SAND2014-17876PE

*Exceptional service in the national interest*

Sandia National Laboratories

# Mobile Malware Analysis:

## Examining suspicious applications on Android and iOS

Michael Bierma, Yung Ryn (Elisha) Choe

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2014-4080C

# Outline

Android

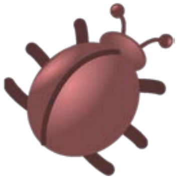
- Background
- Architecture
- Results

iOS

- Background
- Challenges of iOS research
- Solutions
- Current work

# Android Operating System

Over 1 million applications available on Google Play[1]



92% of mobile malware[2]

Device fragmentation

- 19 Current API levels
- Over 11,000 different Android devices[3]

- How can these applications be analyzed efficiently?

[1] http://www.phonearena.com/news/Androids-Google-Play-beats-App-Store-with-over-1-million-apps-now-officially-largest_id45680
[2] http://www.juniper.net/us/en/local/pdf/additional-resources/3rd-jnpr-mobile-threats-report-exec-summary.pdf
[3] http://opensignal.com/reports/fragmentation-2013/

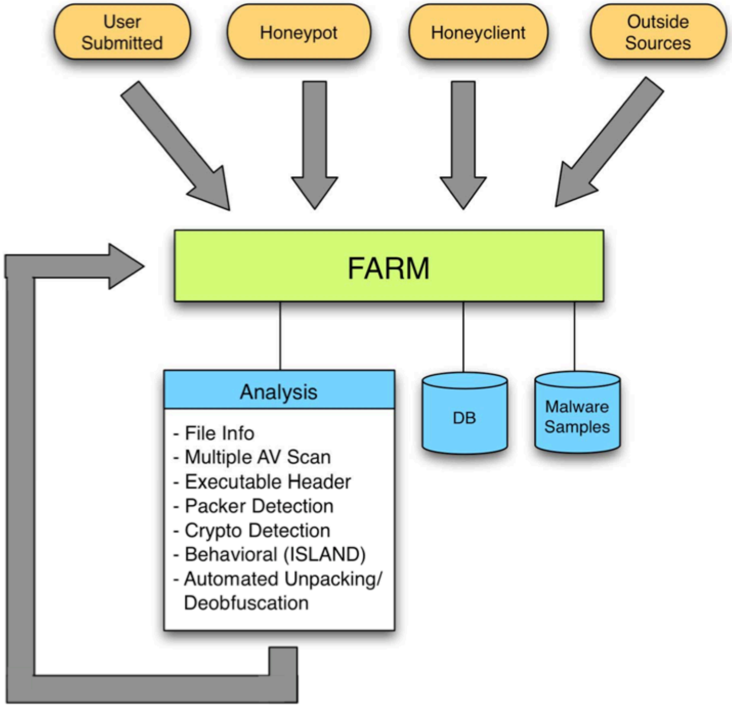
# Dynamic Analysis

- Benefits:
  - Able to detect bugs introduced during runtime
  - Can detect vulnerabilities too complex for static analysis
  - Flexibility in handling application performance across APIs

- Challenges:
  - Scalability
  - Code coverage
  - Tracing bugs to specific lines of code

# Architecture: FARM

- FARM – Forensic Analysis Repository for Malware
  - Malicious and potentially malicious applications (malware samples)
  - Metadata related to results of malware analyses



# Architecture: Kane

- Kane – Commodity cluster for malware analysis
  - 480 diskless nodes
  - Per node:
    - 12GB RAM
    - Quad-core 2.8GHz Intel Core i7 CPU
    - 1Gb Ethernet connection

# Architecture: Job Control

- Parallel scheduling
  - 3 Staging Areas
    - Submit job
      - Boot emulator
      - Acquire IP address
      - Load APK
    - Run experiment
      - Exercise app through dynamic analysis
    - Generate results
      - Collect file system diff
      - Collect dynamic analysis logs

