



Mobile Security Research Lab

Security Study of Android Stock Trade Mobile Apps in Hong Kong



WTIA
香港無線科技商會

OUR STUDY TEAM	4
INTRODUCTION	5
OBJECTIVES	8
EVALUATION WORKFLOW	9
APP SELECTION	10
SUMMARY OF EVALUATION	15
DETAIL EVALUATIONS	18
1. ROOT DETECTION	18
2. SECURE COMMUNICATION	18
3. SOURCE CODE OBFUSCATION / ENCRYPTION	20
4. DATA BACKUP	21
5. MALICIOUS CODE INJECTION	22
6. SCREEN HIJACK EXPLOIT	22
7. TAMPERING AND REPACKING ATTACKS	23
8. WEBVIEW SECURITY: DATA STORE IN PLAINTEXT	24
9. WEBVIEW SECURITY: REMOTE CODE EXECUTION	24
10. PRESENTING DIGITAL CERTIFICATE IN PLAINTEXT	25
11. ENCRYPTION ALGORITHM MODE CHECK	26
12. KEYSTROKE LOGGER	27
13. SHARED OBJECT (.SO FILE) SECURITY	27
14. CONTENTPROVIDER DATA LEAK	28
15. SQL INJECTION VULNERABILITY	29
16. DYNAMIC DEBUGGING ATTACK	29
17. APPLICATION SIGNATURE VERIFICATION CHECK	30
18. BROADCASTRECEIVER COMPONENT SECURITY	31
19. EXPOSURE OF RESOURCES FILES	31
20. IN DEVICE DENIAL OF SERVICE ATTACKS	32
21. SERVICE COMPONENT EXPORT	33
22. ACTIVITY COMPONENT SECURITY	33
23. CONTENTPROVIDER COMPONENT SECURITY	34
24. UNRESTRICTED APK DOWNLOAD THROUGH APP	35

25. WORLD READABLE & WRITEABLE FILE	36
CASE STUDY	37
CASE 1 – DIGITAL CERTIFICATE	37
CASE 2 – ENCRYPTION KEY	38
FINDINGS AND CONCLUSIONS	39
TOOLS USED IN THE STUDY	43
ACKNOWLEDGMENT	44

Our Study Team

Study Lead	Paul Chow
Supervisors	Sam Soo Martin Ho
Assistants	Ricky Yung Cyrus Li
Coordinator	Hung Leung (Hong Kong Institute of Vocation Education, Chai Wan)
VTC Students	Chu, Tsz Yeung Yuen, Tsz To Ho, Lok Man Ho, Wai Ki Yu, Ming Shing Wong, Kwan Tat Chow, Tin Tik Liu, Chin Wai Chiu, Shun Shing
Advisors	Frankie Wong (Professional Information Security Association) Frankie Leung (ISC2 HK Chapter) Frankie Li (Dragon Threat Labs) Jayson Li (Bangle Security Limited) Jiande Chen (Bangle Security Limited)

Version: 1.0

Date: August 1, 2017

© Copyright 2017 Mobile Security Research Lab, all rights reserved.

No distribution or republication without permission

Introduction

The initiative of this study comes from several past activities.

In September 2015, Hong Kong Computer Emergency Response Team Coordination Centre (HKCERT) and Professional Information Security Association (PISA) jointly published a study on Transaction Security of Mobile Apps in Hong Kong¹. In this study, 130 Hong Kong online transaction service apps commonly used locally were tested. *Over one-third of them lack adequate encryption security in processing credential or transaction data, and are vulnerable to man-in-the-middle (MITM) attackers, that makes transmitting data leakage.*

Category	No. of Apps
Mobile Banking (流動銀行服務)	32
Cinema Ticketing (戲院訂票)	26
Financial Securities (金融證券)	24
Online Shopping / Group buy (網上商店/團購)	16
Travel Booking Service (旅遊訂位服務)	13
Online Food Ordering (外賣落單)	11
Digital Wallet / Payment Service (電子錢包/付費服務)	8

Fig 3. Number of apps distribution

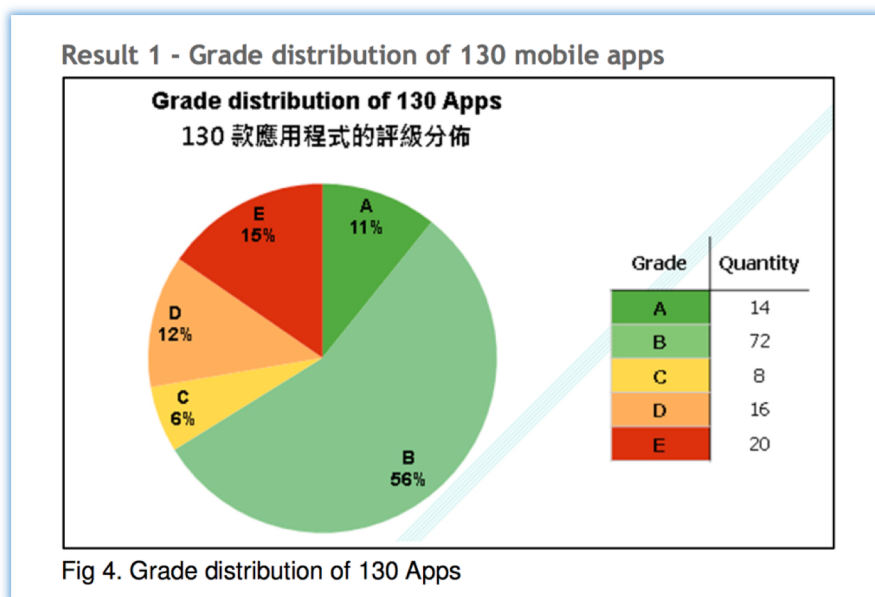


Fig 4. Grade distribution of 130 Apps

¹ HKCERT and PISA "Transaction Security of Mobile Apps in Hong Kong" Study Report. Available: https://www.hkcert.org/my_url/en/blog/15092402

The attacks have become more frequent, the number of affected customers and their loss have increased sharply. According to HKCERT, the number of cybersecurity incidents increased to 6,058 in 2016, up 23% from 2015². According to Securities and Futures Commission (SFC), for the 18 months ended 31 March 2017, 27 cybersecurity incidents resulting in unauthorized trades sum up to more than \$110 million were reported.

In December 2016, a stock trading security incident caused HK\$2M loss in unauthorized transactions. This incidence stroked the alarm and so PISA conducted another study, Mobile App Study on Securities Firms, and presented the result in DRAGONCON2016 event.

- 黑客攻陷證券行App 挪900萬炒高仙股
 - http://hk.on.cc/hk/bkn/cnt/news/20160917/bkn-20160917033054001-0917_00822_001.html
- Hackers targeted stock trade mobile app (STMA), hacked user account and made 9M unauthorized transactions, pushed penny stocks. User lost 2M dollars.



The study result was very shocking. PISA had tested altogether 6 trading relating apps and all of them were insecure.

	Alice	Bob	Coral	Dave	Eve	Frank
Disable allowBackup	✗	✗	✗	✗	✗	✗
addJavascriptInterface	✗	✓	✓	✓	✓	⚠
Obfuscation	✗	✗	✗	✗	✗	✗
Root detection	✗	✗	✗	✓	✓	✓

	Alice	Bob	Coral	Dave	Eve	Frank
End-to-end connection	⚠	✓	⚠	✗	✗	⚠
Encode in Application Layer	✗	✓	✓	✓	✓	✓
Password challenge	✓	✗	⚠	✓	✓	✓

Several questions come up after these studies:

² Hong Kong Computer Emergency Response Team Coordination Centre. “HKPC Warns of Rising Trend of Cybercrime-as-a- Service”. HKCERT Press Centre. Publication date: 16 January 2017. (https://www.hkcert.org/my_url/en/articles/17011601)

- How serious are the security issues on HK mobile apps nowadays?
- How to get the public awareness on the issues?
- How to help to mitigate the current issues?

Mobile Security Research Lab (“**The Lab**”) is being setup with the vision to help the market to understand the security status of the mobile apps, as well as to promote security awareness on the mobile app development.

