บจก.สยามถนัดแฮก

NCSA CTF Boot Camp #2

Mobile Security

Responsible: Mr. Pichaya Morimoto
Version (Date): 1.0 (2024-09-14)
Confidentiality class: Public

whoami

Pichaya (LongCat) Morimoto
Lead Penetration Tester
Siam Thanat Hack Co., Ltd.

Peeratach (Peter) Butto
Penetration Tester
Siam Thanat Hack Co., Ltd.

Yasinthon (Not) Khemprakhon
Penetration Tester
Siam Thanat Hack Co., Ltd.

Disclaimer

- จุดประสงค์ของการบรรยาย นี้เพื่อแบ่งปันความรู้ ทางด้านความปลอดภัยระบบสารสนเทศ
- ไม่สนับสนุนการนำความรู้ทางด้านความปลอดภัยฯ ไปใช้ในทางที่ผิดกฎหมายทั้งหมด
- ตัวอย่างโค้ด และรูปใน การบรรยาย นี้ เป็นระบบจำลองของทางผู้บรรยาย ไม่ใช่ระบบลูกค้า

Agenda (Day 2)

เวลา	รายละเอียด
09.00 - 10.30	Mobile Application Security
10.30 - 10.45	พักเบรก
10.45 - 12.00	Programming for CTF
12.00 - 13.00	พักรับประทาน อาหารกลางวัน
13.00 - 16.00	การแข่งขัน CTF (3 ชม.)
16.00 - 17.00	มอบรางวัลและพิธีปิด

Content Overview

- Mobile App Security 101
  - แฮกโทรศัพท์
  - แฮกแอปในโทรศัพท์
    - Client vs Server
    - Android Component
- Android Testing Setup
- วิเคราะห์ Mobile App แบบ Static
  - วิเคราะห์ไฟล์ APK (Android)
  - mobsf
- วิเคราะห์ Mobile App แบบ Dynamic
  - Android Virtual Device (AVD)
  - Android Debugging Bridge (ADB)
  - Magisk + Burp Suite
- ลองทำแลป !
  - Lab 1: วิเคราะห์ Local Storage
  - Lab 2: วิเคราะห์ PIN API

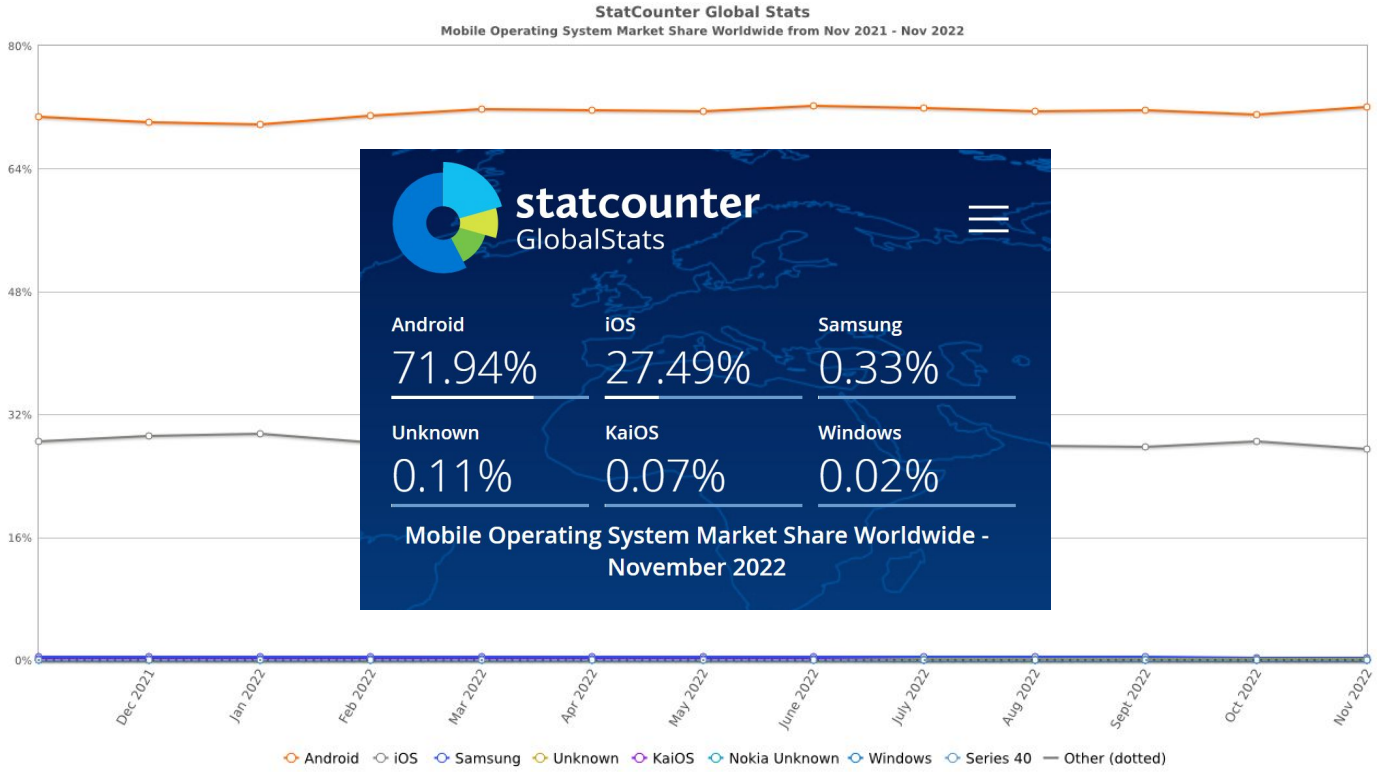
1-hour Mobile App Security 101 ?

- Mobile Application Penetration Testing (MAPT) การทดสอบเจาะระบบแอปพลิเคชันบนโทรศัพท์
- เน้นที่ Android แต่เทคนิคและแนวคิดสามารถประยุกต์ใช้กับแอป iOS ได้

ครอบคลุม:

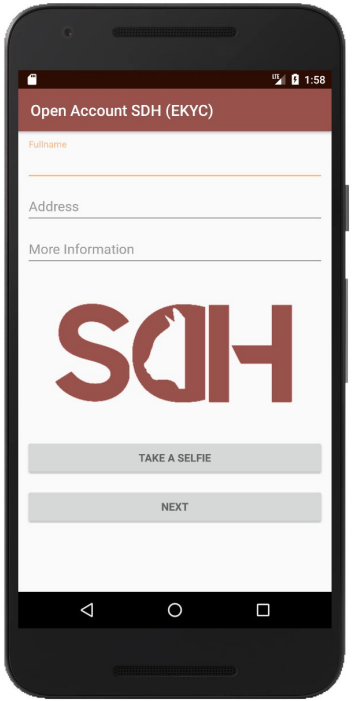
- ความรู้พื้นฐานที่จำเป็น เกี่ยวกับ Mobile Security
- ขั้นตอนการปฏิบัติและกระบวนการต่าง ๆ
  - **ฝึกอบรมสามารถลงมือทำจริง**
  - **ภายใต้สถานการณ์จำลอง**
- ได้เห็นตัวอย่างโจทย์ CTF เกี่ยวกับ Mobile App

Android Market Share



https://gs.statcounter.com/os-market-share/mobile/worldwide

Mobile Application Security 101



Objective:

Find security issues, find flags.

Security issues?:

- Client-side architecture
  - Insecure data storage
  - SQL Injection (SQLite)
  - Design Flaw in Deeplink
  - WebView Client-Side RCE
- Server-side API
  - Broken Access Control
  - SQL Injection
  - Business Logic Flaw

“Think like a hacker, but act like an engineer.”

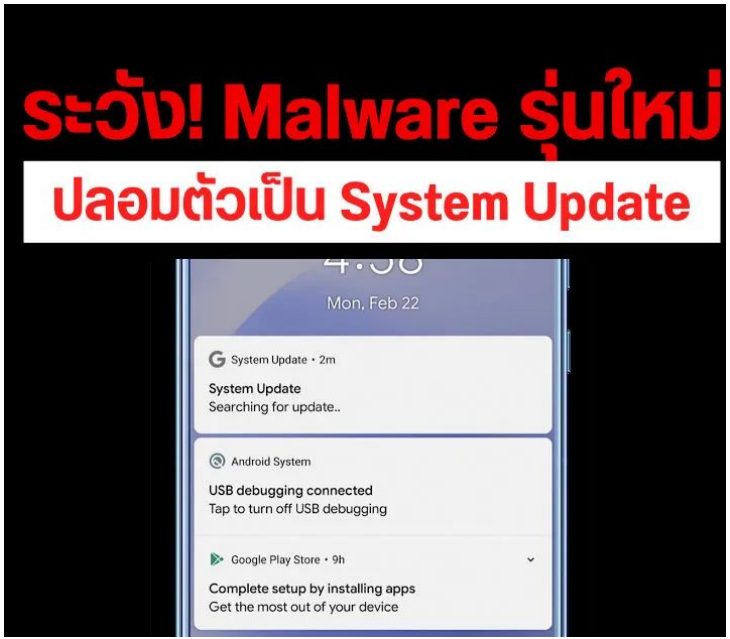
แฮกโทรศัพท์

Android OS Security - Security Concerns

Feature	Observation
Sideloading	Allow installing app from untrusted sources, which can potentially be misused
External Storage	Have a shared external storage that can be accessed by a malicious app
OEM/Bootloader Unlock	Allow unlocking OEM/bootloader, which can potentially be misused

Comparison between android and iOS Operating System in terms of security (2013),
https://www.semanticscholar.org/paper/Comparison-between-android-and-iOS-Operating-System-Ahmad-Musa/dd7632780ffffe1d400d510f0a2bf36aca74dd37

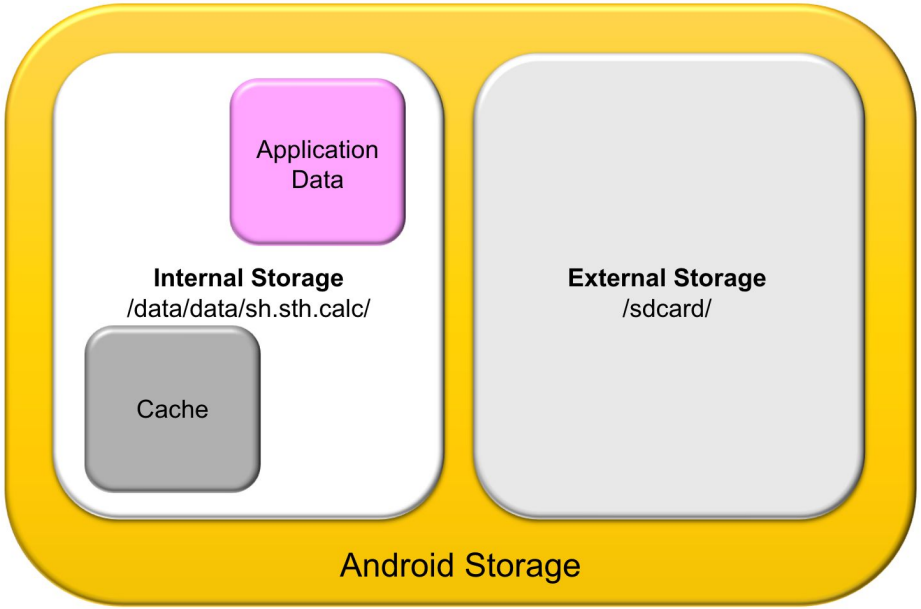
Security Concerns - (1/3) Sideloading



- ➔ Searching for files with specific extensions (including .pdf, .doc, .docx, and .xls, .xlsx);
- ➔ Inspecting the clipboard data;
- ➔ Inspecting the content of the notifications;
- ➔ Recording audio;
- ➔ Recording phone calls;
- ➔ Periodically take pictures (either through the front or back cameras);
- ➔ Listing of the installed applications;
- ➔ Stealing images and videos;
- ➔ Monitoring the GPS location;
- ➔ Stealing SMS messages;
- ➔ Stealing phone contacts;