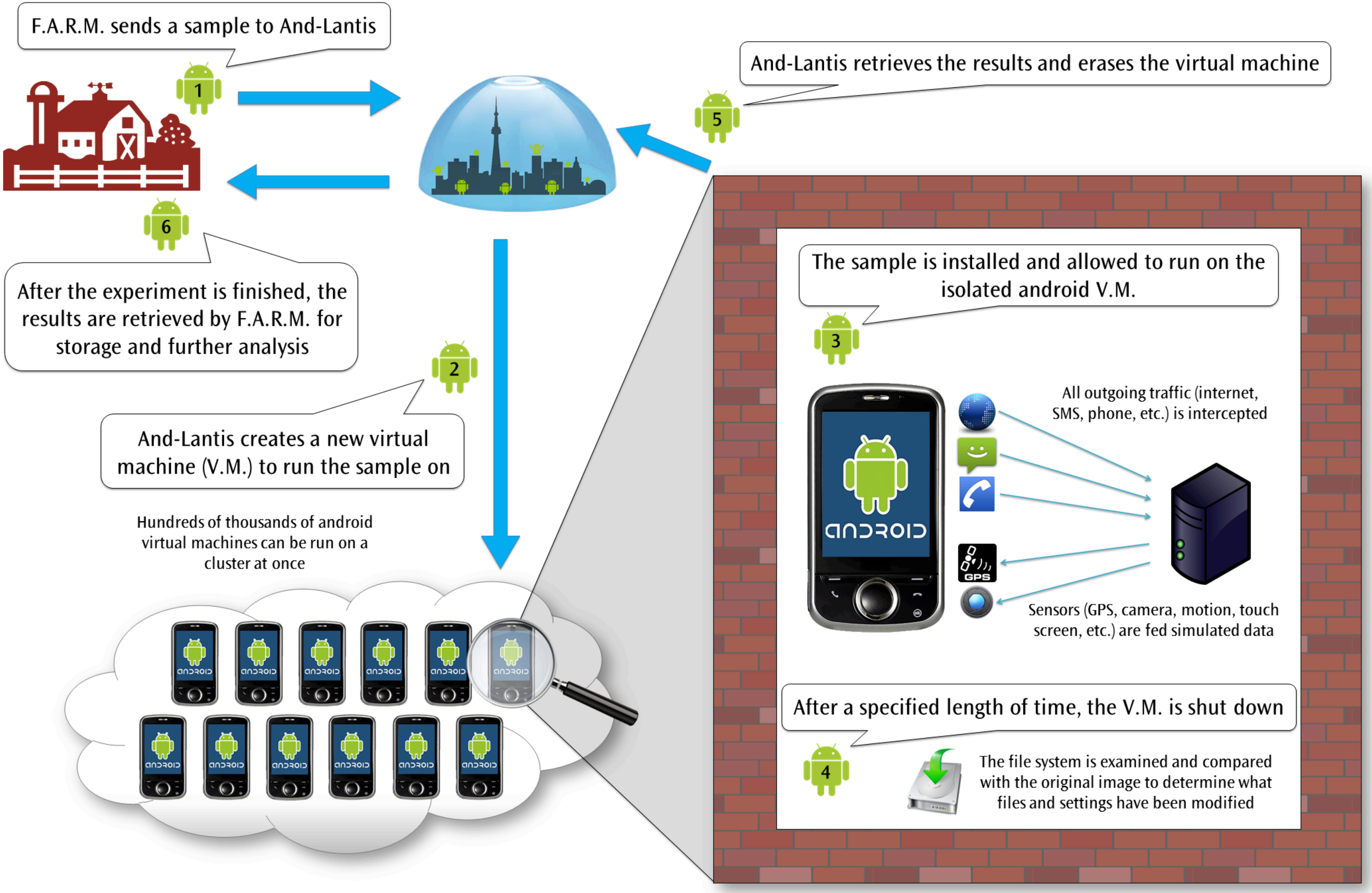
# Architecture: Minimega

- Minimega – large scale Virtual Machine management
  - Simplified VM management
  - Mesh network strategy to allow communication between master node and all VMs
    - Any node can function as master
    - Can easily spin up and communicate with 3 experiments on all available nodes
  - Designed to easily scale to available nodes
  - Robust against node failures

# Architecture: Paddle

- Utilizes the Android’s Monkey infrastructure
  - Drives our dynamic analysis
    - Controls the emulator outside of application code
    - Traverses UI in DFS manner
- Logging for application analysis
  - Interaction patterns
  - Application crashes

# Architecture: Job Flow



1. F.A.R.M. sends a sample to And-Lantis

2. And-Lantis creates a new virtual machine (V.M.) to run the sample on

Hundreds of thousands of android virtual machines can be run on a cluster at once

3. The sample is installed and allowed to run on the isolated android V.M.

All outgoing traffic (internet, SMS, phone, etc.) is intercepted

Sensors (GPS, camera, motion, touch screen, etc.) are fed simulated data

After a specified length of time, the V.M. is shut down

4. The file system is examined and compared with the original image to determine what files and settings have been modified

