Detecting Security Vulnerabilities in Android Applications

محمد مختاری (۹۳۱۳۱۰۱۴)

استاد سمینار: دکتر بابک صادقیان

استاد راهنما: دکتر حمیدرضا شهریاری

فهرست

مقدمه

- چرخه‌ی توسعه‌ی کوتاه نرم‌افزار
- الگوی باز انتشار نرم‌افزارها در Google Play-Store
- عدم پیروی از راهبردهای امنیتی
- از ۲۱۰۷ برنامه معروف بررسی شده توسط مرکز تحقیقات HP نود درصد آن‌ها آسیب‌پذیر بوده‌اند.[2]

مقدمه

OS / Parameter	Android	iOS	Symbian	Blackberry	Windows Phone	WebOS	Ubuntu	Firefox
Package Manager	Google Play, APK	iTunes	Nokia Store	BlackBerry Link	Zune Software (not since Windows 8	OTA deployment, webOS through App store, Web URL, Precentral, .ipk	Ubuntu Touch through App store, Web URL	Firefox OS Packaged Apps
Runs On	Smartphones, tablet, computers, TVs, cars and wearable devices	iPhone, iPad, iPod Touch	Smartphones	Smartphones	personal computers, smartphones, server computers and embedded devices	TVs and Smart watches	Personal computers, Servers, smartphones, tablet computers (UbuntuTouch), smartTVs (Ubuntu TV)	Smartphones, Tablet and computers
Market Share[3]	48.8%	17.2%	0.1%	11.1%	19.5%	--	--	--
Market Size[14]	Very High	High	Very low	Low	Medium	Very low	Very low	Very low
Application Store	Google Play	App Store	Nokia Ovi Store	BlackBerry World	Windows Phone Store	Palm App Catalog	Ubuntu Store	Firefox Marketplace, Web URL
Non-English Language Support	Partial	Yes	Yes	Yes	Yes	Partial	Yes	Yes
Virtual Machine	Allowed	Not Allowed	Allowed	Allowed	Allowed	Allowed	Allowed	Not, Only simulator available.
Debugger	Debugger available	Debugger available	Debugger available	Debugger Available	Debugger available	Debugger available	Debugger Available	Debugger available
Cross-Platform Deployment	Android only	iPhone, iPad, iPod Touch	Compile per target	BlackBerry only	Windows Mobile, Windows FU, Windows CE	webOS, Palm only	HTML5 app to be av*ailable web browser.	Web browser on other platform
GUI	Android	Cocoa Touch	Avkon	Cascades	Visual Studio	Graphical (Luna)	Ubuntu SDK	Firefox browser, Firebug
Documentat-ion Available At	www.android.com	www.apple.com/ios/	symbian.nokia.com	us.blackberry.com/apps-software/blackberry7/	www.windowsphone.com	www.hpwebos.com www.openwebosproject.org	www.ubuntu.com	mozilla.org/firefox/os
Tool for Reverse Engineering of App	Apk tool, Dex2jar, JD-Compiler, XDA Auto tool	iRET Toolkit, Windows Explorer, oTool, iExplorer, Class-dump-z	Carbidge.c++, IDA Pro, APP Trk, SISWare, ARM assembler	JD-GUI, Notepad, VSMTool, COD extractor	Decompresser, Visual Studio / Notepad, .Net Decompiler	Binwalk	Bokken	gdb-debugger, b2g-ps
Future Prospect	Very High	High	Low	Low	Medium	Low	Low	Low

آمار سال های ۲۰۰۹ و ۲۰۱۰:
آمار جولای سال ۲۰۱۵:

Q09 Units/1k	3Q09 Market Share (%)
18,314.8	44.6
1,424.5	3.5
7,040.4	17.1
8,522.7	20.7
3,259.9	7.9
1,918.5	4.7
612.5	1.5
41,093.3	100.0

روش های تحلیل نرم افزار

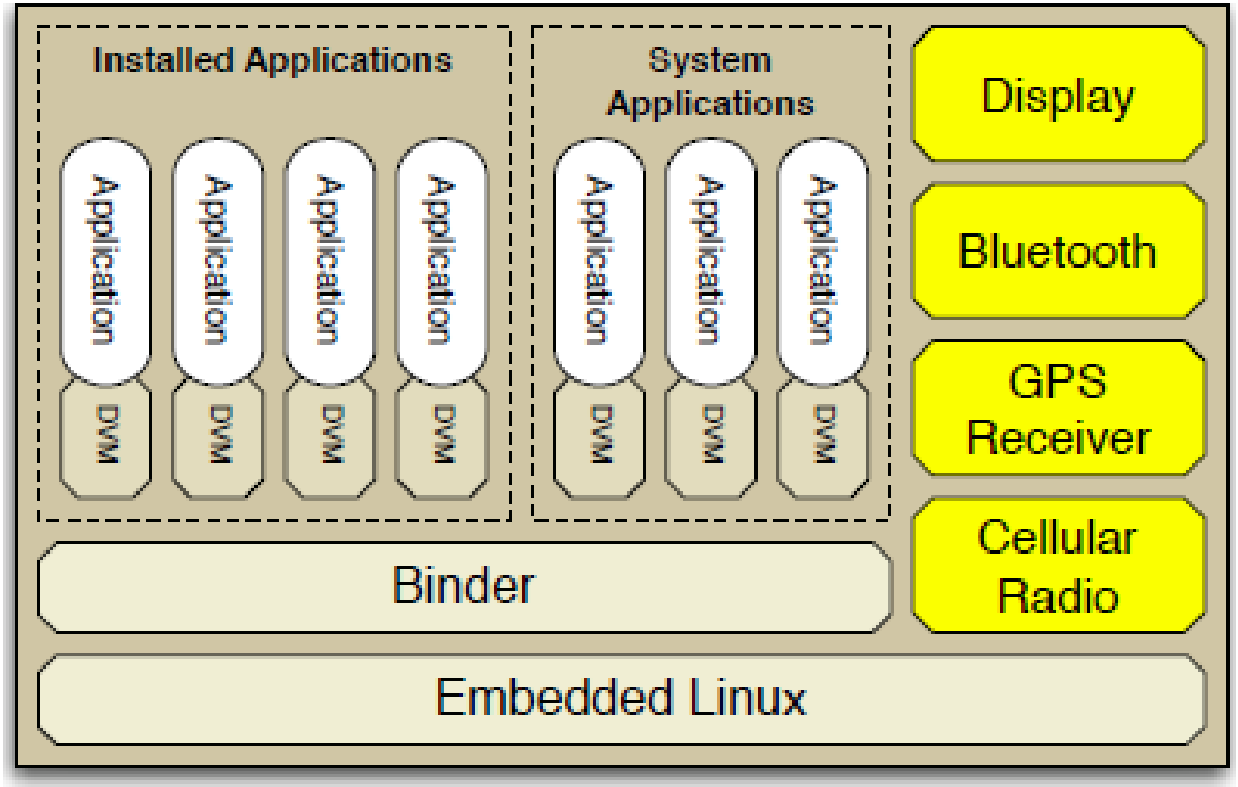
- White Box
- Black Box

- Static Analysis
- Dynamic Analysis

- Symbolic execution
- Taint analysis

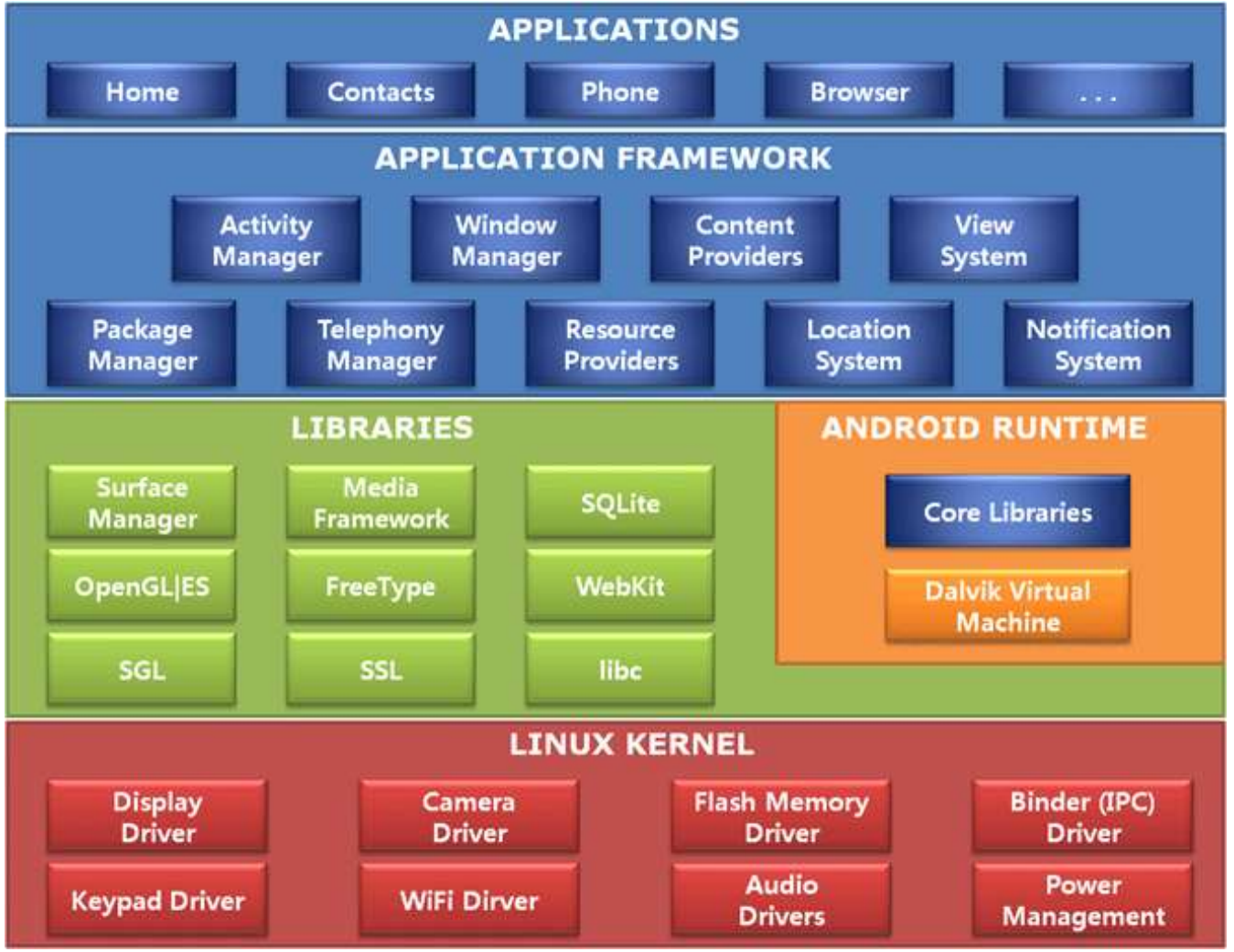
توضیحات پایه

توضیحات پایه [1],[7],[19]



