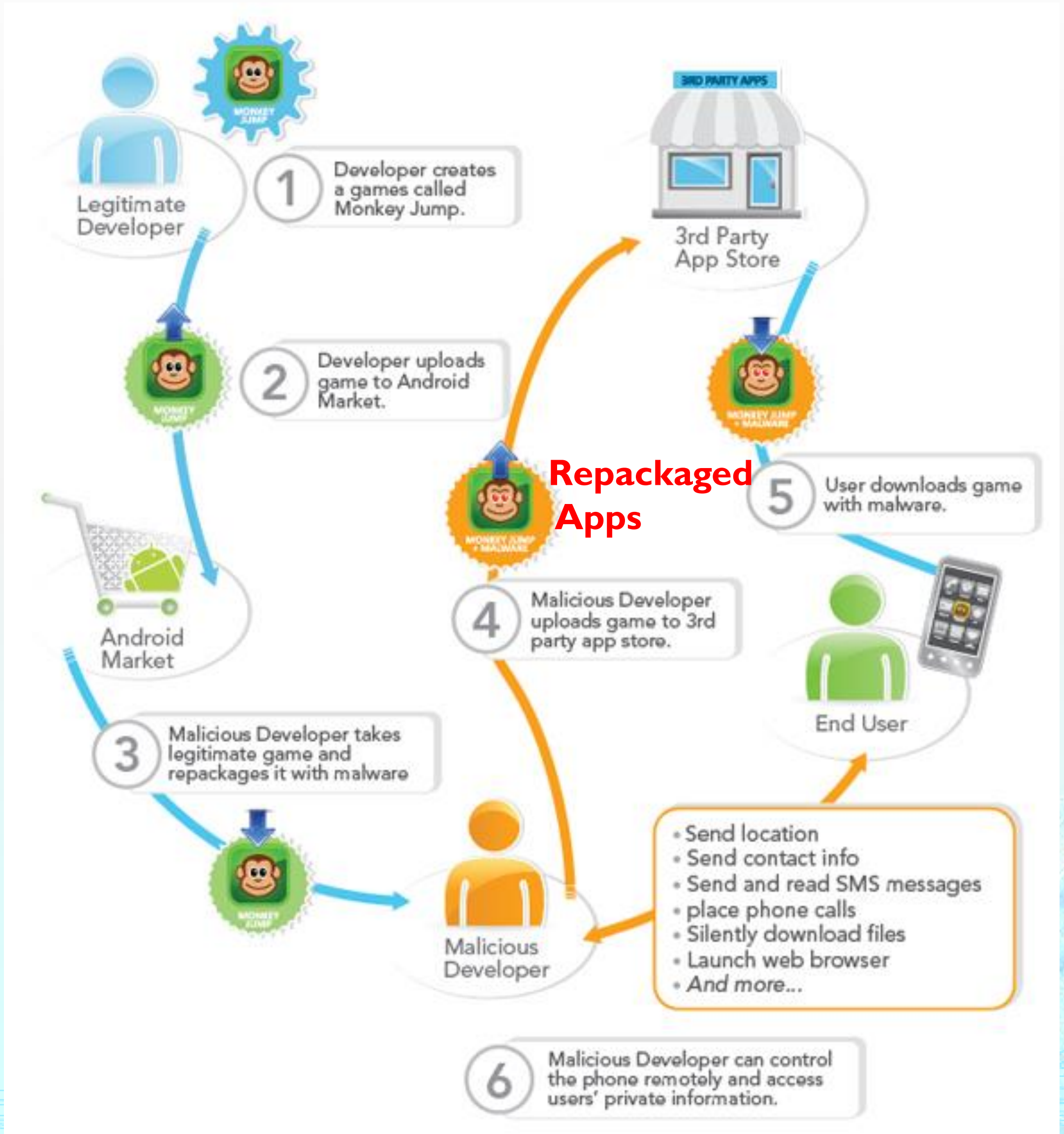
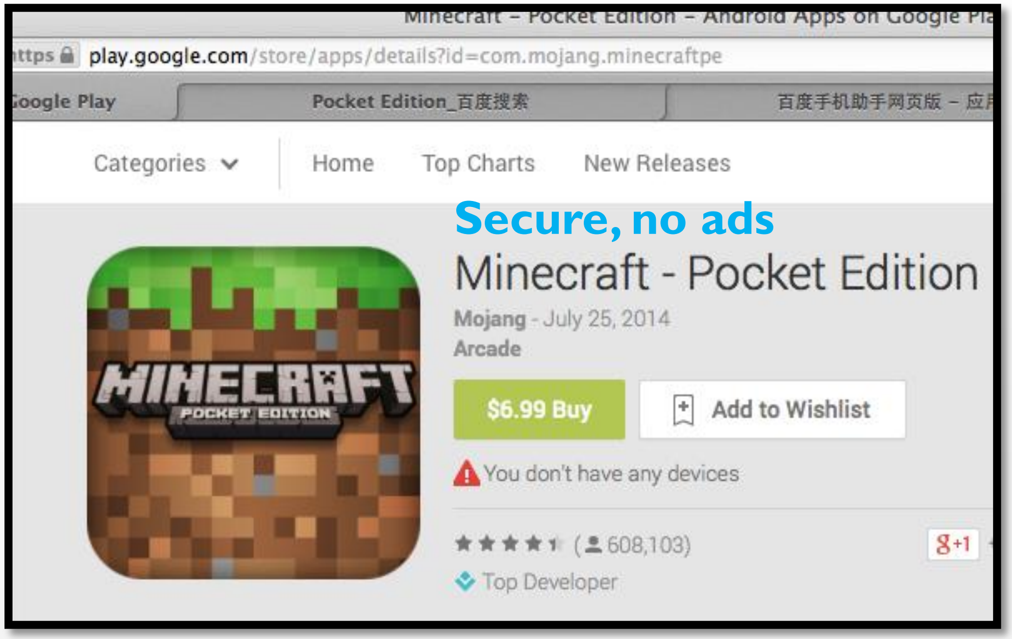
# Android App Protection