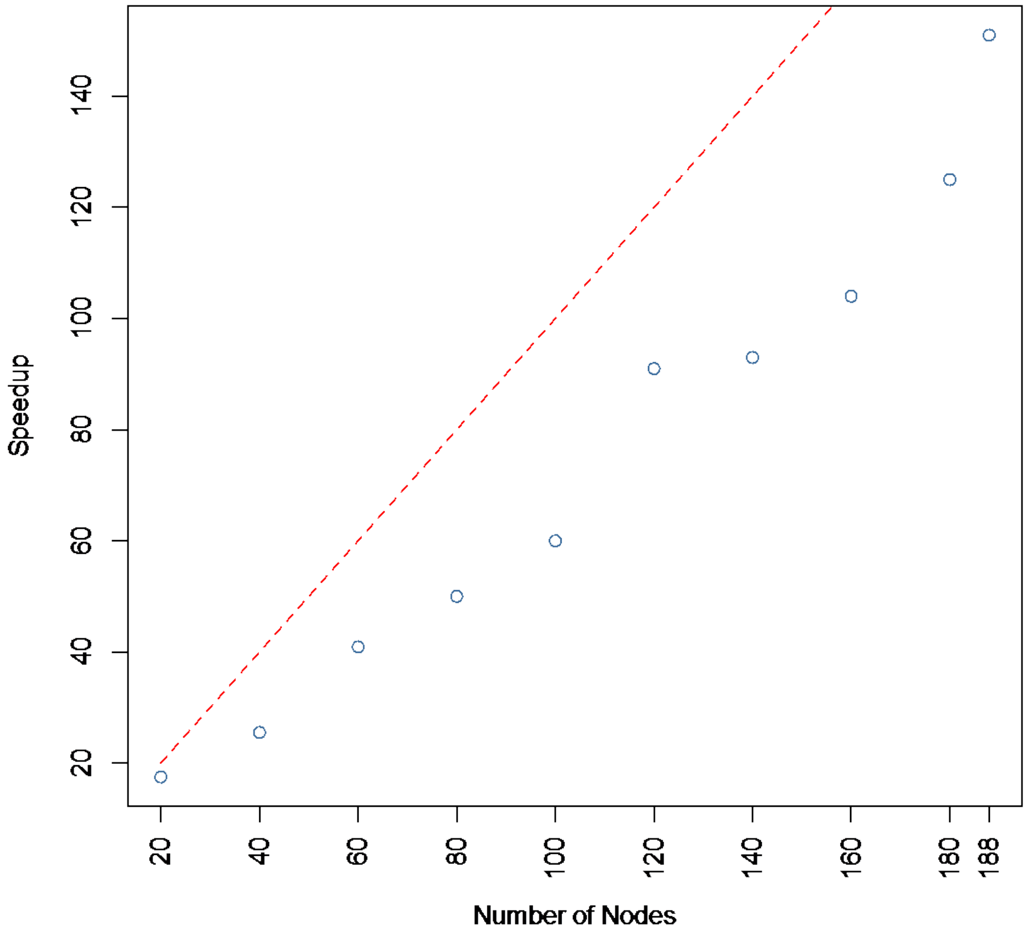
5. And-Lantis retrieves the results and erases the virtual machine

