Progress

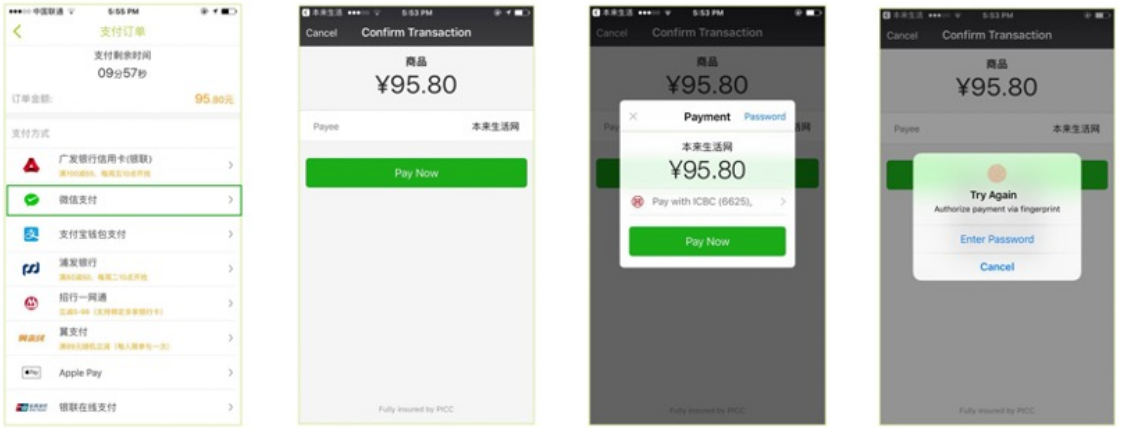
Week 4 of July

2020019252 김나현

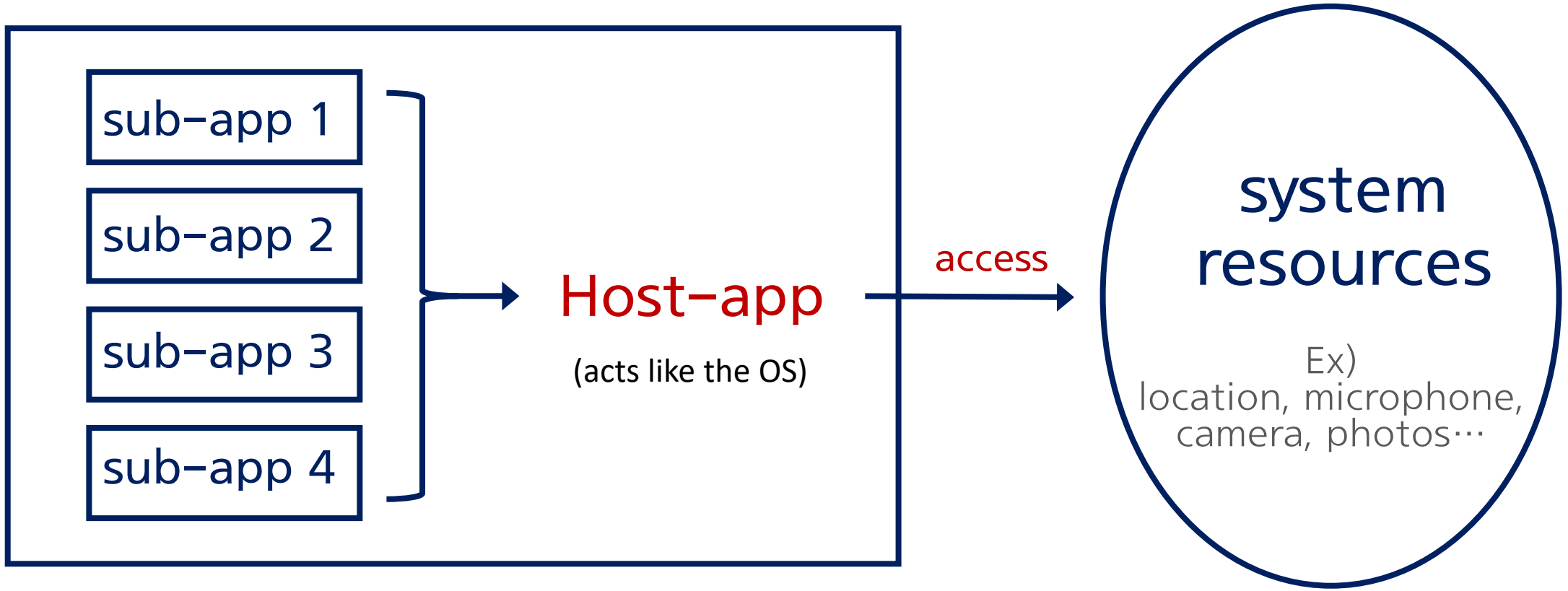
Demystifying Resource Management Risks in Emerging Mobile App-in-App Ecosystems