**Daniel Xiapu Luo**
**csxluo@comp.polyu.edu.hk**
**Department of Computing**
**The Hong Kong Polytechnic University**

# How an attacker turn your app into a malware?


MONKEY JUMP
Legitimate Developer
1 Developer creates a games called Monkey Jump.
3RD PARTY APPS
3rd Party App Store
2 Developer uploads game to Android Market.
MONKEY JUMP + MALWARE
Repackaged Apps
5 User downloads game with malware.
Android Market
4 Malicious Developer uploads game to 3rd party app store.
End User
3 Malicious Developer takes legitimate game and repackages it with malware
Malicious Developer
• Send location
• Send contact info
• Send and read SMS messages
• place phone calls
• Silently download files
• Launch web browser
• And more...
6 Malicious Developer can control the phone remotely and access users' private information.

Source: businessinsider.com


Minecraft – Pocket Edition – Android Apps on Google Pla

ttps play.google.com/store/apps/details?id=com.mojang.minecraftpe

oogle Play | Pocket Edition_百度搜索 | 百度手机助手网页版 – 应

Secure, no ads

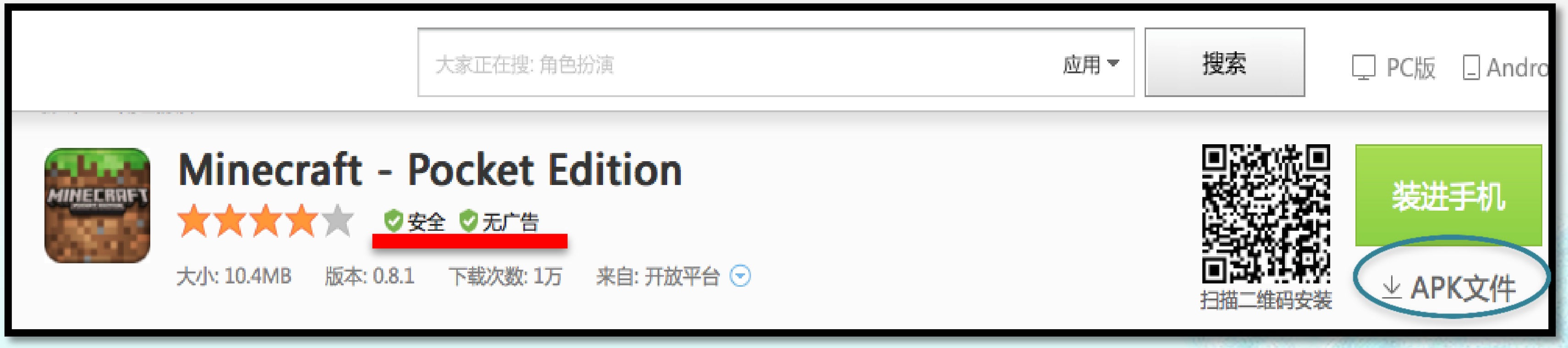
Minecraft - Pocket Edition

Mojang - July 25, 2014
Arcade

You don't have any devices

★★★★½ (608,103)