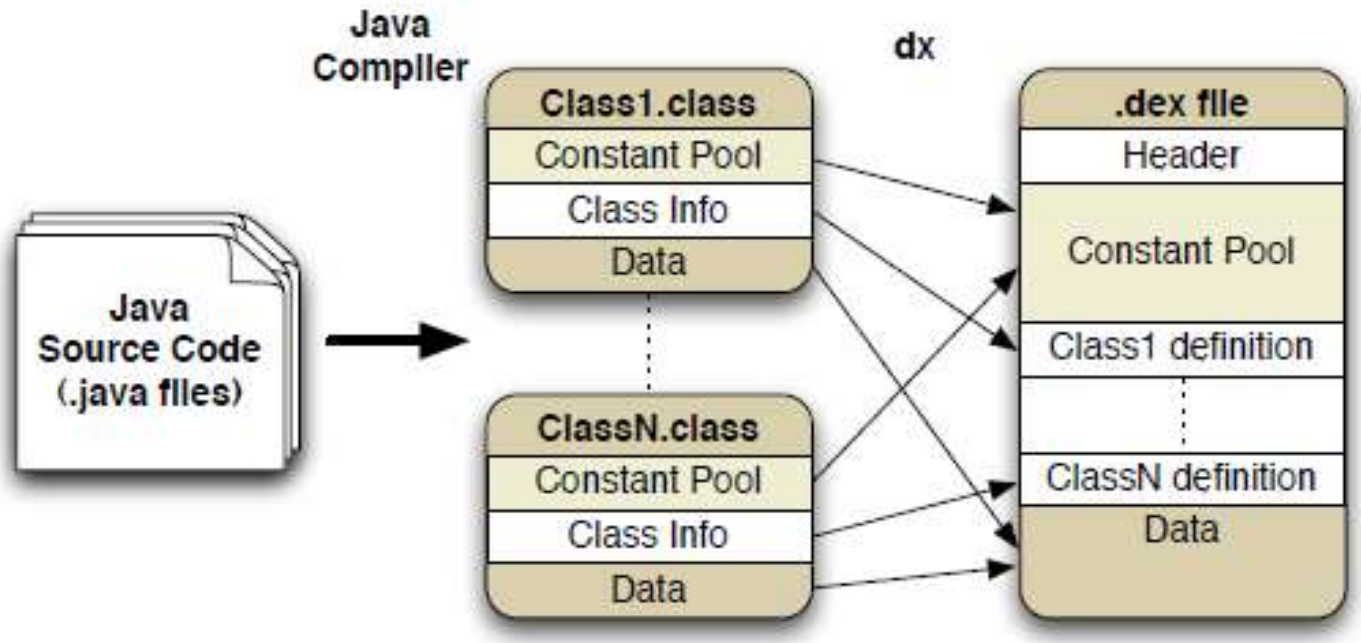
- چهارچوب سیستم عامل
- Linux Kernel (ARM)
- Binder
- Dalvik VM

توضیحات پایه [19],[7],[1]

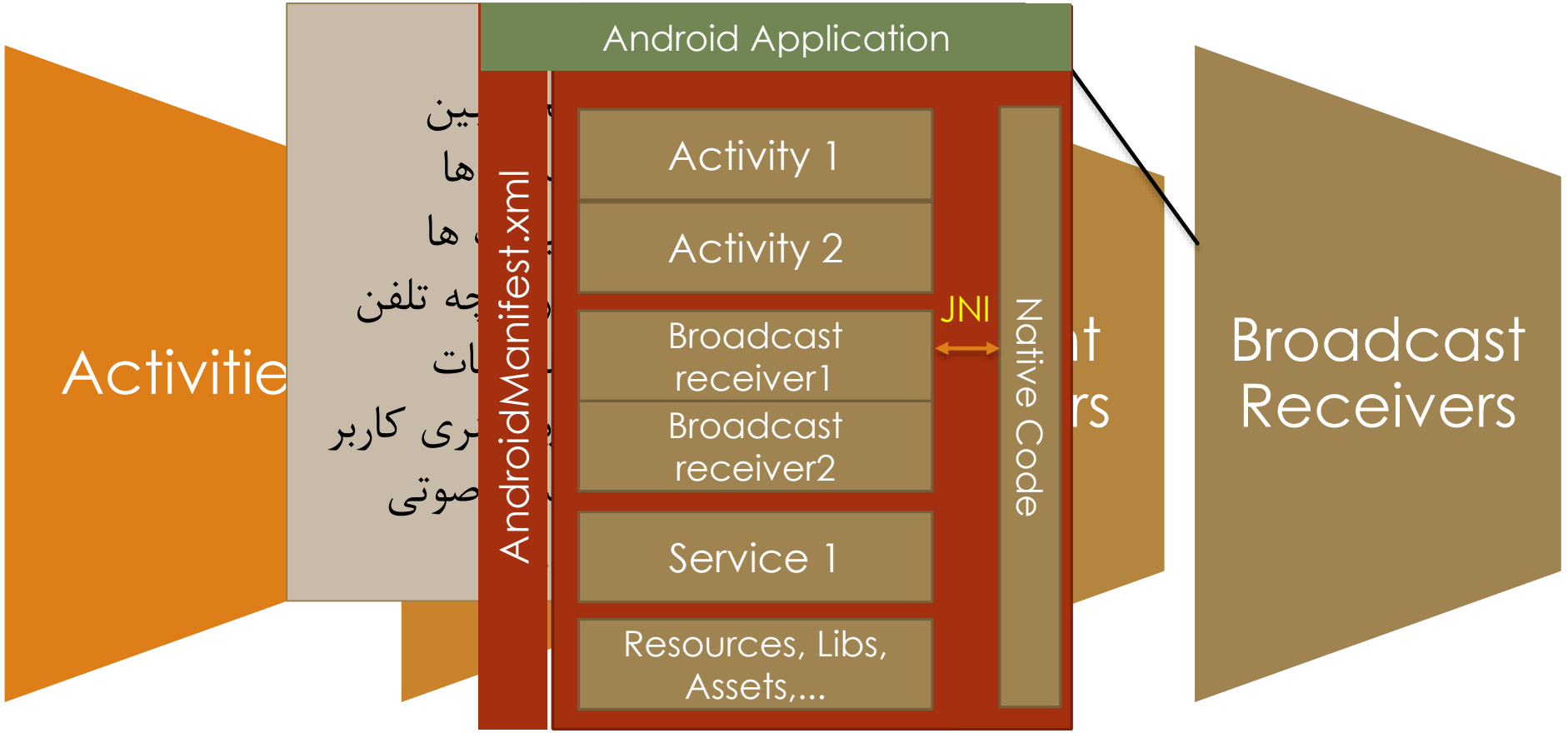


توضیحات پایه – ادامه...