6. After the experiment is finished, the results are retrieved by F.A.R.M. for storage and further analysis

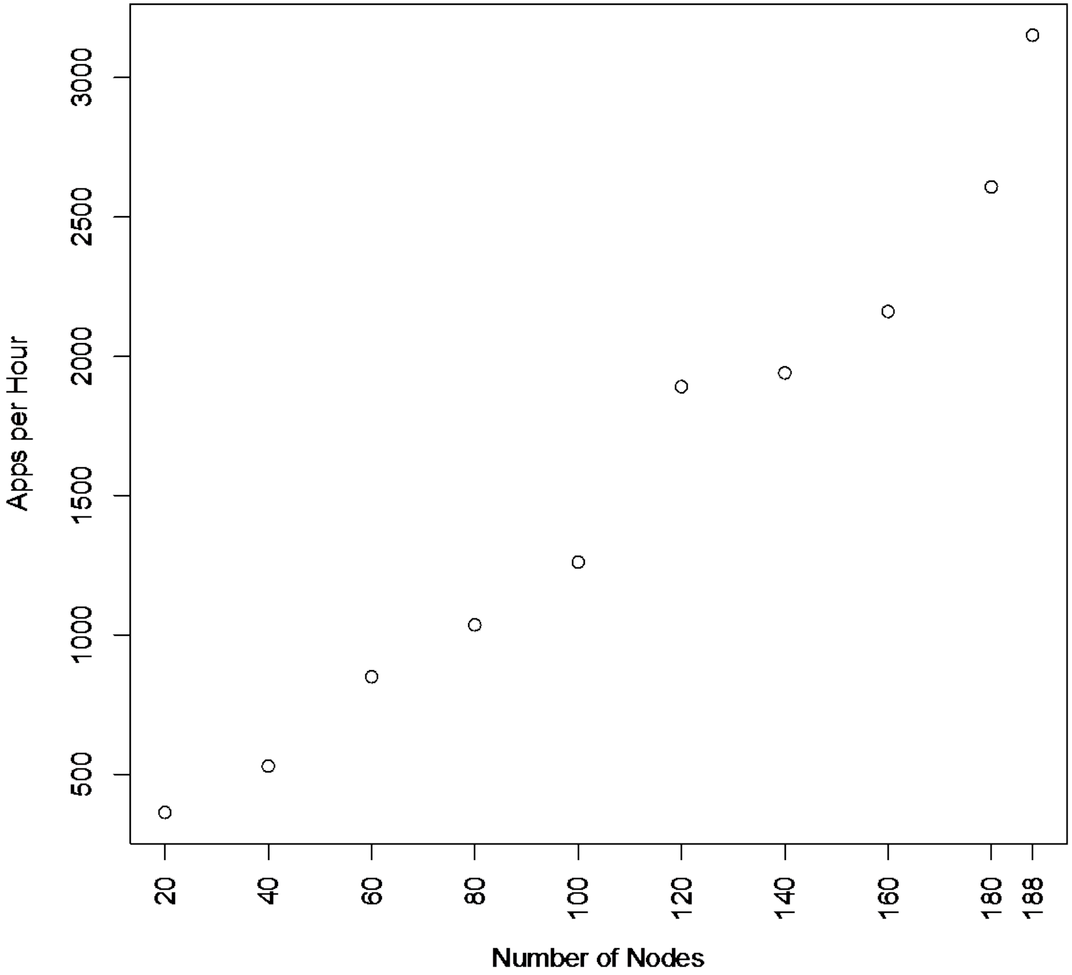
# Dataset

- Malware Genome Project
  - Application dataset compiled by NC State
  - Comprised of 1261 malware samples covering a range of malware families
  - Beneficial for large scale malware testing
    - Performance evaluation
    - Analysis of known malware behavior

