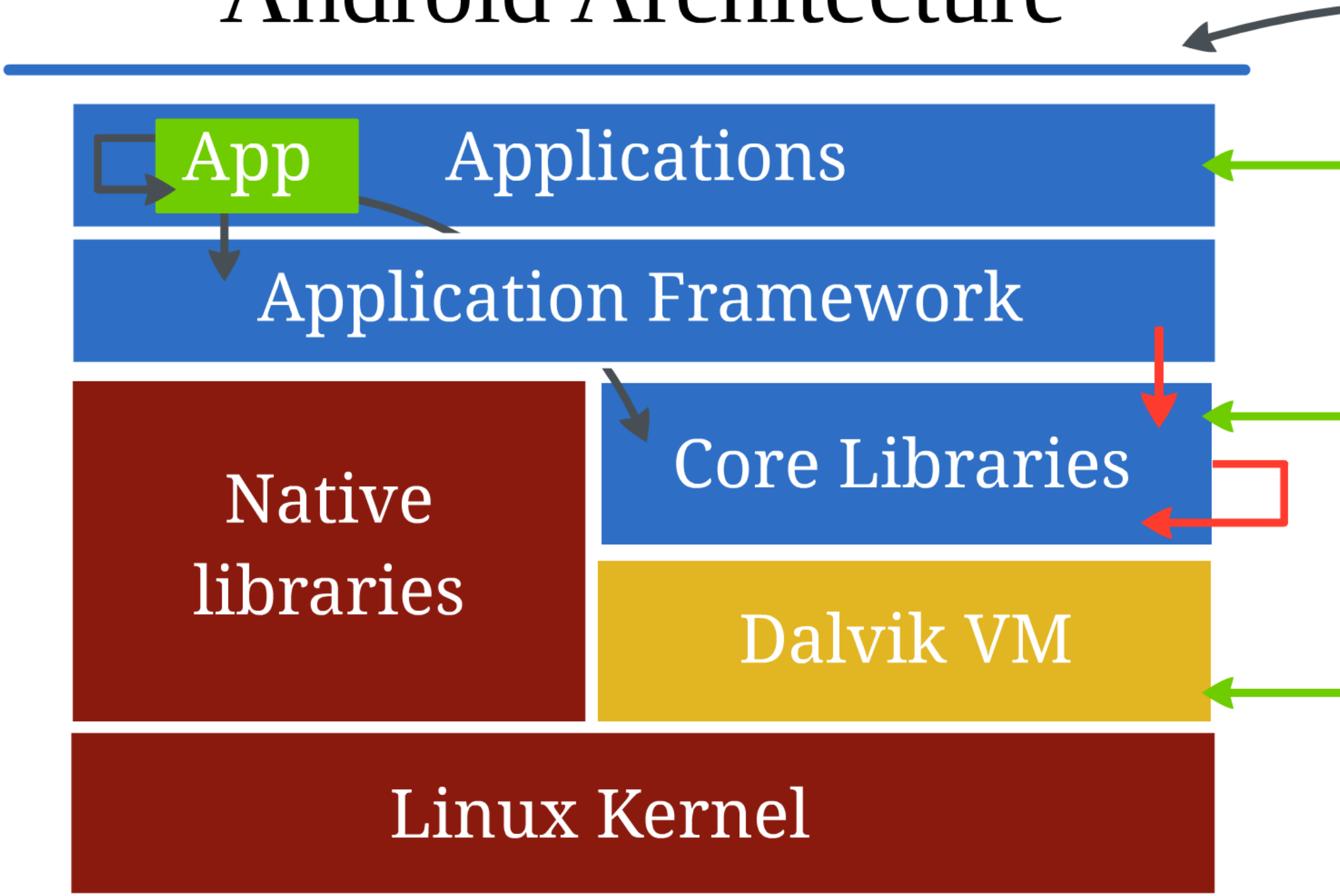
Apps are written in Java, executed by a VM

Building blocks:

- *Activity* Single screen with a UI
- *Service* Background components
- *Receiver* Listener for specific announcements
  e.g., boot completed, sms received

Distributed as signed jar files (.apk)