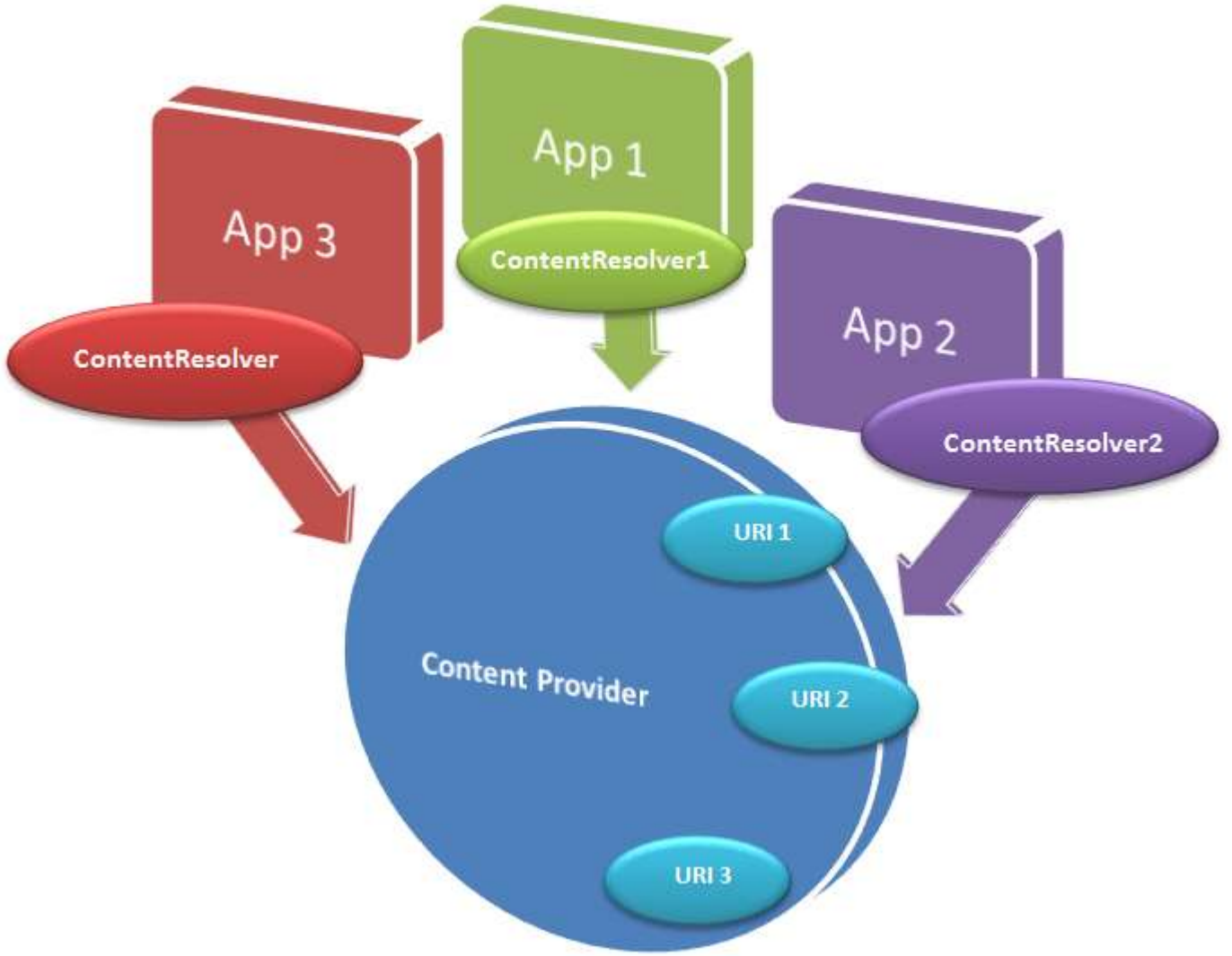
- ساختار برنامه



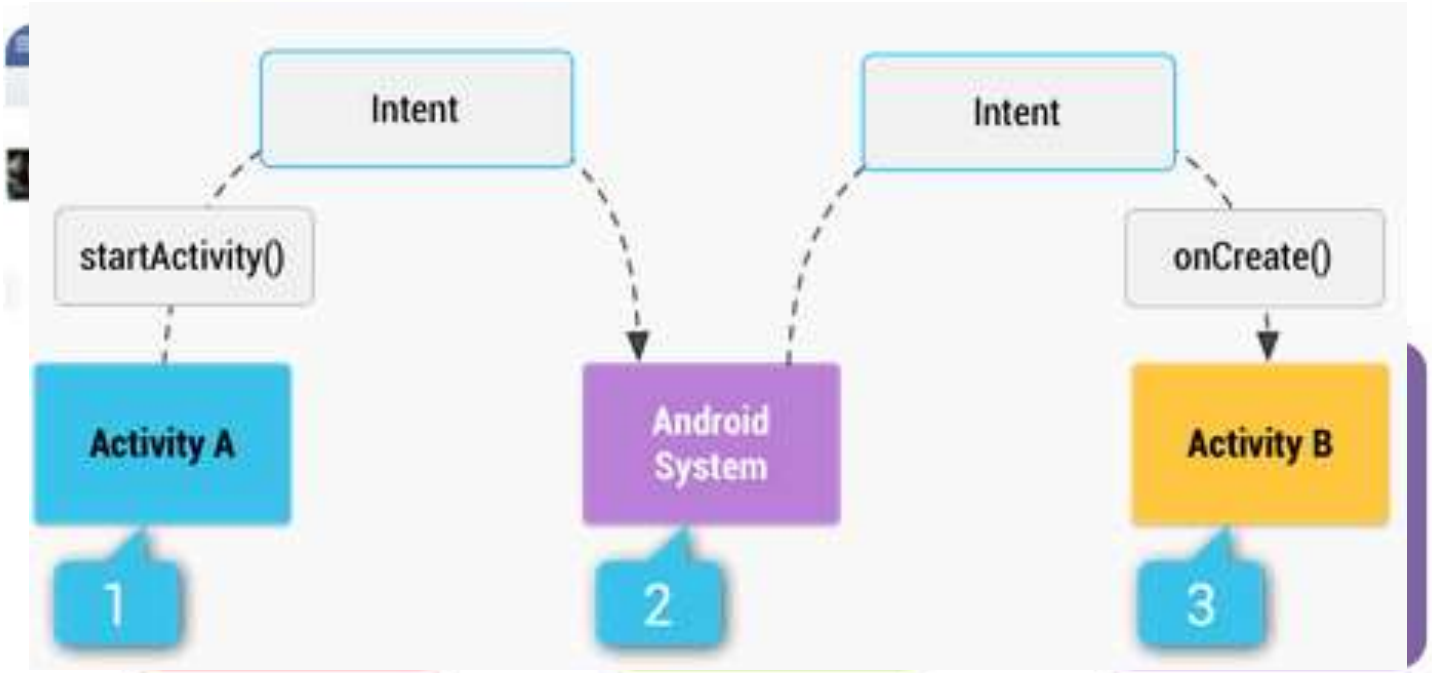
توضیحات پایه– اجزای برنامه در اندروید



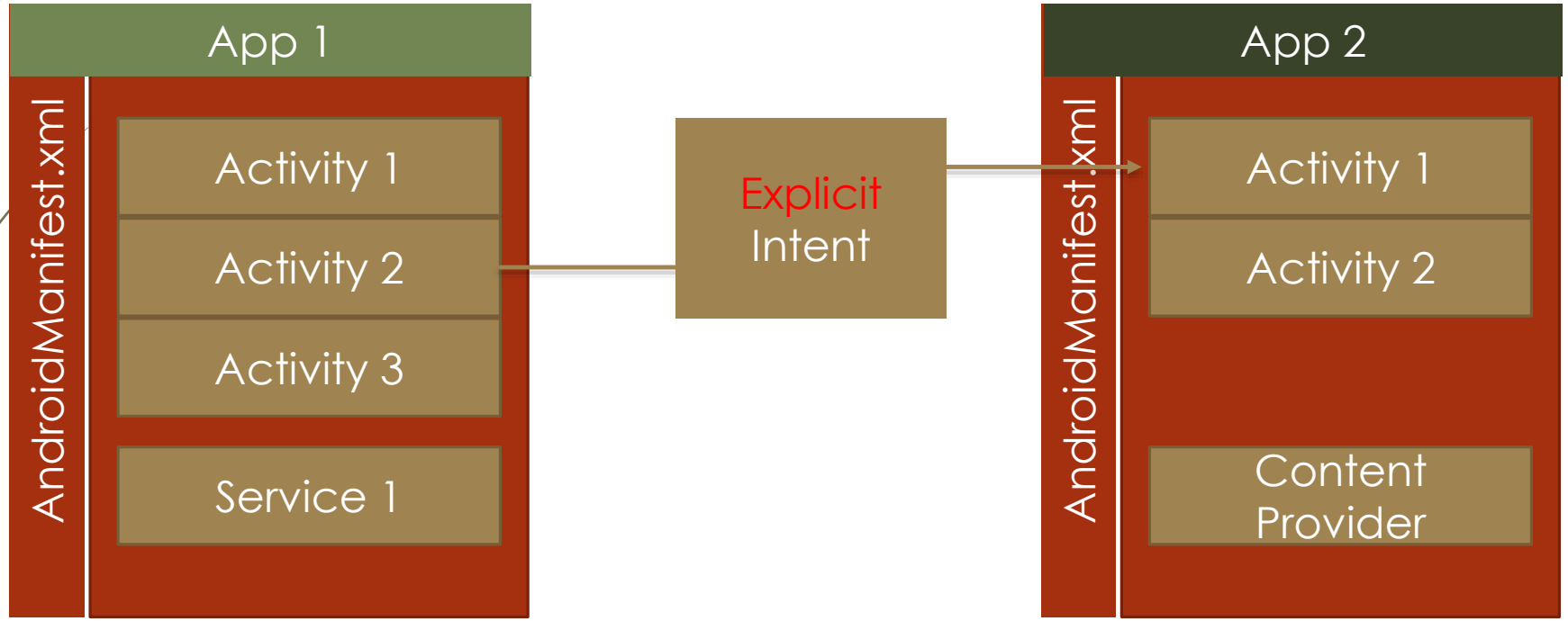
Content Provider – **توضیحات پایه**



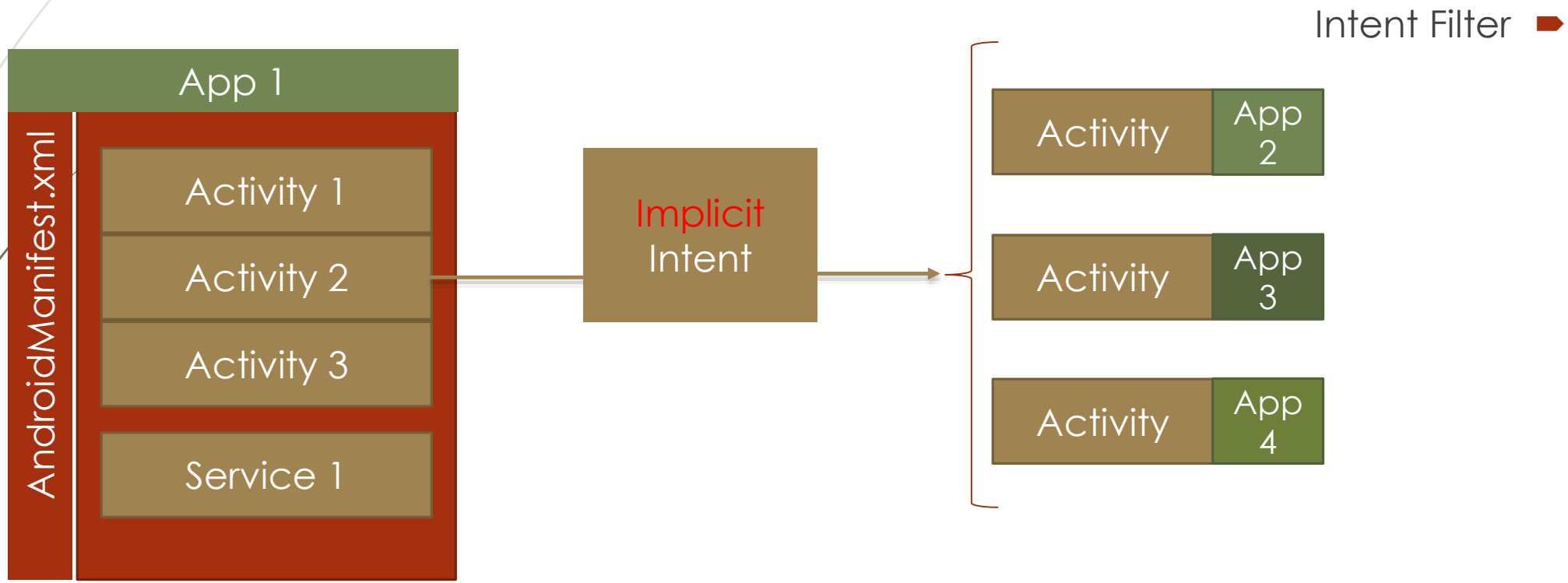
Intent – **توضیحات پایه**



Intent – **توضیحات پایه**



Intent – **توضیحات پایه**



Mechanism	Description	Security Issue
Linux mechanisms		
POSIX users	Each application is associated with a different user ID (or UID).	Prevents one application from disturbing another
File access	The application's directory is only available to the application.	Prevents one application from accessing another's files
Environmental features		
Memory management unit (MMU)	Each process is running in its own address space.	Prevents privilege escalation, information disclosure, and denial of service
Type safety	Type safety enforces variable content to adhere to a specific format, both in compiling time and runtime.	Prevents buffer overflows and stack smashing
Mobile carrier security features	Smart phones use SIM cards to authenticate and authorize user identity.	Prevents phone call theft
Android-specific mechanisms		
Application permissions	Each application declares which permission it requires at install time.	Limits application abilities to perform malicious behavior
Component encapsulation	Each component in an application (such as an activity or service) has a visibility level that regulates access to it from other applications (for example, binding to a service).	Prevents one application from disturbing another, or accessing private components or APIs
Signing applications	The developer signs application `.apk` files, and the package manager verifies them.	Matches and verifies that two applications are from the same source
Dalvik virtual machine	Each application runs in its own virtual machine.	Prevents buffer overflows, remote code execution, and stack smashing

معرفی تعدادی از آسیب پذیری ها

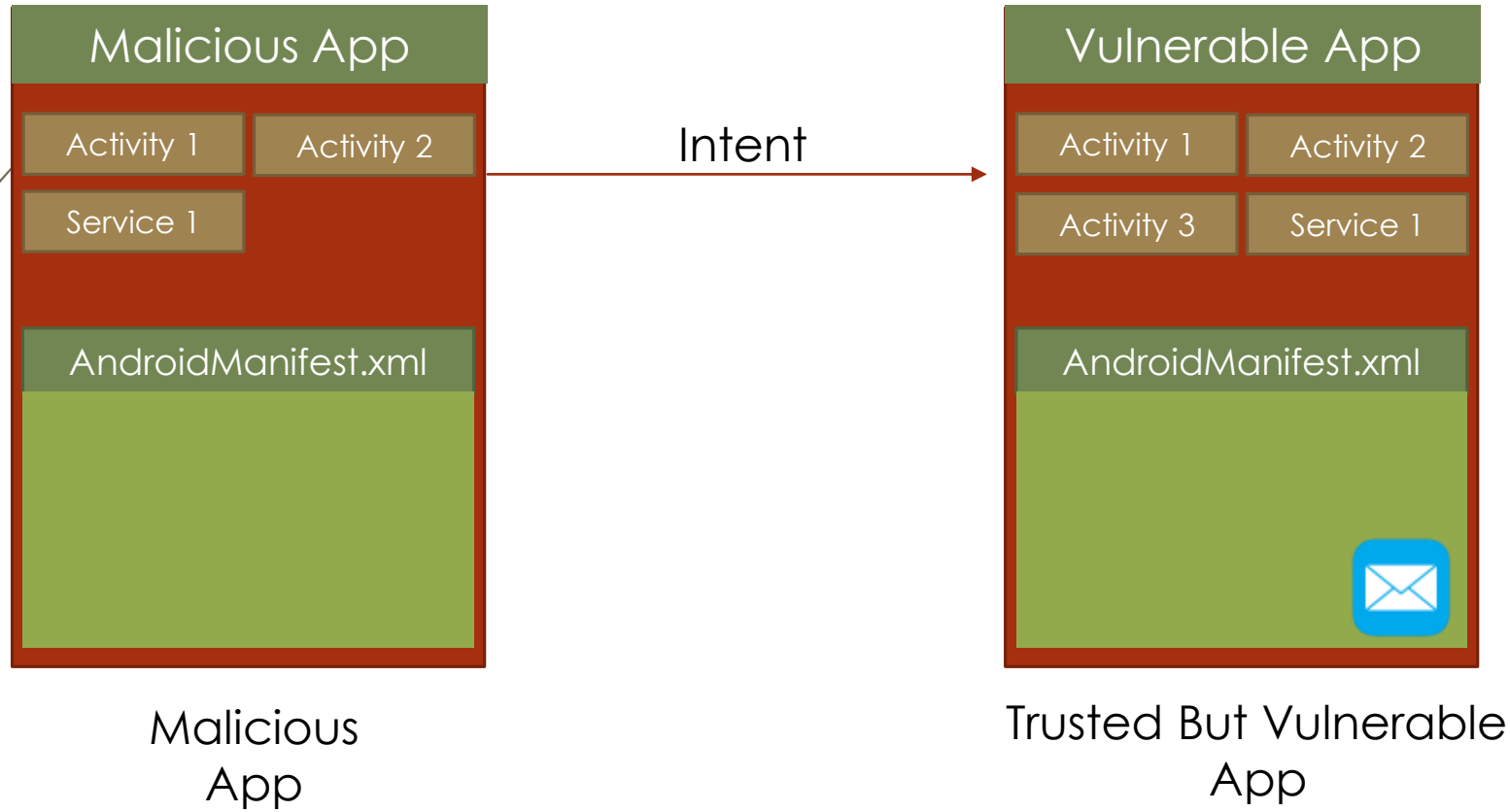
معرفی آسیب پذیری ها [3]

- نشت توانایی (Capability leak)
- پیمایش دایرکتوری در ارائه‌دهنده‌ی محتوا (Content Provider directory traversal)
- پیاده سازی اشتباه کلاس برای ارتباط SSL (X509TrustManager implemented improperly)
- مجوز دسترسی عمومی فایل (Public file access permission)
- ثبت کردن اطلاعات حساس (Log sensitive information)

معرفی آسیب پذیری های معمول [3]