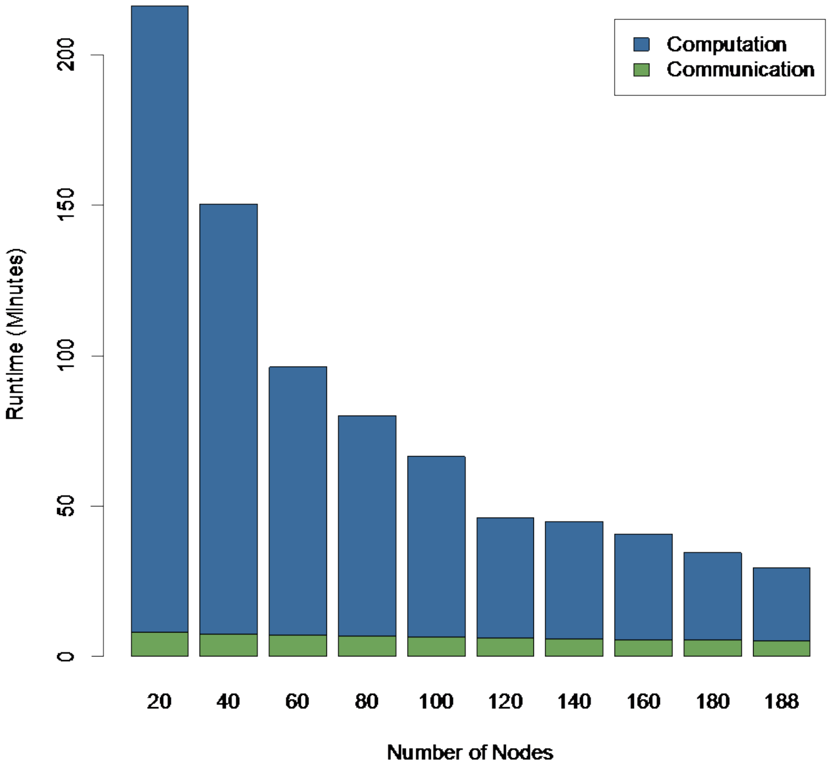
# Performance: Speedup

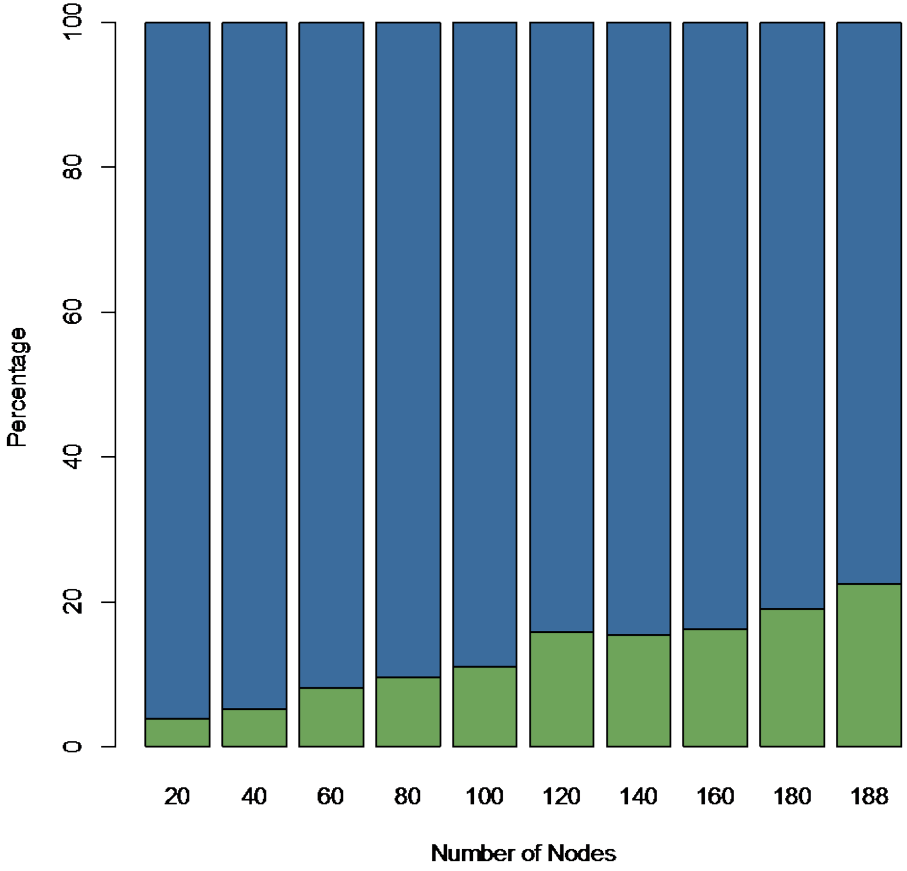


# Performance: Throughput



# Performance: Breakdown





# Findings

- *AndroidOS/DroidKrungFu.A*
  - */data/media/0/txtbooks/legacy APK payload*
- *AndroidOS/Anserver.A*
  - *anserva.db APK payload*
  - *Application asks for user confirmation before installation (more features promised)*
- *j.SMSHider*
  - Root access acquired

# Android Recap

- Highly scalable dynamic analysis platform for Android application analysis
  - Developers
    - Bug testing
    - Fragmentation-proofing Android applications
  - Security Researchers
    - Quickly analyze thousands of applications in parallel
    - Run a single application in a variety of different emulated environments

- Andlantis v2 is in development

# iOS Ecosystem

Over 1.3 million apps available on App Store[1]

Closed source

- Tightly controlled distribution

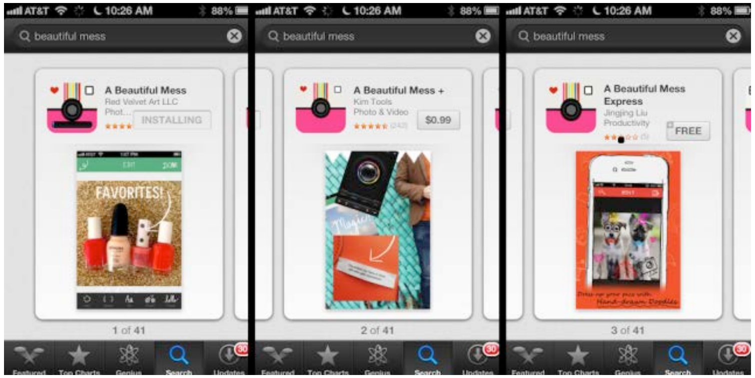
Developers

- 275,000[2]
- 45% more revenue per user vs Android[3]

[1] https://www.apple.com/pr/library/2014/09/09Apple-Announces-iOS-8-Available-September-17.html
[2] https://www.apple.com/about/job-creation/
[3] http://www.dazeinfo.com/2014/05/27/ios-users-32-likely-make-app-purchases-android-users-engaged/

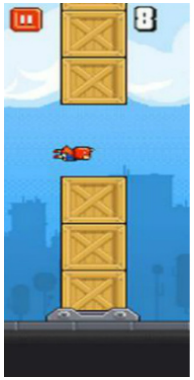
# Clones

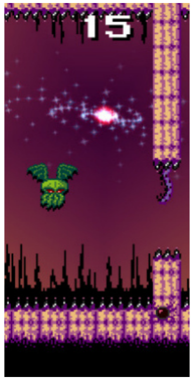
- Re-upload exact same application as another user

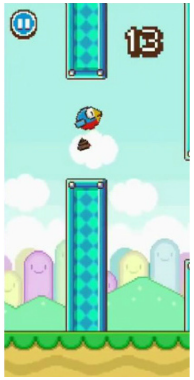


- Commercial templates

- Fake apps (UI duplication)







- How can do detect these cloned applications?