2020, CCS(Computer and Communications Security)

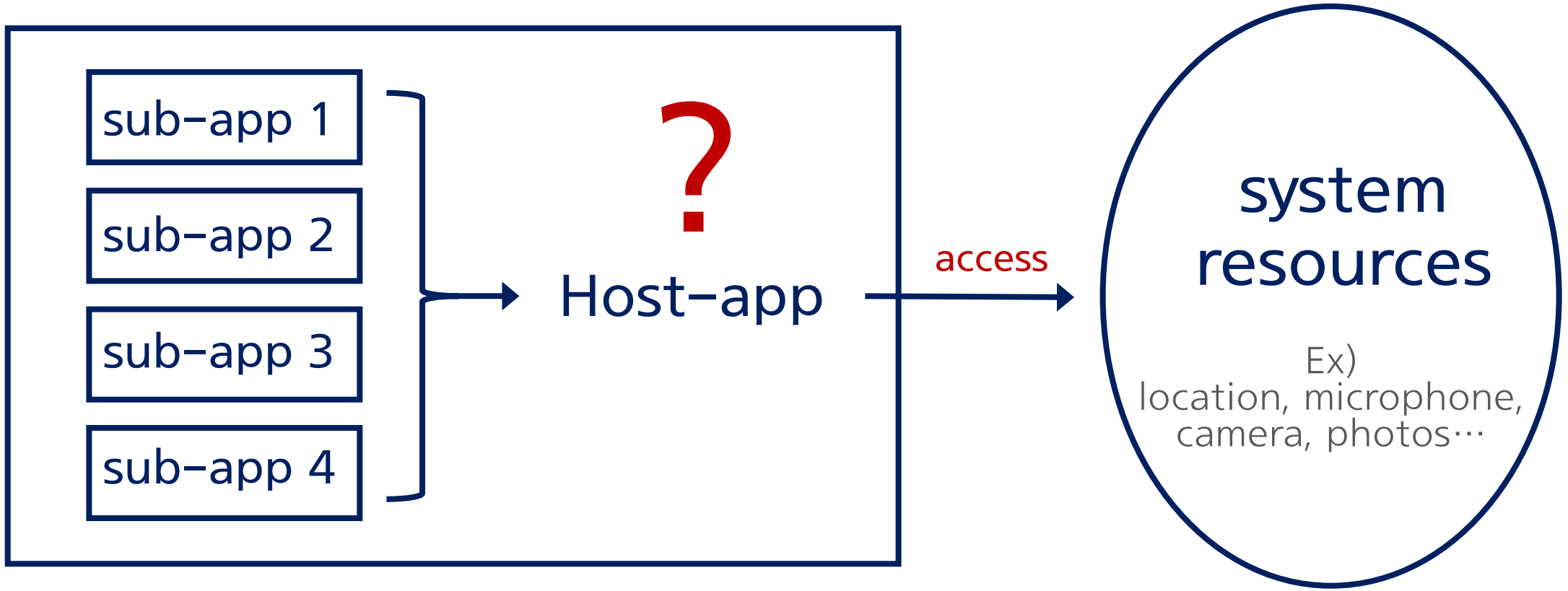
widely used “App-In-App” paradigm



App-In-App Ecosystems



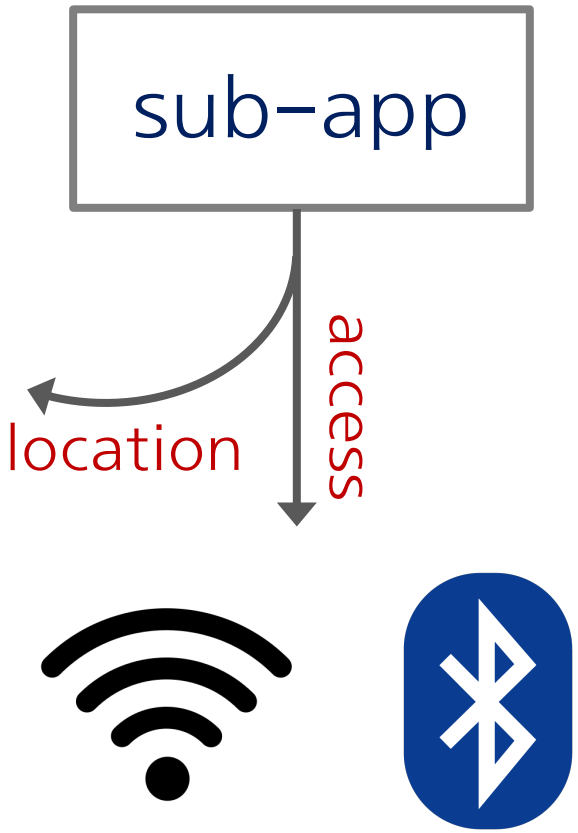
App-In-App Ecosystems



Security analysis on resource management

1. System Resource Exposure
2. Sub-window Deception
3. Sub-app Lifecycle Hijacking

1. System Resource Exposure



Inconsistency!

