Andrubis – 1,000,000 Apps Later

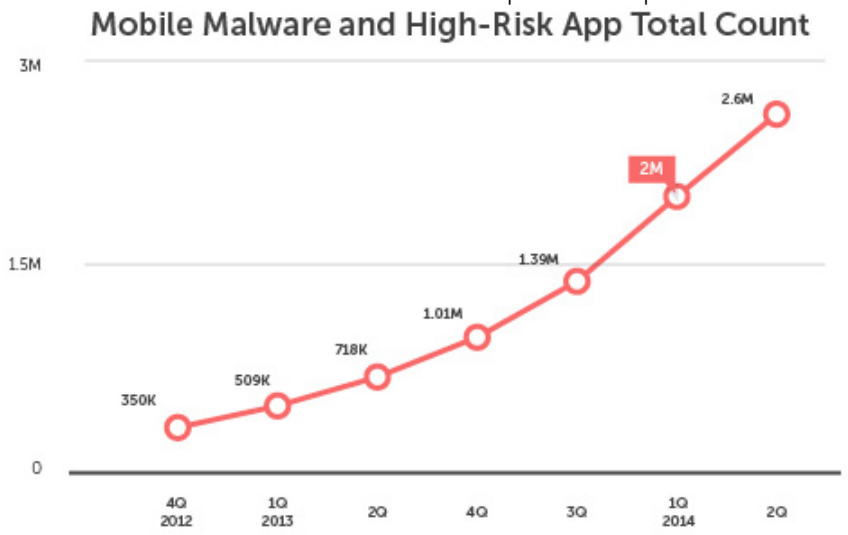
A View on Current Android Malware Behaviors

Martina Lindorfer, Matthias Neugschwandtner, Lukas Weichselbaum,
Yanick Fratantonio, Victor van der Veen, Christian Platzer

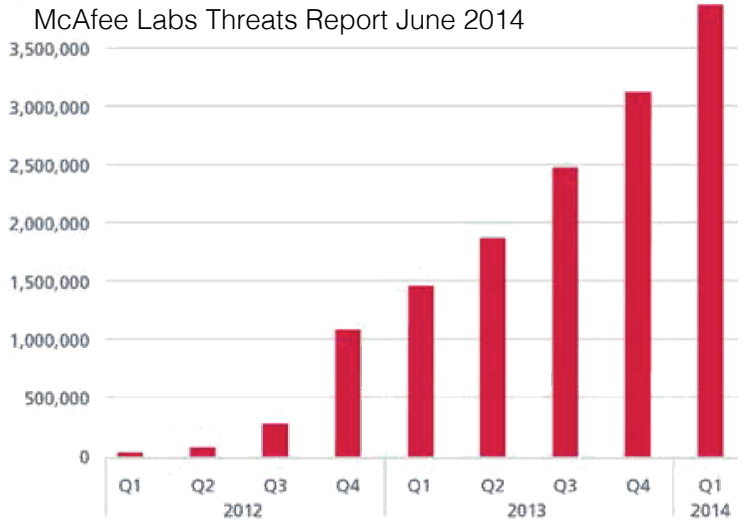
Vienna University of Technology
University of California, Santa Barbara
VU University Amsterdam

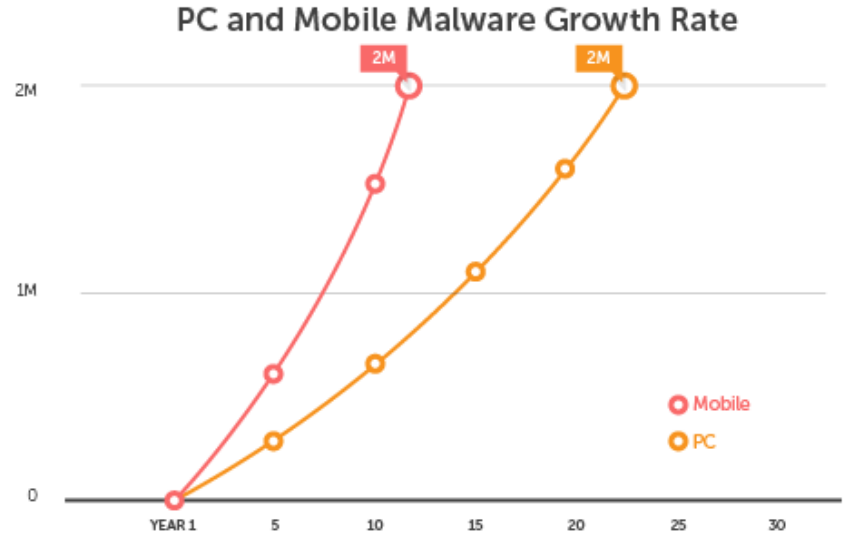
Android Malware Pandemic?