The first activity of **the Lab** is to extend the two studies mentioned above to cover a larger number of apps. Regarding this study

- It was a joint effort between Hong Kong Wireless Technology Industry Association (WTIA)³ and **the Lab**.
- It was started in April 2017 and targeted to complete within 3 months.
- 140 Android apps were selected (please refer to the section ‘App Selection’ for the details).
- The selected apps were installed on 6 Android Phones (same model) for analysis.
- 25 security perspectives (named evaluation criteria in this report) were used to evaluate the security posture of the apps. Please note that all these criteria are focus on apps coding aspects, evaluations on other components, such as hardware, Android OS, etc. are excluded. In minimize the impact of other components (such as Android OS), the study was conducted on the same hardware and software platform.
- 90 man-days were consumed with 2 supervisors leading 9 VTC IVE students.

The evaluation results are detailed in the following sections in this report.

The study will not stop. **The Lab** plans to repeat the same study in a regularly basis, such as annually. Depending on the time and resources availabilities, **the Lab** targets to extend the study to iOS stock trade mobile apps and other categories, referencing to the categories listed in the study, Transaction Security of Mobile Apps in Hong Kong. The first goal will aim to online payment apps, which is becoming prevalent in the public.

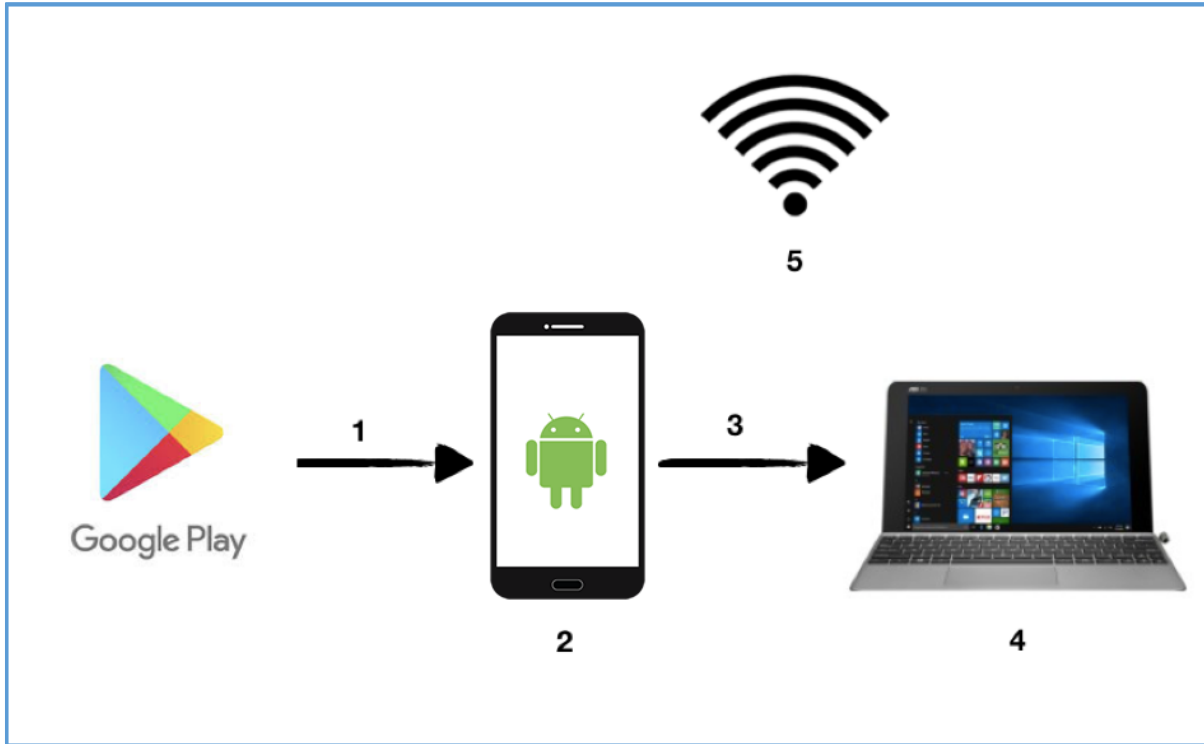
³ Hong Kong Wireless Technology Industry Association (<http://wtiahk.org/>) is a not-for profit, politically-neutral trade association dedicated to the wireless and mobile industry.

Objectives

The objectives of the study are

- to understand the current security posture of the stock trade mobile apps (STMA) on Android device in Hong Kong
- to raise the awareness of the public, apps owners and developers on the security issues of these mobile apps
- to educate VTC IVE Chai Wan students about the importance of Mobile App Security and the ethical practice of being a IT Security Professional
- to find out any tools to speed up the Mobile Apps scanning to reduce the manpower involved

Evaluation Workflow



1: Downloaded the selected 140 STMA apps from Google Play Store

2: Apps are installed and run on the 6 Android devices (Same model) with Android 6.0.1

3: USB connection to extract the apps' apk file to PC

4: PC (windows/ Mac) running VM (Kali-Linux with tools, detail tools please refer to the section 'Tools Used in this Study')

5: Wi-Fi for backend server connections

App Selection

The analyzed mobile apps were selected from the official **Android Google Play Store** in the period from March to June 2017 with the following criteria:

- 1) It provides online transaction that would manipulate personal information, credential and password and stock trading information.
- 2) It was provided by the participants⁴ of Hong Kong Exchanges and Clearing Limited (HKEX).

The selected 140 Android apps are listed as below (in arbitrary order).

	App Name	Package Name	Version
1	AMTD18 Sec	hk.com.amtd.imobility	1.3.4-release.2
2	Aristo Trader	com.aristo.trade	1.1.2_5
3	beevest securities	com.hkfdt.broker	2.1.0.170525001
4	BOCI Securities Limited	com.megahub.boci.mtrader.activity	5
5	BOCOM.HK (Securities)	com.hkmb	1.1.4
6	Bright Smart Securities (AA)	com.aastocks.android.bs	1.24
7	Bright Smart Securities (ET)	com.brightsmart.android	3.0.3.6
8	Bright Smart Securities (MH)	com.megahub.brightsmart.mtrader.activity	3.3
9	CASH RTQ	com.cashonline.cashrtq	2.8.1
10	cfkab	com.zscfappview.zjsj	1.01
11	Chief Sec(MH)	com.megahub.chief.mtrader.activity	4.4
12	CITICS Mobile	ttd.android.wininvest.citic	3.8.8
13	ConvoyMT	com.megahub.convoy.mtrader.activity	3.1
14	CSL Securities Limited - etnet	com.convoy.android	3.0.2.3
15	CSS Sec	com.aastocks.csss	1.1.1
16	Dah Sing Bank Securities Trade	com.megahub.dsb.stocktrading.activity	A1.2.10
17	e*trade	com.etrade.mobilepro.activity	5.6
18	BEA 東亞銀行	com.mtel.androidbea	1.1.3
19	First HK	com.aastocks.fisl	1.1
20	Fulbright Financial Group	com.megahub.fulbright.mtrader.activity	6.5
21	Get Nice Securities	com.aastocks.getn	1.1
22	gotrade	com.gotrade.mobile	1.9.4
23	hooray securities	hk.com.hooray.imobility	1.4.5-release.2
24	IB TWS	atws.app	8.3.877
25	icbc	com.icbc.mobile.abroadbank	1.0.2.4

⁴ HKEX participates URL : <https://www.hkex.com.hk/eng/plw/Search.aspx?selectType=SE>