Android Architecture



Related Work

Droidbox

Injects trace methods into bytecode

- Only a small subset of API calls
- Break signature

Droidbox

Adds tracing code to core libraries

- Only a small subset of API calls
- Only for Android 2.1

Android profiler

Method tracer for developers

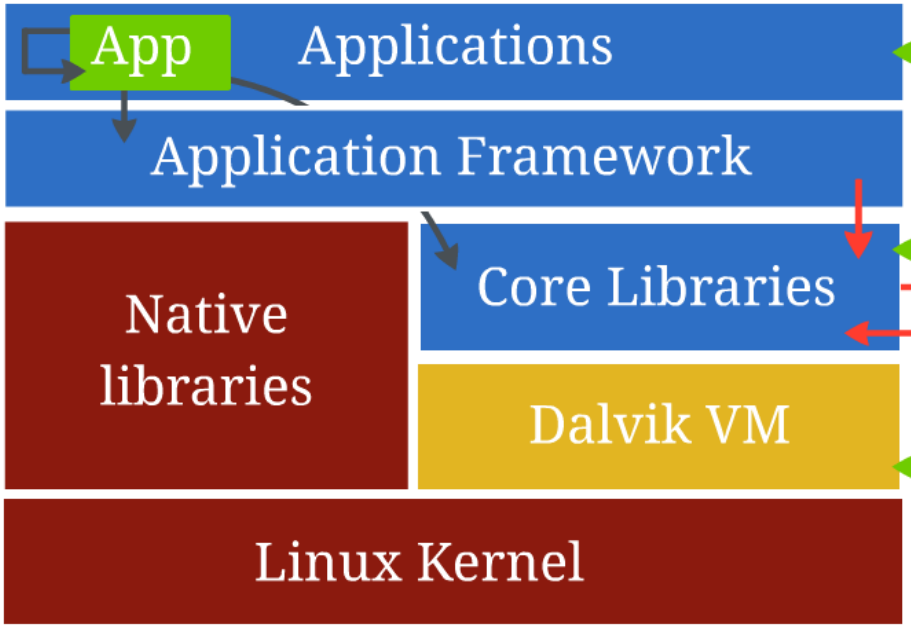
- No object resolution
- Limited start/stop control
- Bloated

DroidScope

Uses VMI to reconstruct instructions

- Bound to an emulator
- Not open source at the time

Android Architecture



Related Work

Droidbox

Injects trace methods into bytecode
- Only a small subset of API calls
- Break signature

Droidbox

Adds tracing code to core libraries
- Only a small subset of API calls
- Only for Android 2.1

Android profiler

Method tracer for developers
- No object resolution
- Limited start/stop control
- Bloated

DroidScope

Uses VMI to reconstruct instructions
- Bound to an emulator
- Not open source at the time

Android Profiler

- Trace app internally
- Trace app to application framework
- Trace app to core libraries

- ~~Trace application framework to core libraries~~
- ~~Trace core libraries internally~~

Extend Android Profiler to suit our needs

TraceDroid

Extending Android's Profiler implementation

Hook on method invocations

- Fetch parameters from stack frame
- Lookup and invoke .tostring() for Objects
- Convert signatures and descriptors