Reference: https://droidsans.com/new-malware-fake-system-update/,
https://www.zimperium.com/blog/new-advanced-android-malware-posing-as-system-update/

Security Concerns - (2/3) External Storage



1. App-specific files
2. **Media**
3. **Documents and other files**
4. App preferences
5. Database

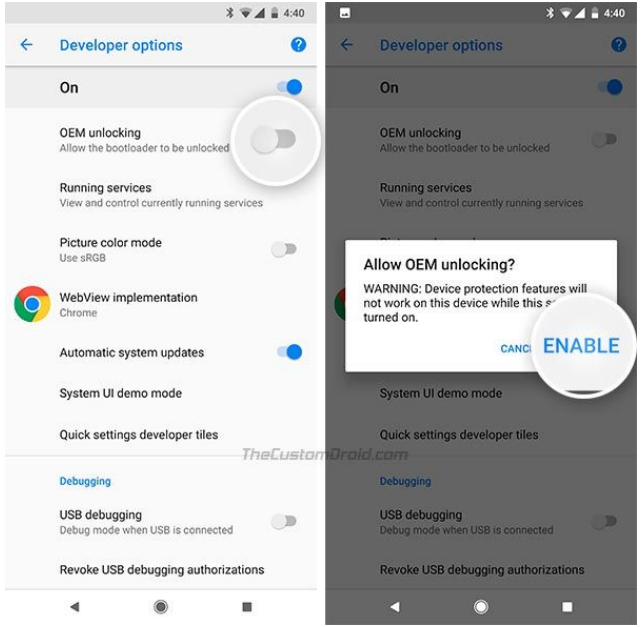
Reference: https://developer.android.com/training/data-storage

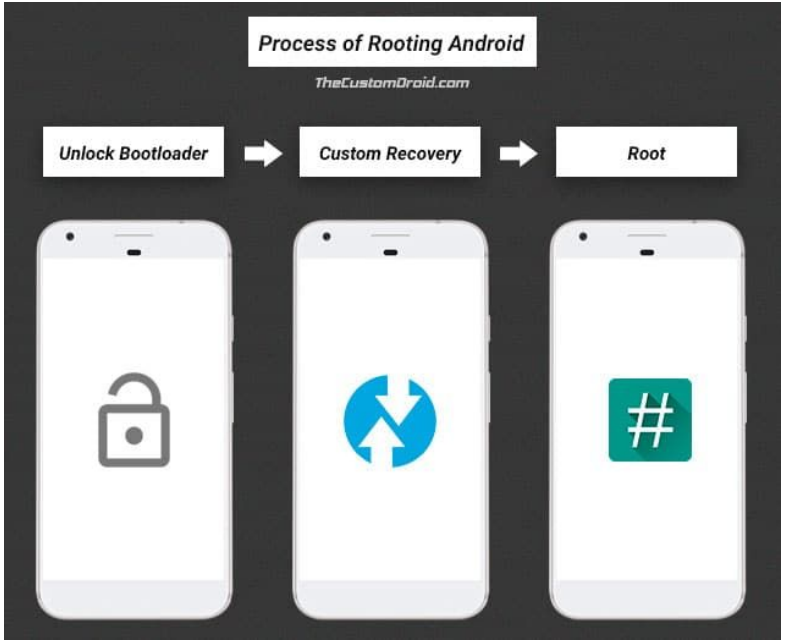
Security Concerns - (2/3) External Storage

ประเภทของข้อมูล		แอปอื่น เข้าถึงได้ ?	ลบแอปหาย ?
App-specific files	Files meant for your app's use only	ไม่ได้ *	ใช่
Media	Shareable media files (images, audio files, videos)	ได้	ไม่
Documents and other files	Other types of shareable content, including downloaded files	ได้	ไม่
App preferences	Key-value pairs	ไม่ได้ *	ใช่
Database	Structured data	ไม่ได้ *	ใช่

Reference: https://developer.android.com/training/data-storage

Security Concerns - (3/3) OEM/Bootloader Unlock



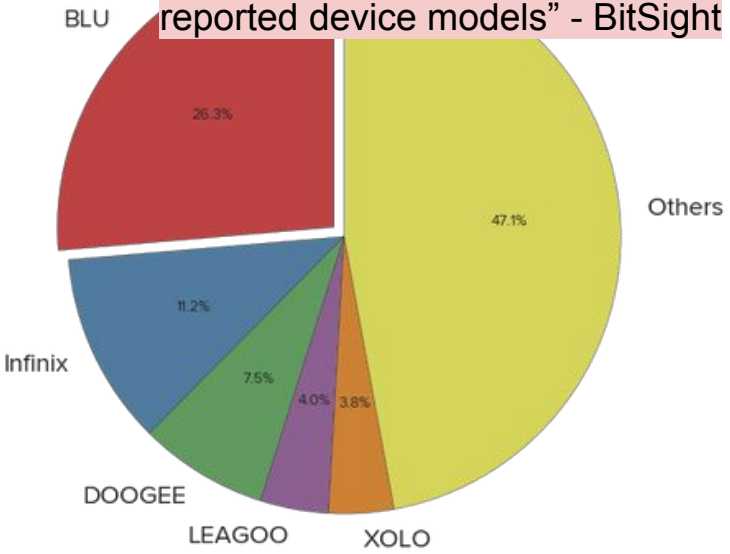


- ผู้ใช้งาน Root เครื่องโทรศัพท์พัง App Sandboxing
- คนขายโทรศัพท์ Pre-Install มัลแวร์หรือแอปที่มีช่องโหว่

Reference: https://www.thecustomdroid.com/enable-oem-unlocking-on-android/

Security Concerns - (3/3) OEM/Bootloader Unlock

"We have observed over 2.8 million distinct devices, across roughly 55 reported device models" - BitSight



Reference: https://www.kb.cert.org/vuls/id/624539

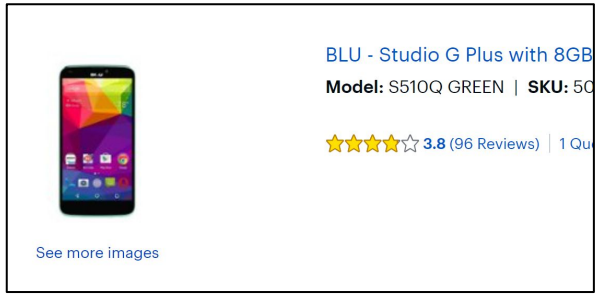
Android OTA Update RCE Vulnerability

CWE-494: Download of Code Without Integrity Check - CVE-2016-6564

Android devices with code from Ragentek contain a privileged binary that performs over-the-air (OTA) update checks.
Additionally, there are multiple techniques used to hide the execution of this binary. This behavior could be described as a rootkit.

This binary, which resides as /system/bin/debugs, runs with root privileges and does not communicate over an encrypted channel.
The binary has been shown to communicate with three hosts via HTTP:

- oyag[.]lhzbdvm[.]com
- oyag[.]prugskh[.]net
- oyag[.]prugskh[.]com



Security Concerns - (3/3) OEM/Bootloader Unlock

Android OTA Update RCE Vulnerability

Server responses to requests sent by the debugs binary include functionalities to execute arbitrary commands as root, install applications, or update configurations.

Examples of a request sent by the client binary:

```
POST /pagt/agent?data={"name":"c_regist","details":{...}} HTTP/1. 1
Host: 114.80.68.223
Connection: Close
```

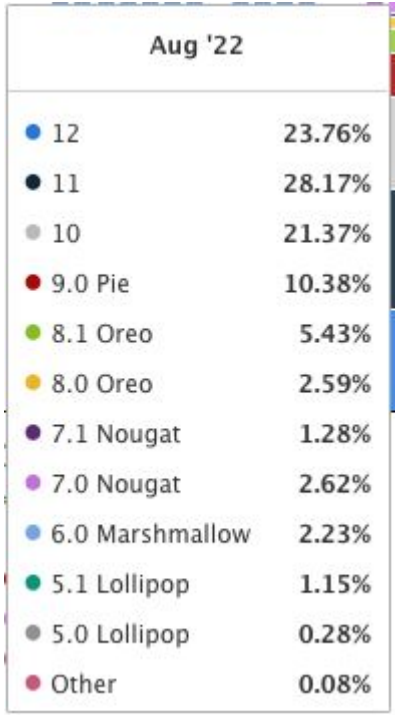
An example response from the server could be:

```
HTTP/1.1 200 OK
{"code": "01", "name": "push_commands", "details": {"server_id": "1" ,
"title": "Test Command", "comments": "Test", "commands": "touch /tmp/test"}}
```

Reference: https://www.kb.cert.org/vuls/id/624539

Android API Level, Android Version

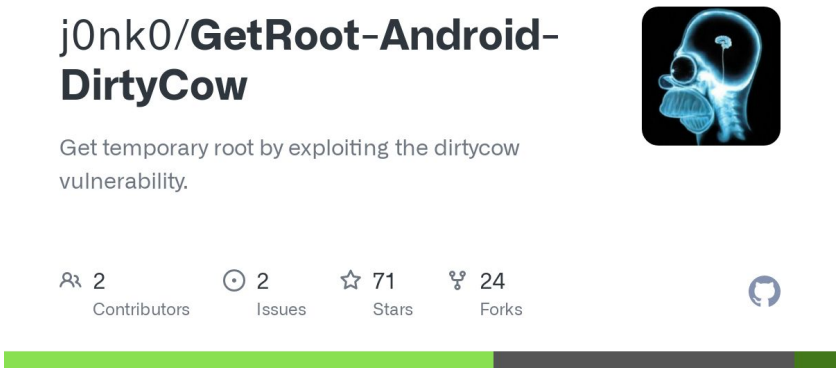
Platform Version	API Level	VERSION_CODE
Android 13	33	TIRAMISU
Android 12	32	S_V2
	31	S
Android 11	30	R
Android 10	29	Q
Android 9	28	P
Android 8.1	27	O_MR1
Android 8.0	26	O
Android 7.1.1 Android 7.1	25	N_MR1
Android 7.0	24	N
Android 6.0	23	M