26	kamfai 金輝	com.aastocks.kfsl	1
27	KE Trade PRO (HK)	hk.com.ayers.kimeng.trade	1.1.1
28	KGI HK Mobile Trader(AAStocks)	com.aastocks.kgi	2.3
29	Kingsway Financial	com.megahub.kingsway.mtrader.activity	1
30	Marigold	com.hk.phillip.B2BPoemsMobile.MG	1.01
31	market info	com.hangseng.androidpws	1.4
32	Mason Securities Mobile	com.megahub.guoco.mtrader.activity	3.4
33	PBHK Stock Trading	ttl.android.winvest.pub	1.1.1
34	POEMS HK	com.hk.poems.poemsMobileFX	1.909
35	ruibang trader	com.hee.rsl	1.1.2_5
36	Securities Trader	com.aastocks.tanrich	1.3
37	SHKF eMO!	com.gt.shk	2.8
38	Simsen@Mobile	com.megahub.simsenplus.mtrader.activity	3.1
39	South China Mobile Trade (AA)	com.aastocks.sout	2
40	SPTTrader Pro HD	hk.com.sharppoint.spmobile.sptraderprohd	11.4.1
41	Success Securities	com.aastocks.susl	1
42	Tai Shing EZ-Trade (MegaHub)	com.megahub.taishing.mtrader.activity	3.1
43	TaitakMtrade	com.megahub.taitakasia.mtrader.activity	3.2
44	WF Securities	com.megahub.wingfung.mtrader.activity	4.1
45	中信建投國際港股快車手機版	com.konsonsmx.csc	2.1.3
46	中原證券	com.megahub.centaline.mtrader.activity	3
47	中天交易宝	com.tsci.wtl	1.1.3
48	中投証券(香港)	com.konsonsmx.cis	1.2.9
49	中期證券	hk.com.ayers.cifco.trade	1.0.8
50	中泰國際全球融易通	com.tsci.qli	2.1.4
51	中潤證券	com.nationalresources.android	3.0.2.1
52	亨達投顧	com.livebricks.index.hantec	2.0.0
53	京華山一港股快車	com.tsci.cpy	1.0.8
54	信誠交易通	com.konsonsmx.pru	1.2.8
55	偉祿美林證券	qianlong.qlmobile.weilu.hk	V3.19 B001 (20170412-2)
56	元富財經	com.masterlink.finapp	3.1.0
57	兆安證券	com.megahub.siuon.activity	3
58	光证国际	com.zscfappview.gzgj	2.9
59	公平交易宝(Tele-Trend)	com.tsci.mrs	1.0.3
60	六福交易宝	com.tsci.lff	1.0.4
61	兴港通通专业版(Tele-Trend)	com.telekonson.ind	1.0.7
62	利星行證券	com.megahub.lsh.mtrader.activity	1
63	利通天下(Tele-Trend)	com.tsci.hni	1.1.4

64	北京証券	com.tsci.psl	2.0.9
65	北京首通證券	hk.com.bjsg.imobility	1.3.10-release.2
66	華金証券(國際)有限公司	com.ettrade.ssplus.android.huajin	2.7.3
67	卓誠證券	com.megahub.wellhonest.mtrader.activity	1.1
68	協聯證券有限公司	hk.com.ayers.cbl.trade	1.1.3
69	博大證券	com.partnerscap.android	3.0.3.0
70	君陽財富通	com.megahub.junyang.mtrader.activity	1
71	國泰君安國際	com.konsonsmx.gtja	2.2.8
72	國農落盤易	com.aastocks.trinity	1.3
73	國金證券(香港)有限公司	com.konsonsmx.gsl	1.0.9
74	基石證券	com.megahub.csl	1
75	大新銀行	com.MobileTreeApp	1.13
76	大眾証券	ttl.android.wininvest.pub	1.1.1
77	大華繼顯(香港)有限公司	com.uob.android	3.0.0.5
78	天元金融	com.ettrade.ssplus.android.tianyuan	2.8.0
79	太平洋國際證券有限公司	hk.com.pis.imobility	1.4.0-release.2
80	太陽國際證券 v2.0	com.aastocks.sins	2.1
81	好盈證券有限公司	hk.com.hooray.imobility	1.4.5-release.2
82	威靈頓金融	hk.com.ayers.weling.trade	1.1.1
83	安信國際港股快車手機版(Tele-Trend)	com.konsonsmx.css	2.1.1
84	宏大金融控股	com.tsci.gcs	2.0.5
85	宏高證券	com.megahub.wocom	1
86	富盈交易通	com.ettrade.ssplus.android.galaxytreasure	2.7.4
87	富途牛牛	cn.futu.trader	7.5.544
88	實德期貨(SP)	com.successmobile	1
89	寶通證券亞洲有限公司	com.aastocks.kcg	1
90	展兆投資有限公司	hk.com.celetio.imobility	1.3.5-release.2
91	山證國際交易寶(Tele-Trend)	com.tsci.gld	2.0.0
92	平安證券	com.pingan.android	2.4.2.1
93	恒泰證券有限公司	com.cairh.khapp.htgpkh	3.1.1.16063017
94	摩根富國證券	com.megahub.morganfuelgo	1
95	敦沛中亞證券有限公司	net.metaquotes.metatrader4	400.1091
96	日發證券有限公司	com.tsci.str	1.5.2
97	智易東方證券有限公司	hk.com.geos.imobility	1.3.8-release.2
98	東英證券	com.megahub.orientalpatron.mtrader.activity	1
99	林達證券有限公司	com.lamtex.android	3.0.0.1
100	永豐金中港通	com.tdx.AndroidYFJ	3.2

101	永隆銀行一點通_2.2.0	com.wlb.android	2.2.0
102	法興輪證	com.dbpower.warrant.sg	1.7
103	浚承資本	com.hangfat.android	3.0.2.2
104	海富交易宝	com.tsci.fwb	1.0.1
105	海通國際(etnet)報價交易版	com.haitong.android	1.3.0.10
106	渣打證券(香港)有限公司	com.scb.breezebanking.hk	4.6.1
107	潮商證券	com.tdx.AndroidNewCSHK	1
108	灝天金融	com.glorysky.android	1.3.1.9
109	申萬宏源環球期貨	com.megahub.sws.fso.mtrader.activity	1.2.1
110	發利證券	com.prime.android	1.3.0.3
111	益高證券	com.megahub.yicko.mtrader.activity	1
112	福亿交易宝(Tele-Trend)	com.tsci.fmx	1.0.8
113	萬盛金融控股	com.visioncap.android	3.0.3.1
114	美高證券	com.aastocks.metro	1.2
115	群益掌中寶	com.csc.android	2.4.1.4
116	股票快選	com.cathaysec.filter	1.2.2
117	勝利移動	com.tsci.vic	1.0.3
118	華信證券	com.aastocks.csys	1.1
119	華僑永亨証券有限公司	com.winghang	1.6.1
120	華南永昌	com.aastocks.hnsl	1.1
121	華融國際證券	com.megahub.huarong.mtrader.activity	1
122	華邦股票通	com.tsci.qhs	1.1.4
123	財華證券	com.gotrade.fn3620	1.5.8
124	財通交易宝	com.tsci.cts	1.5.2
125	越秀証券	com.konsonsmx.yxs	1.2.1
126	金利豐證券有限公司	com.megahub.kingston.mtrader.activity	2
127	金力証券	hk.com.ayers.kilm.trade	1.1.1
128	金唐國際證券	com.megahub.jti.mtrader.activity	1
129	金橋証券	hk.com.ayers.gobr.trade	1.1.1
130	金洋證券(et)	com.goldjoy.android	3.0.3.2
131	金裕富證券	com.aastocks.grsec	1
132	中華金融	com.aastocks.sbic	1.1
133	長雄證券	hk.com.ayers.elong.trade	1.0.8
134	长江証券港股快車	com.konsonsmx.cjs	2.1.5
135	雙龍證券(et)	com.msec.android	3.0.2.3
136	首控證券	com.aastocks.fcfl	1.2
137	高富金融	com.gtcapital.android	3.0.2.0
138	鴻鵬資本證券有限公司手機服務	com.aastocks.grcs	1



Security Study of Android Stock Trade Mobile Apps


139	鼎成證券	com.megahub.gransing.mtrader.activity	1
140	鼎新証券	com.tdx.tdxandroidv3DXXG	1.05

Summary of Evaluation

All apps were evaluated by automation tool (22 criteria) and 73 of them were picked randomly for further investigation (additional 3 criteria).