سناریوی اول

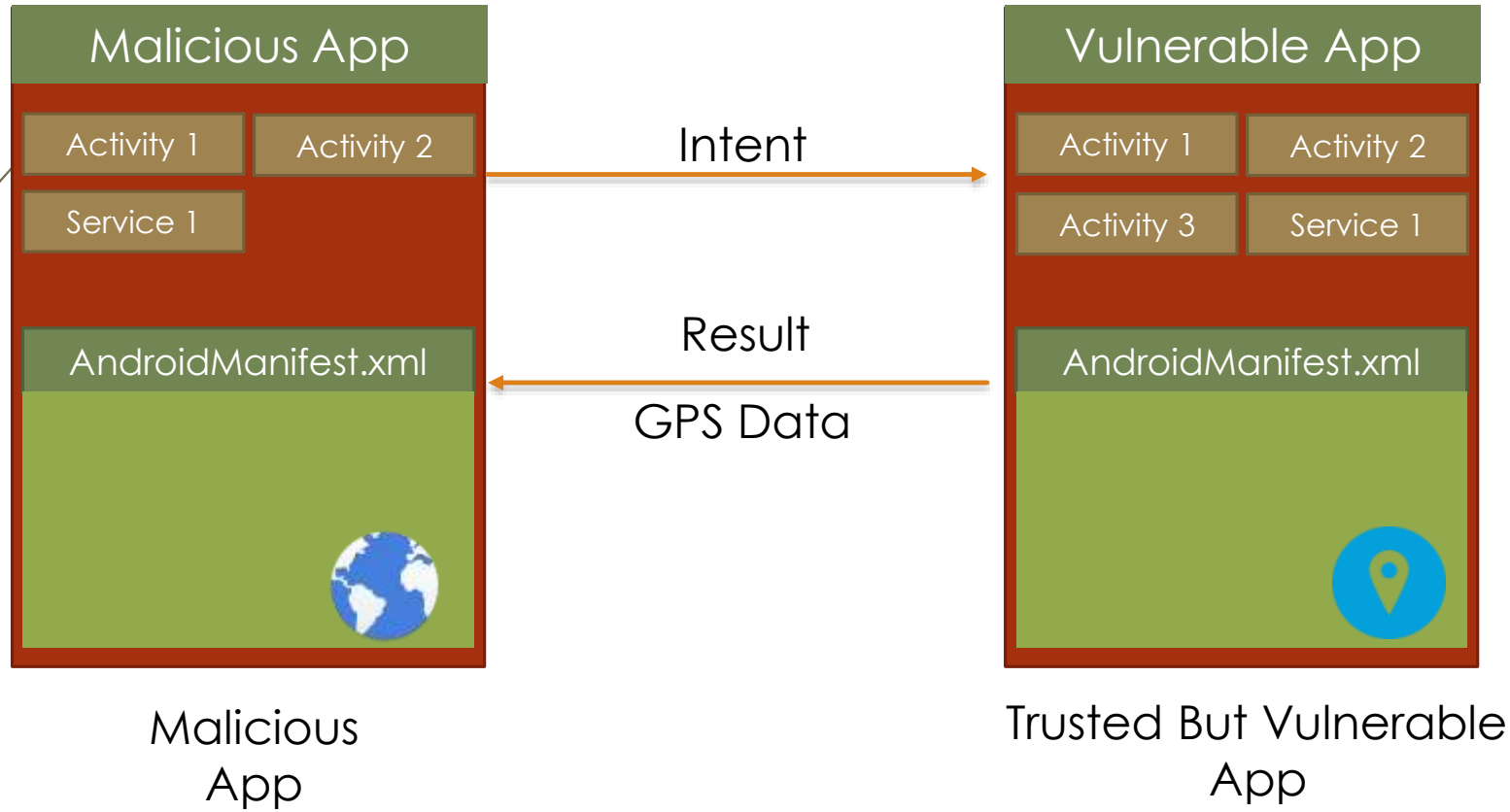
- نشت توانایی (Capability leak)



معرفی آسیب پذیری های معمول [3]

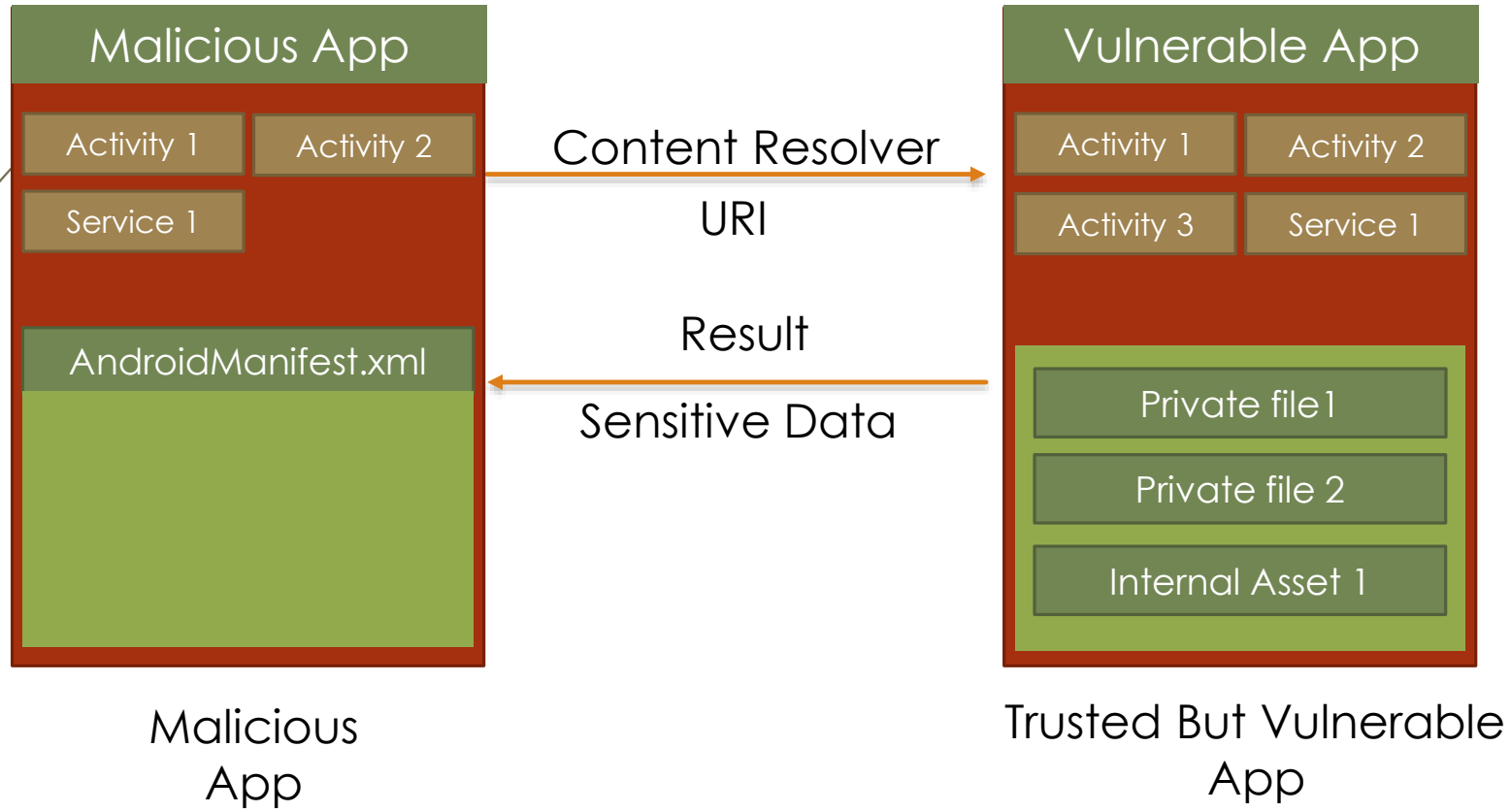
سناریوی دوم

- نشت توانایی (Capability leak)



معرفی آسیب پذیری های معمول [3]

- پیمایش دایرکتوری در ارائه‌دهنده‌ی محتوا (Content Provider directory traversal)



معرفی آسیب پذیری های معمول [3]

- پیاده سازی اشتباه کلاس برای ارتباط SSL (X509TrustManager implemented improperly)

No Op

Trusting Self-Signed Only

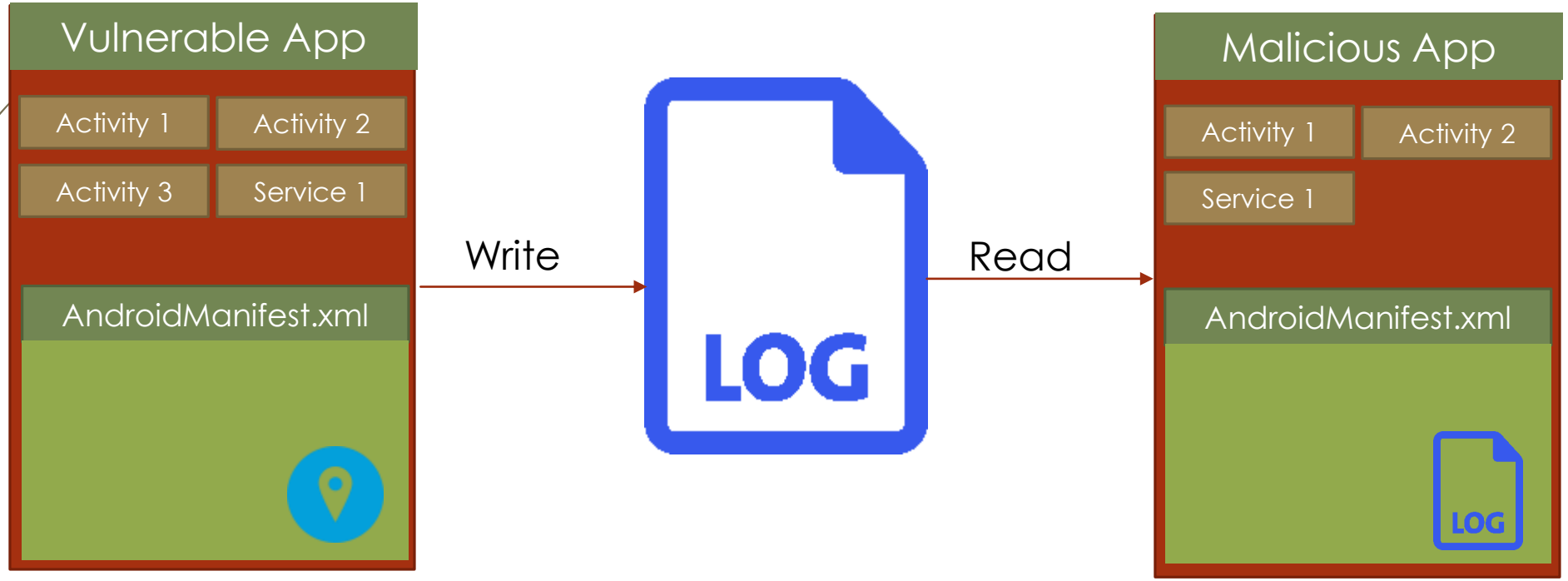
Checking Chain's Validity Only

معرفی آسیب پذیری های معمول [3]

- مجوز دسترسی عمومی فایل (Public file access permission)

معرفی آسیب پذیری های معمول [3]

- ثبت کردن اطلاعات حساس (Log sensitive information)



کارهای انجام شده

A Study of Android Application Security (2011)

- اولین تلاش برای دیکامپایل کردن برنامه های اندروید
- ارائه ی یک دیکامپایلر برای دالویک به نام ded
  - تبدیل کدهای DVM به JVM

- یافتن تعداد زیادی استفاده از اطلاعات شخصی (IMEI، tracking و ...)
- نیافتن مدرکی دال بر استفاده ی نادرست از امکانات تلفنی
- ۵۱ درصد از برنامه های مورد مطالعه حاوی کتابخانه های تبلیغ
- عدم پیروی از راهبردهای امنیتی در توسعه ی نرم افزار

A Study of Android Application Security (2011)