TrendMicro: The Mobile Landscape Roundup 1H 2014



McAfee Labs Threats Report June 2014





Enter Sandbox

iseclab

SandDroid - An automatic Android application analysis sandbox.

An Android Application Sandbox System for Suspicious Software Detection

Thomas Bläsing, Leonid Batyuk, Aubrey-Derrick Schmidt,
Seyit Ahmet Camtepe, and Sahin Albayrak
Technische Universität Berlin - DAI-Labor
{thomas.blaesing, leonid.batyuk, aubrey.schmidt}@dai-labor.de
{ahmet.camtepe, sahin.albayrak}@dai-labor.de

ANANAS – A Framework For Analyzing Android Applications

Thomas Eder, Michael Rodler, Dieter Vymazal, Markus Zeilinger
Department Secure Information Systems
University of Applied Sciences Upper Austria
{thomas.eder, michael.rodler}@students.fh-hagenberg.at
{dieter.vymazal, markus.zeilinger}@fh-hagenberg.at

Tracedroid

A System Call-Centric Analysis and Stimulation Technique to Automatically Reconstruct Android Malware Behaviors

Alessandro Reina
Dept. of Computer Science
Università degli Studi di Milano
ale@security.di.unimi.it

Aristide Fattori
Dept. of Computer Science
Università degli Studi di Milano
aristide@security.di.unimi.it

Lorenzo Cavallaro
Information Security Group
Royal Holloway, University of London
lorenzo.cavallaro@rhul.ac.uk

FORESAFE MOBILE SECURITY

VisualThreat

CopperDroid

MS

Mobile-Sandbox: Having a Deeper Look into Android Applications

Michael Spreitzenbarth, Felix Freiling
Friedrich-Alexander-University
Erlangen, Germany
michael.spreitzenbarth, felix.freiling@cs.fau.de

Florian Echtler, Thomas Schreck
Siemens CERT
Munich, Germany
florian.echtler, t.schreck@siemens.com

Johannes Hoffmann
Ruhr-University Bochum
Bochum, Germany
johannes.hoffmann@rub.de

DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis

Lok Kwong Yan[†‡] Heng Yin[†]

[†]*Syracuse University*
Syracuse, New York, USA

[‡]*Air Force Research Laboratory*
Rome, New York, USA

{loyan, heyin}@syr.edu

JoeSandbox Mobile

Enter Sandbox

SandDroid - An automatic Android application analysis sandbox.

An Android Application Sandbox System for Suspicious Software Detection

Thomas Bläsing, Leonid Batyuk, Aubrey-Derrick Schmidt,
Seyit Ahmet Camtepe, and Sahin Albayrak
Technische Universität Berlin - DAI-Labor
{thomas.blaesing, leonid.batyuk, aubrey.schmidt}@dai-labor.de
{ahmet.camtepe, sahin.albayrak}@dai-labor.de

ANANAS – A Framework For Analyzing Android Applications

Thomas Eder, Michael Rodler, Dieter Vymazal, Markus Zeilinger
Department Secure Information Systems
University of Applied Sciences Upper Austria
{thomas.eder, michael.rodler}@students.fh-hagenberg.at
{dieter.vymazal, markus.zeilinger}@fh-hagenberg.at

Tracedroid

A System Call-Centric Analysis and Stimulation Technique to Automatically Reconstruct Android Malware Behaviors

Alessandro Reina
Dept. of Computer Science
Università degli Studi di Milano
ale@security.di.unimi.it

Aristide Fattori
Dept. of Computer Science
Università degli Studi di Milano
aristide@security.di.unimi.it

Lorenzo Cavallaro
Information Security Group
Royal Holloway, University of London
lorenzo.cavallaro@rhul.ac.uk

DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis

Lok Kwong Yan[†‡] Heng Yin[†]