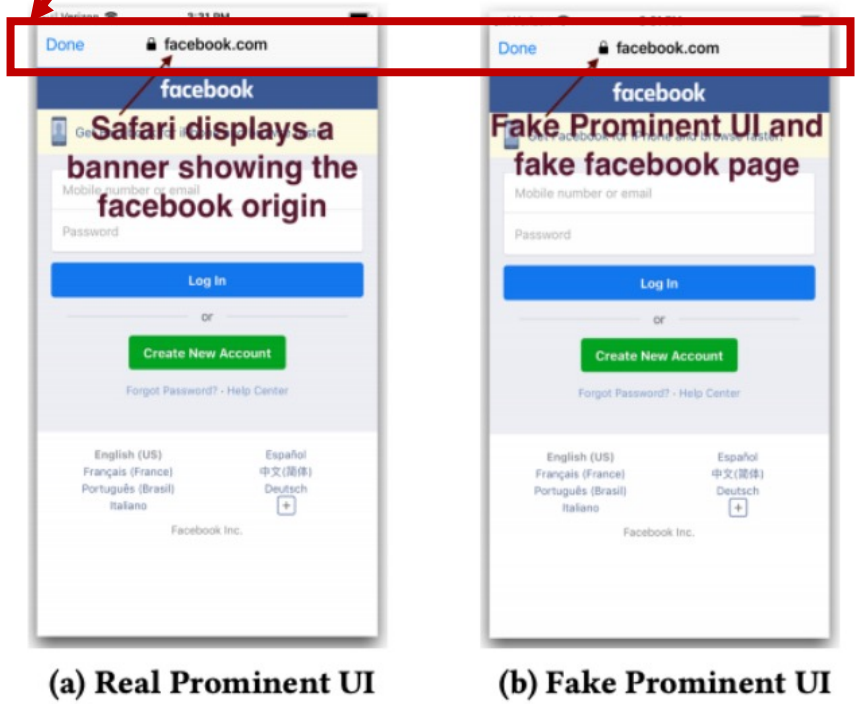
system API : protected

sub-app API : unprotected

(escaped sub-app API)