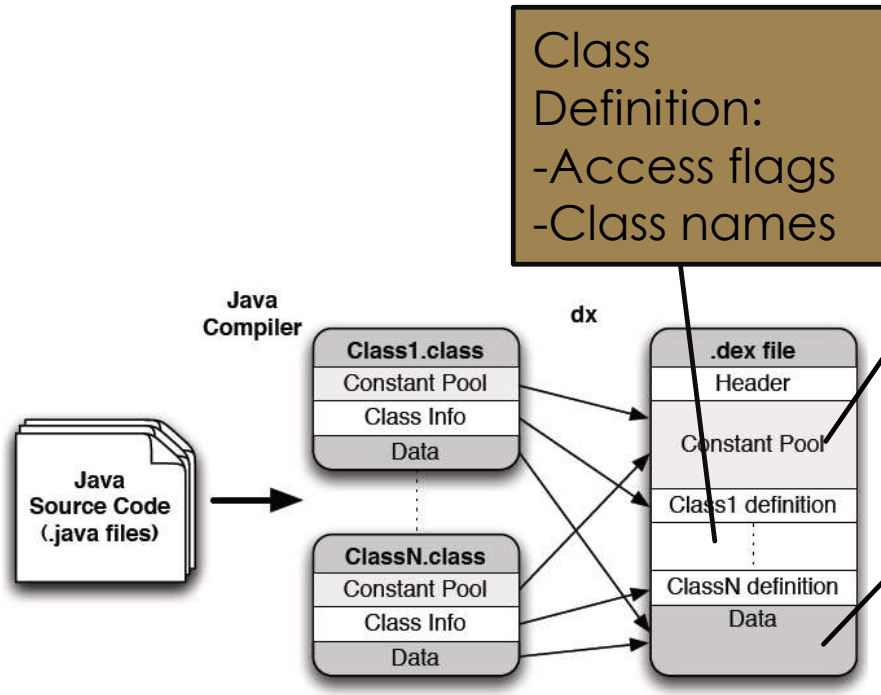


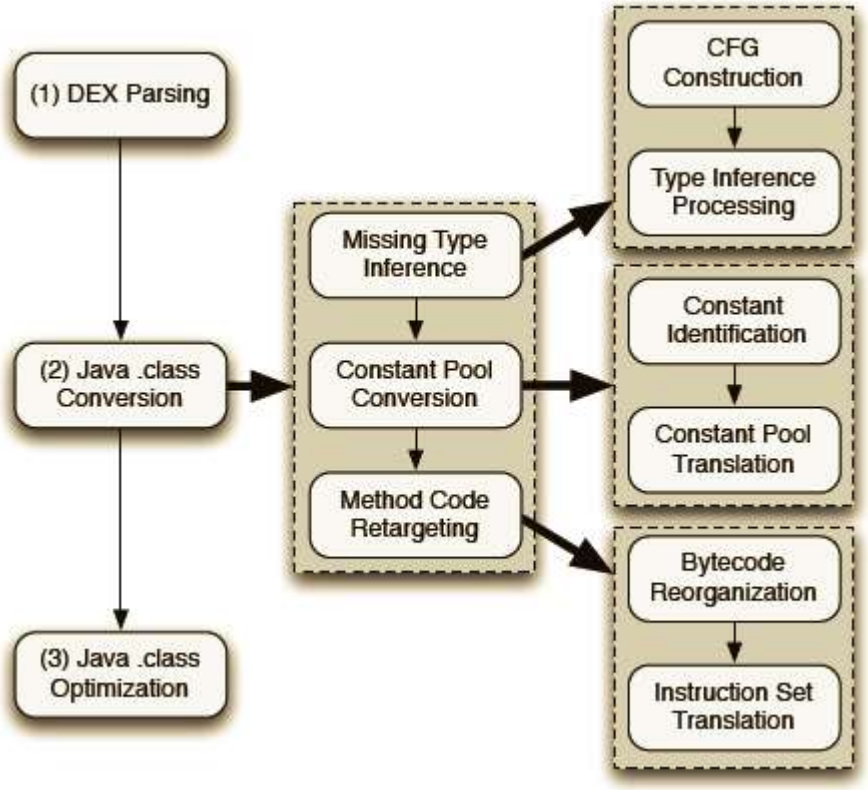
Figure 2: Compilation process for DVM applications

Class Definition:
-Access flags
-Class names

Constant Pool:
-References to other classes
-Method names
-Numerical constants

Data:
-Method code
-Info related to methods
-Variables

A Study of Android Application Security (2011)



- فرآیند retargeting ۳ مرحله ای:
  - بازیافتن اطلاعات نوع داده ها
  - تبدیل constant pool
  - تبدیل بایت کد

Dalvik	Java
add-int d_0, s_0, s_1	iload s'_0 iload s'_1 iadd istore d'_0

A Study of Android Application Security (2011)

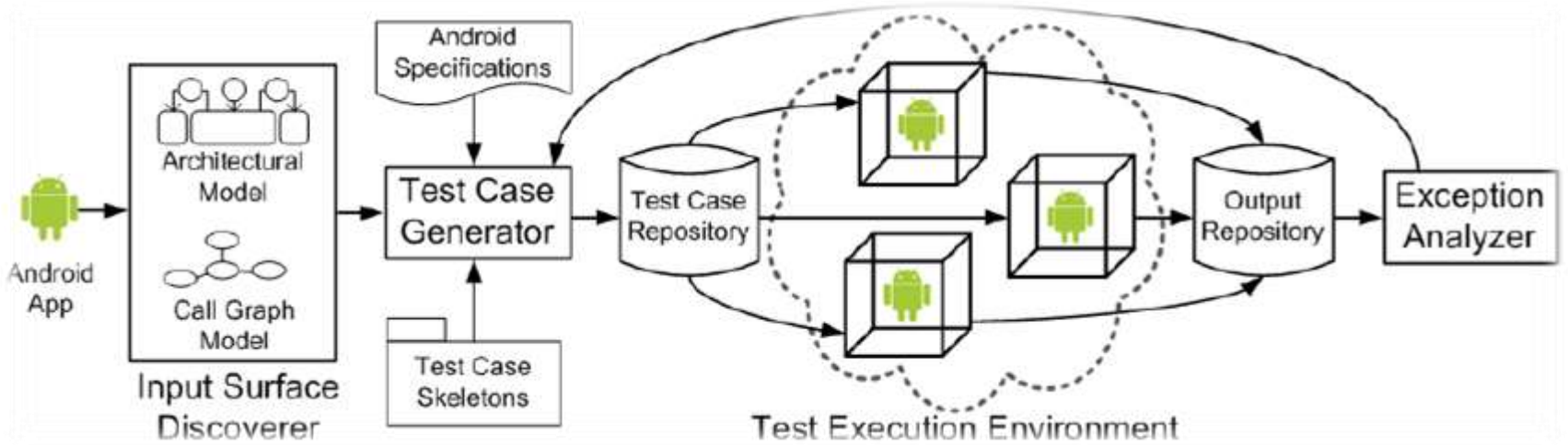
Table 1: Studied Applications (from Android Market)

Category	Total Classes	Retargeted Classes	Decompiled Classes	LOC
Comics	5627	99.54%	94.72%	415625
Communication	23000	99.12%	92.32%	1832514
Demo	8012	99.90%	94.75%	830471
Entertainment	10300	99.64%	95.39%	709915
Finance	18375	99.34%	94.29%	1556392
Games (Arcade)	8508	99.27%	93.16%	766045
Games (Puzzle)	9809	99.38%	94.58%	727642
Games (Casino)	10754	99.39%	93.38%	985423
Games (Casual)	8047	99.33%	93.69%	681429
Health	11438	99.55%	94.69%	847511
Lifestyle	9548	99.69%	95.30%	778446
Multimedia	15539	99.20%	93.46%	1323805
News/Weather	14297	99.41%	94.52%	1123674
Productivity	14751	99.25%	94.87%	1443600
Reference	10596	99.69%	94.87%	887794
Shopping	15771	99.64%	96.25%	1371351
Social	23188	99.57%	95.23%	2048177
Libraries	2748	99.45%	94.18%	182655
Sports	8509	99.49%	94.44%	651881
Themes	4806	99.04%	93.30%	310203
Tools	9696	99.28%	95.29%	839866
Travel	18791	99.30%	94.47%	1419783
Total	**262110**	**99.41%**	**94.41%**	**21734202**

top 50 free applications

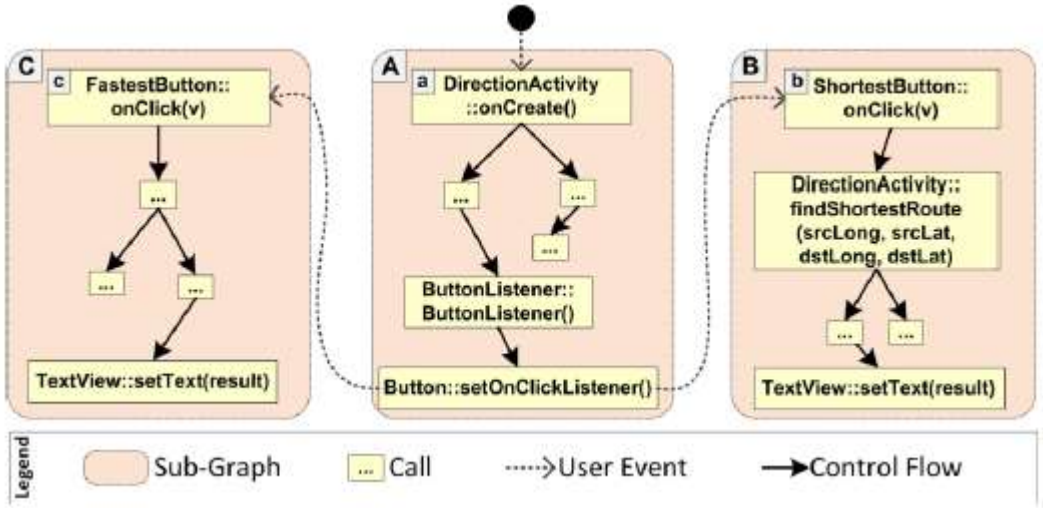
A Whitebox Approach for Automated Security Testing of Android Applications on the Cloud (2012)

- روش مقیاس پذیر و هوشمند برای تست فازی
- استفاده از رایانش ابری
- فازینگ هوشمند با استفاده از روش هایی برای استخراج معماری نرم افزار
- مهندسی معکوس با استفاده از یکی از ابزارهای dexer, baksmali, dex2jar, apktool

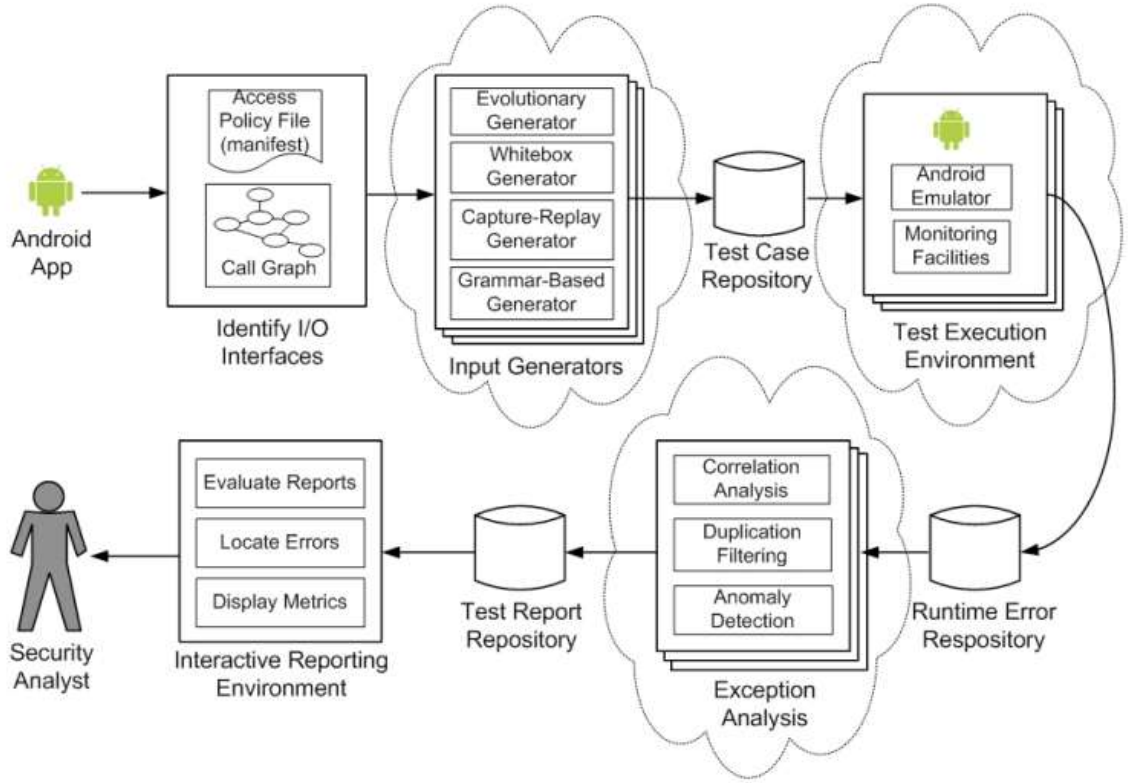


Testing Android Apps Through Symbolic Execution (2012)