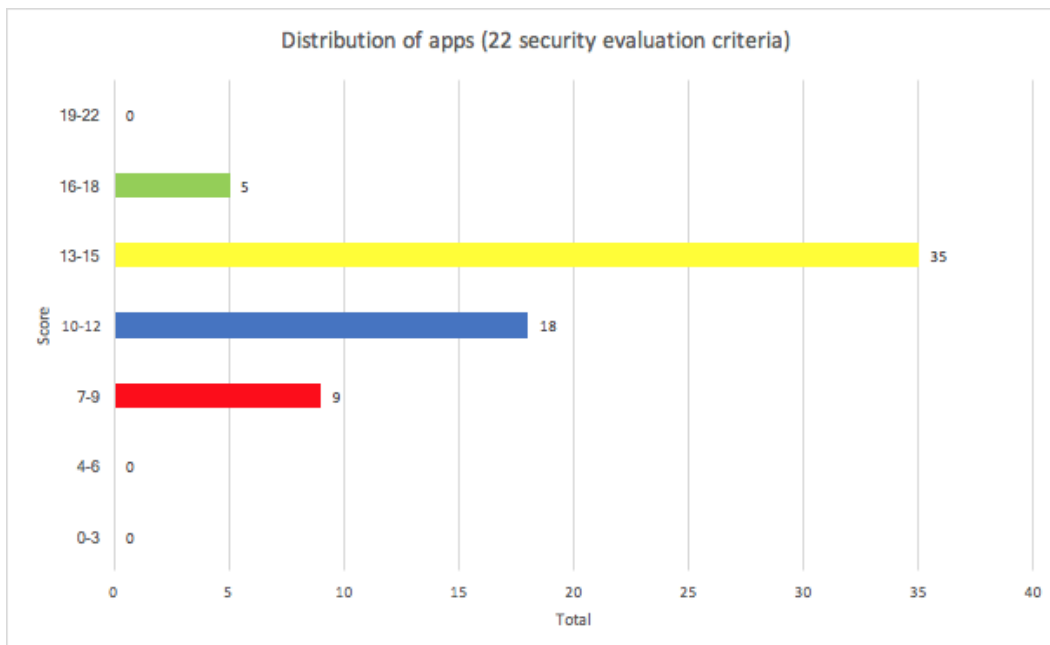
	No. of Apps	No. of Security criteria
Analyzed with automation tool + manual analysis	73	25
Analyzed with automation tool only	67	22
Total	140	

The following table shows the number of apps classified as Secure  or Insecure  under each criteria, with descending order of Insecure percentage.

Security criteria	Severity			Total	 %
Malicious Code Injection	High	0	140	140	100%
Dynamic Debugging Attack	Medium	0	140	140	100%
Source Code Obfuscation / Encryption	High	3	137	140	98%
WebView Security: Data Store in Plaintext	High	11	129	140	92%
Tampering and Repacking Attacks	High	18	122	140	87%
Root Detection	High	10	63	73	86%
Screen Hijack Exploit	High	24	116	140	83%
Application Signature Verification Check	Medium	25	115	140	82%
Data Backup	High	16	57	73	78%
Encryption Algorithm Mode Check	High	71	69	140	49%
Secure Communication	High	41	32	73	44%
BroadcastReceiver Component Security	Medium	85	55	140	39%
Presenting Digital Certificate in Plaintext	High	97	43	140	31%
Exposure of Resources Files	Medium	102	38	140	27%
In Device Denial of Service Attacks	Medium	104	36	140	26%
Service Component Export	Medium	105	35	140	25%
WebView Security: Remote Code Execution	High	109	31	140	22%
Activity Component Security	Medium	114	26	140	19%
Keystroke Logger	High	123	17	140	12%

Shared Object (.so file) Security	High	131	9	140	6%
ContentProvider Component Security	Medium	136	4	140	3%
SQL Injection Vulnerability	High	140	0	140	0%
ContentProvider Data Leak	High	140	0	140	0%
World Readable & Writeable File	Medium	140	0	140	0%
Unrestricted APK Download through app	Medium	140	0	140	0%

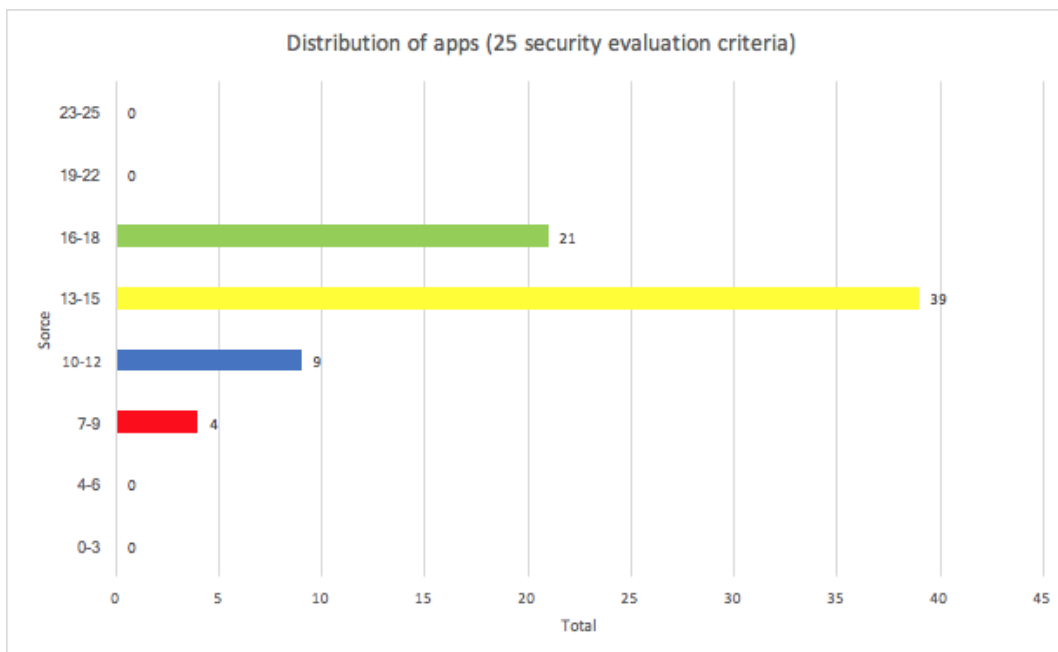
For all 67 apps which were evaluated in 22 criteria, one score is added to the app if it can pass the evaluation (i.e. Secure). The following chart shows the apps under different score ranges out of 22.



Highlights:

- There are 27 apps (40.3%) have a score below 12.
- The best app has a score 17, which means it has 6 criteria classified as Insecure.
- The worse app has a score 7.
- The average score is 13.

For the 73 apps which were evaluated with additional 3 criteria, one score is added to the app if it can pass the evaluation (i.e. Secure). The following chart shows the apps under different score ranges out of 25.



Highlights:

- There are 13 apps (17.8%) have a score below 12.
- The best app has a score 17, which means it has 6 evaluation classified as Insecure.
- The worse app has a score 7.
- The average score is 14.

Detail Evaluations

1. Root Detection

Purpose	<ul style="list-style-type: none"> Evaluate whether a rooted device can be detected by the app. Hackers can perform lots of malicious activities such as installing malware, modifying the device setting, monitoring app activities (to get confidential information) on a rooted device.
Severity	<ul style="list-style-type: none"> High
OWASP Mobile Top 10, 2016	<ul style="list-style-type: none"> M8 Code Tampering
What to verify	<ul style="list-style-type: none"> Can the app detect the Android device had been rooted? If the Android device had been rooted, will the app stop and exit?
How to verify	<ul style="list-style-type: none"> Install and run the app on a rooted Android device and analyze its behavior.
Result	<ul style="list-style-type: none"> 63 out of 73 (86%) apps do not have root detection. 5 out of 10 (50%) apps (which have root detection) stop processing, the other 5 apps allow user to continuous the operation after displaying warning message.