[†]*Syracuse University*
Syracuse, New York, USA

[‡]*Air Force Research Laboratory*
Rome, New York, USA

{loyan, heyin}@syr.edu

Mobile-Sandbox:
Having a Deeper Look into Android Applications

Michael Spreitzenbarth,
Felix Freiling
Friedrich-Alexander-University
Erlangen, Germany
michael.spreitzenbarth,
felix.freiling@cs.fau.de

Florian Echtler,
Thomas Schreck
Siemens CERT
Munich, Germany
florian.echtler,
t.schreck@siemens.com

Johannes Hoffmann
Ruhr-University Bochum
Bochum, Germany
johannes.hoffmann@rub.de

Our Contributions

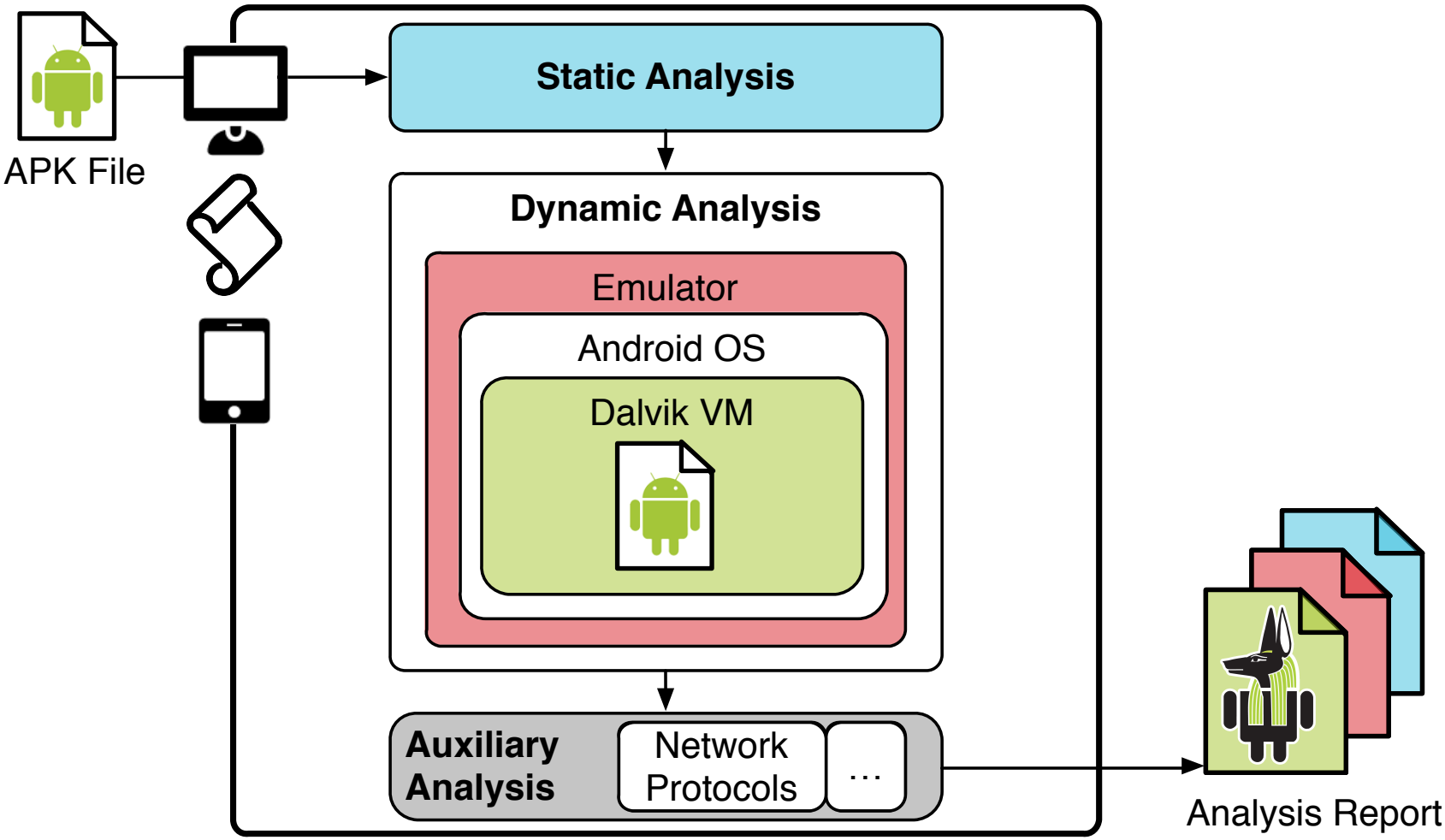
- Take advantage of our existing Anubis infrastructure