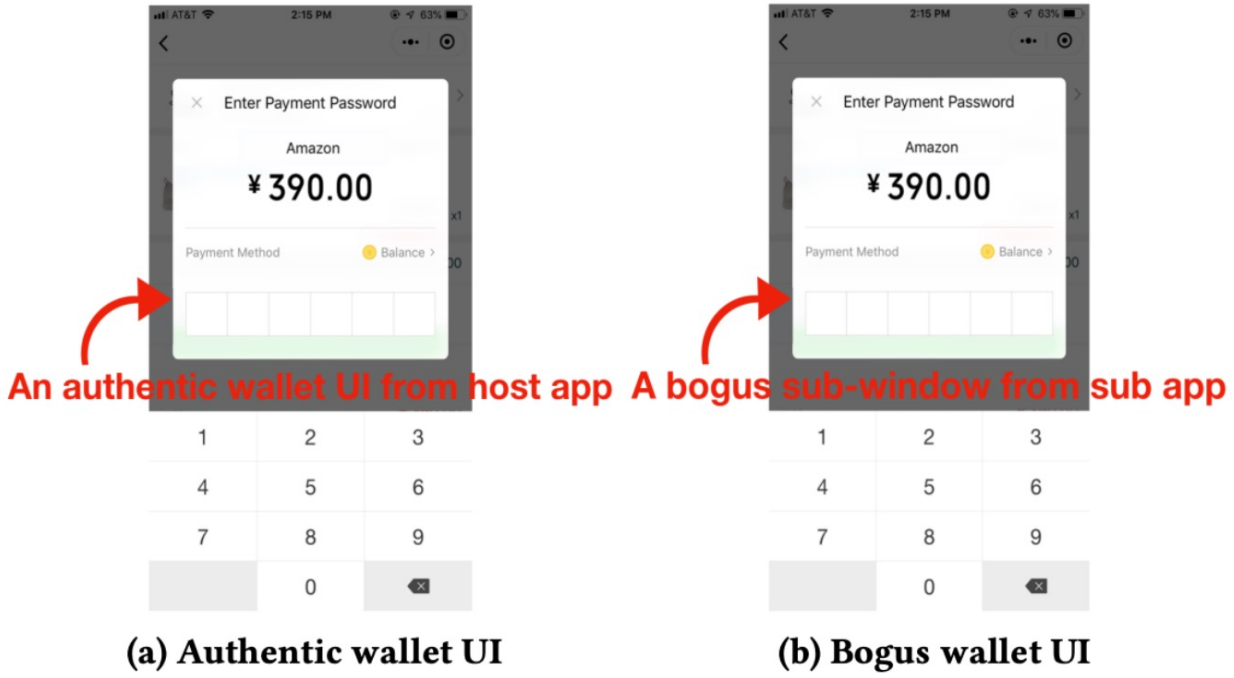
2. Sub-window Deception

When the user navigates to out-of-scope URL in a Web App
the browser displays a Prominent UI



(a) Real Prominent UI (b) Fake Prominent UI

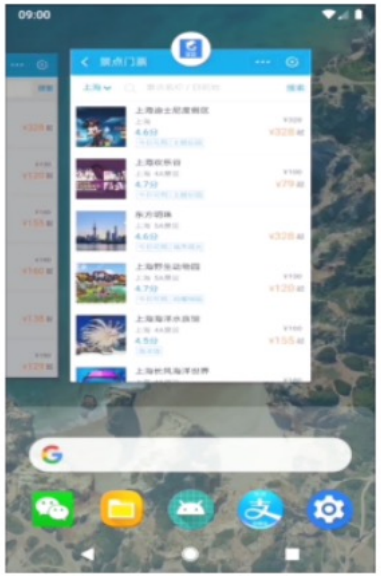
Browsers' Prominent UI confusion



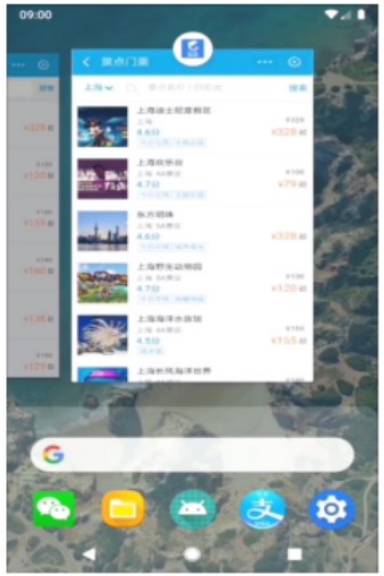
(a) Authentic wallet UI (b) Bogus wallet UI

Mobile Wallet UI confusion

3. Sub-app Lifecycle Hijacking



(a) Real sub-app task



(b) Bogus sub-app task

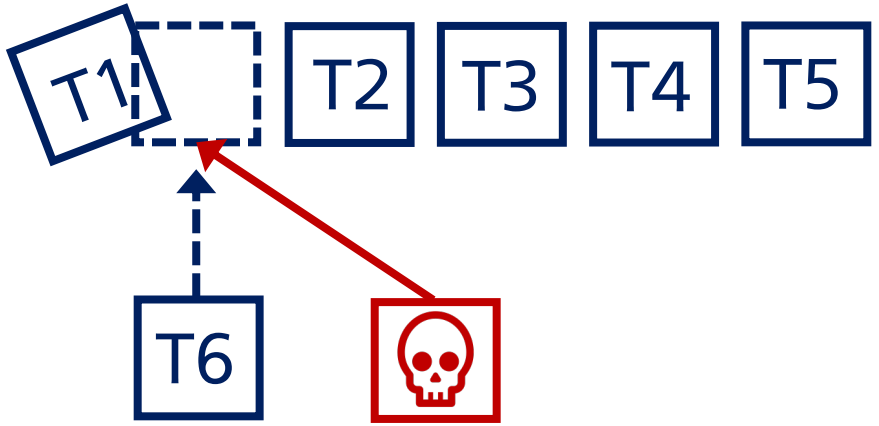
Recents screen takeover

Recents screen?