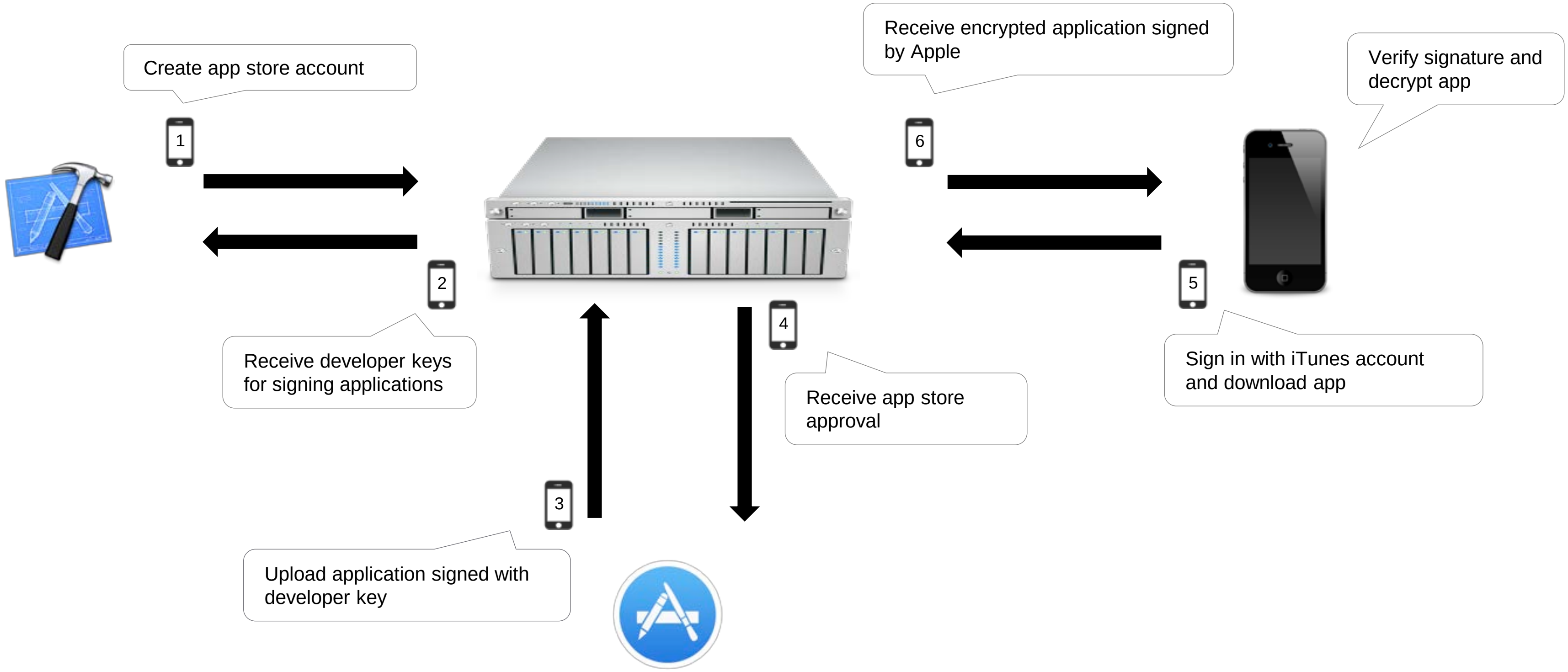
# Motivations

Reduction of app store spam

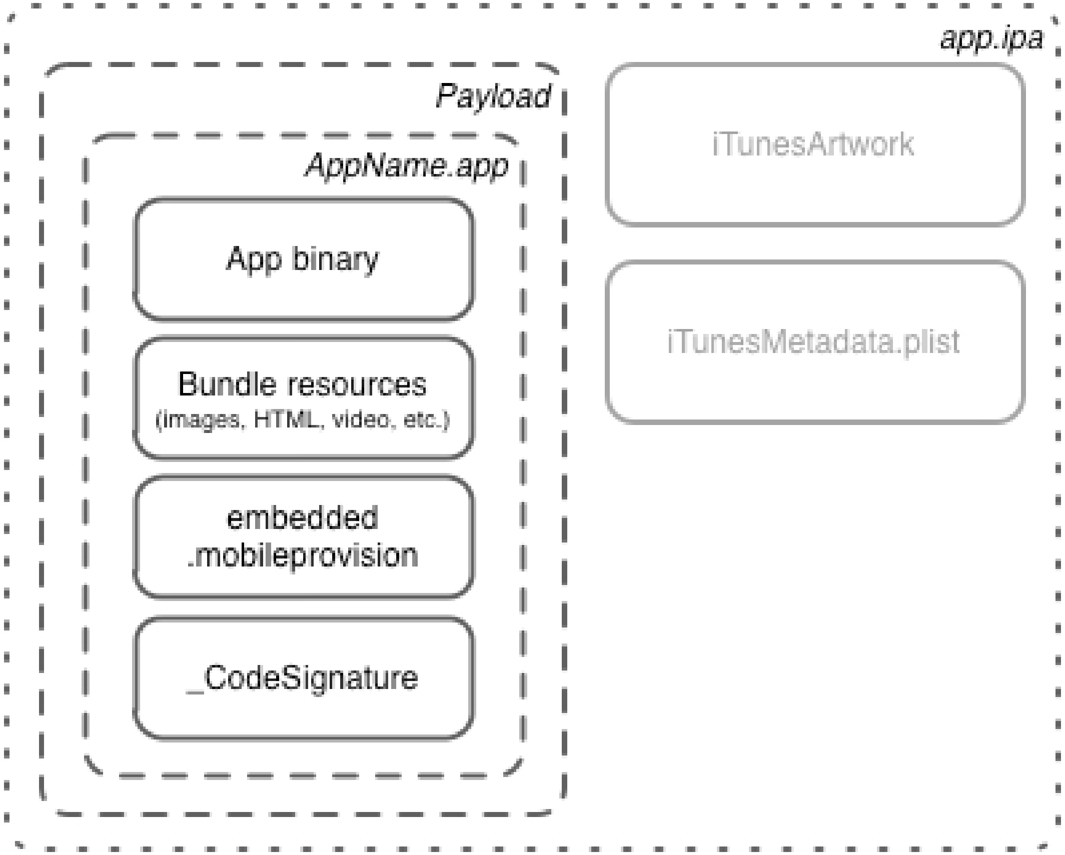
Prevention of developer revenue loss

Understanding of code reuse/libraries in iOS apps

# iOS Development


1
Create app store account
2
Receive developer keys for signing applications
3
Upload application signed with developer key
4
Receive app store approval
5
Sign in with iTunes account and download app
6
Receive encrypted application signed by Apple
Verify signature and decrypt app