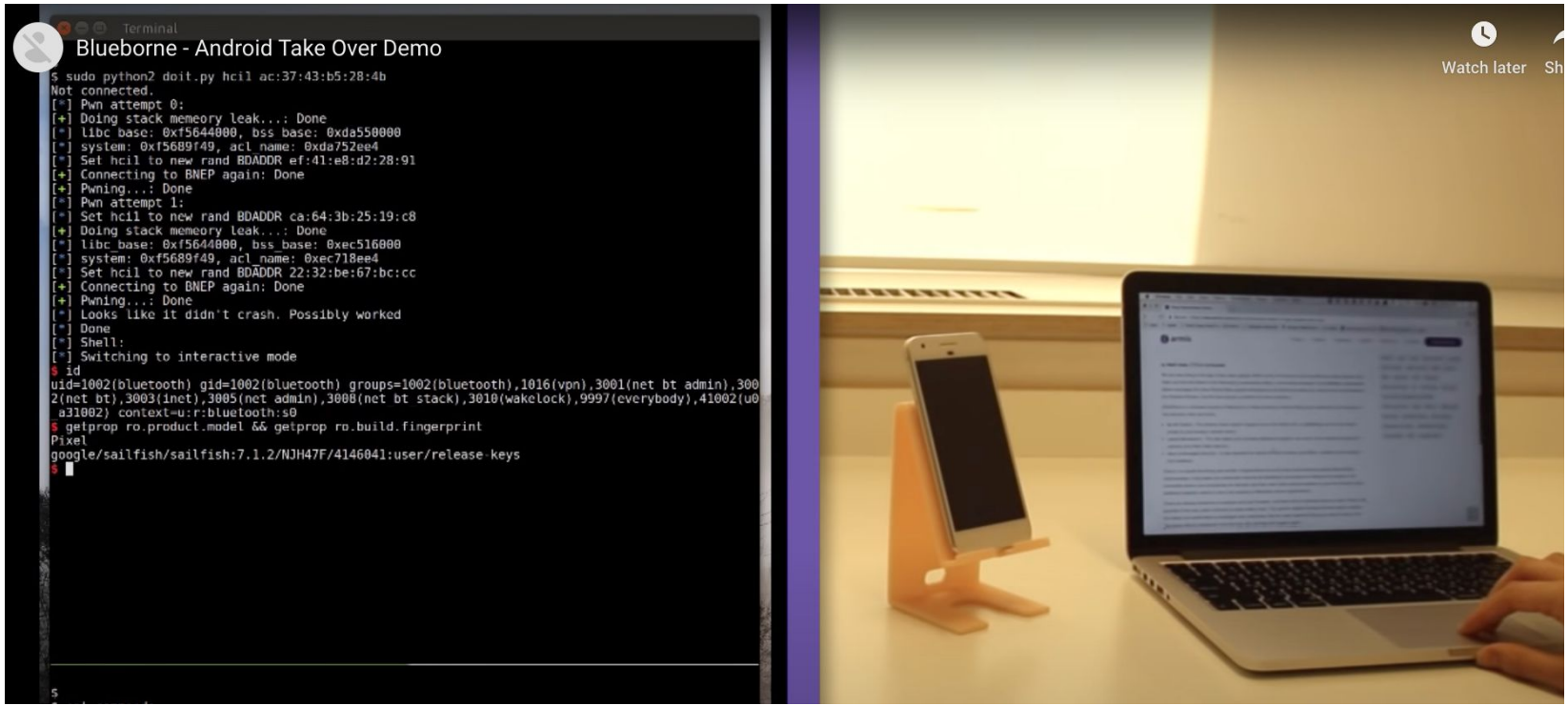
https://developer.android.com/guide/topics/manifest/uses-sdk-element
https://developer.android.com/about/dashboards
https://www.statista.com/statistics/921152/mobile-android-version-share-worldwide/

Android LPE via DirtyCOW (CVE-2016-5195)

```
adb shell
./G1tR0oT
id
uid=0(root) gid=0(root) groups=0(root),1004(input),1007(log)
```



BlueBorne Bluetooth RCE (CVE-2017-0785)



https://www.youtube.com/watch?v=Az-l90RCns8&t=41s

Android OS Security - Security Concerns

Feature	Observation
Sideloading	Allow installing app from untrusted sources, which can potentially be misused
External Storage	Have a shared external storage that can be accessed by a malicious app
OEM/Bootloader Unlock	Allow unlocking OEM/bootloader, which can potentially be misused

Comparison between android and iOS Operating System in terms of security (2013),
https://www.semanticscholar.org/paper/Comparison-between-android-and-iOS-Operating-System-Ahmad-Musa/dd7632780ffffe1d400d510f0a2bf36aca74dd37

แฮกแอปในโทรศัพท์

STH Penetration Testing Methodology (STHPTM)

1 Information Gathering
- Attack surface analysis
- Application interactions
- Service scanning
- Passive/active recons

- Mobile Client Reverse Engineering (Static Analysis)
- Anti-Reverse Engineering Bypass

2 Vulnerability Identification
- Application fuzzing
- Automated tools
- Manual security testing
- Business logic testing
- AuthN/AuthZ testing

- Client-Side Inspection
- Local Storage
- IPC (Inter-Process Communication)

3 Exploitation
- Proof of concept
- Create exploit code
- Create attack scenario
- Use public exploit code

- Runtime In-Memory Hooking (frida)
- Binary Patching

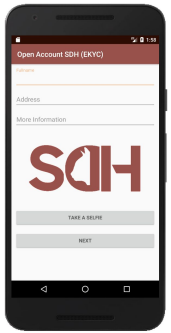
4 Documentation
- Preliminary report
- Bug tracker document
- Penetration test report
- Revisited pentest report

Application Architecture - Mobile App (Zoom-Out)

1 ผู้ใช้งาน (End User) ↔ 2 แอปพลิเคชันบนโทรศัพท์ (Mobile Client) ↔ 3 เซิร์ฟเวอร์ (Mobile Web API)

Client-side architecture:

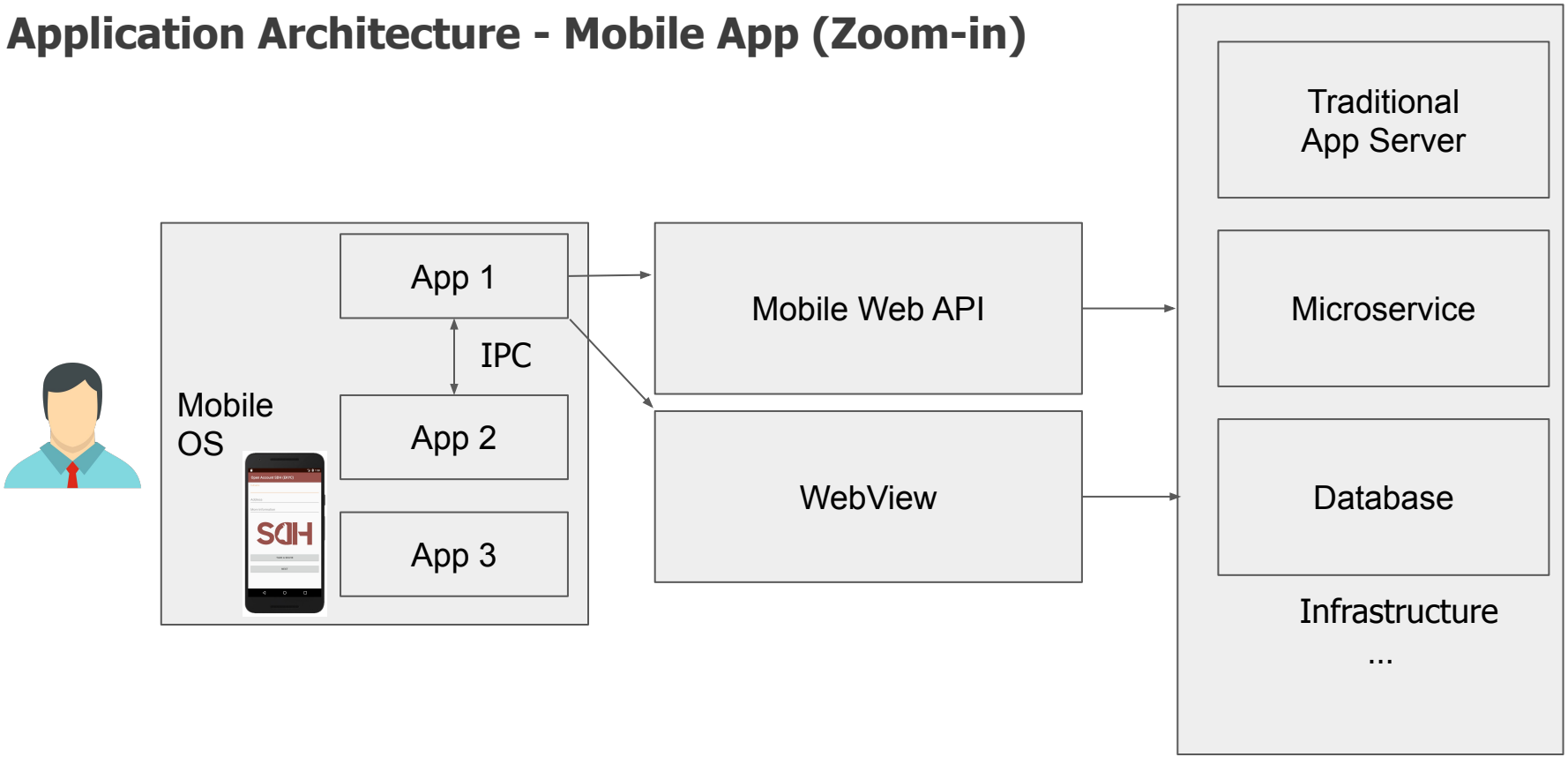
- Insecure data storage
- SQL Injection (SQLite)
- Design Flaw in Deeplink
- WebView Client-Side RCE



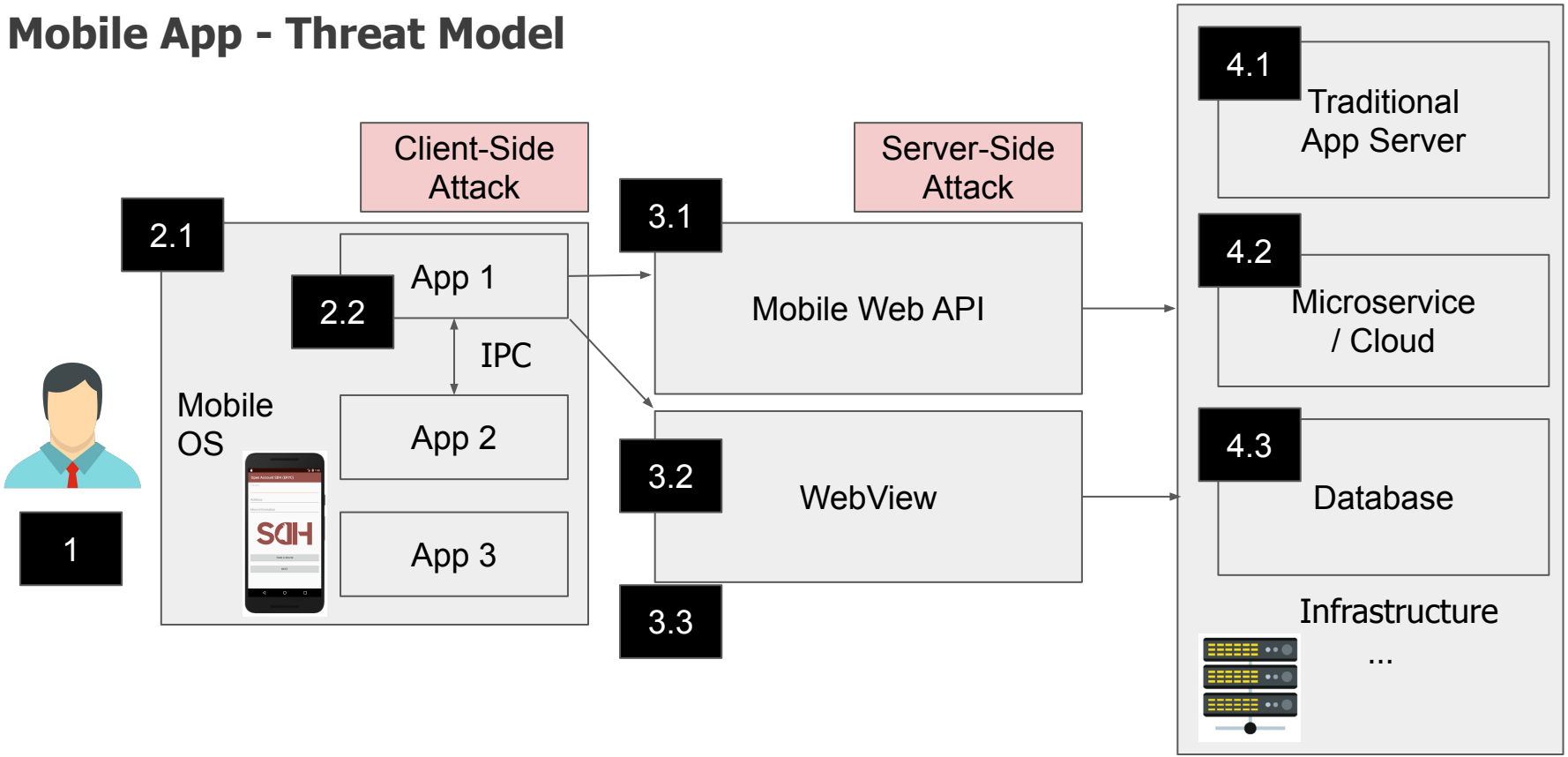
Server-side API:

- Broken Access Control
- SQL Injection
- Business Logic Flaw

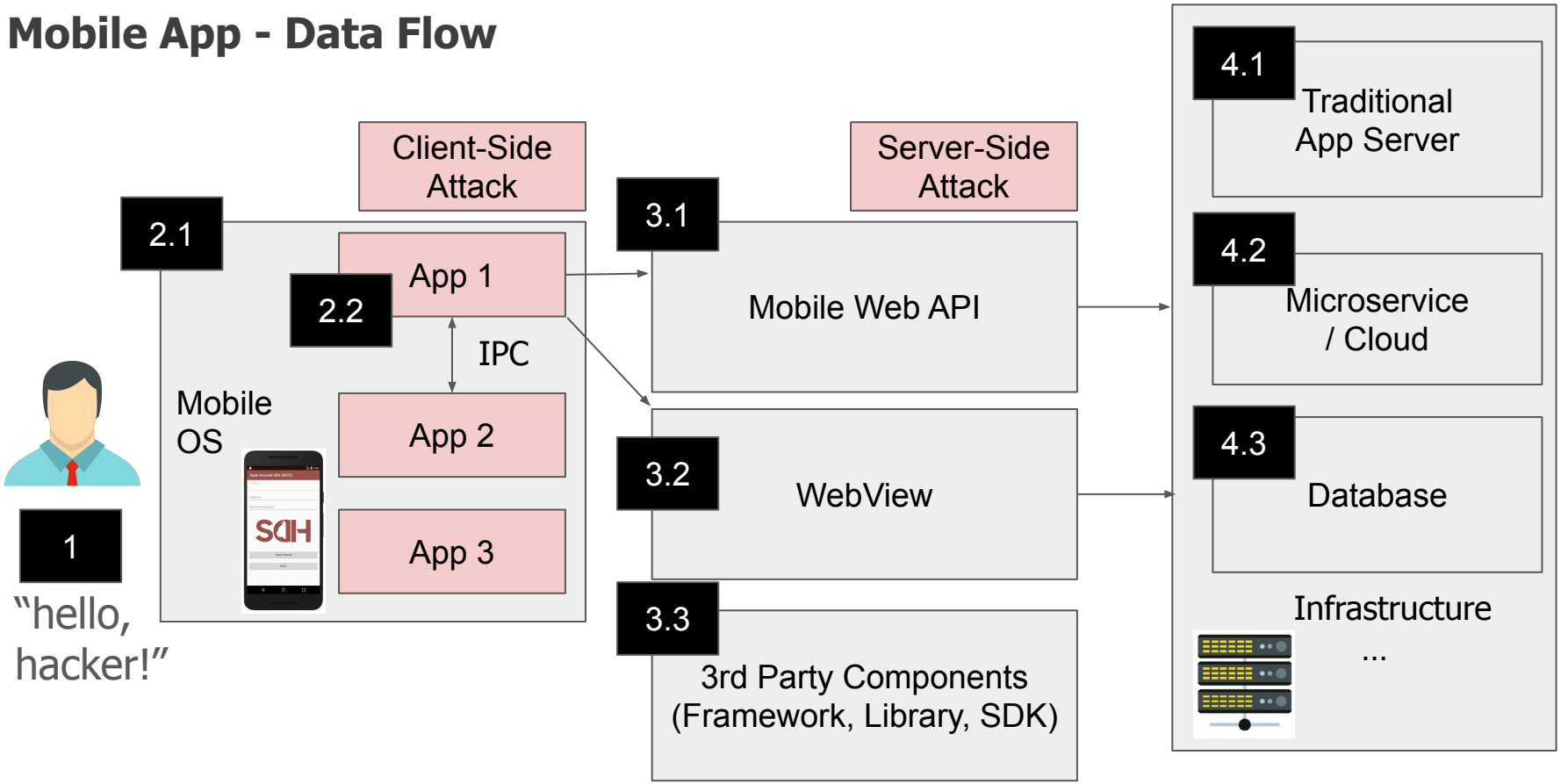
Application Architecture - Mobile App (Zoom-in)



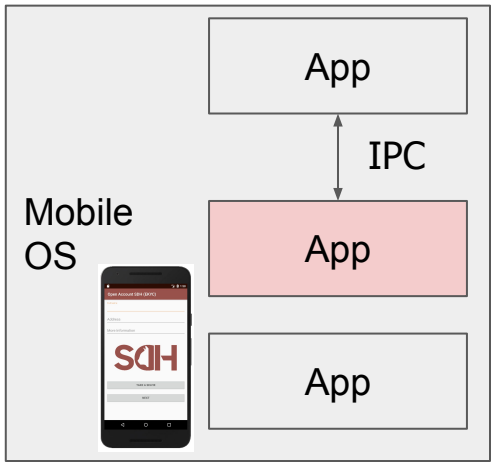
Mobile App - Threat Model

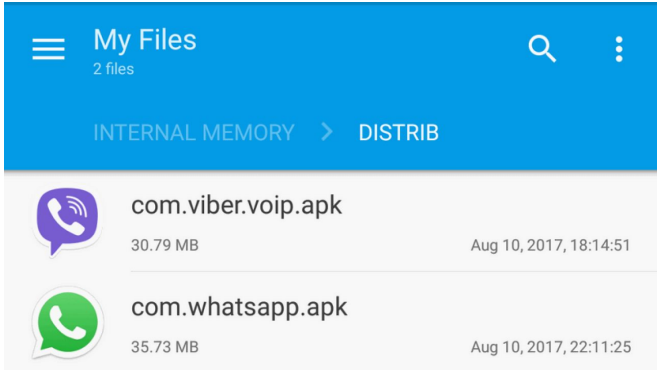


Mobile App - Data Flow



Mobile App Basic - Mobile Client



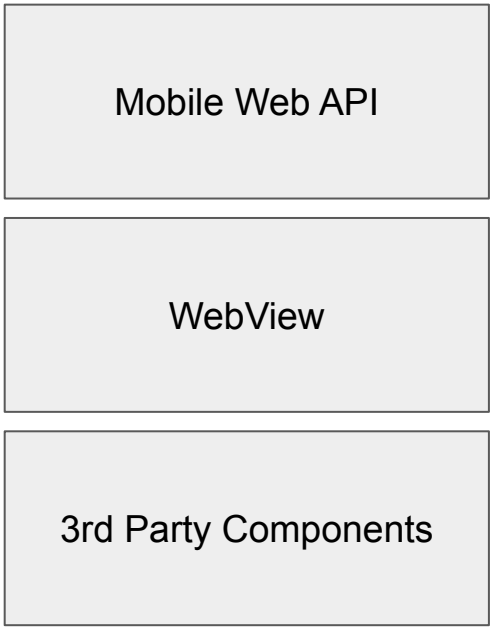


Mobile OS:
- Android, iOS

App Installer File:
- **Android:** Google Play
  - .apk
  - .aab
- **iOS:** App Store
  - .ipa

IPC:
- Deeplink
- Intent

Mobile App Basic - Mobile API Server

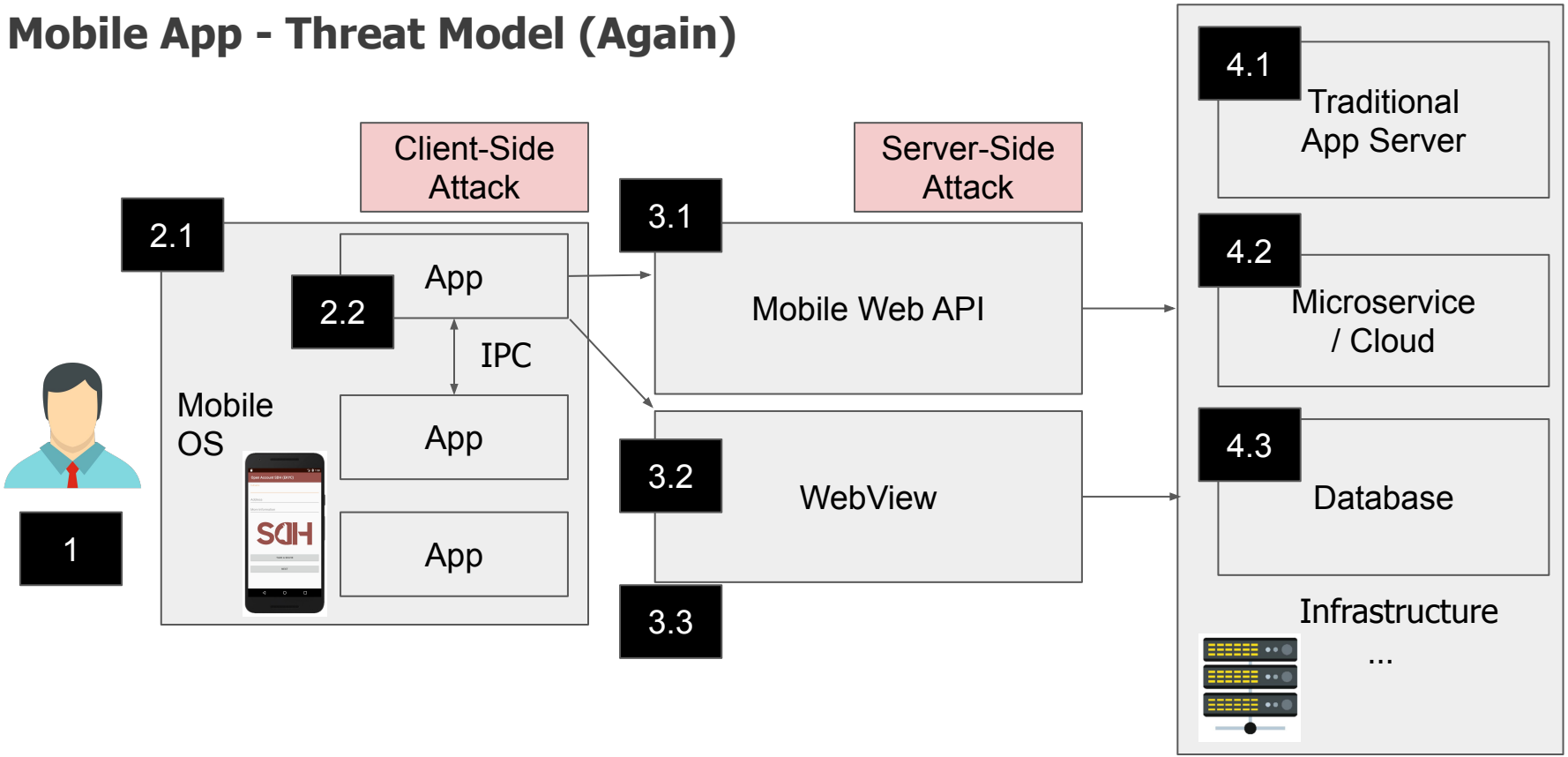


Mobile Web API
API Endpoint
Web Service
- https://mobile.sdh.p7z.pw/api/news/1
- https://mobile.sdh.p7z.pw/api/endpoint