Top Developer

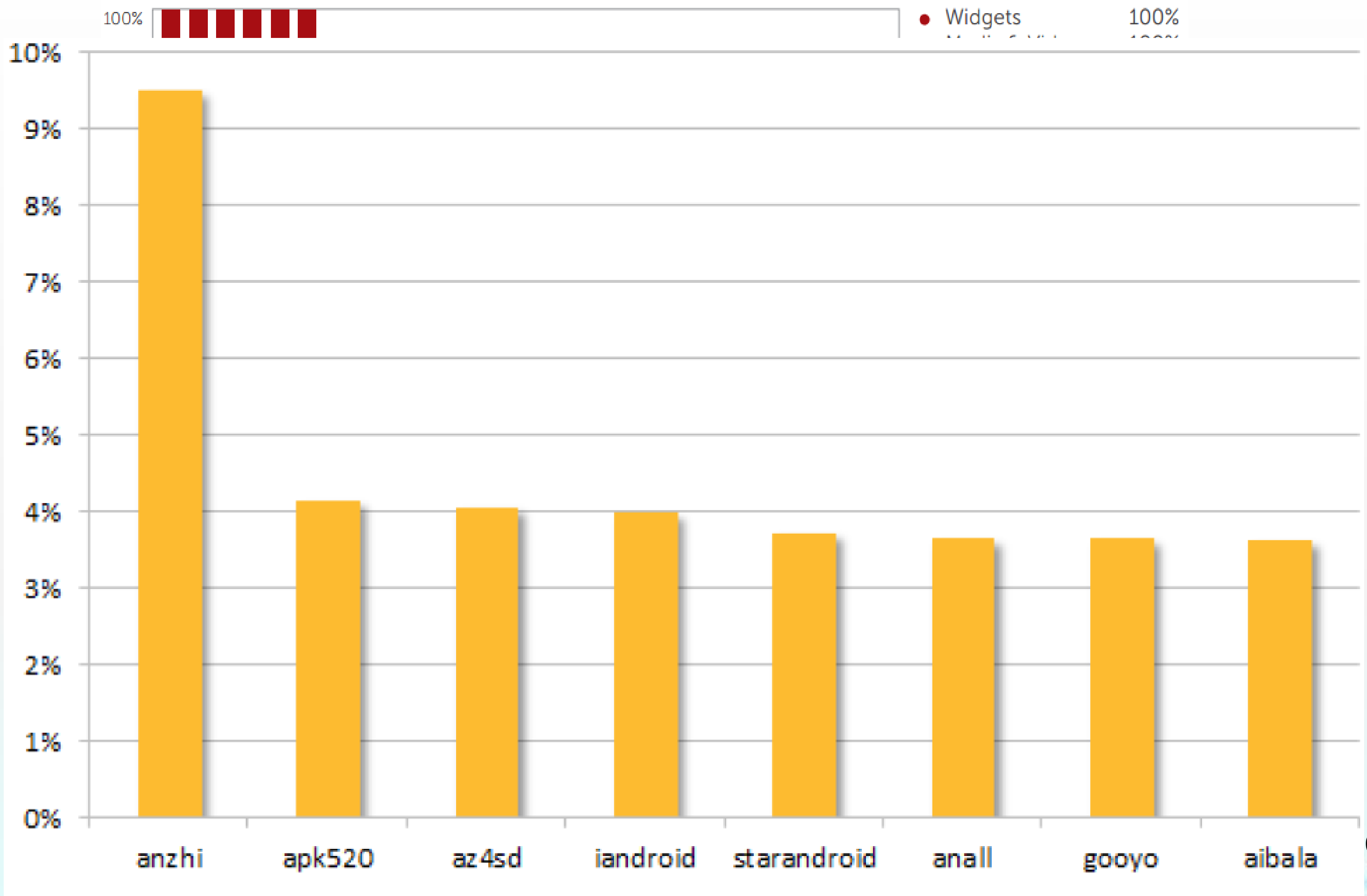


Minecraft - Pocket Edition

★★★★☆ 安全 无广告

大小: 10.4MB 版本: 0.8.1 下载次数: 1万 来自: 开放平台

Save $6.99, but get ads