# iOS App


app.ipa
Payload
AppName.app
App binary
Bundle resources
(images, HTML, video, etc.)
embedded
.mobileprovision
_CodeSignature
iTunesArtwork
iTunesMetadata.plist

# Challenges

Acquiring applications

Fairplay DRM

Analysis

# Acquiring Applications

POST https://p6-buy.itunes.apple.com/WebObjects/MZBuy.woa/wa/buyProduct

```
<plist version="1.0">
  <dict>
    <key>appExtVrsId</key>
    <string>368952942</string>
     <key>kbsync</key>
     <data> AAQAA+xdY4vyNxmG284QjYs41Yg++toup14SNmI6W+leqqznHO5z+W2Tesj2d3E2fg4d
              gX9d+F9f3W311xTMTwSyuJCyGLp0MckbMjvyQxrnrnRM1Cgl0bwrANeogub4zFywanVL Gu1kxjhEbhaZvHjO6EiHCE1Q/gUb
             +kFQtKpMJ8b62T44n2PHRkZSj74QD5M0wmEe/X8O
             9SWggfujrsxPzYKQNb6wqKG8BzvCzR4kiLVTEle7mS+OesS5oVPKnG2gNZbLORrb6boq P0FM/
             8bCebAzbGNB0TgYbr3AjqgQPJgQnD7i8of0okrt0dw7p2+XjY+GcMbBwQam4THn
             tcb7UXYbQRfSqsT05pdrDCT+lsJ+hLBpkgMj3fo18BjpHGvAcoc/OsIGYyYWWjkxoQB5 Lcvsri8kS8PYPeS1qSlXI6VzqTUbdpR4b0LDrgmdn+M0z/
             y5R5f8srqZN7QqNixEKjjV
             ijsqmaJ9JzdRAEiQPQj1x9pCv0bZHKPf26uVZW7RABaDc3uRTJM7YvmjykGQYg5JaCLg 6kFAFAv1pIvCpdgqcO7ZL5jz9q8S7iboFtQ/D/
            mLlTDHfDnvSYg7u9JWXn7iB+ibkzYc
            2kmL5Xa76FE7bhIGQoHHGRNMbleHFxN+bgfNMAm9ni54Y6ipAGtO4V/mNISPwY/DWlPc
           +44R0cIb01zxfELMUuhsyNRi5UlJGAKF8bSGe9EYcTnJ5gN7svjAc53aLYHvaZyA3fLO
           AD1zRGfMrzBIl7CUVzZ8MM0/wISmD0aLyOa7zACBnkZ93VLgXPEs0NNa20qtrtIgOuE5
    </data>
    <string>Tab_iphone|Titledbox_Top Free Apps|Lockup_2|Buy</string>
    <key>price</key>
    <string>0</string>
    <key>pricingParameters</key>
    <string>STDQ</string>
    <key>productType</key>
    <string>C</string>
    <key>salableAdamId</key>
    <string>803181003</string>
  </dict>
</plist>
```

GET http://a1301.phobos.apple.com/us/r1000/005/Purple6/v4/54/e3/e3/54e3e335-4bfa-27a0-8366-2938c266a1b7/mzps8089721949445276176.D2.dpkg.ipa