- Build an Android analysis sandbox that …
  - is suitable for large-scale analysis
  - allows us to collect a comprehensive dataset of Android malware and goodware
  - can be easily integrated into other tools and services
  - is publicly available
    - As a web service: https://anubis.iseclab.org
    - For batch submissions via API: http://anubis.iseclab.org/Resources/submit_to_anubis.py
    - As a mobile app: https://play.google.com/store/apps/details?id=org.iseclab.andrubis

Outline

- **Andrubis System Overview**
- Andrubis As A Service
- Android Malware Landscape
- Future Work and Conclusion

System Overview



Public Analysis Features

- Static Analysis
  - Parse meta information from Android manifest
    - Requested permissions
    - Activities
    - Services
    - Registered Broadcast Receivers
  - Extract available methods from bytecode
    - Used permissions
    - Use of DEX and native code loading
  - Useful during stimulation

Public Analysis Features

- Dynamic Analysis
  - Run app in QEMU-based environment
  - Instrumented Dalvik VM
    - Log file system, network, phone (calls & SMS), crypto and dynamic code loading activity
  - Taint tracking to identify data leaks
  - Stimulation
    - Invoke all Activities, Services and Broadcast Receivers
    - Simulate common events (e.g. SMS receipt)
    - Application Exerciser Monkey

- Auxiliary Analysis
  - Network capture outside QEMU
  - Extraction of high-level network protocol features

Advanced Analysis Features

- Method Tracing
  - Extension of the Dalvik VM profiler
  - Outputs list of executed methods
  - Use Cases:
    - Basic code coverage computation
    - Permissions actually used during dynamic analysis
    - Behavioral signatures and classification

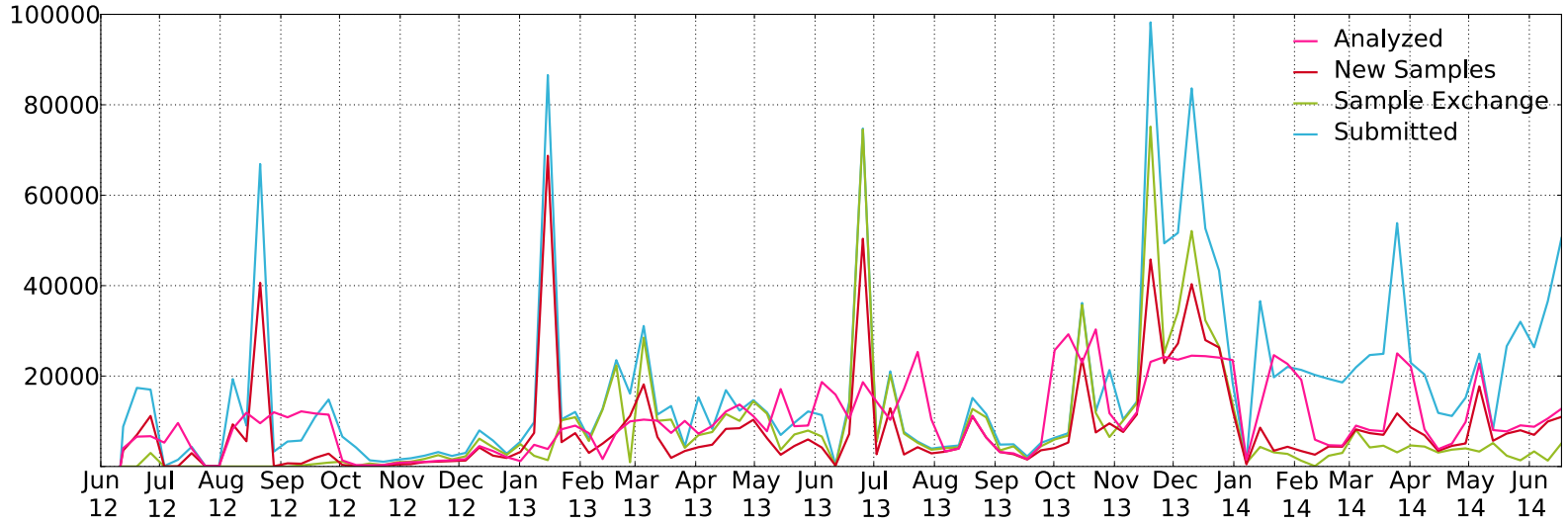
- System-Level Analysis
  - QEMU VMI
  - Outputs list of executed system calls
  - Use Cases:
    - Analysis of native libraries, e.g. root exploits