~~*foo(Ljava/lang/String;Z[[J)V*~~

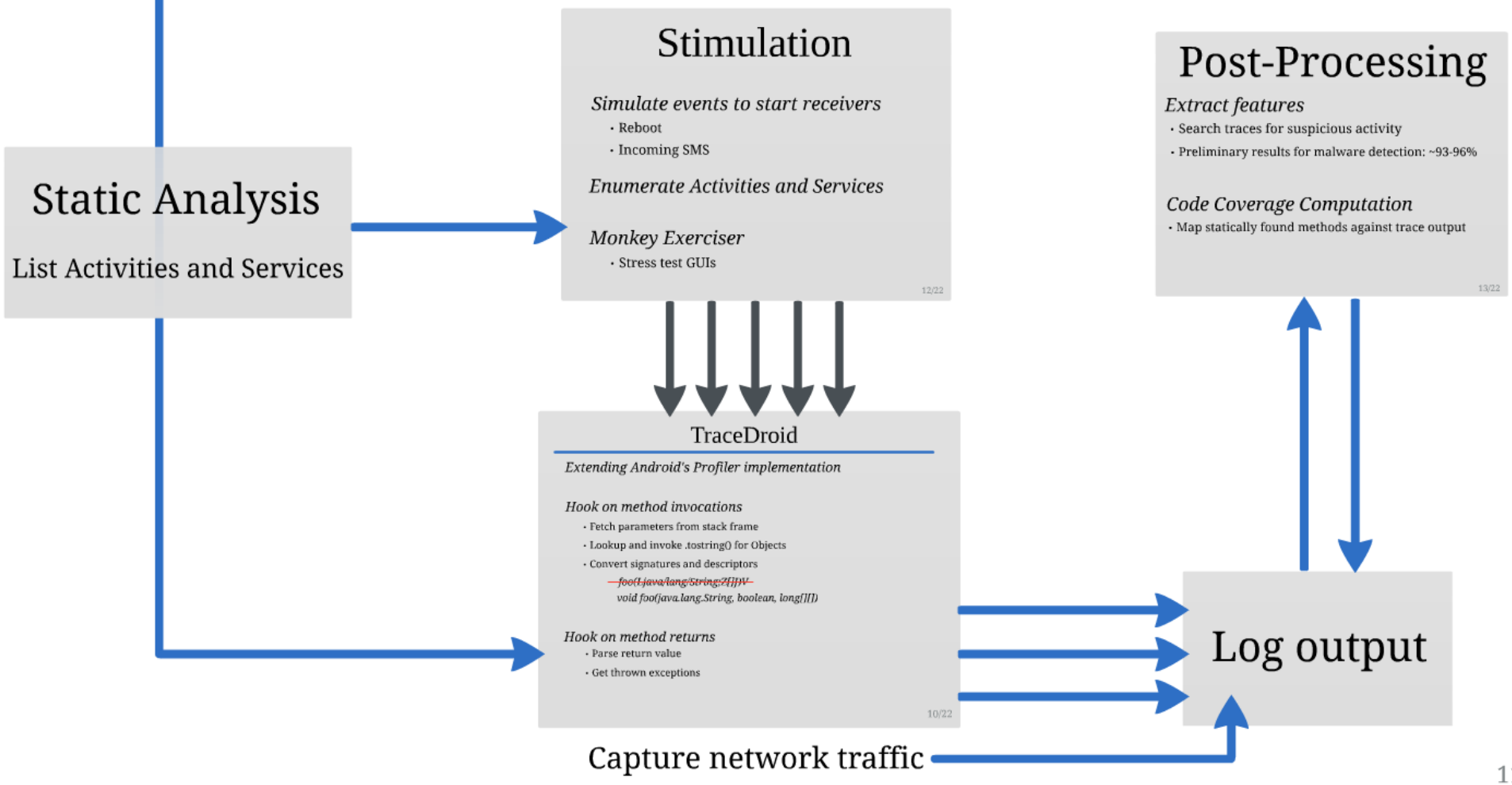
void foo(java.lang.String, boolean, long[][])