*Third-party app stores hosting the most malware from January to June 2013*

Source:Symantec

d

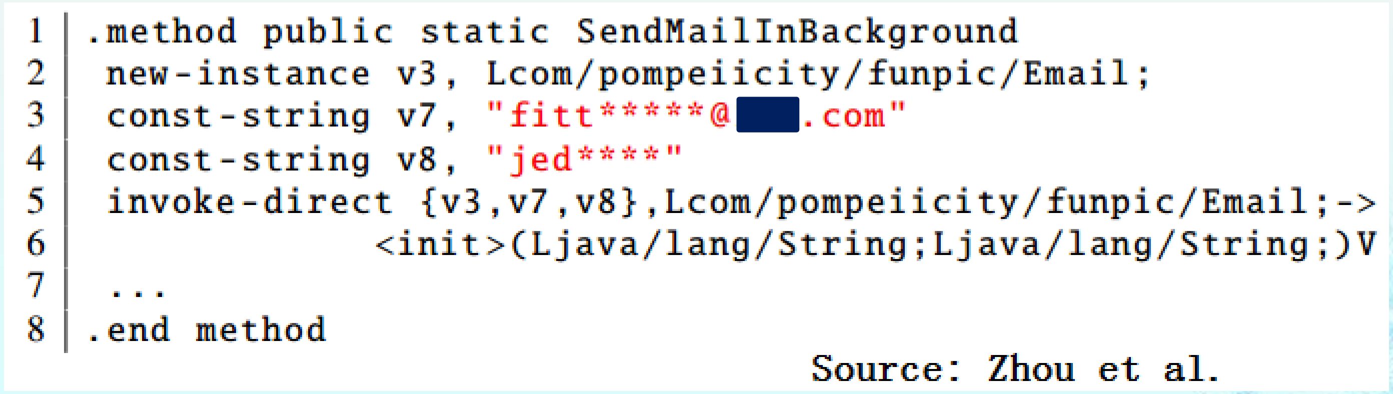
**•90% of the apps of Business, Music & Audio, etc.**