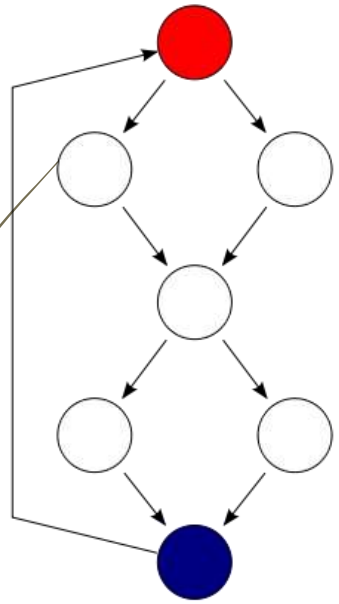
- ارائه ی اولین سیستم اجرای نمادین برنامه های اندروید
- توسعه ی Java pathfinder
- استفاده از stub برای حل چالش استفاده از JPF در اندروید
- استفاده از کلاس های ساختگی برای حل مشکل واگرایی مسیرها



A Framework for Automated Security Testing of Android Applications on the Cloud (2012)



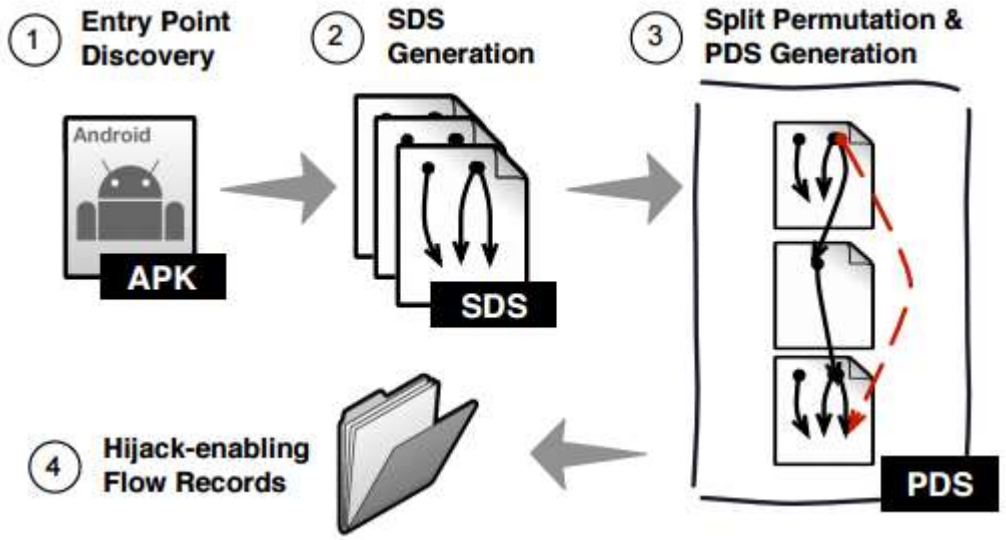
CHEX: Statically Vetting Android Apps for Component Hijacking Vulnerabilities (2012)



- Component Hijacking Examiner
- تعریف آسیب پذیری های «ربودن مولفه»
  - permission redelegation and leakage
  - intent spoofing
  - private data leakage
- یک روش تحلیل ایستا
- از منظر تحلیل جریان داده
- با استفاده از گراف وابستگی سیستم (System Dependence Graph)

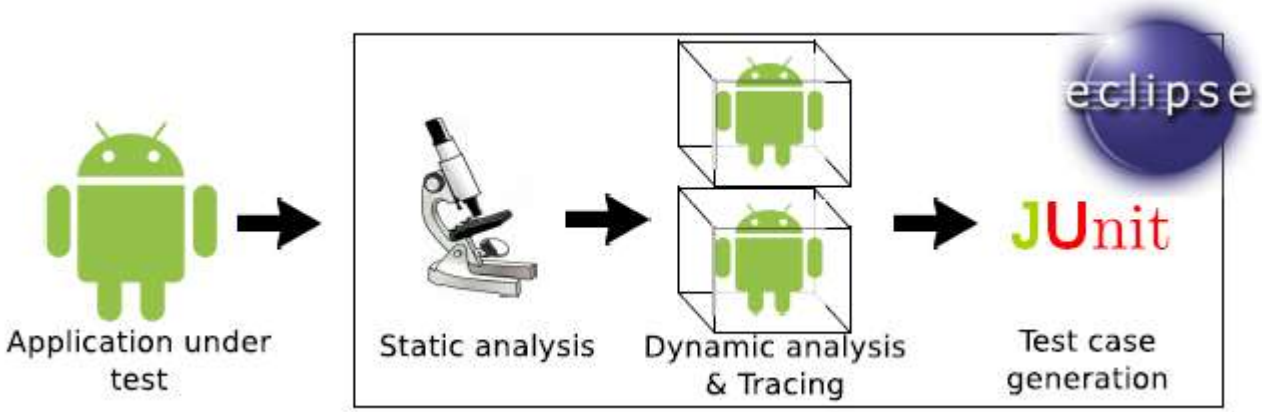
CHEX: Statically Vetting Android Apps for Component Hijacking Vulnerabilities (2012)

- ارائه ی Dalysis
  - سیستم عمومی برای تحلیل ایستای بایت کدهای نرم افزار اندروید
  - تبدیل مستقیم بایت کد دالویک به کد SSA
- ارزیابی روی ۵۴۸۶ نرم افزار اندروید
  - شناسایی۲۵۴ نرم افزار آسیب پذیر
- با میانه زمان اجرای ۳۷.۰۲ ثانیه برای هر نرم افزار
- یک روش sound برای شناسایی نقاط ورودی نرم افزار
  - مثبت درست ۸۱ درصد



Security Testing of the Communication among Android Applications (2013)

- تست ارتباطات بین برنامه های اندروید
- تست روی ۳ برنامه معروف



Security Testing of the Communication among Android Applications

```
<application android:icon="@drawable/game_logo" android:label="@string/app_name">
<activity android:name=".newGameImportActivity">
        <intent-filter>
                <action android:name="android.intent.action.VIEW"></action>
                <category android:name="android.intent.category.BROWSABLE"></category>
                <data android:scheme="http" android:host="*" android:pathPattern=".*\\.game" />
        </intent-filter>
</activity>
</application>
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.INTERNET" />
```

Security Testing of the Communication among Android Applications

- حالت های نتیجه مقایسه:
  - بدون خطا
  - یک خطای ثابت
  - خطاهای متفاوت

Subject	Total Activities	Activities with Intent Filter	Test Intents	Invalid Intents	Failing Tests
AnkiDroid	16	2	61	50	5
Jamendo	14	14	188	150	51
OpenSudoku	10	5	48	44	11

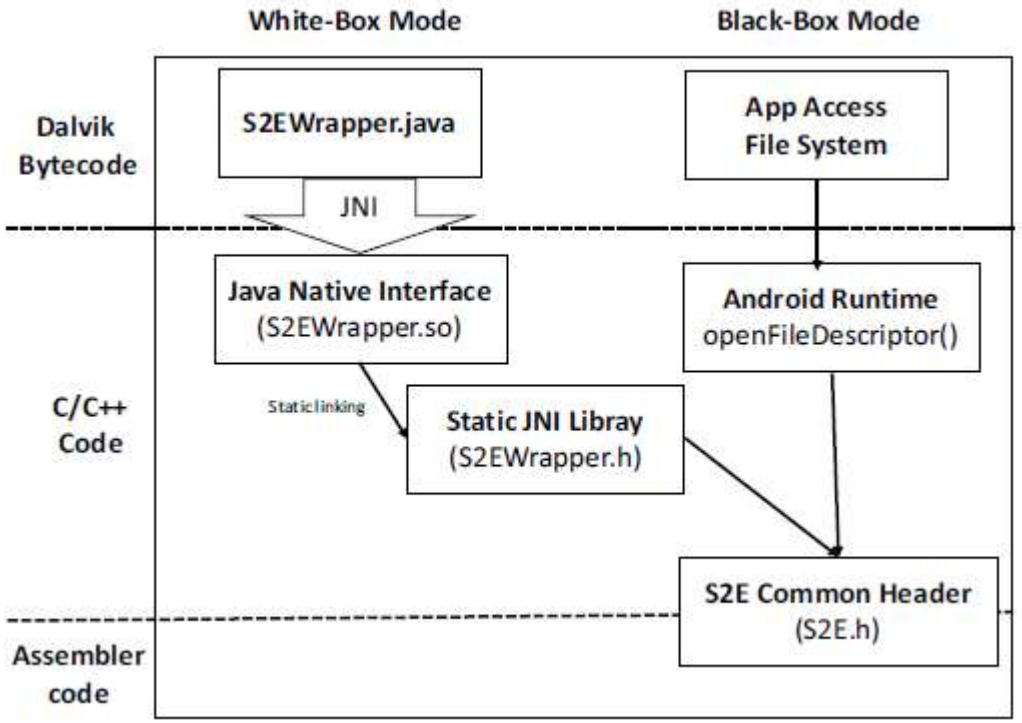
CRAXDroid: Automatic Android System Testing by Selective Symbolic Execution (2014)

- پلتفرم برای تست برنامه‌های اندروید به نام CRAXDroid
- با قابلیت تولید کد بهره‌بردار
- راحت تر کردن روند تست برای توسعه دهندگان و مدیران بازارها
- ورودی از فازر رابط کاربری
- خطاهای منجر به بسته شدن (Crash) ثبت می شوند.

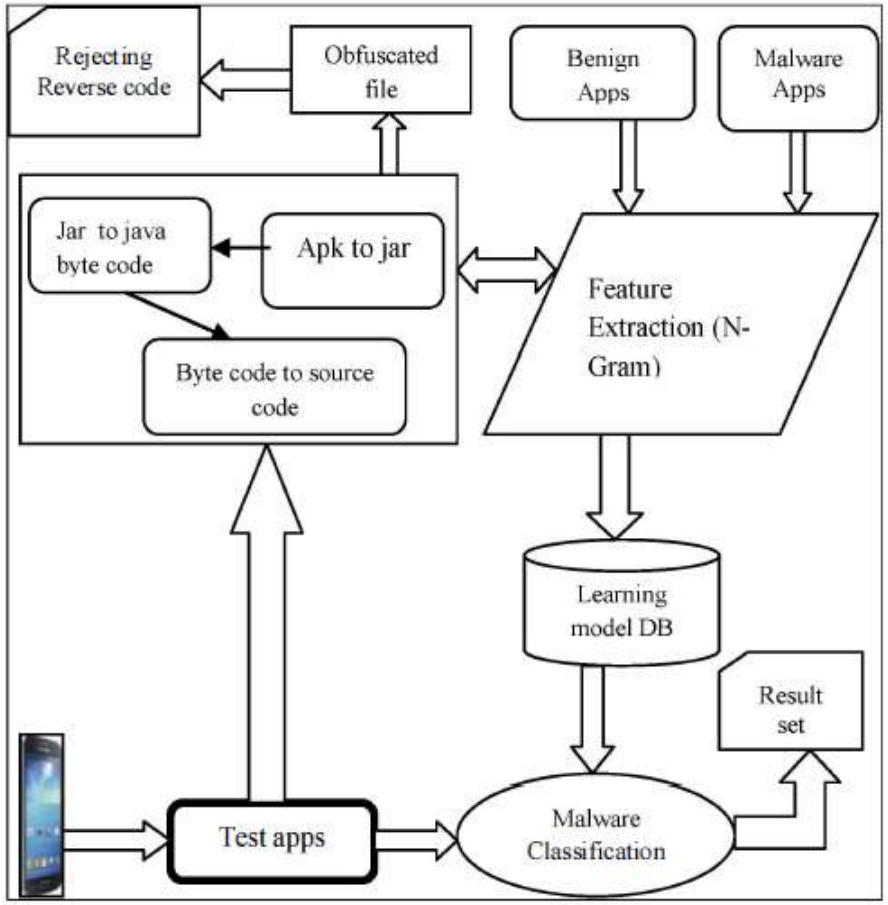
	Data size(bytes)	Time (minutes)
AndFTP	2	60.4
Scp	2	6000.1

Data leakage detection

CRAXDroid - Architecture



Detecting Software Vulnerabilities in android Using Static Analysis (2014)