Hook on method returns

- Parse return value
- Get thrown exceptions

TraceDroid Analysis Platform

Automated analysis



Stimulation

Simulate events to start receivers

- Reboot
- Incoming SMS

Enumerate Activities and Services

Monkey Exerciser

- Stress test GUIs

Post-Processing

Extract features

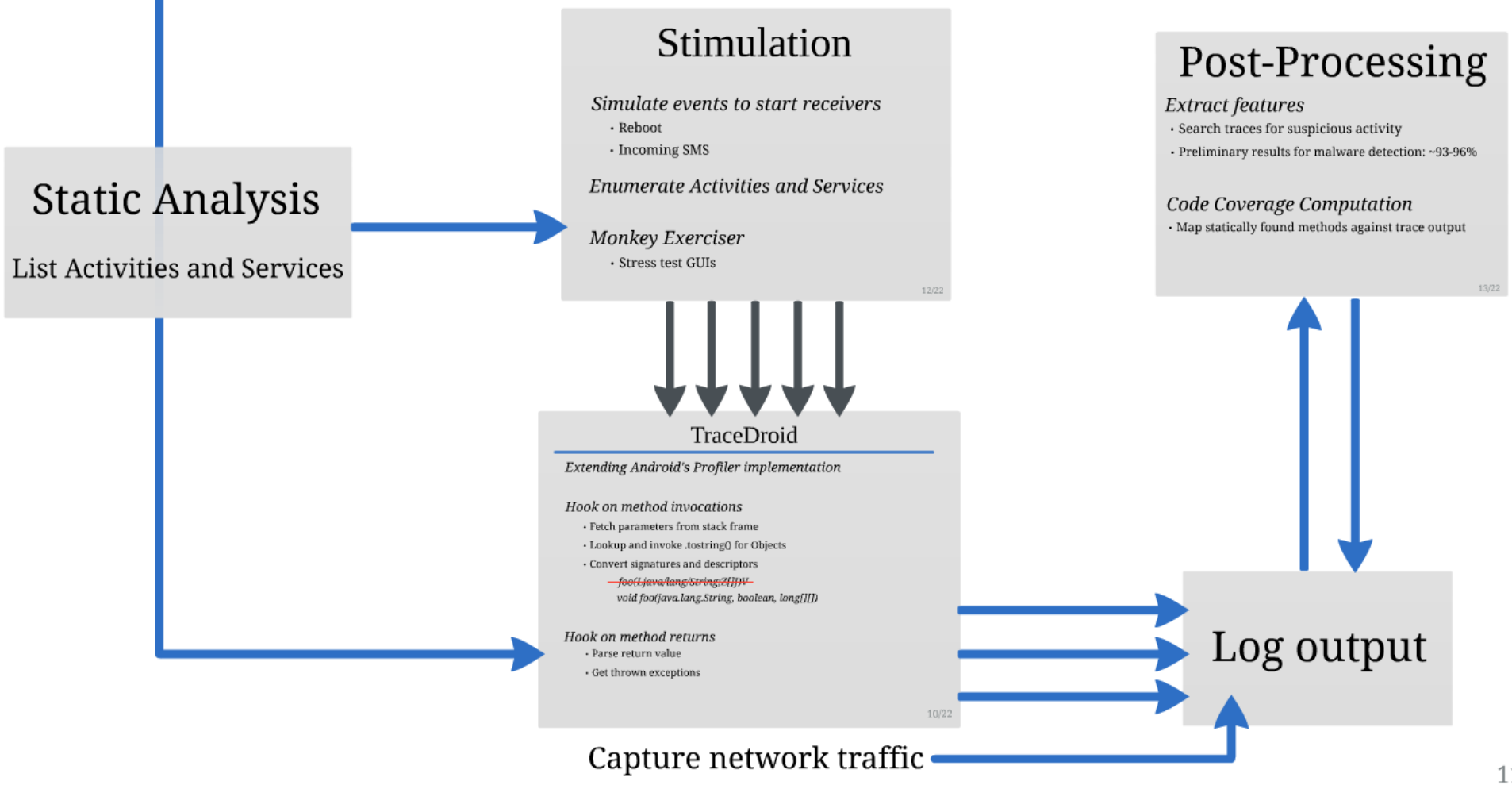
- Search traces for suspicious activity
- Preliminary results for malware detection: ~93-96%