**•...**

# What if your mobile app is reverse-engineered by others?

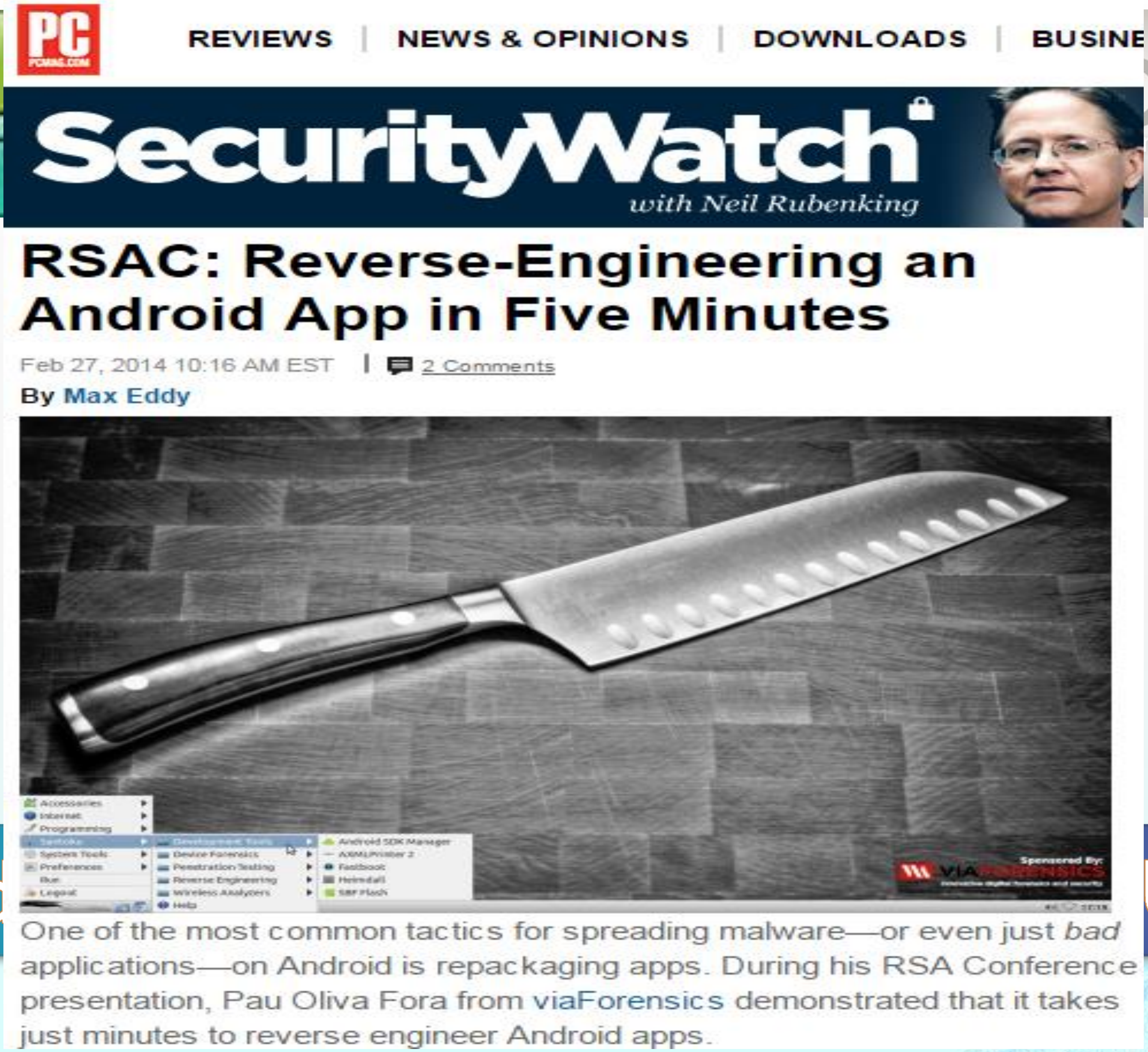
- Core business logic and major algorithms could be learnt by your competitors.

- Credentials in apps.



```
.method public static SendMailInBackground
 new-instance v3, Lcom/pompeiicity/funpic/Email;
 const-string v7, "fitt*****@[redacted].com"
 const-string v8, "jed****"
 invoke-direct {v3,v7,v8},Lcom/pompeiicity/funpic/Email;->
              <init>(Ljava/lang/String;Ljava/lang/String;)V
 ...
.end method
```

Source: Zhou et al.



## RSAC: Reverse-Engineering an Android App in Five Minutes

Feb 27, 2014 10:16 AM EST | 2 Comments

By Max Eddy

One of the most common tactics for spreading malware—or even just *bad* applications—on Android is repackaging apps. During his RSA Conference presentation, Pau Oliva Fora from viaForensics demonstrated that it takes just minutes to reverse engineer Android apps.

## Outline

- Catch Me If You Can
- You Can Run But You Cannot Hide
- Suggestions

# Catch Me If You Can

- Goal
  - Raise the bar for attackers

m4

## Android App Protection

- Techniques used by packers
  - Obfuscation
  - Dynamic class loading
  - Java reflection
  - Dex file modification
  - Native code
  - Emulator detection
  - Anti-debug
  - …

**Hide the code**

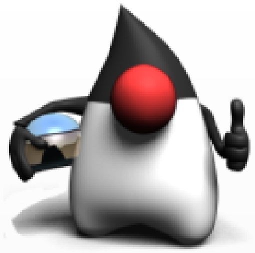
## Obfuscation

- Transform the code to make it difficult to understand or change while keeping its functionalities.
  - Renaming identifier
  - Equivalent expression
  - Encrypting data
  - Splitting and merging functions
  - Complicating control flow
  - Inserting bogus codes
  - …