: system-owned UI to list recently accessed task

Ex)

Maximum: 5

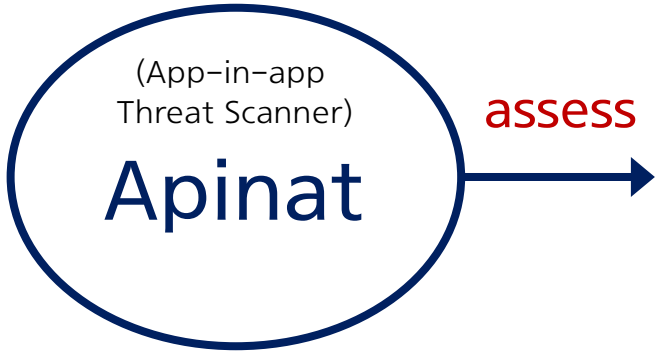


APINA flaws in the wild

by an automatic analysis tool, Apinat

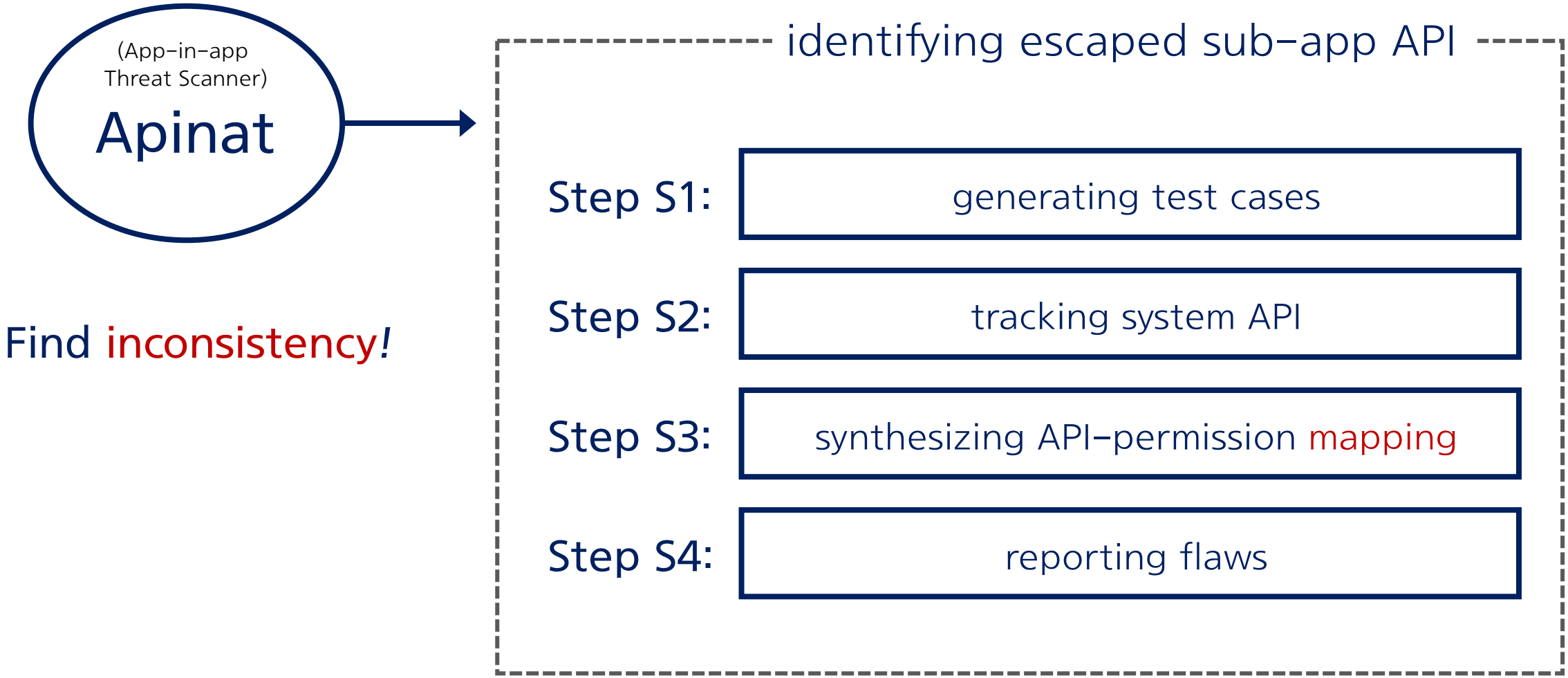
(short for App-in-app Threat Scanner)

APINA flaws in the wild



Host App	Functionality (of the host itself, except sub-apps)	Number of Sub-App	Number of Downloads
WeChat	Instant Messaging, Social Networking, Social Media, VoIP, Mobile Wallet	1M+	200M+
Alipay	Mobile Wallet, Finance/Investment Management, Shopping, News, Social Networking, Utility Bill Management, Travel	120,000+	1B+
Chrome	Web Browser	N/A†	1B+
Safari	Web Browser	N/A†	N/A†
TikTok	Video Sharing, Instant Messaging, Personal Blog, Game Center	N/A†	500M+
JinRiTouTiao	News Feeding, live streaming	N/A†	1M+
QQ	Instant Messaging, Video/Audio Chatting, File Transfer, Personal Blog, Game Center, Mobile payment	N/A†	10M+
Firefox	Web Browser	N/A†	100M+
Opera	Web Browser	N/A†	100M +
Baidu	Search engine, News, Short Videos, Voice Recognition, Readings	N/A†	230M+
DingTalk	Address Book, Mobile Office Tool Box, Video Conference	20,000	500K+
Total	-	1M+	2.6B+