Code Coverage Computation

- Map statically found methods against trace output

TraceDroid Analysis Platform

Automated analysis



TraceDroid Analysis Platform

Inspection tool

Quickly analyze >100K lines of trace output

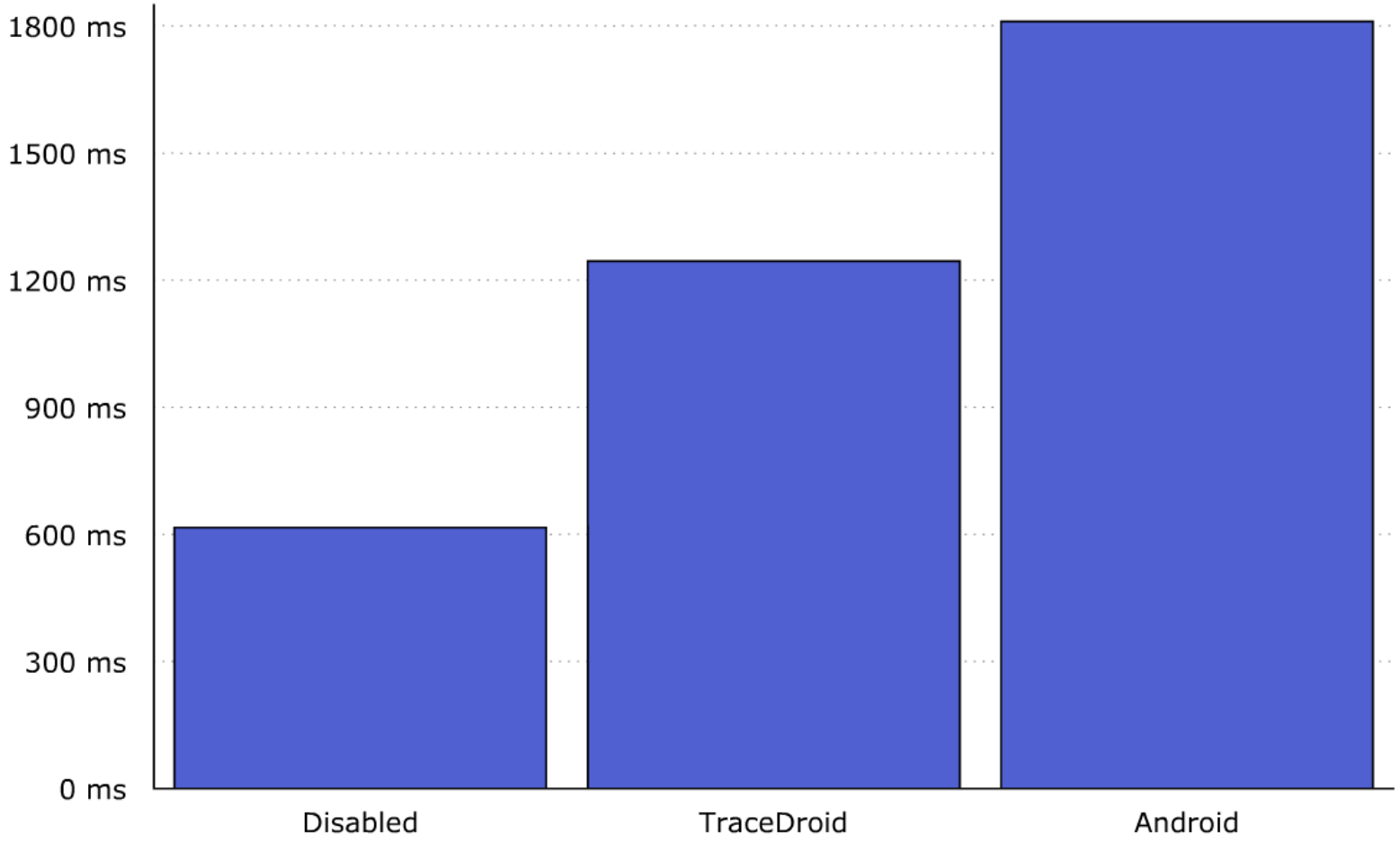
Load trace output into Python objects

- Interactive shell
- Call graphs for control flow analysis

TraceDroid is fast

Benchmark: browse to 8 cached webpages

Visit each page 10 times before computing average load time



Speedup of 1.45 compared to original profiler

Simulation Effectiveness

Compare automated analysis against manual input (180 seconds)

	Manual		*TraceDroid*
• 17x benign	38.49%	−2.45%	36.04%
• 18x malicious	27.61%	+3.79%	31.40%