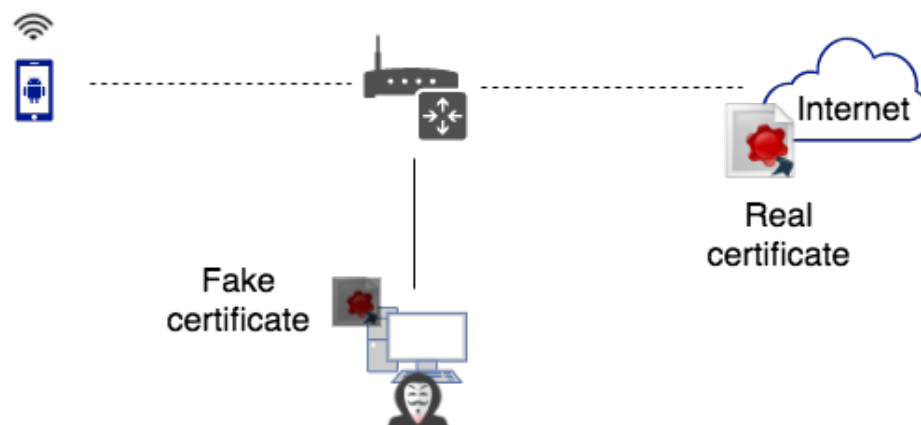
2. Secure Communication

Purpose	<ul style="list-style-type: none"> Evaluate whether the data is transferred in a secure way between servers and mobile devices.
Severity	<ul style="list-style-type: none"> High
OWASP Mobile Top 10, 2016	<ul style="list-style-type: none"> M3 Insecure Communication

What to verify	<ul style="list-style-type: none">• Does the app use SSL/TLS on internet communication?• Does the app encode/encrypt sensitive data on internet communication?• Is the SSL/TLS implementation safe from man-in-the-middle (MITM) attack?
-----------------------	--

How to verify	<ul style="list-style-type: none">• Setup a testing environment to perform Man-in-the-Middle (MITM) attack (see Figure 1 below)• The testing Android device is connected to internet through a proxy server so that the network traffic between client and server can be captured.<ol style="list-style-type: none">1. Test if the app use SSL/TSL connection. Capture the network traffic between server and the app to see if SSL/TSL is using while transferring data.2. Test SSL certification is implement in the app. The proxy CA was installed in the testing device as trusted root CA. See if there is any network traffic can be captured between server and the app.• For a secure mobile app, it should<ul style="list-style-type: none">○ communicate with the server (via internet) using SSL/TLS;○ be able to verify the correctness of the digital certificate;○ deny to establish a SSL connection when an incorrect certificate is detected
----------------------	---

Fig 1. Man in the Middle Attack



Result	<ul style="list-style-type: none"> • All tested 73 apps use HTTPS (i.e. SSL) for internet communication. • 14 out of 73 apps display sensitive data (username and/or password) in plain text (inside HTTPS) • 32 out of 73 (44%) apps' SSL implementation are under the risk of man-in-the-middle (MITM) attack. • 13 out of 73 apps' certificate is not unique (refer to Case Study 1)
---------------	---

3. Source Code Obfuscation / Encryption

Purpose	<ul style="list-style-type: none"> • Evaluate whether the app's APK can be reversed back to source code so that the logic, algorithm or traffic can be studied.
----------------	--

Severity	<ul style="list-style-type: none"> • High
-----------------	--

OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M5 Insufficient cryptography
--------------------------------	--

What to verify	<ul style="list-style-type: none"> • Is the app protected from reverse engineering back to source code? • If the source code of the app can be obtained successfully,
-----------------------	---

	can it be studied easily (i.e. no obfuscation or weak obfuscation)?
How to verify	<ul style="list-style-type: none"> • Decode the app using different tools (APKtool, APK-deguard.com, MobSF, dex2jar). • Study the source code if decoding can be done successfully.
Result	<ul style="list-style-type: none"> • 137 out of 140 (98%) apps are under the risk of reverse engineering back to source code.

4. Data Backup

Purpose	<ul style="list-style-type: none"> • Evaluate whether app data able to backup or restore without restriction. • Hackers may get sensitive information from the backup data.
Severity	<ul style="list-style-type: none"> • High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M2 Insecure Data Storage
What to verify	<ul style="list-style-type: none"> • Does the app allow backup and restoration?
How to verify	<ul style="list-style-type: none"> • Decode the app and examine the android:allowBackup attribute in the AndroidManifest.xml file.
Result	<ul style="list-style-type: none"> • 57 out of 73 (78%) apps are under the risk of data backup.

5. Malicious Code Injection

Purpose	<ul style="list-style-type: none"> Evaluate whether the app can be injected with malicious code. Hackers can inject malicious code into the target process to hook, monitor and obtain sensitive information such as stealing login account, password, alter the target account and amount of transfers through dynamic injection.
Severity	<ul style="list-style-type: none"> High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M8 Code Tampering
What to verify	<ul style="list-style-type: none"> Can the app detect the malicious code injected and stop running?
How to verify	<ul style="list-style-type: none"> Try to inject malicious code.
Result	<ul style="list-style-type: none"> All 140 (100%) apps are under the risk of malicious code injection.

6. Screen Hijack Exploit

Purpose	<ul style="list-style-type: none"> Evaluate whether the app client interface potentially being hijacked Screen hijacking is a form of malicious code that modifies or replaces app program interface to gather information such as username, password, banking and email authentication.
Severity	<ul style="list-style-type: none"> High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M2 Insecure Data Storage
What to verify	<ul style="list-style-type: none"> Can the app detect the screen hijack exploit and stop running?

How to verify	<ul style="list-style-type: none"> • Use tools to check if the app is vulnerable from screen hijacking.
Result	<ul style="list-style-type: none"> • 116 out of 140 (83%) apps are under the risk of screen hijack exploit.

7. Tampering and Repacking Attacks

Purpose	<ul style="list-style-type: none"> • Evaluate whether app can be repackaged and run after its source code, resource files and other parts being tampered. • Hackers may repack and create phishing app to steal user's login ID and password and intercepting SMS verification code.
Severity	<ul style="list-style-type: none"> • High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M8 Code Tampering
What to verify	<ul style="list-style-type: none"> • Can the app detect the tampering and repacking attacks and stop running?
How to verify	<ul style="list-style-type: none"> • Use tools to check if the app can run after tampering and repacking. Will there be any error message or force quit from the app?
Result	<ul style="list-style-type: none"> • 122 out of 140 (87%) apps are under the risk of tampering and repacking attacks.

8. WebView Security: Data Store in Plaintext

Purpose	<ul style="list-style-type: none"> Evaluate whether the data are stored in app directory (databases/WebView.db) in plaintext format and exposing those data might lead to another security issue.
Severity	<ul style="list-style-type: none"> High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M2 Insecure Data Storage
What to verify	<ul style="list-style-type: none"> Does the database WebView.db in app contain the data in plaintext?
How to verify	<ul style="list-style-type: none"> Open WebView.db
Result	<ul style="list-style-type: none"> 129 out of 140 (92%) apps are saving sensitive data in by default android WebView component feature.

9. WebView Security: Remote Code Execution

Purpose	<ul style="list-style-type: none"> Evaluate whether there is any remote code execution vulnerability in Webview component. Function addJavascriptInterface can export Java classes or Java methods and called by JavaScript to achieve interaction between webpage javascript and local Java. Due to there is no limitation on the method call of registered Java class, other unregistered Java classes can be called by the reflex mechanism, which could lead to execution of malicious code in the tampered URL, installation of Trojans in user's mobile phone, sending SMS, contacts or SMS being stolen, and even smart phones being controlled remotely.
Severity	<ul style="list-style-type: none"> High

OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M7 Client Code Quality
What to verify	<ul style="list-style-type: none"> • Is WebView and function addJavascriptInterface are used in the app.
How to verify	<ul style="list-style-type: none"> • Use tools to check if WebView and function addJavascriptInterface are used in the app.
Result	<ul style="list-style-type: none"> • 31 out of 140 (22%) apps allow the unregistered Java classes function calls.

10. Presenting Digital Certificate in Plaintext

Purpose	<ul style="list-style-type: none"> • Evaluate whether the digital certificate in APK is presented in plaintext. • Plaintext stored digital certificates can be tampered and disabling any other security measures which are rely on certificate validation.
Severity	<ul style="list-style-type: none"> • High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M2 Insecure Data Storage
What to verify	<ul style="list-style-type: none"> • Is digital certificate in APK of app stored in plaintext format?
How to verify	<ul style="list-style-type: none"> • Use tools to examine the digital certificate format.
Result	<ul style="list-style-type: none"> • 43 out of 140 (31%) apps are presenting digital certificate in plaintext.