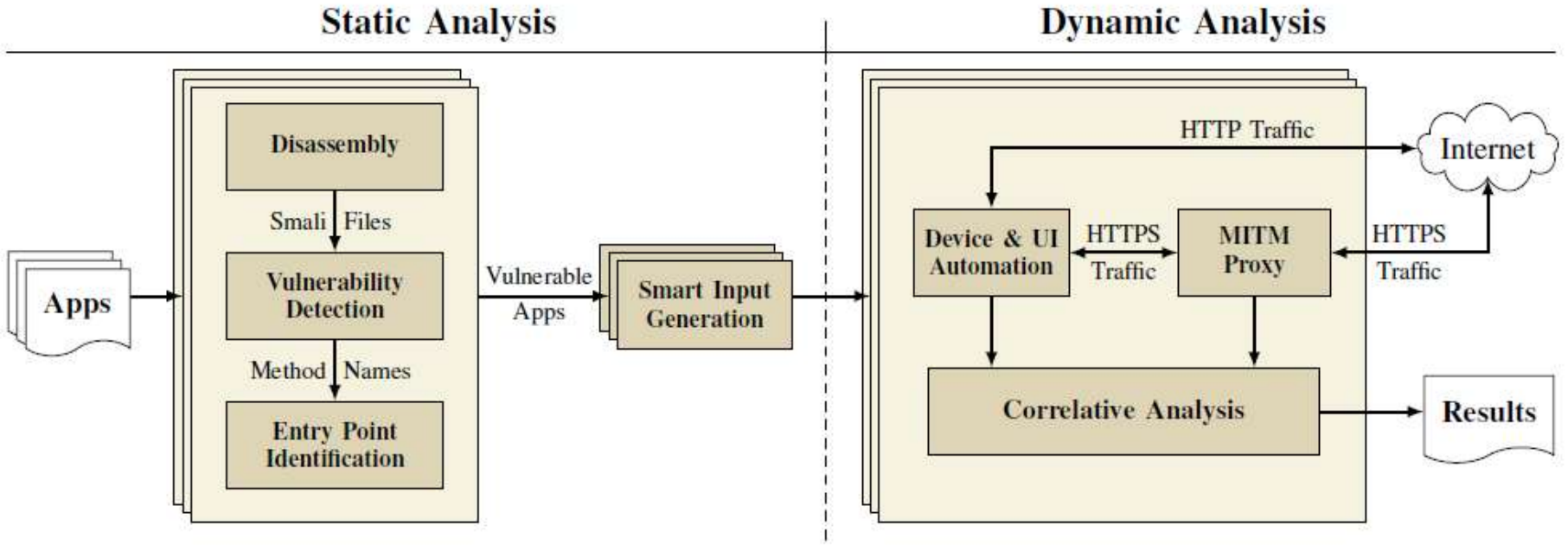


- استفاده از تحلیل ایستا و یادگیری ماشین
  - N-gram
- سریع، کم هزینه و انعطاف پذیر
- هدف اصلی شناسایی دژافزارها با قابلیت شناسایی آسیب‌پذیری‌ها
- مهندسی معکوس APK، استخراج ویژگی‌ها، استفاده از ابزار

SMV-HUNTER: Large Scale, Automated Detection of SSL/TLS Man-in-the-Middle Vulnerabilities in Android Apps (2014)

- SSL MITM Vulnerabilities
- بررسی های قبلی وابسته به تحلیل دستی برای شناسایی یا تایید آسیب پذیری
  - تحلیل ایستا
- چرا ممکن است برنامه ای نیاز به پیاده سازی فرآیند تایید اعتبار داشته باشد؟
  - اشکال در پیاده سازی بستر اندروید نسخه های اولیه
  - نبود ریشه CA مورد نظر در keystore
  - استفاده از گواهی self-signed
  - استفاده از یک کتابخانه با پیاده سازی اشتباه

SMV-HUNTER: Large Scale, Automated Detection of SSL/TLS Man-in-the-Middle Vulnerabilities in Android Apps



SMV-HUNTER: Large Scale, Automated Detection of SSL/TLS Man-in-the-Middle Vulnerabilities in Android Apps

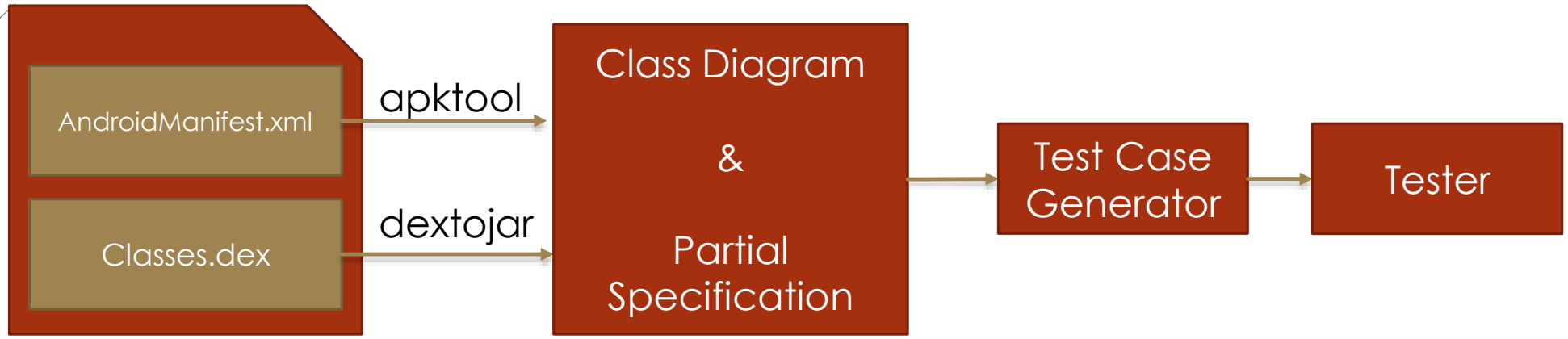
Category	Vulnerable DS1	DS2
Arcade & Action	3	11
Books & Reference	2	12
Brain & Puzzle	2	19
Business	12	149
Cards & Casino	0	6
Casual	1	17
Comics	0	2
Communication	4	19
Education	1	20
Entertainment	3	31
Finance	64	15
Health & Fitness	1	7
Libraries & Demos	0	0
Lifestyle	1	28
Media & Video	1	11
Medical	0	10
Music & Audio	0	22
News & Magazines	1	21
Personalization	0	4
Photography	1	7
Productivity	4	20
Racing	0	2
Shopping	2	16
Social	2	25
Sports	11	24
Sports Games	0	8
Tools	4	24
Transportation	1	13
Travel & Local	6	37
Weather	0	19
Total:	127	599

- $|DS1| = 3165$
- $|DS2| = 20316$

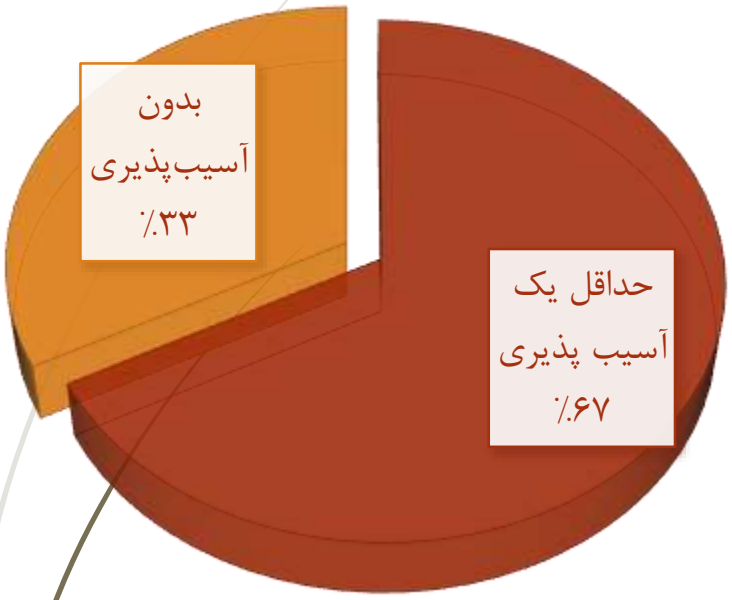
APSET, an Android aPplication SEcurity Testing tool for detecting intent-based vulnerabilities (2015)

- شناسایی آسیب پذیری های intent-based
- استخراج Class diagram و مشخصات جزئی به صورت خودکار از برنامه
  - تولید test case
- بر روی برنامه های تصادفی در Android market
- استفاده از یک SMT Solver برای مسیرهای اجرای test case

APSET, an Android aPplication SEcurity Testing tool for detecting intent-based vulnerabilities (2015)



VulHunter: Toward Discovering Vulnerabilities in Android Applications (2015)

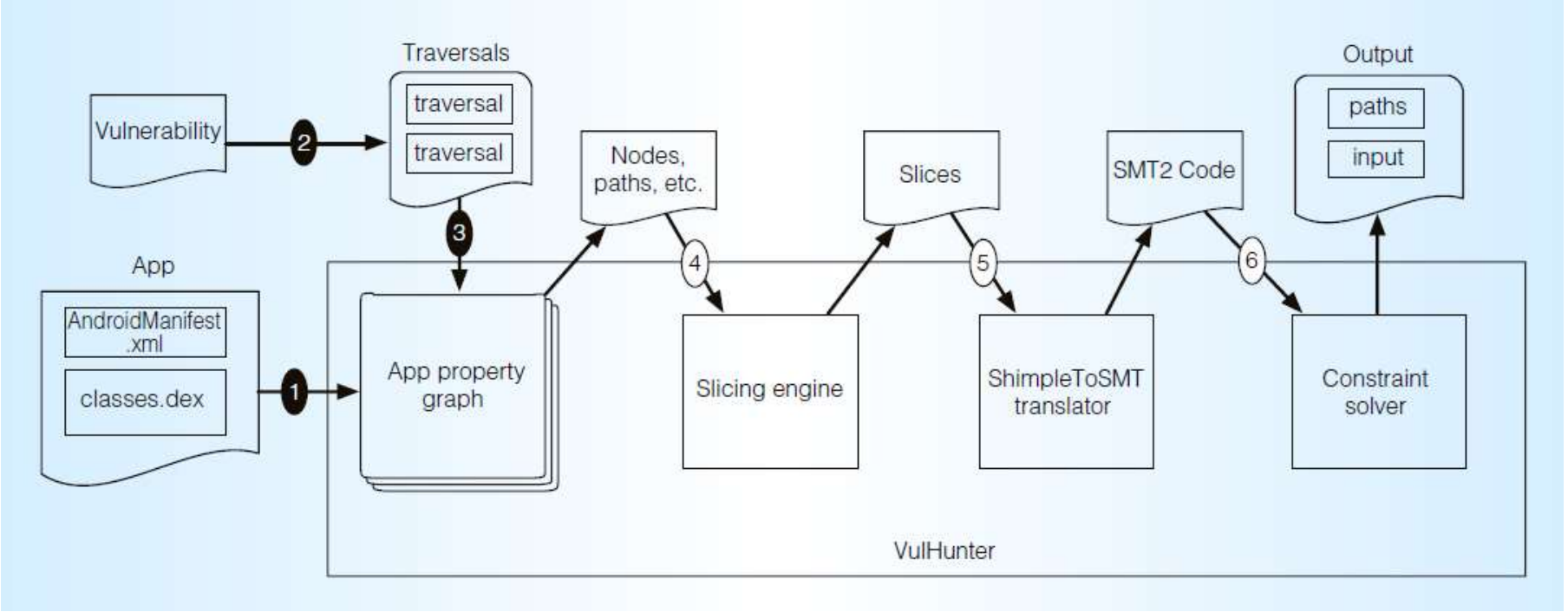


- تحلیل ایستا
- بر اساس مقاله ی F.Yamaguchi
- گراف توصیف برنامه (APG)
- APG شامل:
  - Interprocedure Control-Flow Graph
  - System Dependency Graph
  - Method Call Graph
  - Abstract Syntax Tree