TraceDroid's coverage is about as good as manual analysis

Likely of higher quality due to receiver stimulation

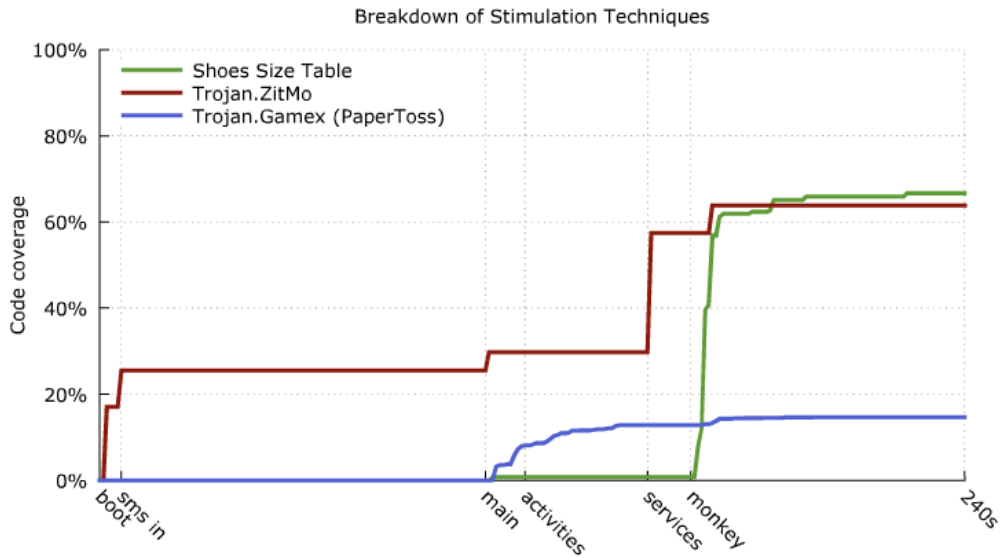
Analysis of ~500 samples	*Code Coverage*	
• 250x benign	35.02%	~33%
• 242x malicious	31.10%	

Simulation Effectiveness

Code coverage of 33% is fairly low

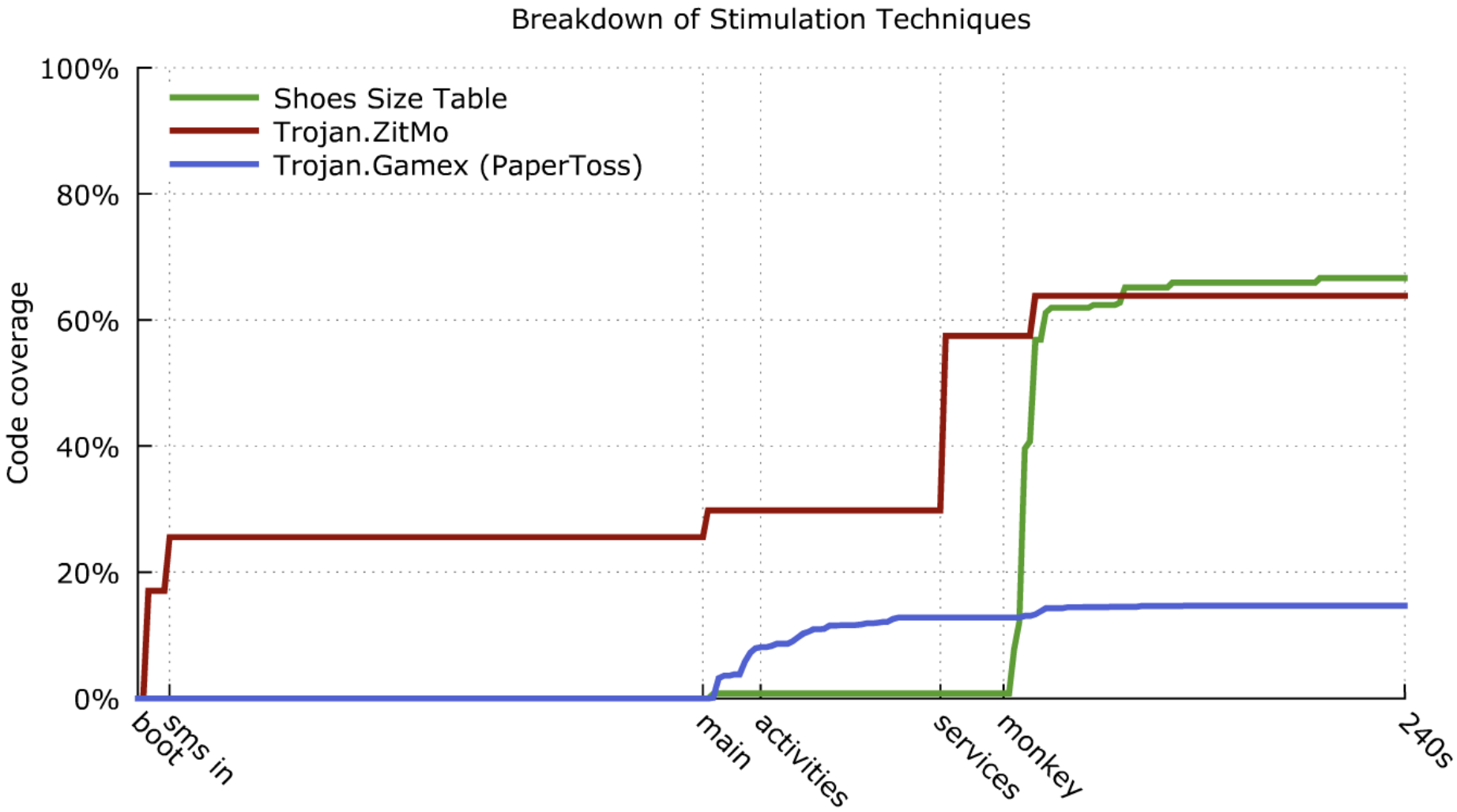
- (third-party) Libraries
- Unreachable code
- Complex applications