ProGuard

DexGuard

## Dynamic class loading



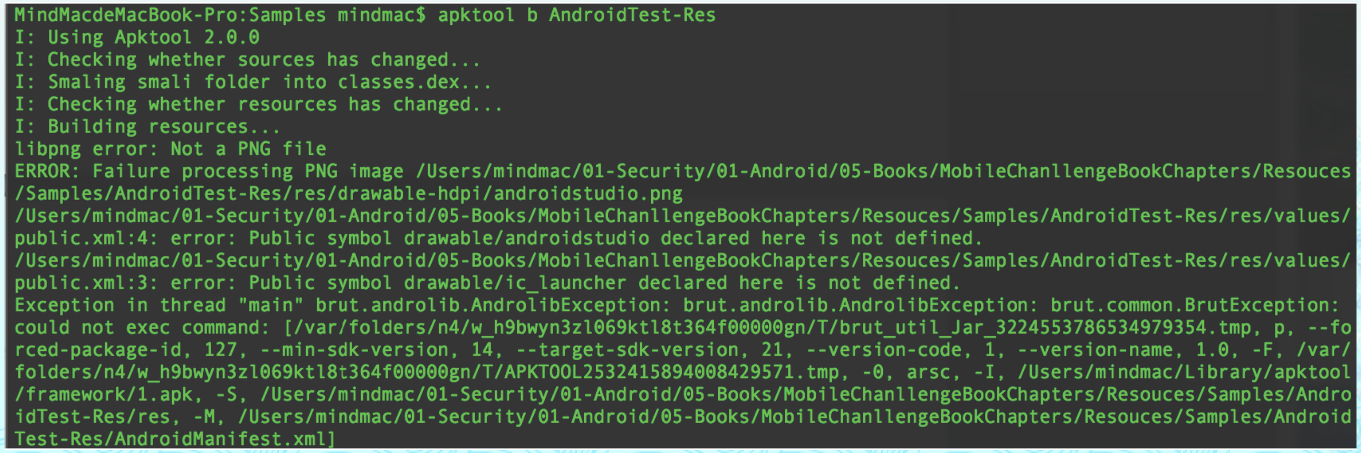
- A feature supported by Java
  - Implement the core business logic in a separated class.
  - The class can be located in the server or released from a native library.
  - Load the class into the runtime when the class is used.

# Java reflection

- A feature supported by Java
- An app can use it to
  - Inspect classes, interfaces, fields and methods at runtime *without* knowing their names,
  - Instantiate new objects dynamically,
  - Invoke methods dynamically,
  - …

# Dex file modification

- Hide the method.
- Bad code to make reverse-engineering tools crash.
  - Opcodes
  - AXML
  - Resource files
  - …