11. Encryption Algorithm Mode Check

Purpose	<ul style="list-style-type: none"> Evaluate whether encryption algorithm mode used by the app. AES/DES are two commonly used symmetric encryption algorithms in android program and the working modes include of ECB, CBC, CFB, and OFB. Encryption data may be expose to chosen-plaintext attack (CPA) on ECB or OFB working mode and this may lead to disclosure of client privacy data, breach of encrypted files, acquisition of transfer data, man-in-the-middle attack and other consequences
Severity	<ul style="list-style-type: none"> High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M5 Insufficient Cryptography
What to verify	<ul style="list-style-type: none"> Is the app working on ECB or OFB mode and under the risk of chosen-plaintext attack (CPA)?
How to verify	<ul style="list-style-type: none"> Use chosen-plaintext attack (CPA) to expose encrypted data to ensure the app is on ECB or OFB working mode
Result	<ul style="list-style-type: none"> 69 out of 140 (49%) apps are working on ECB or OFB mode and under the risk of chosen-plaintext attack (CPA).

12. Keystroke Logger

Purpose	<ul style="list-style-type: none"> Evaluate whether the app is under the risk of keystroke monitoring. Sensitive information in an application majority is data input by user, if input data is monitored or the key position is recorded, it may cause the input data leakage. The default keyboard used in Android system has the risk of keystroke monitoring.
Severity	<ul style="list-style-type: none"> High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M2 Insecure Data Storage
What to verify	<ul style="list-style-type: none"> Is the app protected from monitoring or recording for the input data or key position?
How to verify	<ul style="list-style-type: none"> Use tools to check if the keystroke can be monitored when running the app.
Result	<ul style="list-style-type: none"> 17 out of 140 (12%) apps have potential risk on keystroke monitoring.

13. Shared Object (.so file) Security

Purpose	<ul style="list-style-type: none"> Evaluate whether the app contain shared objects file and can they be cracked and read. Shared Objects(.so file) are the dynamic link library file in an APK and Android uses NDK technology to compile C code to .so file to use directly from Java. Reverse engineer .so file may lead to leakage of assembly code of core function and even source code, this may causing the lost of intellectual property rights, Hacker may repacking the app for financial gain.
----------------	--

Severity	<ul style="list-style-type: none"> • High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M9 Reverse Engineering
What to verify	<ul style="list-style-type: none"> • Are there any shared objects file (.so file) used in the app? • If yes, can the .so file be cracked and read?
How to verify	<ul style="list-style-type: none"> • Check if any .so file is used in the app and try to crack it with tools.
Result	<ul style="list-style-type: none"> • 9 out of 140 (6%) apps contain shared objects file (.so file).

14. ContentProvider Data Leak

Purpose	<ul style="list-style-type: none"> • Evaluate whether the app's ContentProvider can be accessed sensitive data. • Since ContentProvider can be used for data sharing between apps. Strict access control should be implemented. Misconfiguration on authority setting may result in sensitive data leakage or tampering.
Severity	<ul style="list-style-type: none"> • High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M8 Code Tampering
What to verify	<ul style="list-style-type: none"> • Does the app allow data access via ContentProvider?
How to verify	<ul style="list-style-type: none"> • Use tools to check ContentProvider is used and access control of ContentProvider is implemented.
Result	<ul style="list-style-type: none"> • None of 140 (0%) apps can be accessed via ContentProvider.

15. SQL Injection Vulnerability

Purpose	<ul style="list-style-type: none"> Evaluate whether the app there is any SQL injection vulnerability. If read and write authority of ContentProvider component is set incorrectly and no filtering judgment is made for the field parameters of SQL query statement, the app's local database may subject to injection attack. This risk may lead to leakage of sensitive data information stored (such as account name, password and others.) or generate queries abnormalities to crash the app.
Severity	<ul style="list-style-type: none"> High
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M2 Insecure Data Storage
What to verify	<ul style="list-style-type: none"> Does the app use any SQL database? Is the app subject to SQL injection attack?
How to verify	<ul style="list-style-type: none"> Check if the app use SQL database and try to inject SQL query statement.
Result	<ul style="list-style-type: none"> None of 140 (0%) apps are in the risk of SQL injection attack.

16. Dynamic Debugging Attack

Purpose	<ul style="list-style-type: none"> Evaluate whether the app can be attacked by dynamic debugging attack. Hackers can use GDB, IDA, Ptrace and other debuggers to track running program, view and modify the code or data, analyze and tamper the business logic of the app (i.e., business transaction data and flow).
Severity	<ul style="list-style-type: none"> Medium

OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M10: Extraneous Functionality
What to verify	<ul style="list-style-type: none"> • Can the app detect dynamic debugging attack and stop running?
How to verify	<ul style="list-style-type: none"> • Try to use debuggers to track the app.
Result	<ul style="list-style-type: none"> • All 140 (100%) apps are under the risk of dynamic debugging attack.

17. Application Signature Verification Check

Purpose	<ul style="list-style-type: none"> • Evaluate whether the application signature is verified while the app on startup. • Signature is the unique identifier for app developer and signature certificate validation to effectively reduce the piracy rate of app. app without signature certificate may be subject to the APK repack by hacker after decompiled, this may lead to app piracy disclosure, loss of revenue and customer confident, worse scenario may even inject with malicious code and lead to data leakage or malicious attack.
Severity	<ul style="list-style-type: none"> • Medium
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M9 Reverse Engineering
What to verify	<ul style="list-style-type: none"> • Does the app check the application signature on startup?
How to verify	<ul style="list-style-type: none"> • Change the signature certificate and repackage for the app running.
Result	<ul style="list-style-type: none"> • 115 out of 140 (82%) apps do not have signature verification while the app on startup.

18. BroadcastReceiver Component Security

Purpose	<ul style="list-style-type: none"> Evaluate whether the BroadcastReceiver components of the app are subject to export risk. BroadcastReceiver may directly be called and used by the system or third-party app when export authority is set and this may lead to the risks of sensitive information disclosure, bypassed login interface and etc.
Severity	<ul style="list-style-type: none"> Medium
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M8 Code Tampering
What to verify	<ul style="list-style-type: none"> Is the BroadcastReceiver directly called and used by the system or third party app when export authority is set?
How to verify	<ul style="list-style-type: none"> Use tools to check if the setting of BroadcastReceiver is correct.
Result	<ul style="list-style-type: none"> 55 out of 140 (39%) apps' BroadcastReceiver components are subject to export risk.

19. Exposure of Resources Files

Purpose	<ul style="list-style-type: none"> Evaluate whether resource files able to read and altered. Resource files in an APK include of icons, images, JavaScript files, and JavaScript files may contain important display interfaces and execution information in resource files. Exposure of JavaScript file may lead to functional logic leakage and if it is altered, it may be implanted with a phishing page or malicious code and causing user's sensitive information disclosure.
----------------	--

Severity	<ul style="list-style-type: none"> • Medium
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M9 Reverse Engineering
What to verify	<ul style="list-style-type: none"> • Does the app contain any JavaScript files? • Are these files readable outside the app?
How to verify	<ul style="list-style-type: none"> • Extract the APK of the app and check if there is any JavaScript file and is it readable.
Result	<ul style="list-style-type: none"> • 38 out of 140 (27%) apps are under the risk of exposure of resources files.

20. In Device Denial of Service Attacks

Purpose	<ul style="list-style-type: none"> • Evaluate whether the components in app are subject to in device denial of service attacks • Denial of service attack vulnerability may lead to service outage for security protection and monitoring tools, this may also cause the app to be subject to malicious attack and outage, resulting economic loss or loss of customers.
Severity	<ul style="list-style-type: none"> • Medium
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M7 Client Code Quality
What to verify	<ul style="list-style-type: none"> • Does the app can still run while receiving invalid input.
How to verify	<ul style="list-style-type: none"> • Use tools to check if the app crash after receiving invalid data.
Result	<ul style="list-style-type: none"> • 36 out of 140 (26%) apps are subject to in device denial of service attacks.