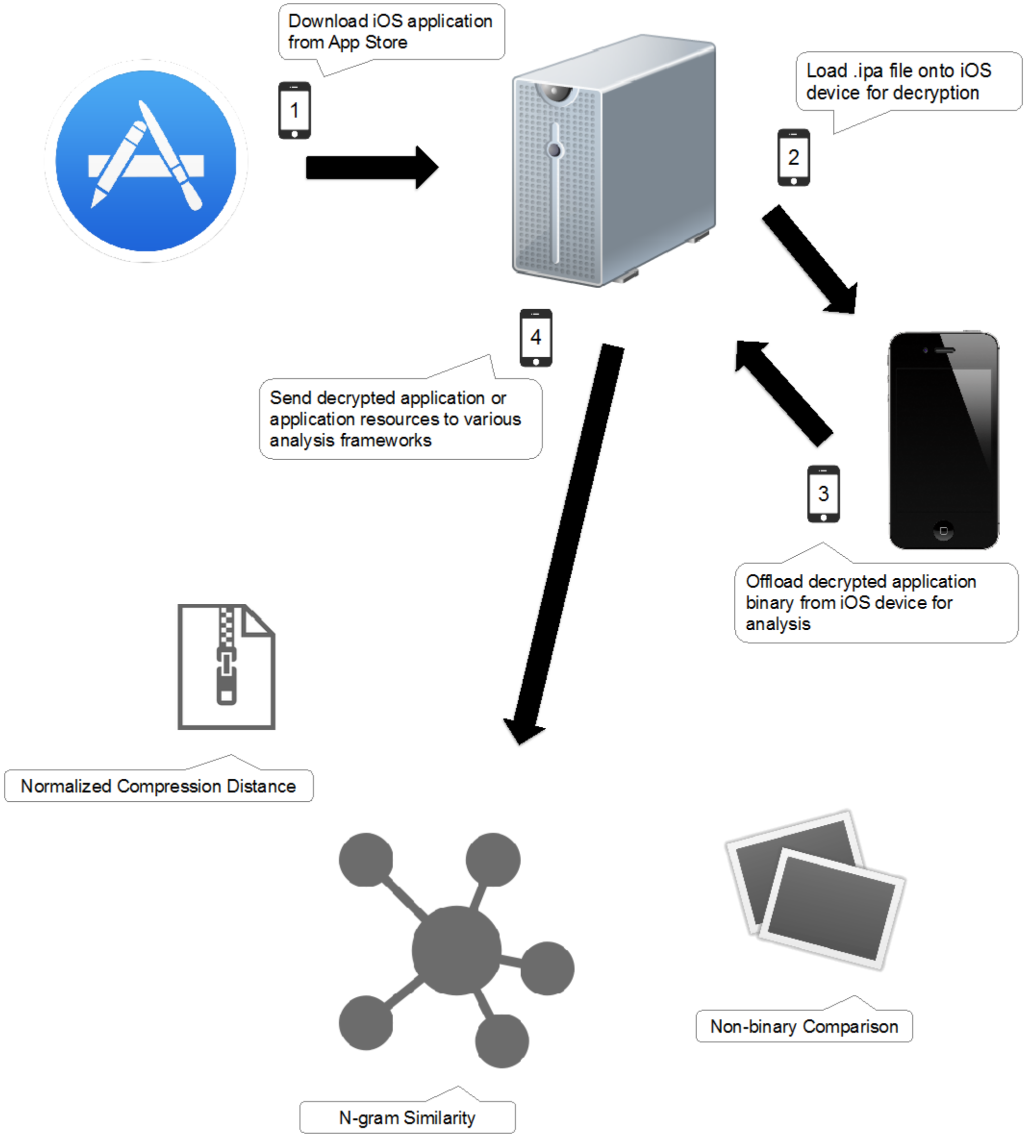
# Fairplay DRM

- Unpublished specs for iOS apps
- Music
    - Unique user key is created for each account
    - Song is encrypted with master key
    - Master key is encrypted with user key
- iOS decryption at load time by kernel
    - Has not been broken

# Fairplay DRM

- Send ipa files to device
- Add *dumpdecrypted* dynamic library and execute application
- Uninstall application and retrieve FAT binary
  - FAT binary includes multiple architectures
- Lipo and store
  - Lipo converts universal binary to single architecture
  - Archive decrypted binary with ipa resources and metadata

# Current Work



Download iOS application from App Store

Load .ipa file onto iOS device for decryption

Offload decrypted application binary from iOS device for analysis

Send decrypted application or application resources to various analysis frameworks

Normalized Compression Distance

N-gram Similarity

Non-binary Comparison

# Normalized Compression Distance

$$NCD(x,y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}$$

- Similarity between two applications

- Pros
  - Easy to apply
  - Fast pairwise comparison
- Cons
  - Limited resolution
  - Scales poorly

# N-Gram Similarity

- Higher resolution comparison than NCD
- Based on minhash
    - n-gram of ARM instructions
- Scalable comparison of application binaries
    - Eliminates pairwise comparisons
    - Similar files get similar hashes

# Non-binary Comparison

- Images
  - App icon
- Detection of UI Duplication
  - Comparison between in category icons
- Dhash (difference hashing algorithm)
  - Convert images to grayscale
  - Reduce image size
  - Compare adjacent pixels

# iOS Recap

- Solved challenges with iOS research
  - Scalability
    - Acquiring apps
    - Fairplay
- Static analysis platform for iOS app analysis
  - Security research
  - Software engineering

# Conclusion

- Scalable analysis for mobile platforms
  - Android
    - Dynamic analysis
    - Malware forensics
  - iOS
    - Static analysis
    - Clone detection

QUESTIONS?

# References

M. Bierma, E. Gustafson, J. Erickson, D. Fritz, and Y. R. Choe, "Andlantis: Large-scale Android Dynamic Analysis," in Proceedings of the 3rd IEEE Mobile Security Technologies Workshop (MoST), 2014.

M. Bierma, N. Ward, K. Wu, and Y. R. Choe, "iOS Clone Detection," Poster Session presented at the 24th USENIX Security Symposium, 2014.