Outline

- Andrubis System Overview
- **Andrubis As A Service**
- Android Malware Landscape
- Future Work and Conclusion

Submission Statistics

- Online since June 2012
- 1,778,997 submissions
  - 95.82% from bulk submitters
- 1,034,999 unique apps
  - 5% of total samples submitted to An(dr)ubis
- Throughput of 3,500 apps per day



Deployment Considerations

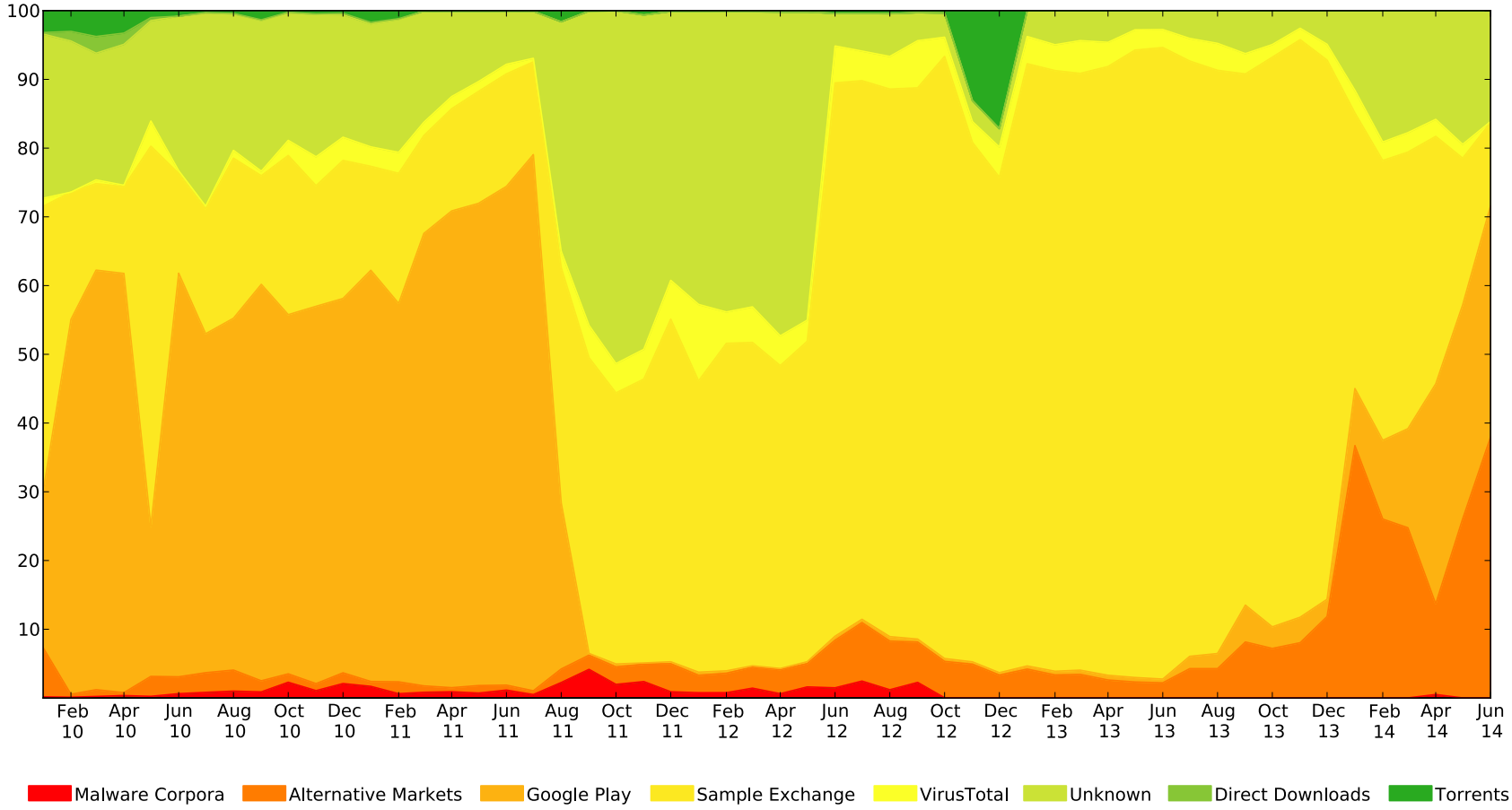
- OS version = trade-off between running …
  - Old version to observe root exploits
  - New version to analyze current apps
- Maintenance effort of constant updates
  - Focus on implementing new features instead
- Andrubis supports API level ≤ 10 (Gingerbread)
- Unsupported API level mainly a concern for GW:
  - 2.11% of benign apps with API level > 10
  - 0.10% of malicious apps of API level > 10
  - Maximize potential “user base” of malware