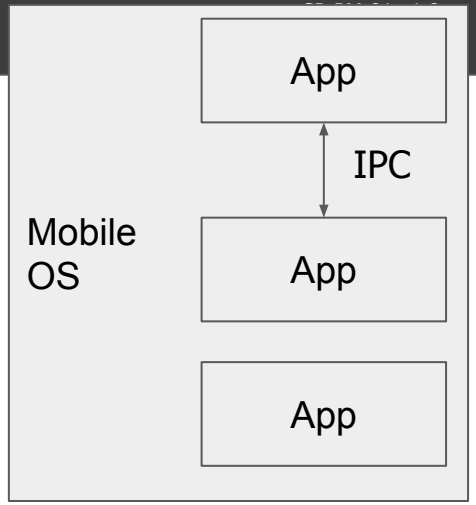
WebView
- https://sdh.p7z.pw/article?id=123
- https://sdh.p7z.pw/promotion/456
- wss://sdh.p7z.pw:1234/api/realtime

3rd Party Components
- Firebase
- Google Analytics
- OneSignal

Mobile App - Threat Model (Again)



Client-Side Attack Surface



UI Input Validation
- Parameter Tampering
- Hidden/Debugging Functionality

Data Storage
- External Storage (SD Card)
- SQLite
- Shared Preferences

IPC Design
- Android
  - Activity
  - Intent
  - Content Provider
  - Broadcast Receiver
  - Service

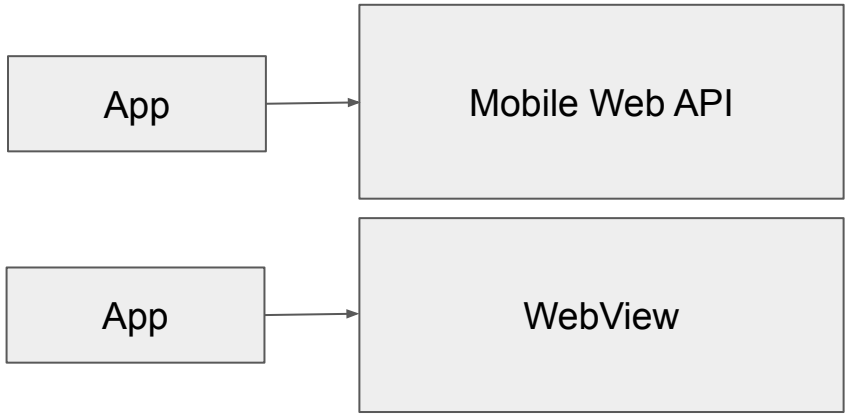
Reverse Engineering
- Root/Jailbreak Detection
- Binary Obfuscation
- Anti Debugging

อื่น ๆ
- Clipboard
- Protocol Scheme Handler
- Application Logging
- Background Screenshot
- Backup file
- Secret Management
- Library / Framework Vulnerabilities

Server-Side Attack Surface

API Communication

- In-Transit Data Encryption (TLS)
- Certificate Validation
- Certificate Pinning
- End-to-End Encryption

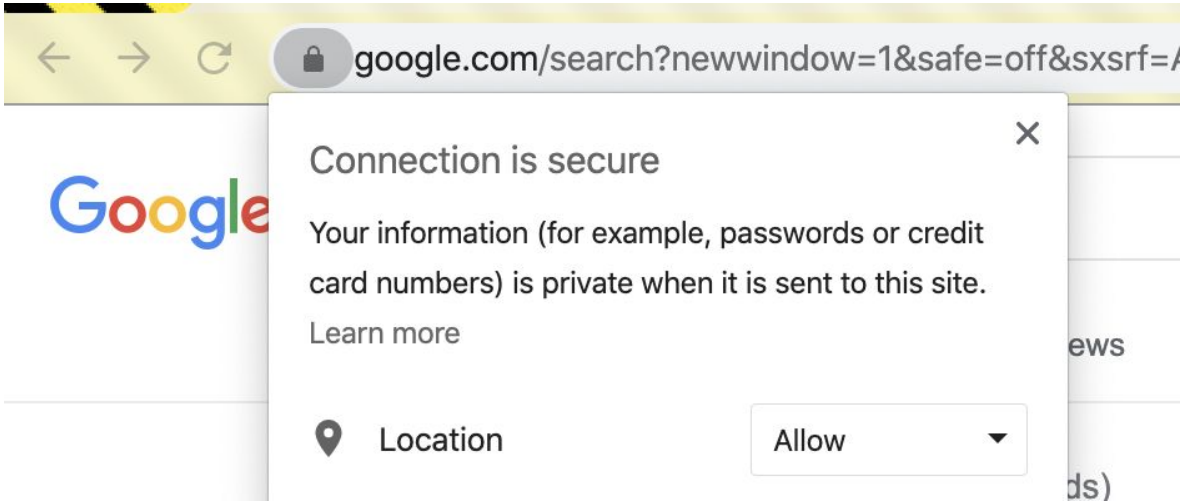


OWASP Top 10 Web/API

- Broken Access Control
- SQL Injection
- Excessive Data Exposure
- Command Injection
- Multi-Step Action Implementation

คำถามยอดฮิต - HTTPS

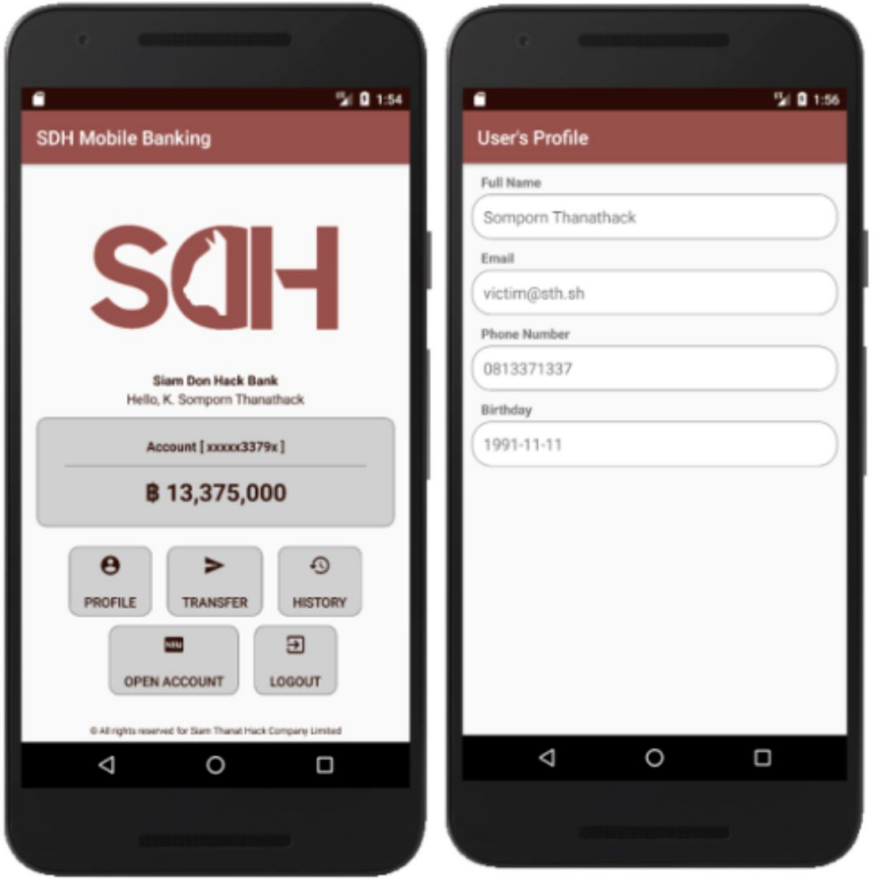
Mobile Web API มี https แล้ว ปลอดภัยไหม?
Google, Facebook ก็ใช้ แปลว่าใช้แล้วเว็บปลอดภัยไม่โดนแฮก?



OWASP

OWASP Mobile Top 10 - 2016

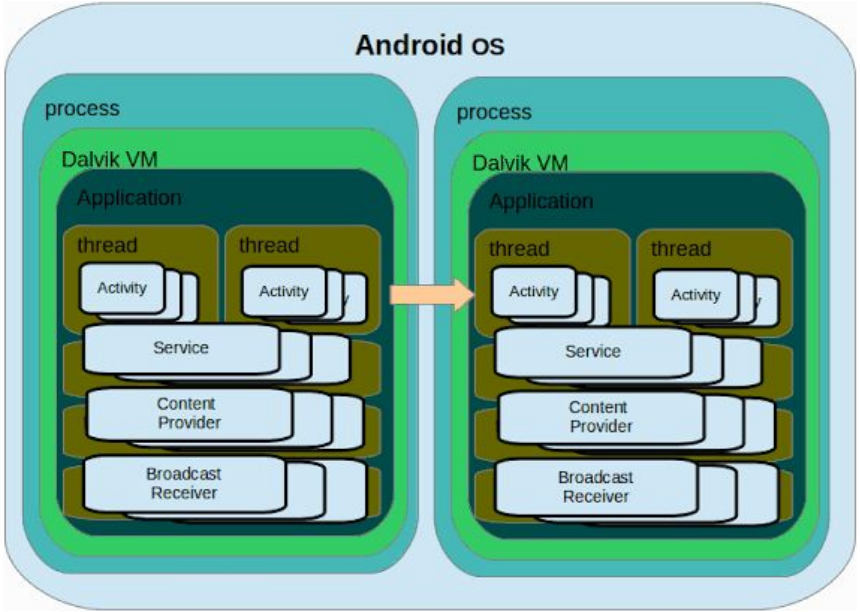
M1:2016 - Improper Platform Usage
M2:2016 - Insecure Data Storage
M3:2016 - Insecure Communication
M4:2016 - Insecure Authentication
M5:2016 - Insufficient Cryptography
M6:2016 - Insecure Authorization
M7:2016 - Client Code Quality
M8:2016 - Code Tampering
M9:2016 - Reverse Engineering
M10:2016 - Extraneous Functionality



Android App Internal (Basic)

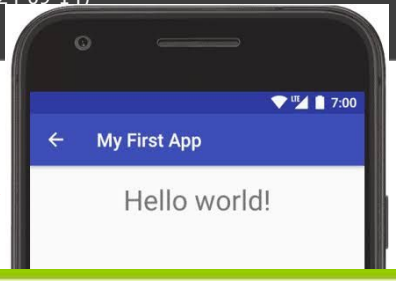
Android Components

1. Activity
2. Service
3. Broadcast Receiver
4. Content Provider



http://thebirdsarecomingcloseyourwindows.blogspot.com/2013/05/android-components.html

Android Activity



1. Activity

onCreate()

onStart()