21. Service Component Export

Purpose	<ul style="list-style-type: none"> Evaluate whether the service components of the app are subject to export risk. Service component usually serves as the progress in the background and it may directly be called and used by the system or third party app when export authority is set.
Severity	<ul style="list-style-type: none"> Medium
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M8 Code Tampering
What to verify	<ul style="list-style-type: none"> Is there service component used in the app? Are the service components directly called and used by the system or third party app when export authority is set?
How to verify	<ul style="list-style-type: none"> Use tools to check if the service component export setting is correct
Result	<ul style="list-style-type: none"> 35 out of 140 (25%) apps' service component are subject to export risk.

22. Activity Component Security

Purpose	<ul style="list-style-type: none"> Evaluate whether activity components of the app are subject to export risk. Activity component is the interface for interaction between Android program and user. Activity component may directly be called and used by the system or third-party app if export authority enabled and this may lead to the risks of bypass login interface, denial of service attacks, call of program interface by third
----------------	---

party and etc.

Severity	<ul style="list-style-type: none"> • Medium
OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> • M8 Code Tampering
What to verify	<ul style="list-style-type: none"> • Is the activity components directly be called and used by the system or third party app when export authority is set?
How to verify	<ul style="list-style-type: none"> • Use tools to check if the activity component security setting is correct.
Result	<ul style="list-style-type: none"> • 26 out of 140 (19%) apps' activity components are subject to export risk.

23.ContentProvider Component Security

Purpose	<ul style="list-style-type: none"> • Evaluate whether the ContentProvider components of the app are subject to export risk. • ContentProvider is a container for sharing data between apps and provide the data set designated by app to third party app. • ContentProvider may directly be called and used by the system or third party app when export authority is set and this may lead to the risks of sensitive information disclosure.
Severity	<ul style="list-style-type: none"> • Medium
OWASP Mobile	<ul style="list-style-type: none"> • M8 Code Tampering

App Risk (2016)

What to verify	<ul style="list-style-type: none"> Is the ContentProvider components directly called and used by the system or third party app when export authority is set?
How to verify	<ul style="list-style-type: none"> Use tools to check if the ContentProvider setting contain any vulnerability.
Result	<ul style="list-style-type: none"> 4 out of 140 (3%) apps' ContentProvider components are subject to export risk.

24. Unrestricted APK Download through app

Purpose	<ul style="list-style-type: none"> Evaluate whether there is vulnerability of downloading APK through this app. The component with the ability to download APK has export vulnerability and the component caller is not authenticated. Hacker can download any APK through the component and disguise the download information (i.e. icon, description and others.) of APK file in the process of download and resulting the malicious app installed by user.
----------------	--

Severity	<ul style="list-style-type: none"> Medium
-----------------	--

OWASP Mobile	<ul style="list-style-type: none"> M10 Extraneous Functionality
---------------------	--

App Risk (2016)

What to verify	<ul style="list-style-type: none"> Check if the app has the ability to download APK .
How to verify	<ul style="list-style-type: none"> Browse the app to see if it is capable of downloading APK .
Result	<ul style="list-style-type: none"> None of 140 (0%) apps' APK are subject to vulnerability

of downloading APK through their app .

25. World Readable & Writeable File

Purpose	<ul style="list-style-type: none"> Evaluate whether the files in the app can be read and write by any other app. Creating world-readable or writable files is dangerous as it could allow other app to have read or write access to that file. If such file contains critical configuration information, account information and other sensitive information, it may result to information leakage.
----------------	--

Severity	<ul style="list-style-type: none"> Medium
-----------------	--

OWASP Mobile App Risk (2016)	<ul style="list-style-type: none"> M2 Insecure Data Storage
---------------------------------------	--

What to verify	<ul style="list-style-type: none"> Are there any files in the app can be read or write by any other apps?
-----------------------	--

How to verify	<ul style="list-style-type: none"> Check if the app creates any file that can be readable or writeable besides the app itself.
----------------------	---

Result	<ul style="list-style-type: none"> None of 140 (0%) apps' files can be read and write by any other app
---------------	---

Case Study

Case 1 – Digital Certificate

For all secure Android apps, they must be digitally signed with a developer certificate. The certificate is a critical component because (1) the same certificate must be used to update the app, (2) Android treats apps signed by the same certificate as one single application. *Losing control of a certificate's private key, or using an insecure private key, would result in severe security consequences.*

For example, if an attacker has an opportunity to obtain the private key of an app,

1. he or she could then create a fake APK file,
2. sign it using the same certificate as the legitimate app, and
3. replace the targeted app with fake app on the device silently using the “Application upgrade” procedure.
4. Or the attacker can make use of the “SharedUserId” option, which allows apps with different package names but signed with the same certificate to share permissions and stored data.

Ideally, for each unique app posted to the Google Play Store, the app developer should generate a unique private key. In other words, the app developer shouldn't use the same certificate for multiple apps, unless he really wants to treat the apps as one single application. In this particular case, he should implement additional access controls among the apps.

More information for this issue can be found in a study in 2014⁵.

In our study, we found that the digital certificate of 13 out of 73 apps were not unique.

⁵ Bad Certificate Management in Google Play Store (<https://researchcenter.paloaltonetworks.com/2014/08/bad-certificate-management-google-play-store/>)

Case 2 – Encryption key

When talking about securing the data, encryption is only the first step. The utmost importance is how safe the key is being protected. If the accessibility of the encryption key is not properly guarded, all of that encryption could be rendered useless.

During the study, we found that one app simply embeds the encryption key inside the app without any protection. *It is a critical security issue because ALL users using that app are using the same key for encryption.*

For example,

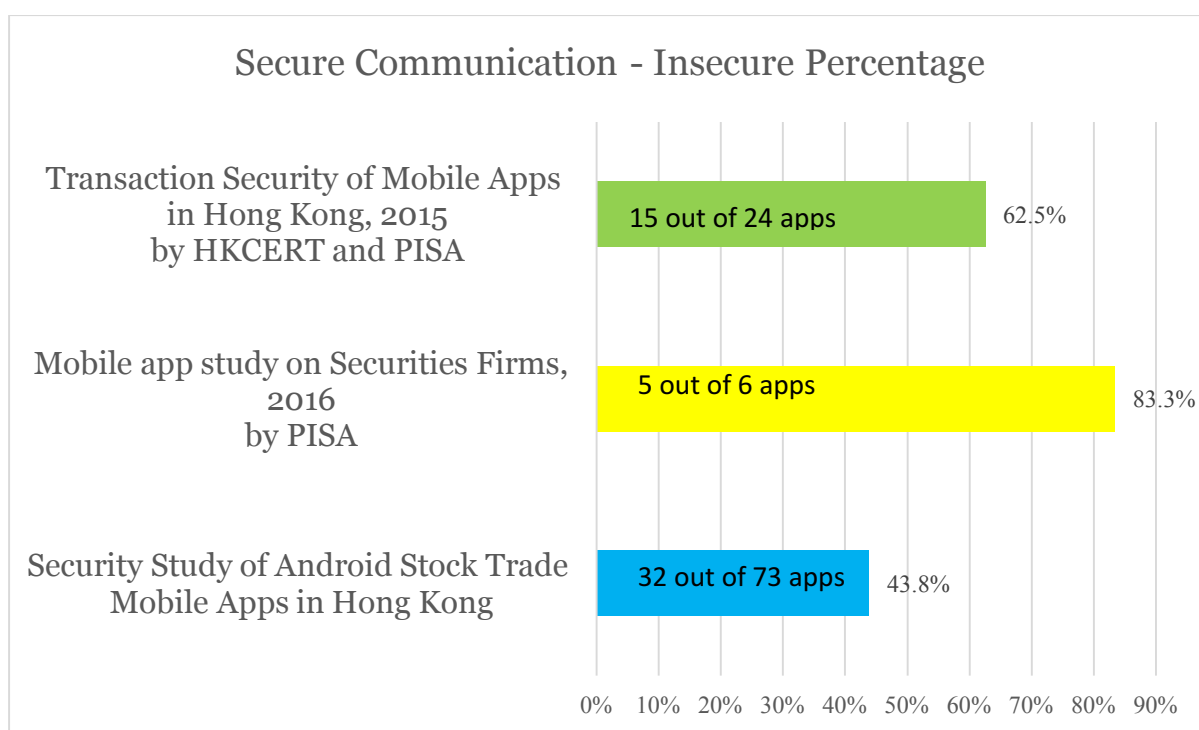
1. the hacker can simply download the legitimate app from Google Play Store,
2. reverse engineering the app and extract the key from the source,
3. with the key on hand, the hacker can decrypt ALL users' data which are protected by the key. The hacking process is reduced to one task only (how to get the users' data)

It is an indication that at least some app developers are not aware of the basic security principles in encryption.