Our Dataset

- Samples from a variety of sources
  - Google Play and alternative market crawls (AndRadar)
    - Main distribution vector for Android apps
  - Torrents & Direct Downloads
  - Sample exchange with other researchers
  - VirusTotal
  - Malware Corpora
    - Genome Project, Contagio, Drebin
  - Anonymous submissions

- Comparison to other tools
  - Based on public malware corpora (mostly outdated)
  - (Subset of) our dataset

Sample Age by Source



Outline

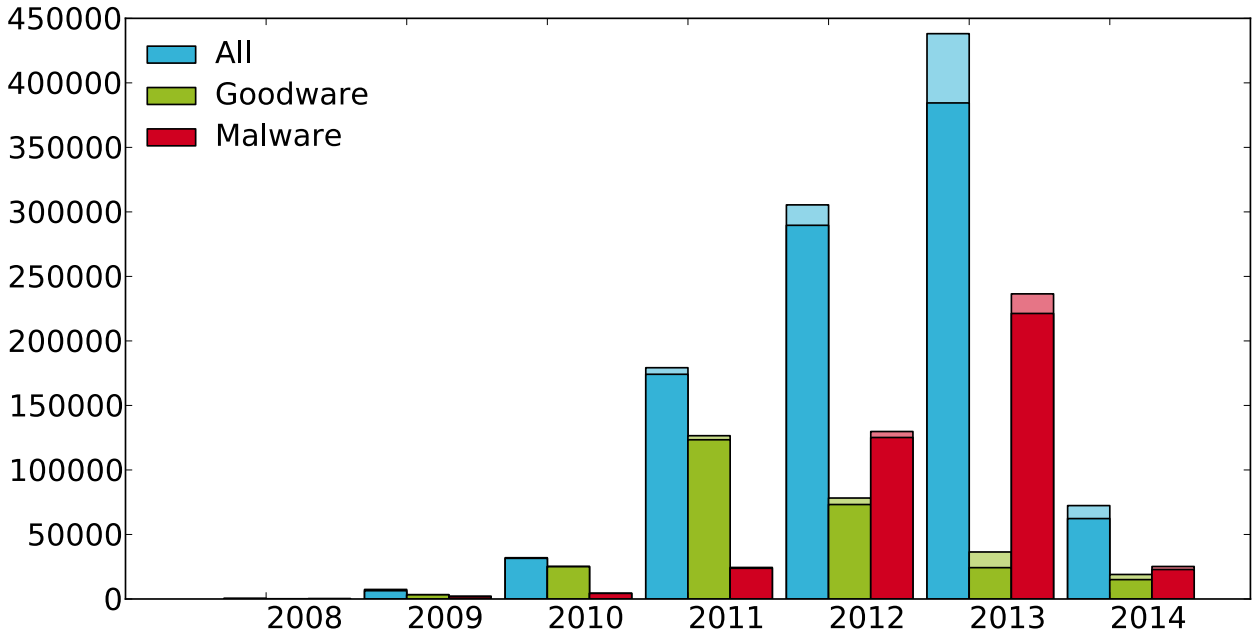
- Andrubis System Overview
- Andrubis As A Service
- **Android Malware Landscape**
- Future Work and Conclusion