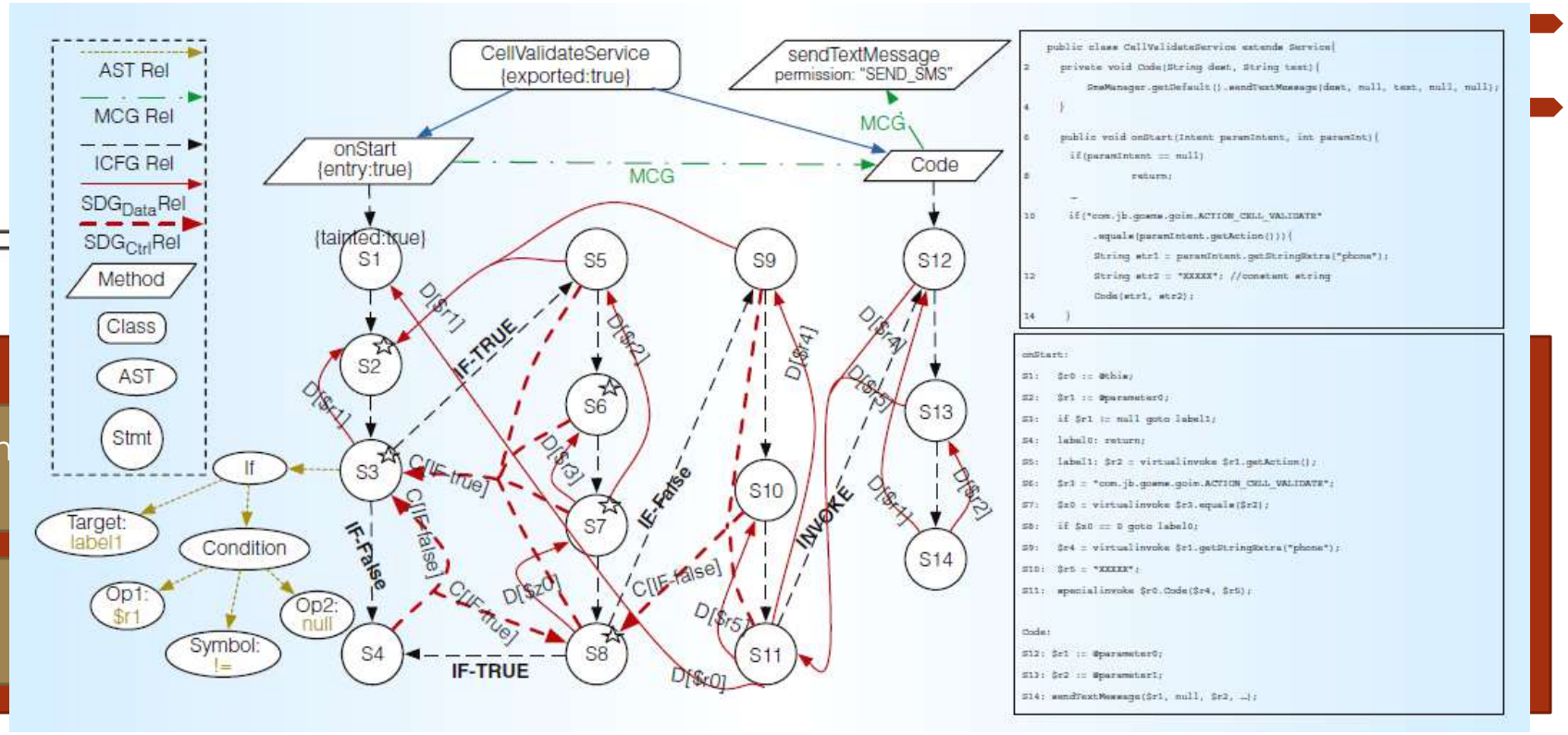
- مدل کردن ۵ آسیب پذیری معروف
- از ۵۵۷ برنامه مورد بررسی، تعداد ۳۷۵ با حداقل یک آسیب‌پذیری

VulHunter - overview

- ۳ بخش اصلی، ۳ بخش اختیاری



VulHunter – Constructing APG



VulHunter – Modeling Vulnerabilities

```
onStart:
S1:   $r0 := @this;
S2:   $r1 := @parameter0;
S3:   if $r1 != null goto label1;
S4:   label0: return;
S5:   label1: $r2 = virtualinvoke $r1.getAction();
S6:   $r3 = "com.jb.gosms.goim.ACTION_CELL_VALIDATE";
S7:   $z0 = virtualinvoke $r3.equals($r2);
S8:   if $z0 == 0 goto label0;
S9:   $r4 = virtualinvoke $r1.getStringExtra("phone");
S10:  $r5 = "XXXXX";
S11:  specialinvoke $r0.Code($r4, $r5);


Code:
S12: $r1 := @parameter0;
S13: $r2 := @parameter1;
S14: sendTextMessage($r1, null, $r2, …);
```

$$(N_{Method}\{\text{name} : \text{“onStart”}\}) - [R_{ICFG}]^{+} \rightarrow (N_{Stmt}\{\text{type} : \text{“invoke”}, \text{callee\_name} : \text{“sendTextMessage”}\}) \quad (1)$$

VulHunter – Results

Table 1. Five common vulnerabilities. For each vulnerability, we give both the number of Common Vulnerabilities and Exposures reports in each year and the number of vulnerable apps within the 577 apps under examination.

Vulnerability type	2011	2012	2013	2014	Vulnerable apps
Capability leak	1	63	3	6	4
Content provider directory traversal	0	0	0	9	3
X509TrustManager implemented improperly	1	7	2	6	337
Public file access permission	3	6	6	3	133
Log sensitive information	0	2	1	2	20

جمع بندی

نام روش	قابلیت شناسایی	نوع تحلیل
Ded	-	ایستا، دیکامپایل، دستی
Automated Security Testing on the Cloud	نشت توانایی، پیمایش در ارائه‌دهنده‌ی محتوا، آسیب‌پذیری‌های ارزیابی ناقص پارامترها	پویا، فازر، اجرای نمادین
CHEX	ربودن مولفه	ایستا، گراف کنترل جریان
Testing Communications among android apps	نشت توانایی، دیگر آسیب‌پذیری‌های ارزیابی ناقص پارامترها	ترکیبی، فازر
CraxDroid	نشت توانایی، دیگر آسیب‌پذیری‌های ارزیابی ناقص پارامترها	پویا، فازر، اجرای نمادین
Static Analysis	-	ایستا، یادگیری ماشین
SMV Hunter	پیاده سازی اشتباه ارتباط SSL	ترکیبی، گراف کنترل جریان، فازر
APSET	Intent based، نشت توانایی	پویا، فازر
Vulhunter	۵ دسته	ایستا، گراف کنترل جریان

مسائل باز

- تشخیص آسیب‌پذیری با تحلیل ایستا در جایی که منطق برنامه با عبارت منظم کار می کند.
- دسته بندی دقیق و مورد قبول در زمینه ی آسیب پذیری های نرم افزارها در اندروید
- بررسی آسیب پذیری های Native با روش ایستا

پروژه کارشناسی ارشد

- بهبود شناسایی آسیب پذیری های امنیتی در نرم افزارهای اندروید با استفاده از تحلیل ایستا
  - توسعه ی روش Vulhunter برای شناسایی آسیب‌پذیری های کدهای Native در برنامه های اندروید (برنامه های توسعه داده شده با استفاده از NDK)
  - رویکرد WhiteBox

- دو رویکرد:
  1. تحلیل روی کد منبع نرم افزارهای اندروید
  2. تحلیل با دریافت APK و دیکامپایل کردن بخش Native

پروژه کارشناسی ارشد – ادامه...

1. بررسی مزایا و معایب دو رویکرد
2. انتخاب یکی از دو رویکرد و تغییر ساختار گرافی برای پشتیبانی از بخش Native
3. ساخت گراف نمونه برای یک نرم افزار آسیب پذیر
4. مدل کردن یک آسیب پذیری برای پیمایش در گراف و انجام پیمایش در گراف نرم افزار
5. پیاده سازی ابزار جهت خودکار کردن عملیات استخراج گراف برای نرم افزار آسیب پذیر
6. ارزیابی نرم افزار با توجه به رویکرد انتخاب شده در مرحله ی ۲
   a) اعمال بر روی نرم افزارهای OpenSource جهت تشخیص آسیب پذیری
   b) اعمال بر روی نرم افزارهای محبوب در بازار اندروید جهت تشخیص آسیب پذیری
7. تایید دستی آسیب پذیری های مشخص شده توسط ابزار
8. بهبود عملکرد ابزار در صورت لزوم

منابع

1. Enck, W., Octeau, D., McDaniel, P., & Chaudhuri, S. (2011, August). **A Study of Android Application Security.** In *USENIX security symposium* (Vol. 2, p. 2).
2. Seltzer, L. **HP research finds vulnerabilities in 9 of 10 mobile apps**. http://www.zdnet.com/article/hp-research-finds-vulnerabilities-in-9-of-10-mobile-apps/, 2013.
3. Qian, C., Luo, X., Le, Y., & Gu, G. (2015). **VulHunter: Toward Discovering Vulnerabilities in Android Applications.** *Micro, IEEE*, *35*(1), 44-53.
4. Yeh, C. C., Lu, H. L., Chen, C. Y., Khor, K. K., & Huang, S. K. (2014, June). **CRAXDroid: Automatic Android System Testing by Selective Symbolic Execution.** In *Software Security and Reliability-Companion (SERE-C), 2014 IEEE Eighth International Conference on* (pp. 140-148). IEEE.
5. Avancini, A., & Ceccato, M. (2013, May). **Security testing of the communication among Android applications.** In *Proceedings of the 8th International Workshop on Automation of Software Test* (pp. 57-63). IEEE Press.
6. Salva, S., & Zafimiharisoa, S. R. (2014). **APSET, an Android aPplication SEcurity Testing tool for detecting intent-based vulnerabilities.** *International Journal on Software Tools for Technology Transfer*, *17*(2), 201-221.
7. Google, Inc. **Android Developer's Guide**. http://developer.android.com/, 2015.

منابع

8. Greenwood, D. S. J. S. G., & Khan, Z. L. L. (2014). SMV-HUNTER: Large Scale, **Automated Detection of SSL/TLS Man-in-the-Middle Vulnerabilities in Android Apps.**
9. Sbîrlea, D., Burke, M. G., Guarnieri, S., Pistoia, M., & Sarkar, V. (2013). **Automatic detection of inter-application permission leaks in Android applications.** *IBM Journal of Research and Development*, *57*(6), 10-1.
10. Lu, L., Li, Z., Wu, Z., Lee, W., & Jiang, G. (2012, October). **Chex: statically vetting android apps for component hijacking vulnerabilities.** In *Proceedings of the 2012 ACM conference on Computer and communications security* (pp. 229-240). ACM.
11. Mahmood, R., Esfahani, N., Kacem, T., Mirzaei, N., Malek, S., & Stavrou, A. (2012). **A whitebox approach for automated security testing of Android applications on the cloud**. In *Automation of Software Test (AST), 2012 7th International Workshop on* (pp. 22-28). IEEE.
12. Malek, S., Esfahani, N., Kacem, T., Mahmood, R., Mirzaei, N., & Stavrou, A. (2012). **A framework for automated security testing of Android applications on the cloud.** In *Software Security and Reliability Companion (SERE-C), 2012 IEEE Sixth International Conference on* (pp. 35-36). IEEE.
13. Salva, S., & Zafimiharisoa, S. R. (2014). **APSET, an Android aPplication SEcurity Testing tool for detecting intent-based vulnerabilities.** *International Journal on Software Tools for Technology Transfer*, *17*(2), 201-221.

منابع

14. Mirzaei, N., Malek, S., Păsăreanu, C. S., Esfahani, N., & Mahmood, R. (2012). **Testing android apps through symbolic execution.** *ACM SIGSOFT Software Engineering Notes*, *37*(6), 1-5.
15. Dhaya, R., & Poongodi, M. (2014, May). **Detecting software vulnerabilities in android using static analysis.** In *Advanced Communication Control and Computing Technologies (ICACCCT), 2014 International Conference on* (pp. 915-918). IEEE.
16. La Polla, M., F. Martinelli, and D. Sgandurra, ***A Survey on Security for Mobile Devices.* Communications Surveys & Tutorials**, IEEE, 2013. **15**(1): p. 446-471.
17. Bala, K., S. Sharma, and G. Kaur, ***A Study on Smartphone based Operating System***. International Journal of Computer Applications, 2015. **121**(1).
18. Shabtai, A., Fledel, Y., Kanonov, U., Elovici, Y., Dolev, S., & Glezer, C. (2010). **Google android: A comprehensive security assessment**. *IEEE Security & Privacy*, (2), 35-44.
19. Ho Han, D. **Android, at a glance**. http://www.cubrid.org/blog/dev-platform/android-at-a-glance/, 2013.


THANKS

سوال؟