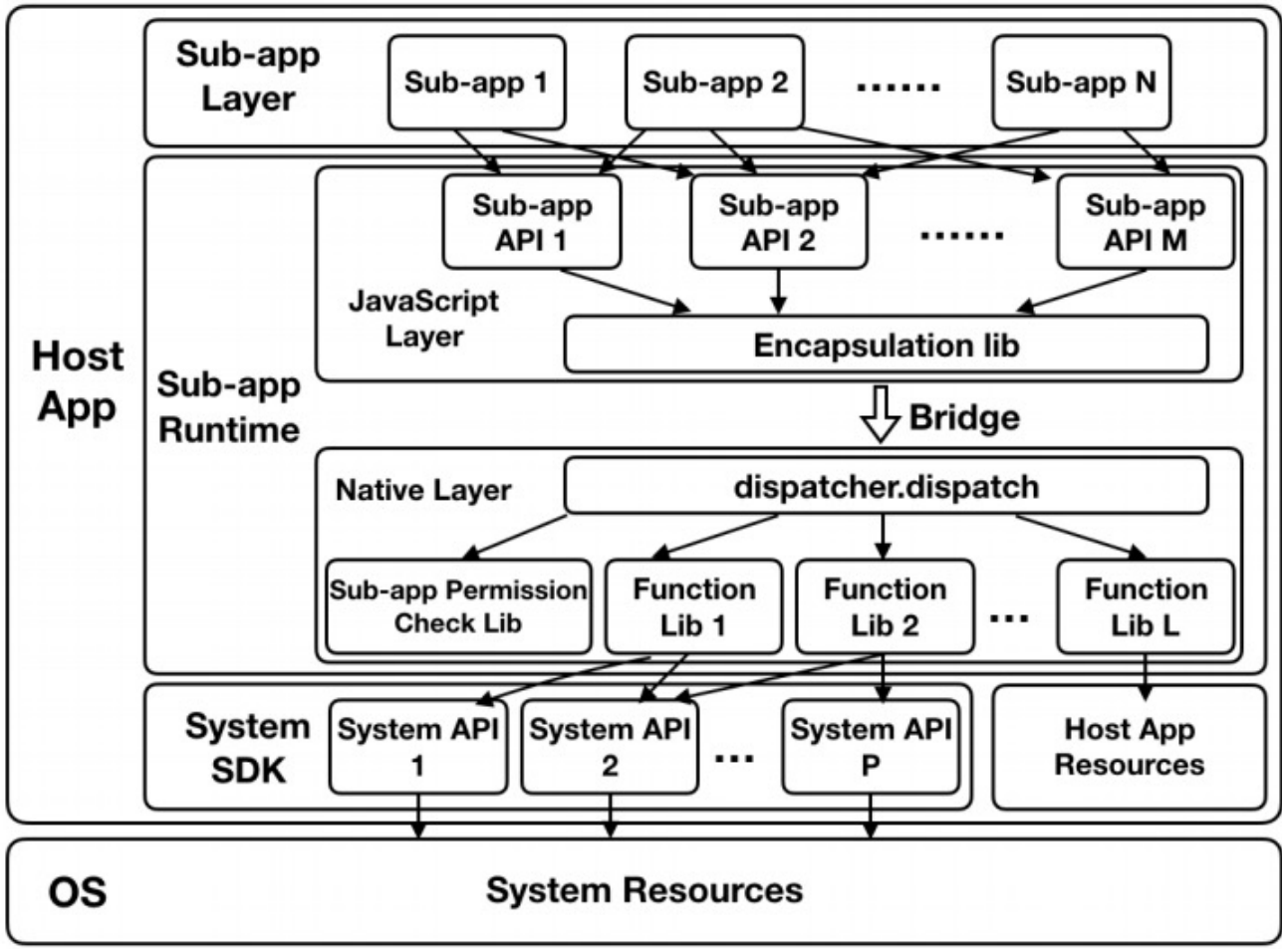
Identifying System Resource Exposure



Find inconsistency!