Simulation effects vary per app

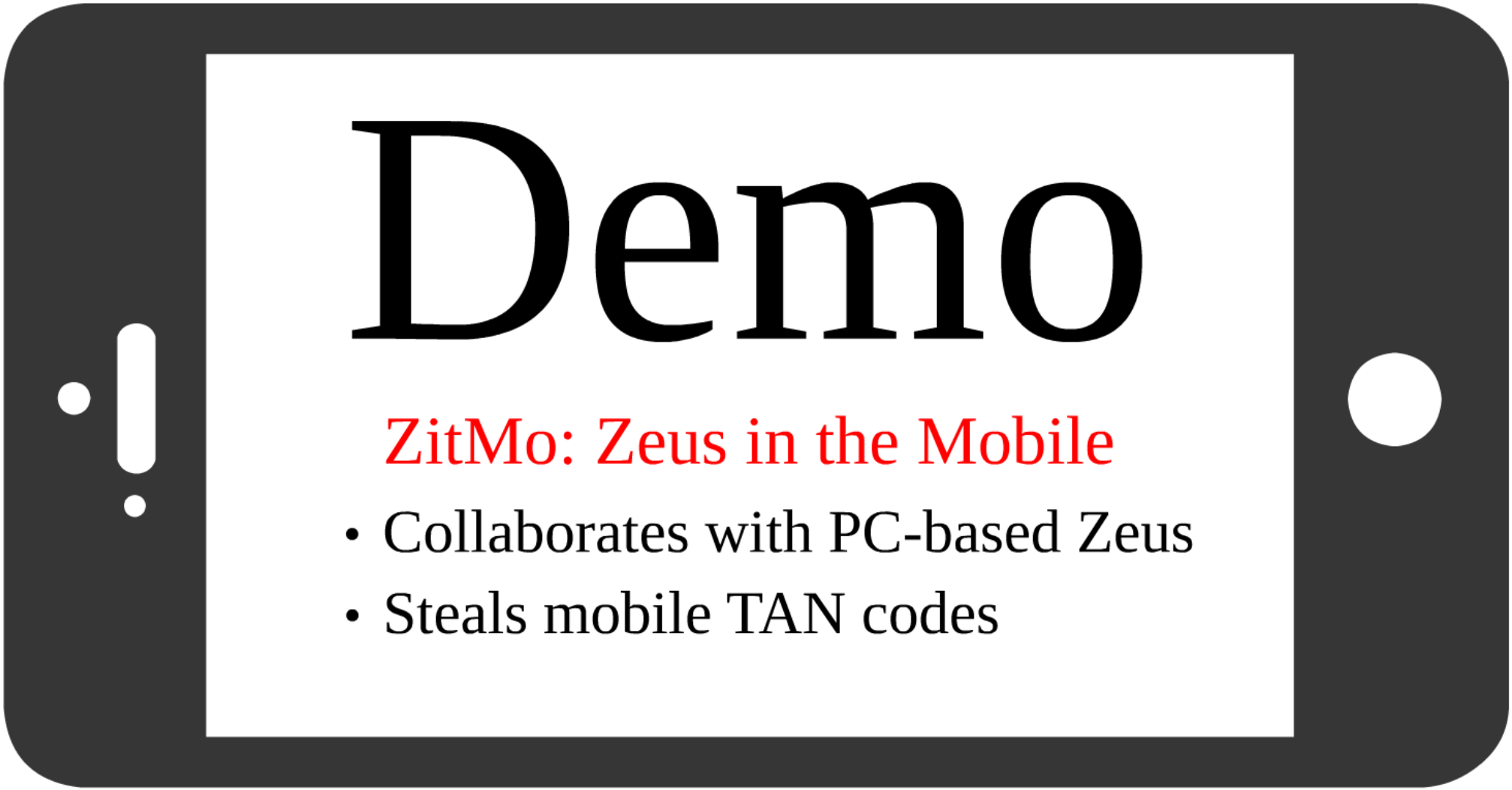


Monkeys suck at gaming

Simulation effects vary per app



Monkeys suck at gaming


Demo
ZitMo: Zeus in the Mobile
• Collaborates with PC-based Zeus
• Steals mobile TAN codes

Conclusions

TraceDroid

- Fast and comprehensive Android method tracer
- Automated analysis of unknown applications
- Quickly identify suspicious applications
- Interactive environment to ease post-analysis

http://tracedroid.few.vu.nl/

Submit your .apk for automated analysis

.tar.gz output containing:

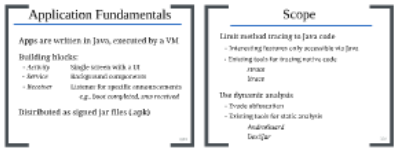
- method traces
- network dump
- call graph

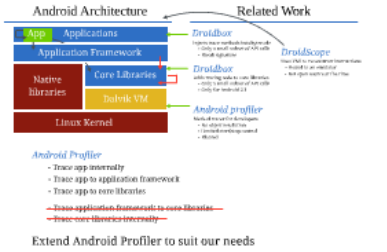