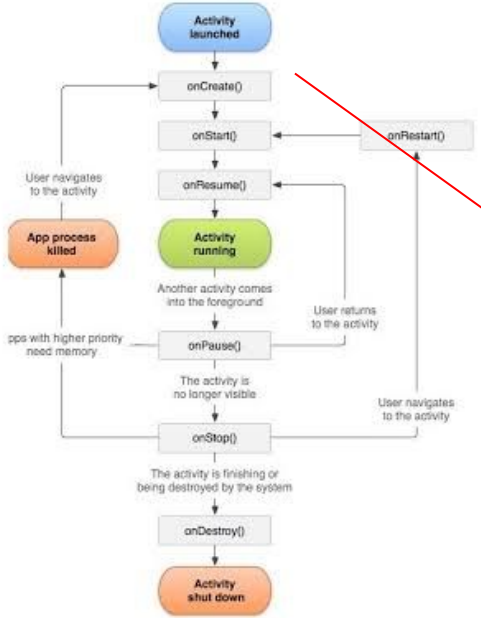
onRestart()

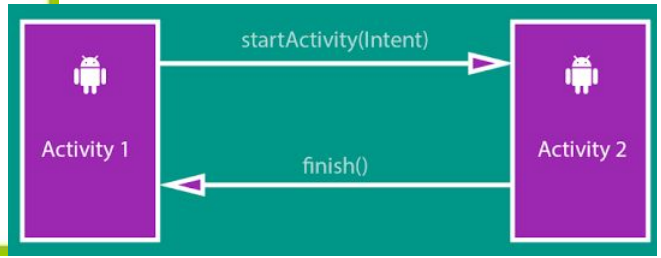
onResume()

onPause()

onStop()

onDestroy()



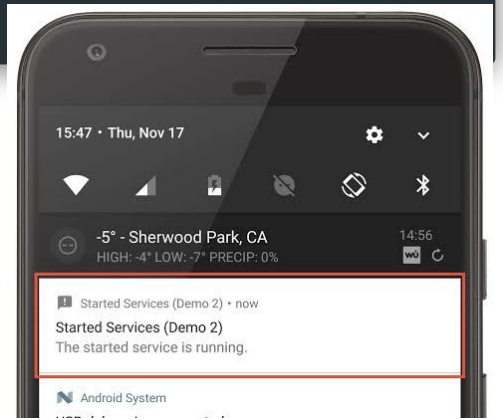


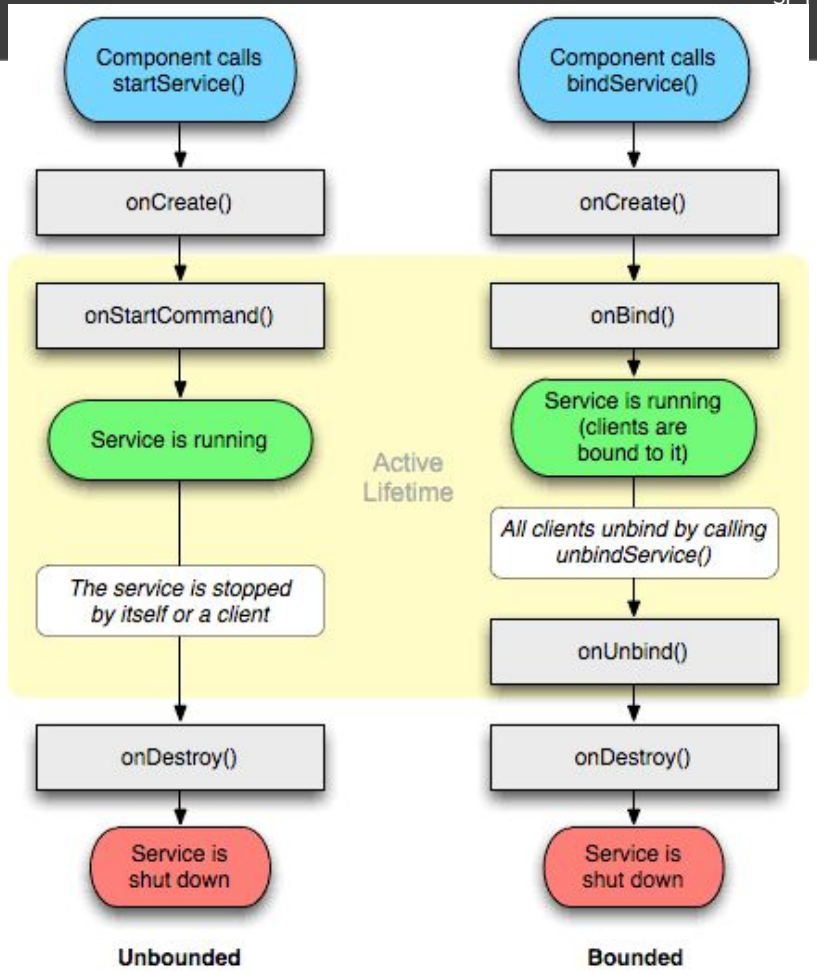
https://www.youtube.com/watch?v=UJN3AL4tiqw

Android Service

2. Service

```
override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
    val notification = initialNotificiaton()
    startForeground(ID, notification)
    initiateBackgroundWork(intent, flags)
    return START_STICKY
}
```





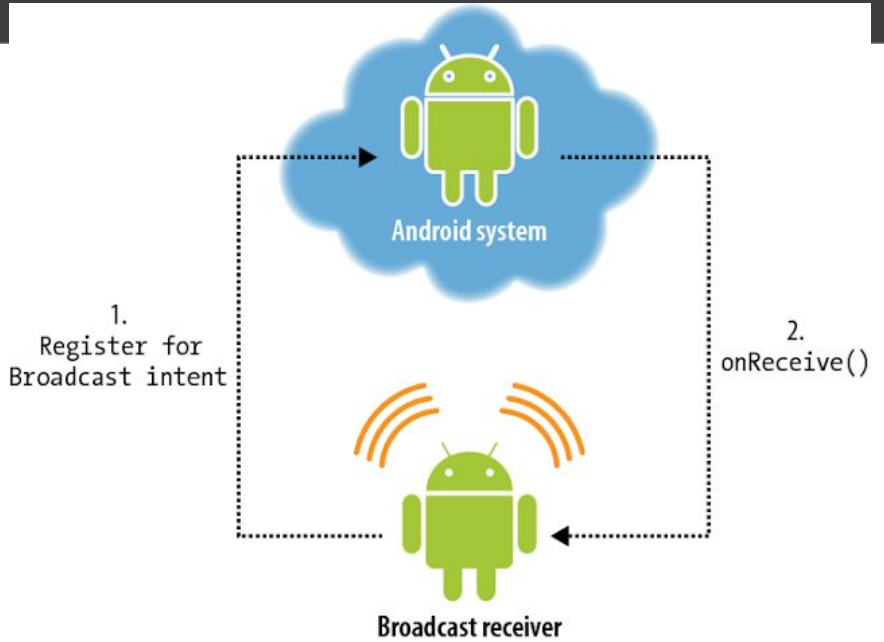
https://riptutorial.com/android/example/3505/lifecycle-of-a-service
https://www.hellsoft.se/how-to-service-on-android---part-2/
https://docs.microsoft.com/en-us/xamarin/android/app-fundamentals/services/service-notifications

Android Broadcast Receiver

3. Broadcast Receiver

```
public class MyReceiver extends BroadcastReceiver {
    public MyReceiver() {}

    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context, "Action: " + intent.getAction(), Toast.LENGTH_SHORT).show();
    }
}
```

```
Intent intent = new Intent();
intent.setAction("com.example.Broadcast");
intent.putExtra("MyData", 1000);
sendBroadcast(intent);
```




```
# trigger a broadcast and deliver it to a component
adb shell am activity/service/broadcast -a ACTION -c CATEGORY -n NAME

# for example send to specific package (this goes into one line)

adb shell am broadcast -a
android.intent.action.BOOT_COMPLETED -c android.intent.category.HOME -n
package_name/class_name

# for example send to all packages
adb shell am broadcast -a android.intent.action.BOOT_COMPLETED
```

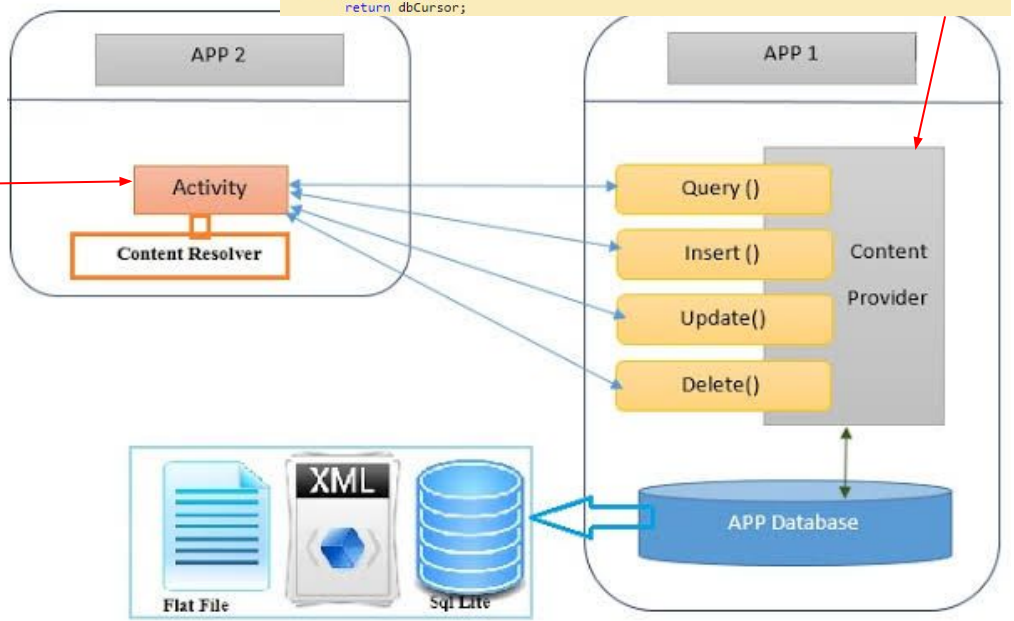
https://en.proft.me/2017/03/10/android-broadcastreceiver-tutorial/

Android Content Provider

4. Content Provider

```
public class ContentProviderForSQLite extends ContentProvider
{
    public static String DB_PATH = "/storage/emulated/0/Pictures/OpinionMining/";

    public static String DB_NAME = "%s";// This has to come through table name parameter

    private SQLiteDatabase db;


    @Override
    public Cursor query(Uri uri, String[] projection, String selection,
            String[] selectionArgs, String sortOrder)
    {
        DB_NAME=String.format(DB_NAME, pathSegs.get(0));

        String tableName=pathSegs.get(1);
        ////////////////////////////////// Executing and returning Query//////////////
        db=SQLiteDatabase.openDatabase(DB_PATH + DB_NAME, null, SQLiteDatabase.OPEN_READONLY);
        Cursor dbCursor = db.query(tableName, projection, selection, selectionArgs, null, null,
sortOrder);
        return dbCursor;
```

```
////////////////// Insert Query//////////////////////////////////////

            Log.i("Insert Query","calling");
            String []newRow=new String[]{null,"tension","-2"};
            String
s="content://com.integratedideas.opinionmining/OpinionSQLiteDatabase/OpinionDataTable";
            Uri uri=new Uri.Builder().build().parse(s);
            Cursor cursor = getContentResolver().query(uri, null, null,null, null);
            String []columns=cursor.getColumnNames();
            String cols="";
            ContentValues values=new ContentValues();
            for(int i=0;i<columns.length;i++)
            {
                cols=cols+ " "+columns[i];
                values.put(columns[i], newRow[i]);
            }
            Log.i("Columns are:",cols);
            uri=getContentResolver().insert(uri, values);
            Log.i("Result of Insert",uri.toString());
            ////////////////////////////////////////////////////////////////////////////
```



https://www.codeproject.com/Articles/818578/An-Absolute-Beginner-s-Guide-to-Building-and-Acces