How to tracking system API

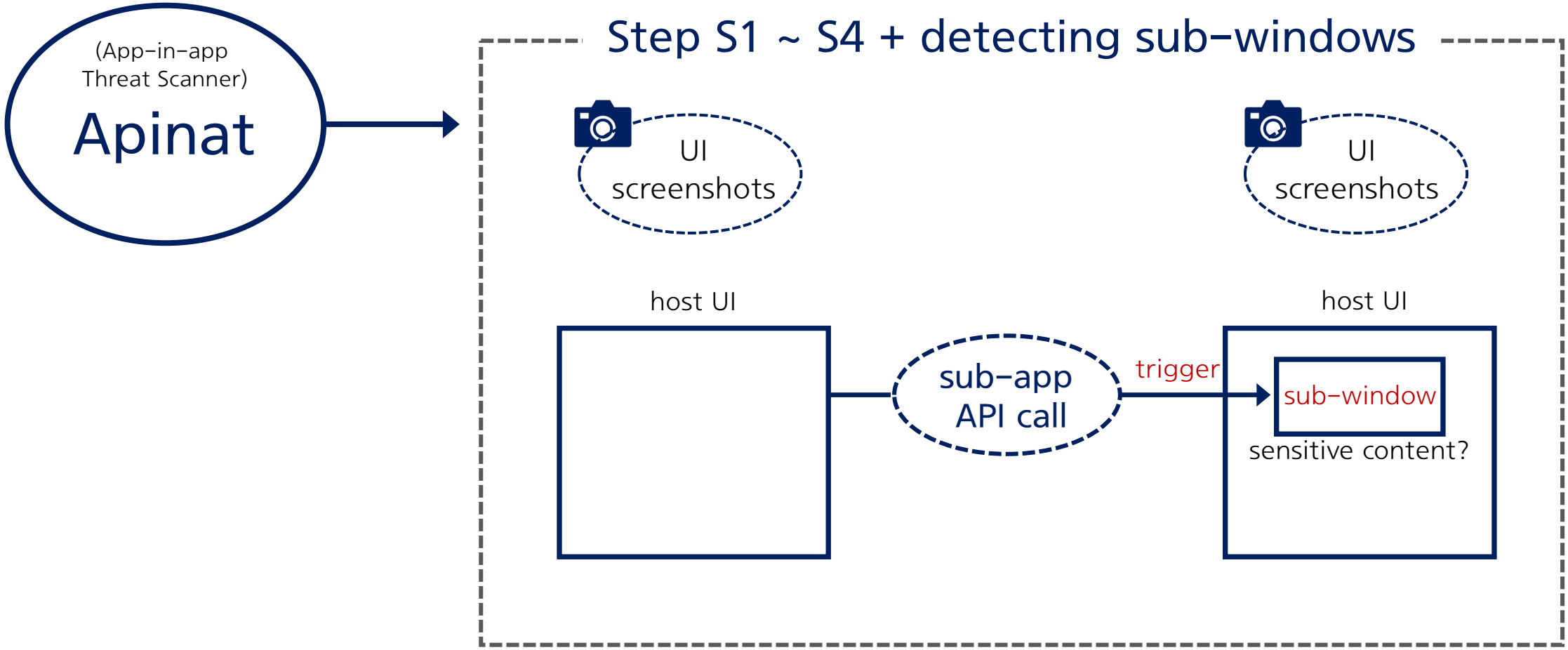
(1) Find the dispatcher.dispatch function in the native layer. (2) Track execution of the dispatcher.dispatch function



WebView & React Native

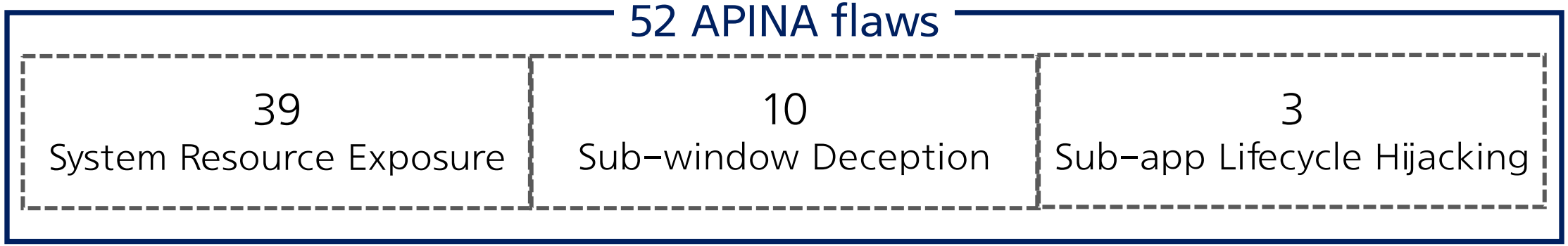
```
addJavascriptInterface(obj, 'dispatcher')
```