Contact me if you would like to analyze a batch

No source or inspect tool available yet

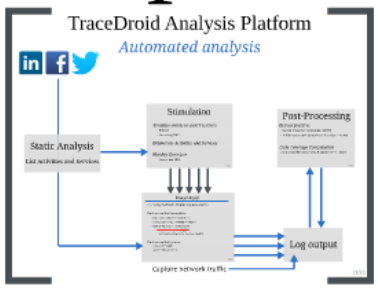
http://tracedroid.few.vu.nl

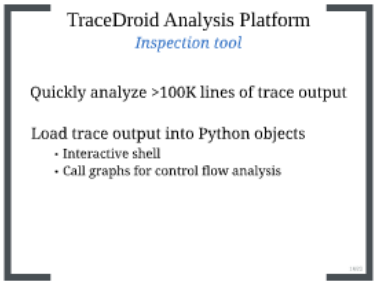
Introduction

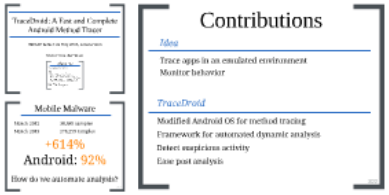


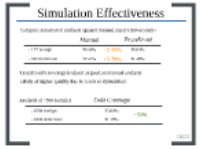


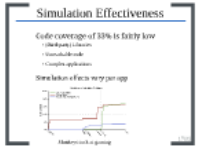
Implementation











Evaluation

Conclusions

http://tracedroid.few.vu.nl/

Submit your .apk for automated analysis
.tar.gz output containing:
- method traces
- network dump
- call graph

Contact me if you would like to analyze a batch

No source or inspect tool available yet

Demo

ZitMo: Zeus in the Mobile
- Collaborates with PC-based Zeus
- Steals mobile TAN codes