Dataset Classification

- No ground truth for majority of samples
  - Besides public malware corpora
- Andrubis itself performs no classification
  - Although we are experimenting with machine-learning approaches
- We rely on AV labels for this evaluation
  - Goodware: 27.90%
  - Malware: 41.15%
  - Unlabeled: 30.95%
- Unlabeled set contains mainly adware
  - Also possible false positives
- Very inconsistent AV labeling
  - Found even Google app labeled as MW by AVs

Dataset by Release Date

- Based on four dates:
  - Last modification date of the APK file (ZIP header)
  - Release date of the minimum required SDK
  - Publication date in alternative markets/Google Play
  - First submission date to Andrubis



Key Observations

- Trends in MW/GW development from 2010-2014
- Static analysis alone becomes increasingly difficult
  - Ubiquitous use of reflection, especially in GW
  - Increasing use of dynamic code loading

- Common assumptions about MW/GW:
  - Malicious apps request more permissions than benign apps, but use less of them
  - Dynamic code loading is an indicator for malware

Requested/Used Permissions

- MW requests 12.99 permissions, uses 5.31 of them
- GW requests 5.85 permissions, uses 4.50 of them
- Requested permissions increased for both
- Decreased permission usage ratio
  - Only 13.38% in GW in 2014
  - Side-effect of dynamic code loading
  - Bad development practices
- Numbers based on static extraction of used permissions
  - Permissions used during dynamic analysis from method tracer logs

App Interdependencies

- Apps can share their UID
  - Share data, run in the same process and inherit permissions
- Allows collusion attack
  - Spread malicious payload over benign looking apps
- Allows privilege escalation by taking advantage of already installed benign apps
  - Circumvent signature system with Master Key vulnerability
  - Use publicly available test keys
  - Even gain system privileges with android.uid.system UID
- Only used in few GW (1.14%) and MW (0.29%) app

Other Findings from Static Analysis

- Application names
  - MW often uses legitimate looking package names
    - Repackaging/posing as popular benign apps
    - Generic names (e.g, com.app.android)
  - “Random” names (e.g.; rpyhwytfysl.uikbvktgwp) reused amongst thousands of apps

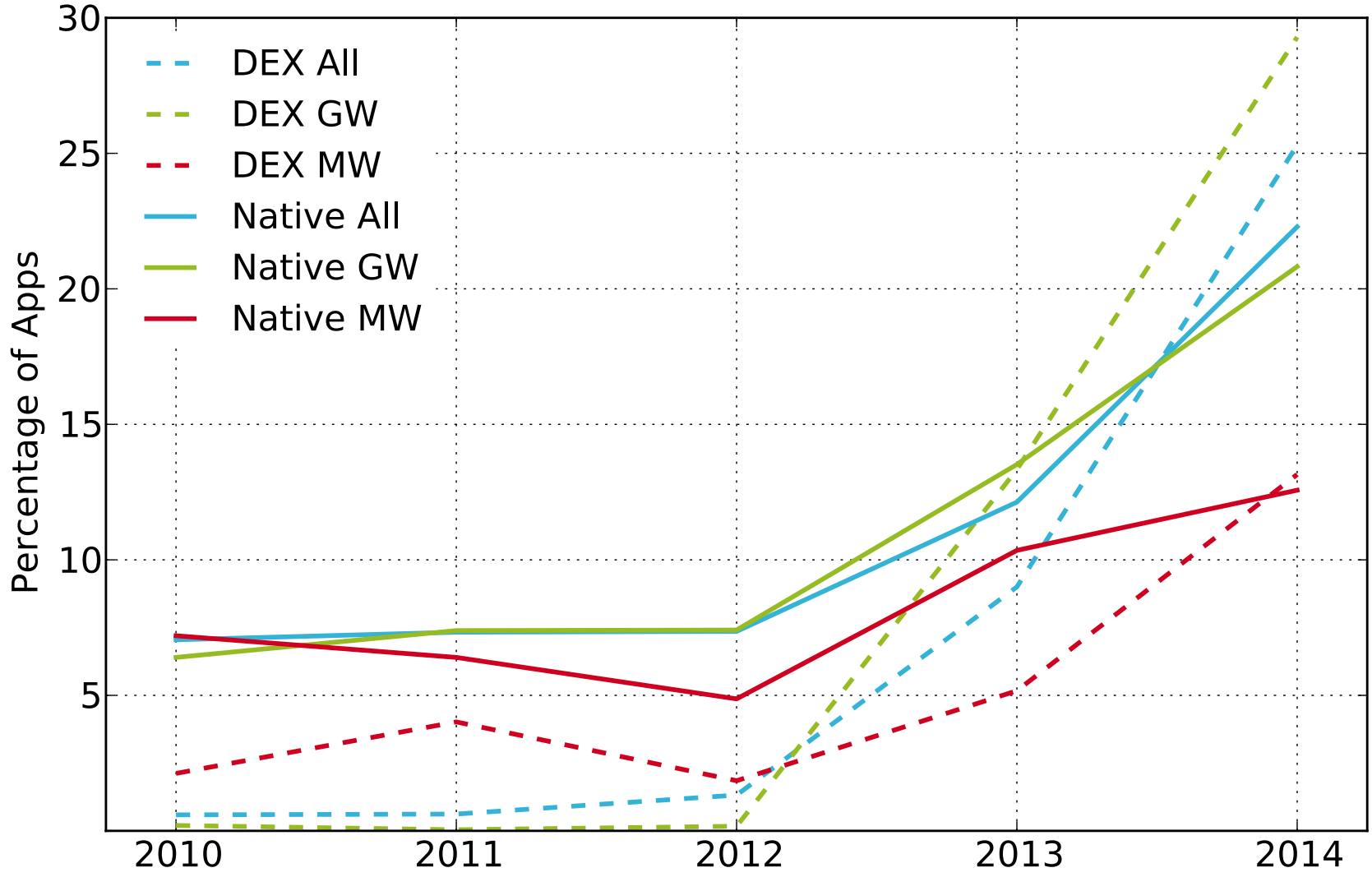
- Decreasing use of public test keys to sign apps
  - Should not be used by legitimate developers
  - 8.92% of MW (down from 65.29% in 2010), 2.26% of GW