Finding UI Deception Flaws



Measurement of Impact

Type of APINA Flaw	iOS	Android
System Resource Exposure	A, D, J, Q, T, W	A, D, J, Q, T, W
Sub-window Deception	A, B, D, W, S, Q	A, B, C, D, F, W, O, Q
Sub-app Lifecycle Hijacking	N/A	A, W, Q



In 11 highly popular host apps

APINA risks come from…

App-level

host app's limited capabilities in managing OS resources

→ lacks the full OS-level knowledge on system resource protection

OS-level

lack of OS-level support, missing of sub-app API standard

→ mobile OSes have confusing, conflicting security policies

Mitigation strategy

for Escaped Sub-app API

1 full OS-level knowledge ⟶ difficult

for Escaped Sub-app API

1 full OS-level knowledge ⟶ difficult

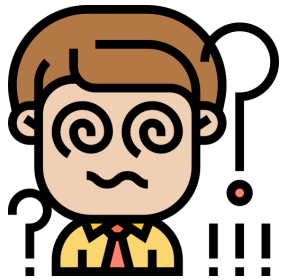
Help identify escaped sub-app APIs

2 Apinat + 3 NLP ⟶ deriving more complete knowledge about API permission policies

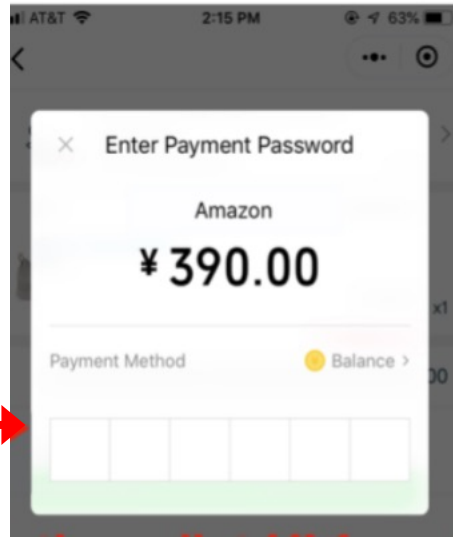
+ API-permission mapping

for Sub-window deception

: UI isolation between the sub-app and the host



A wallet UI from host-app?
Or A bogus sub-window from sub-app?



for Sub-app lifecycle hijacking

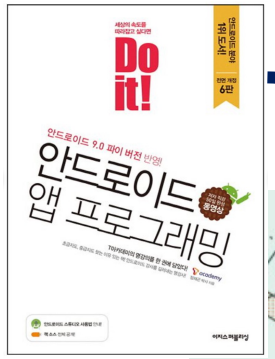
: Placing all sub-app resources in the phone's internal storage

External Storage

Internal Storage

Study Status

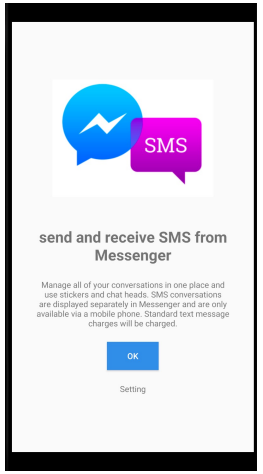
Android App Programming

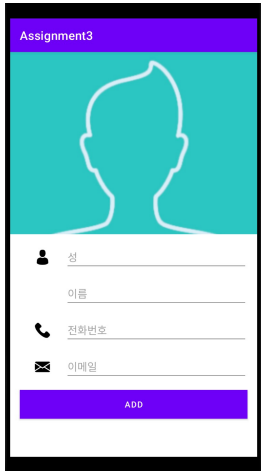


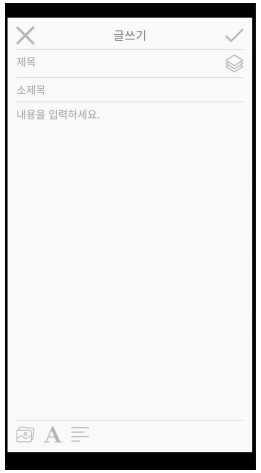
Study process

1. studying individually,
2. performing the assignments and uploading the output to GitHub
3. sharing what we studied during regular meetings

Assignments







Android App Reverse Engineering

Android App Reverse Engineering 101

What I've done

Table of Contents

https://www.ragingrock.com/AndroidAppRE/app_fundamentals.html

Weekly Plan

Mobile App programming

Read it all and finish all the assignments quickly

Reverse Engineering

Android App Reverse Engineering 101

Finish tutorial quickly

Study the small instruction set

https://source.android.com/devices/tech/dalvik/dalvik-bytecode#instructions

* Reading Papers

App in the Middle: Demystify Application Virtualization in Android and its Security Threats