AndroidManifest.xml

AndroidManifest.xml - Permission

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        package="sh.sth.sdhbank.mobilebanking">

    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.USE_BIOMETRIC" />

    <application
          android:allowBackup="true"
          android:icon="@mipmap/ic_launcher"
          android:label="@string/app_name"
          android:roundIcon="@mipmap/ic_launcher_round"
          android:supportsRtl="true"
          android:theme="@style/AppTheme"
          android:requestLegacyExternalStorage="true"
          tools:ignore="GoogleAppIndexingWarning"
          tools:replace="android:icon">
       <activity
```

AndroidManifest.xml - Activity

```
    <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main"
            android:theme="@style/AppTheme.NoActionBar">
    </activity>
    <activity
            android:name=".PinActivity"
            android:configChanges="orientation|keyboardHidden|screenSize"
            android:label="@string/app_name"
            android:theme="@style/Theme.Design.NoActionBar">
    </activity>
    <activity
            android:name=".LoginActivity"
            android:label="@string/app_name">
        <intent-filter>
            <action android:name="android.intent.action.MAIN"/>
            <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
    </activity>
  </application>
</manifest>
```

Mobile Web API vs WebView

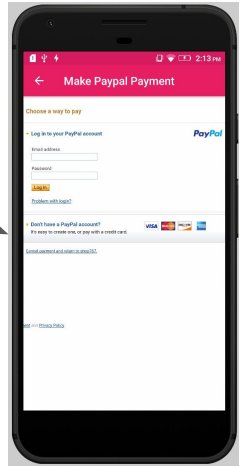
Mobile Web API

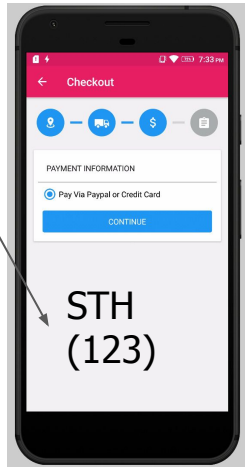
WebView

Mobile app sends a request to API:
https://mobile.sdh.p7z.pw/api/user/1

API responds back:
{"name": "STH", "user_id": 123}

Mobile app loads web page:
https://mobile.sdh.p7z.pw/page/payment



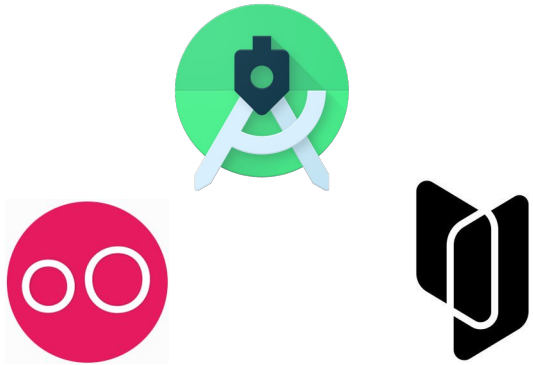


Content Overview

- Mobile App Security 101
  - แฮกโทรศัพท์
  - แฮกแอปในโทรศัพท์
    - Client vs Server
    - Android Component
- Android Testing Setup
- วิเคราะห์ Mobile App แบบ Static
  - วิเคราะห์ไฟล์ APK (Android)
  - mobsf
- วิเคราะห์ Mobile App แบบ Dynamic
  - Android Virtual Device (AVD)
  - Android Debugging Bridge (ADB)
  - Magisk + Burp Suite
- ลองทำแลป !
  - Lab 1: วิเคราะห์ Local Storage
  - Lab 2: วิเคราะห์ PIN API

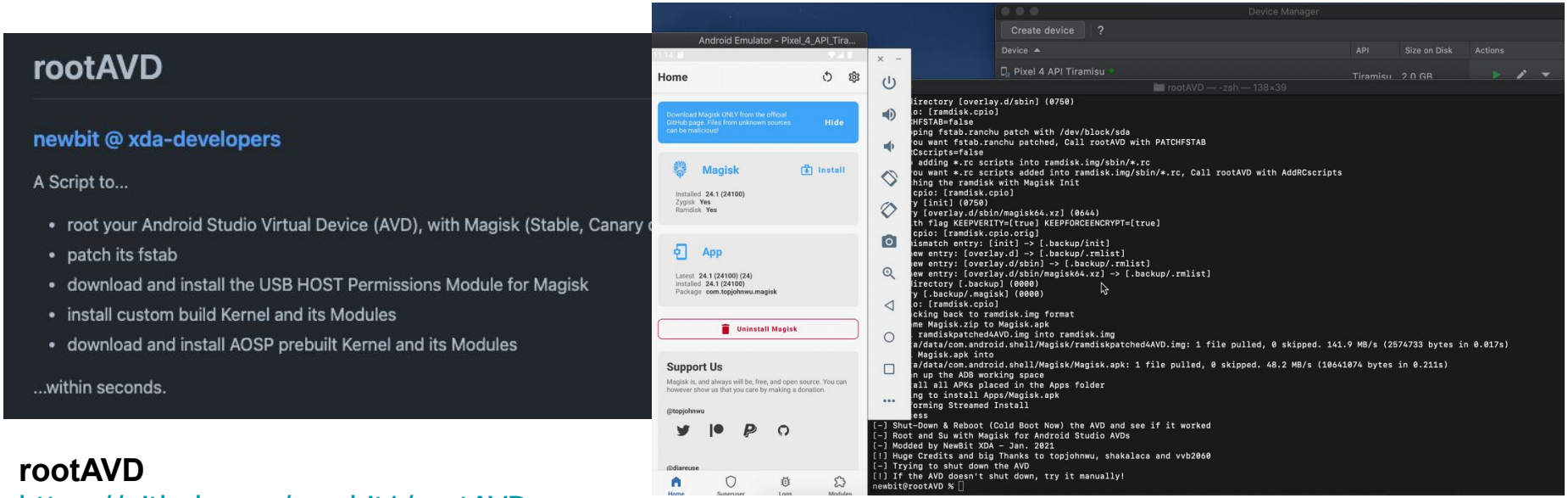
Android Testing Setup

Testing on a Emulator



Emulator
Free emulators: Android Virtual Device (AVD)
Commercial emulators: Genymotion, Corellium

Root Android ด้วย Magisk - Emulator



rootAVD
https://github.com/newbit1/rootAVD

ขั้นตอนการติดตั้ง:
https://github.com/newbit1/rootAVD#install-magisk
https://infosecwriteups.com/intercepting-android-emulator-ssl-traffic-with-burp-using-magisk-bc948dca68f9

Android Studio Installation

1 ดาวโหลดและติดตั้ง JDK

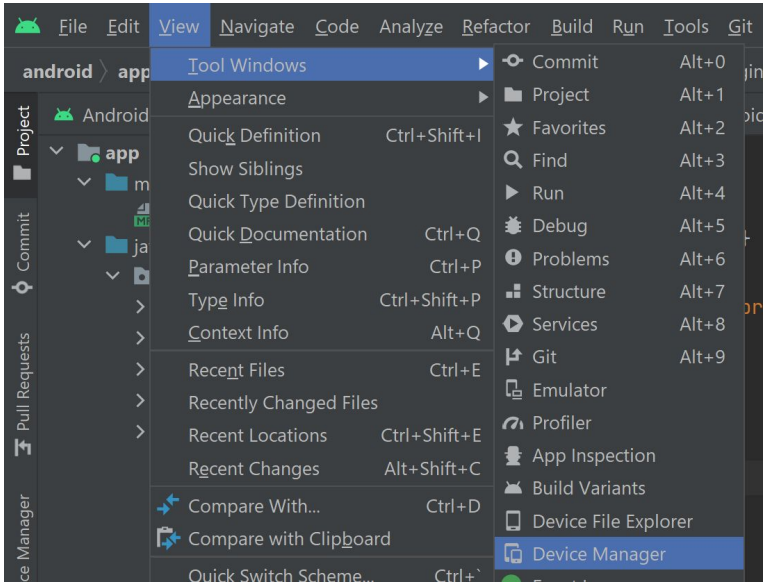
https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.exe

2 ดาวโหลดและติดตั้งโปรแกรม Android Studio

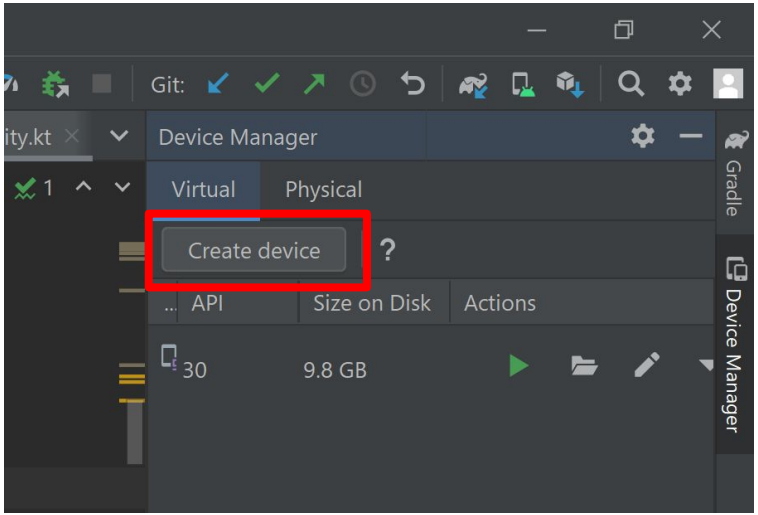
https://redirector.gvt1.com/edgedl/android/studio/install/2021.1.1.23/android-studio-2021.1.1.23-windows.exe

Android Studio - AVD Usage

1 เปิดโปรแกรม Android Studio เลือก View -> Tool Windows -> Device Manager

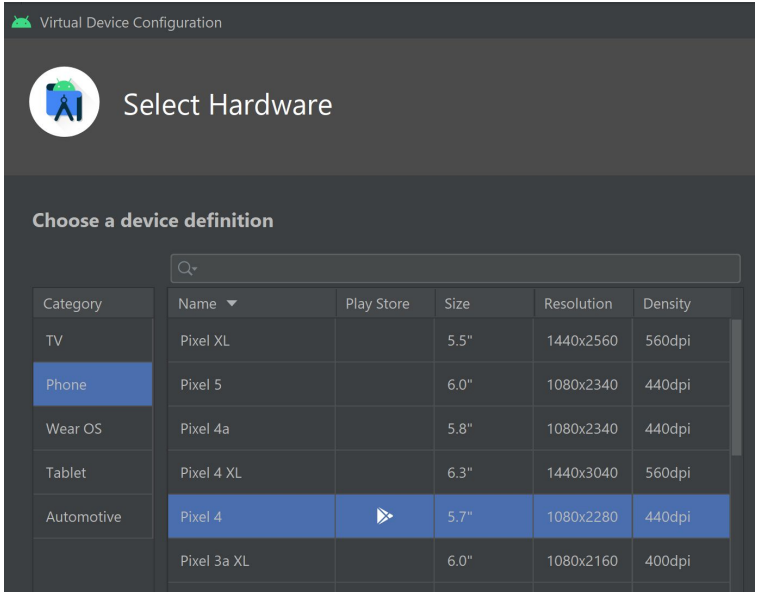


2 เลือก Create device

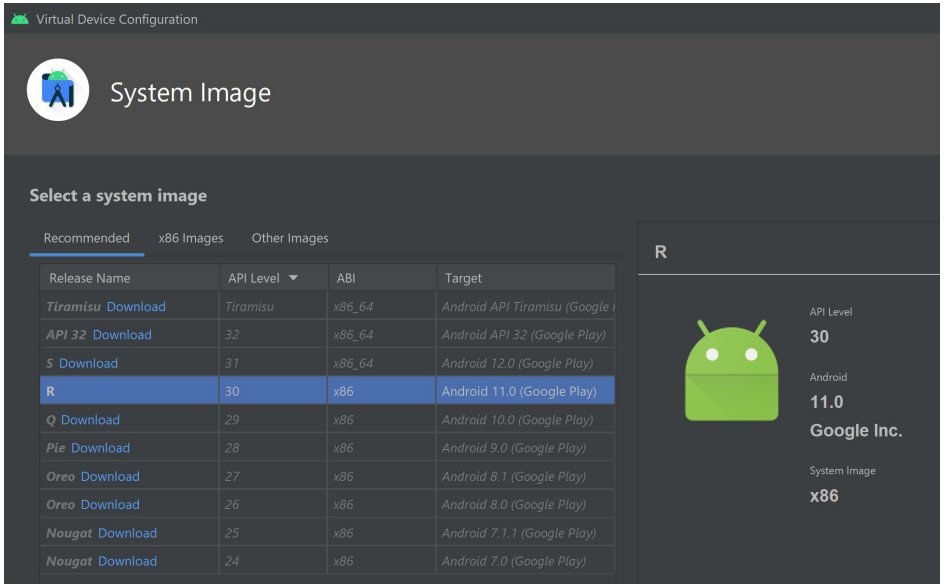


Android Studio - AVD Usage

3 เลือก Pixel 4 -> Next



4 เลือก API Level 30, ABI x86, Android 11 (กด Download ถ้ายังไม่เคยลง) -> Next -> Finish



Super User in AVD

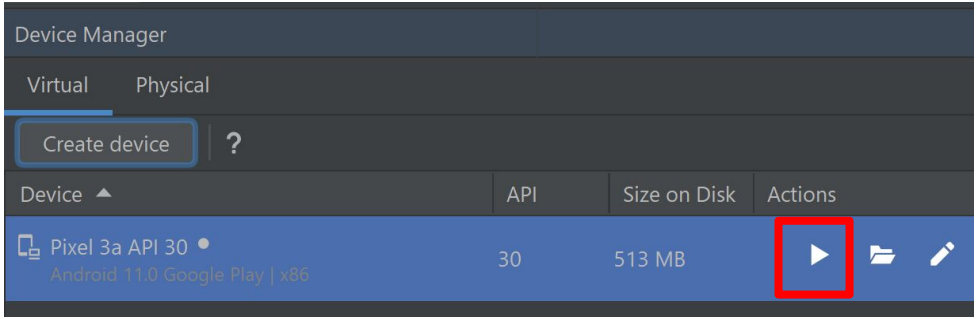
1 ดาวน์โหลดไฟล์ rootAVD

https://github.com/newbit1/rootAVD/archive/refs/heads/master.zip

2 เปิดโปรแกรม cmd และสั่ง cd ไปที่ folder rootAVD-master และใช้คำสั่ง

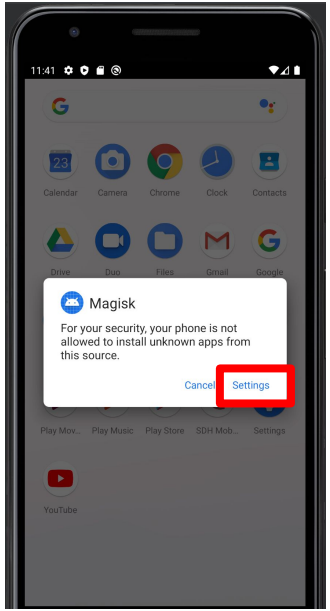
```
$ rootAVD.bat
%LOCALAPPDATA%\Android\Sdk\system-images\android-30\google_apis_playstore\x86\ramdisk.img
```

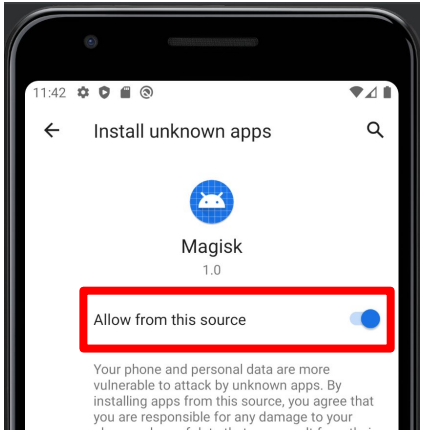
3 เปิด Emulator



Super User in AVD

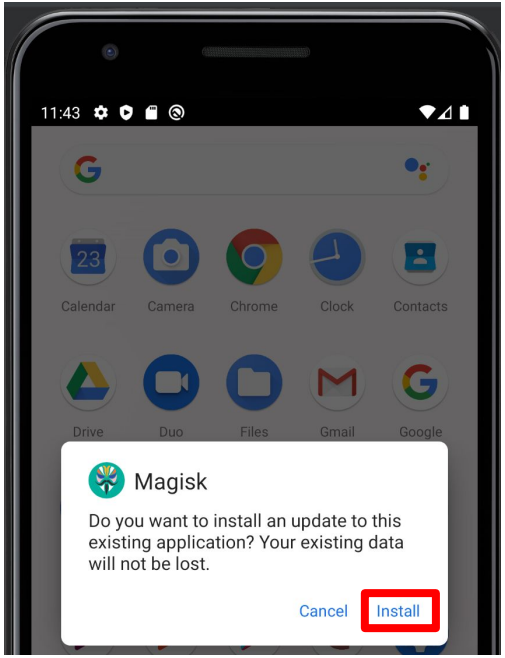
4 รอจน emulator ปิด กดเปิด emulatorใหม่จาก นั้นเปิดแอป Magisk -> ok -> Settings -> Allow from this source -> กด back



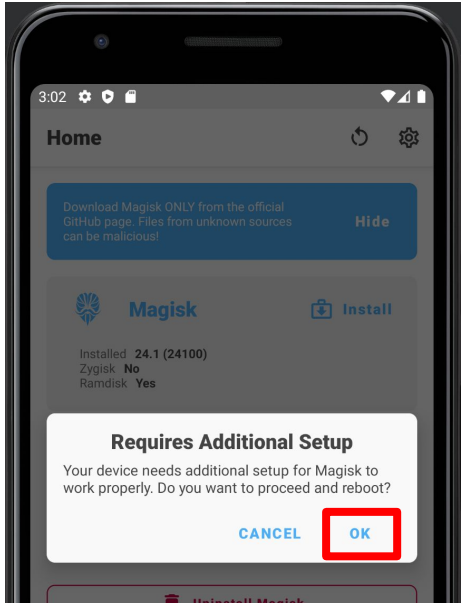


Super User in AVD

5 กด Install



6 เปิดแอป Magisk -> ok -> รอ reboot



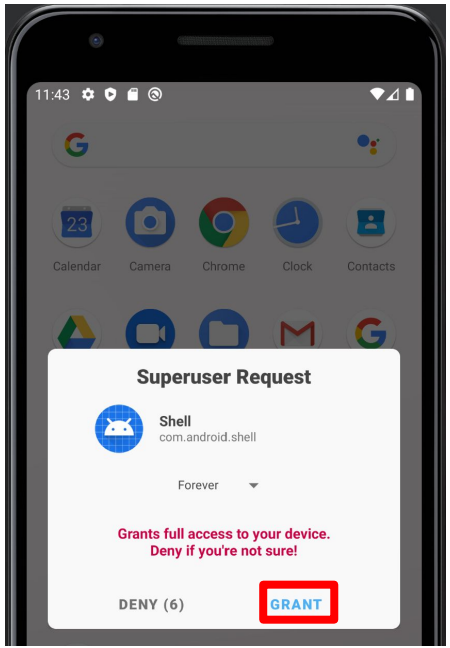
7 เปิดโปรแกรม cmd และใช้คำสั่งดังนี้