- Master Key vulnerabilities not widely exploited
  - Only ~1.500 MW samples

Dynamic Code Loading

- Significantly increased, especially in GW
  - 30% of GW load DEX classes
  - 20% of GW load native code
  - 13% of MW load DEX or native code
- Static detection of dynamic code loading important for selecting samples
  - Successful in detecting DEX loads (>97% of apps)
  - Less successful in detecting native code (54% GW, 83% MW)
- Custom libraries more dangerous than libraries shipped with the OS
  - GW increasingly ships its own native libraries (84%)

Dynamic Code Loading Trend



Other Findings from Dynamic Analysis

- Increasing use of external storage (SD card)
  - Contrary to Google's policy
  - Especially prevalent in malware (30%)
    - New monetization vector (Cryptolocker)
- Almost no apps perform phone calls
  - Not revealed by static analysis in any app
- Almost no benign apps send SMS (0.26%)
- Unsurprisingly 15% of MW send SMS
  - Only revealed through static analysis in ~ 80% of apps
  - Up to 120 SMS to premium number during one analysis run

Other Findings from Dynamic Analysis

- More MW than GW leak data: 43% vs. 14%
  - Mostly to the network, very few per SMS
  - Recently MW started leaking information per e-mail
    - Forwarding incoming SMS
    - Leaking contacts
- Data leakage increased overall from 14% to 50%

- Increased usage of crypto API in GW (11% to 79%)
- MW adopting stronger cryptographic algorithms
  - DES almost completely replaced with AES and Blowfish
- Static analysis determined crypto usage in 43% of MW

Outline

- Andrubis System Overview
- Andrubis As A Service
- Android Malware Landscape
- **Future Work and Conclusion**

Limitations and Future Work

- Dynamic analysis evasion
- GUI Stimulation
  - More intelligent, user-like input
  - Targeted input for phishing attempts of banking apps, …
- Lack of metadata
  - Crawling markets with AndRadar
- Lack of ground truth
  - Classification of Android malware
- Dated public datasets and lack of comparability
  - Planning to release public dataset
  - Sharing of samples and/or reports on request

Conclusion

- Large-scale analysis system for Android apps
- Static and dynamic analysis on Dalvik VM and system level
- Publicly available at https://anubis.iseclab.org and via our Android app
- Operating for the past 2+ years
- Dataset of > 1,000,000 Android apps
- Identified trends in the Android malware landscape
- Dynamic analysis increasingly important

Questions?

andrubis@iseclab.org
https://twitter.com/iseclaborg

mlindorfer@iseclab.org
http://www.iseclab.org/people/mlindorfer