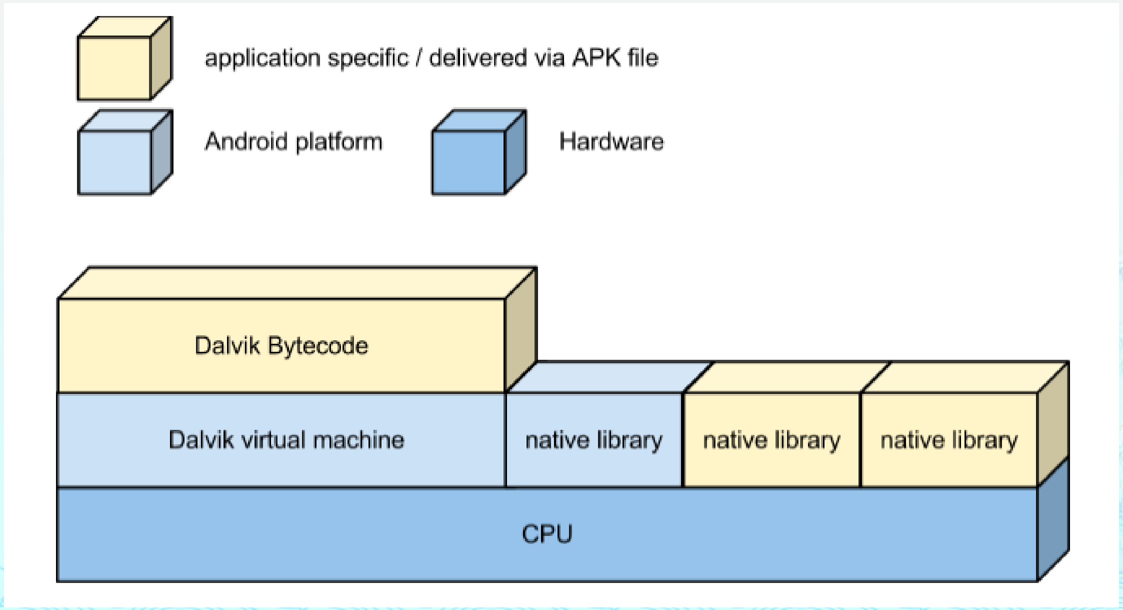
```
MindMacdeMacBook-Pro:Samples mindmac$ apktool b AndroidTest-Res
I: Using Apktool 2.0.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
libpng error: Not a PNG file
ERROR: Failure processing PNG image /Users/mindmac/01-Security/01-Android/05-Books/MobileChanllengeBookChapters/Resouces
/Samples/AndroidTest-Res/res/drawable-hdpi/androidstudio.png
/Users/mindmac/01-Security/01-Android/05-Books/MobileChanllengeBookChapters/Resouces/Samples/AndroidTest-Res/res/values/
public.xml:4: error: Public symbol drawable/androidstudio declared here is not defined.
/Users/mindmac/01-Security/01-Android/05-Books/MobileChanllengeBookChapters/Resouces/Samples/AndroidTest-Res/res/values/
public.xml:3: error: Public symbol drawable/ic_launcher declared here is not defined.
Exception in thread "main" brut.androlib.AndrolibException: brut.androlib.AndrolibException: brut.common.BrutException:
could not exec command: [/var/folders/n4/w_h9bwyn3zl069ktl8t364f00000gn/T/brut_util_Jar_3224553786534979354.tmp, p, --fo
rced-package-id, 127, --min-sdk-version, 14, --target-sdk-version, 21, --version-code, 1, --version-name, 1.0, -F, /var/
folders/n4/w_h9bwyn3zl069ktl8t364f00000gn/T/APKTOOL2532415894008429571.tmp, -0, arsc, -I, /Users/mindmac/Library/apktool
/framework/1.apk, -S, /Users/mindmac/01-Security/01-Android/05-Books/MobileChanllengeBookChapters/Resouces/Samples/Andro
idTest-Res/res, -M, /Users/mindmac/01-Security/01-Android/05-Books/MobileChanllengeBookChapters/Resouces/Samples/Android
Test-Res/AndroidManifest.xml]
```

Source: Hu et al.

# Native code

- App can invoke native code through Java native interface (JNI).

- Native code can modify the dex file in the memory.



Source: A. Blaich

## Emulator detection

- The adversary can observe how an app executes by running it in an emulator (e.g., Qemu).

- Emulator is a software that usually has fixed configuration. So it is different from a real smartphone.
  - Device ID
    - 000000000000000
  - …

# Outline

- Catch Me If You Can
- You Can Run But You Cannot Hide
- Suggestions

# You Can Run But You Cannot Hide

- Can we extract the dex file from a packed app?
  - Yes!
  - **DexHunter**
    - Yueqian Zhang, Xiapu Luo, and Haoyang Yin, *DexHunter: Toward Extracting Hidden Code from Packed Android Applications*, Proceedings of the 20th European Symposium on Research in Computer Security (**ESORICS**), Vienna, Austria, Sept. 2015.
    - **Paper: http://www4.comp.polyu.edu.hk/~csxluo/DexHunter.pdf**
    - **Source code and demo: https://github.com/zyq8709/DexHunter**
- Key insight
  - Dex file will be loaded and run by Android runtime, including Dalvik virtual machine (DVM) and the new Android Runtime (ART), which controls everything.

# Products under Investigation

- 360 http://jiagu.360.cn/
- Ali http://jaq.alibaba.com/
- Baidu http://apkprotect.baidu.com
- Bangcle http://www.bangcle.com/
- Tencent http://jiagu.qcloud.com/
- ijiami http://www.ijiami.cn/

## Summary

- Anti-debugging
  - Anti-ptrace, Anti-JWDP ….
  - But they cannot detect DexHunter.
- Encrypt and hide dex code
- Dynamically modify dex code
- Modify validate values in dex after using them
- Hook functions to prevent dumping
- …
- But DexHunter can still recover the hidden dex code.