Findings and Conclusions

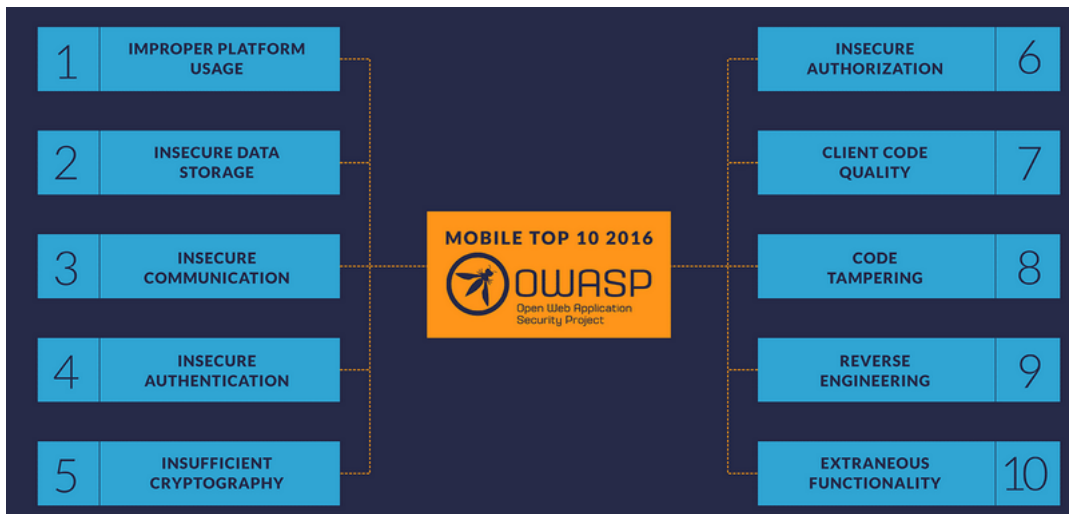
There are several things we observed during the study and result analysis (please refer to the section **Summary of Evaluation** for the result summary).

1. When comparing the Secure Communication (one of the 25 evaluation criteria) against the two previous studies (Transaction Security of Mobile Apps in 2015 and Mobile App Study on Securities Firms in 2016, mentioned in the **Introduction**). The situation is improved but still more than one third of the apps (43.8%) are insecure.



2. For the 25 evaluation criteria, only 4 criteria are fully passed for all selected apps. They are passed because most of the apps didn't use those related features
 - SQL Injection vulnerability: SQL database usually use in server and some apps may also use it. In this study, most of the STMA don't use it.
 - ContentProvider Data Leak: ContentProvider is a method in Android that normally is used to provide content among different apps. In this study, most of the STMA don't use it.

- World Readable & Writeable File: It is an old method for information sharing. In this study, all of the STMA don't use it.
 - Unrestricted APK Download through app: It is usually used in marketing apps. In this study, all of the STMA don't use it.
3. All 25 evaluation criteria can be mapped to OWASP Mobile Top 10 2016⁶. OWASP Mobile Top 10 2016 is the result of OWASP Mobile Security Project that publish the 10 most common mobile app vulnerabilities and ways to avoid them. In order to maintain the apps' security quality, SFC should regulate that the apps should be audited by third party according some security standard, such as OWASP Mobile Top 10 2016.



4. Most of the 140 STMA didn't implement Two Factor Authentication (2FA). Some apps use dual passwords authentication but dual passwords is not 2FA. 2FA is an importance security method because it increases the difficulty of password hacking, and lots of data breaches are password related. Using the 25 evaluation criteria as example, if 2FA is not implemented, the hackers may get the password easily using some of the following vulnerabilities
- Malicious Code Injection
 - Dynamic Debugging Attack
 - Source Code Obfuscation / Encryption
 - WebView Security: Data Store in Plaintext

⁶ OWASP Mobile Top 10 https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

- Root Detection
- Screen Hijack Exploit
- Data Backup
- Secure Communication
- WebView Security: Remote Code Execution
- Keystroke Logger

We suspect that the above vulnerabilities did contribute a portion in the \$110 million unauthorized trades mentioned in **Introduction** above.

2FA had been well adopted in online banking and SFC also recommend to enforce the use of 2FA in its consultation paper (Consultation Paper on Proposals to Reduce and Mitigate Hacking Risks Associated with Internet Trading, May 2017).

5. For top 5 severity of the 25 evaluation criteria, over 86% apps are not passed. The security quality is not satisfied. The app developer should seriously consider to adopt security control in their Software Development Life Cycle (SDLC). One of the consideration can be the OWASP Mobile Top 10 2016 mentioned above.
6. The STMA personal data privacy issues are not evaluated in this study. We recommend the STMA developer to follow the Best Practice Guide for Mobile App Development by PCPD⁷.

During the study preparation stage, we estimated that 1.5 man days were required for each app manual evaluation. Since lot of manpower is required, we cooperated with Dr. Hung Leung of VTC IVE Chai Wan to engage 9 students to participate this study as the students' in-term project. The result of this study demonstrates that cooperation between business sector and education sector did go to a win-win situation.

⁷ Best Practice Guide for Mobile App Development by PCPD
https://www.pcpd.org.hk/tc_chi/resources_centre/publications/guidance/files/Mobileapp_guide_c.pdf

Besides the in-term engagement, we also use automation tools to replace and to verify the works previously need to do manually. The time saving is very significant, from 1.5 man days to less than 1 hour (report generation less than 10 minutes) for an app evaluation.

The Lab strives to deliver the importance and impact of the mobile app security to the market stack holders, such as the app owners, the developers, etc. As a follow-up action, **The Lab** had published a **Best Practices – Secure Mobile Development for Android and iOS⁸** as a code of measure for a secure app development and usage.

If you have any questions on the study and report, or have interest to learn more about **The Lab**, please contact us at info@msr-lab.com.

⁸ Best Practices – Secure Mobile Development for Android and iOS http://www.msr-lab.com/Secure_Mobile_Development_Best_Practices.pdf

Tools Used in the Study

- apktool, a tool for reverse engineering Android apk files (<https://ibotpeaches.github.io/Apktool/>)
- MobSPA, an automated scanning tool (<http://www.bangle.com.hk/Product>)
- MobSF, an automated pen-testing framework (<https://github.com/MobSF/Mobile-Security-Framework-MobSF>)
- JD-GUI, Java decompiler GUI tool (<https://github.com/java-decompiler/jd-gui>)
- Burp Suite, graphical tool for testing Web application security (<https://portswigger.net/burp>)
- Xposed, Android framework for modules (<http://repo.xposed.info/module/de.robv.android.xposed.installer>)
- Inspeckage, dynamic analysis tool on Xposed Framework (<https://github.com/ac-pm/Inspeckage>)

Acknowledgment

- Dr. Hung Leung from Hong Kong Institute of Vocation Education (Chai Wan) – Supported this study by arranging 9 students from VTC as assistants.
- John C Y Chiu JP, Honorary Chairman of Wireless Technology Industry Association - Acts as advisor in the wireless industry for this study.
- Bangle Security Limited – Provided free license for the automatic Mobile Penetration scanning tools used in this study.
- Frankie Li from Dragon Threat Labs – Provided some trading accounts and acts as technical advisor for this study.
- Frankie Wong from Mobile Security Special Interest Group of Professional Information Security Association – Conducted the two studies which we based on, and acts as technical advisor for this study.
- Frankie Leung, President of (ISC)² Hong Kong Chapter and Program Director of Professional Information Security Association - Acts as advisor for this study.
- SC Leung, Senior Consultant, Hong Kong Computer Emergency Response Team Coordination Centre - Acts as advisor for this study report.
- Mike Lo, Risk Advisory Manager, Deloitte China - Acts as advisor for this study report.