## Outline

- Catch Me If You Can
- You Can Run But You Cannot Hide
- Suggestions

## Suggestions

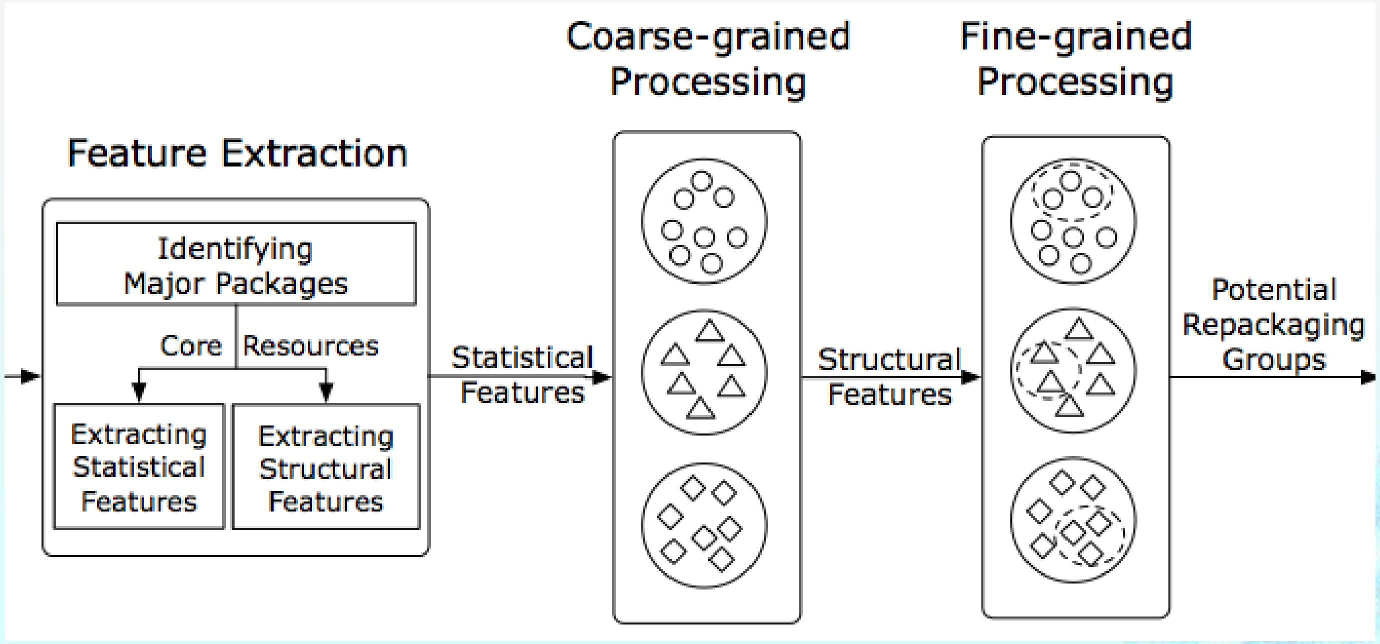
- Do *not* assume that your app cannot be reverse-engineered by others.
- Do not put secrete into your app.
- Protect your apps using various techniques
  - Strong obfuscation algorithms
  - Implement core business logics into native code and then pack the native code
  - Server side verification
  - Customized hardening services
  - …

## Suggestions

- Detect repackaged apps from markets
  - Simple approach
    - Finding apps with similar descriptions, etc.
  - Advanced approach
    - Detect repackaged apps by comparing their codes.
      - It may be affected by the app hardening techniques.
    - Detect repackaged apps by comparing their resources.

## ResDroid

- A scalable approach to detect repackaged apps by leveraging resource features (e.g., GUI, etc.) instead of code.
  - Use statistical features for the coarse-grained processing
  - Use structural features for the fine-grained processing


Coarse-grained Processing
Fine-grained Processing
Feature Extraction
Identifying Major Packages
Core
Resources
Extracting Statistical Features
Extracting Structural Features
Statistical Features
Structural Features
Potential Repackaging Groups

http://www4.comp.polyu.edu.hk/~csxluo/ResDroid.pdf

# THANKS

Thanks my group members and collaborators for contributing to this research:
Yueqian Zhang,Wenjun Hu,Yuru Shao,Haoyang Yin,Xiaobo Ma,Xian Zhan
DexHunter
Paper: http://www4.comp.polyu.edu.hk/~csxluo/DexHunter.pdf
Source code and demo : https://github.com/zyq8709/DexHunter
ResDroid
Paper: http://www4.comp.polyu.edu.hk/~csxluo/ResDroid.pdf
Our other tools and papers on Android security:
http://www4.comp.polyu.edu.hk/~csxluo