```
$ adb shell
$ su
```

```
C:\Windows\system32>adb shell
generic_x86_64:/ $ su
```

Super User in AVD

8 กด GRANT



9 ทดสอบคำสั่ง id จะได้ root

```
generic_x86_64:/ # id
uid=0(root) gid=0(root) groups=0(root) context=u:r:magisk:s0
generic_x86_64:/ #
```

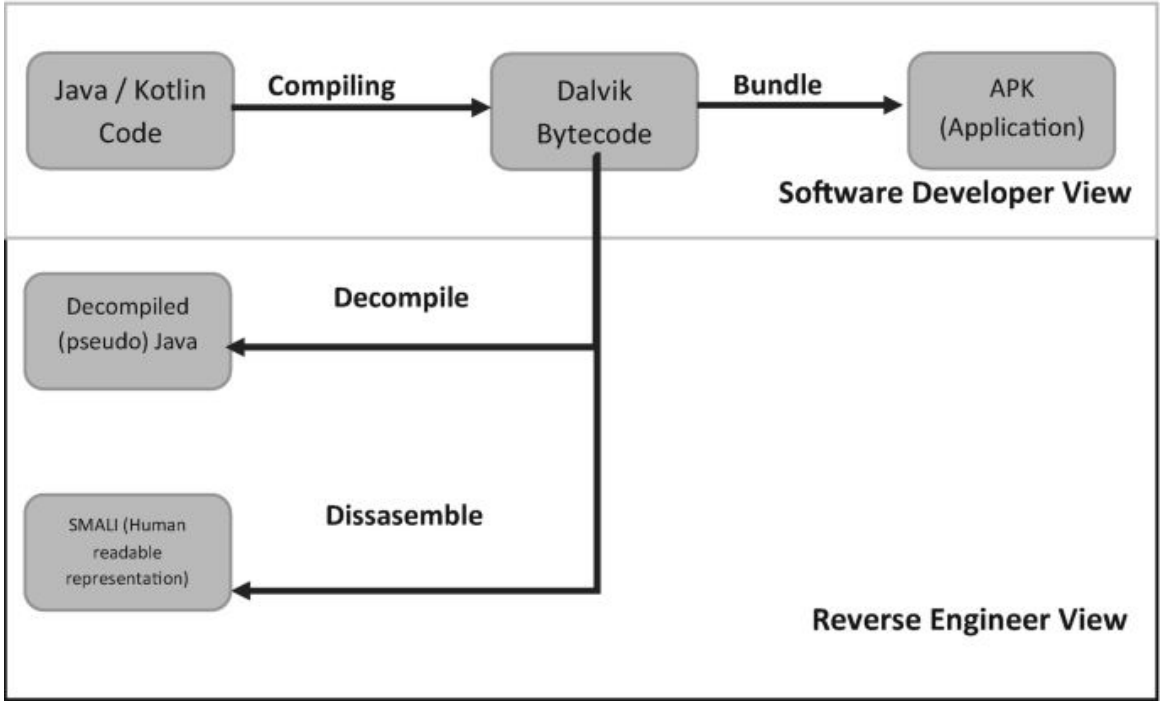
Static Analysis

Disassemble vs Decompile

Disassemble .apk to Smali code:
- apktool
- D2j-dex2smali
- apkstudio

Decompile .apk to Java code:
- jadx
- jadx-gui
- D2j-dex2jar

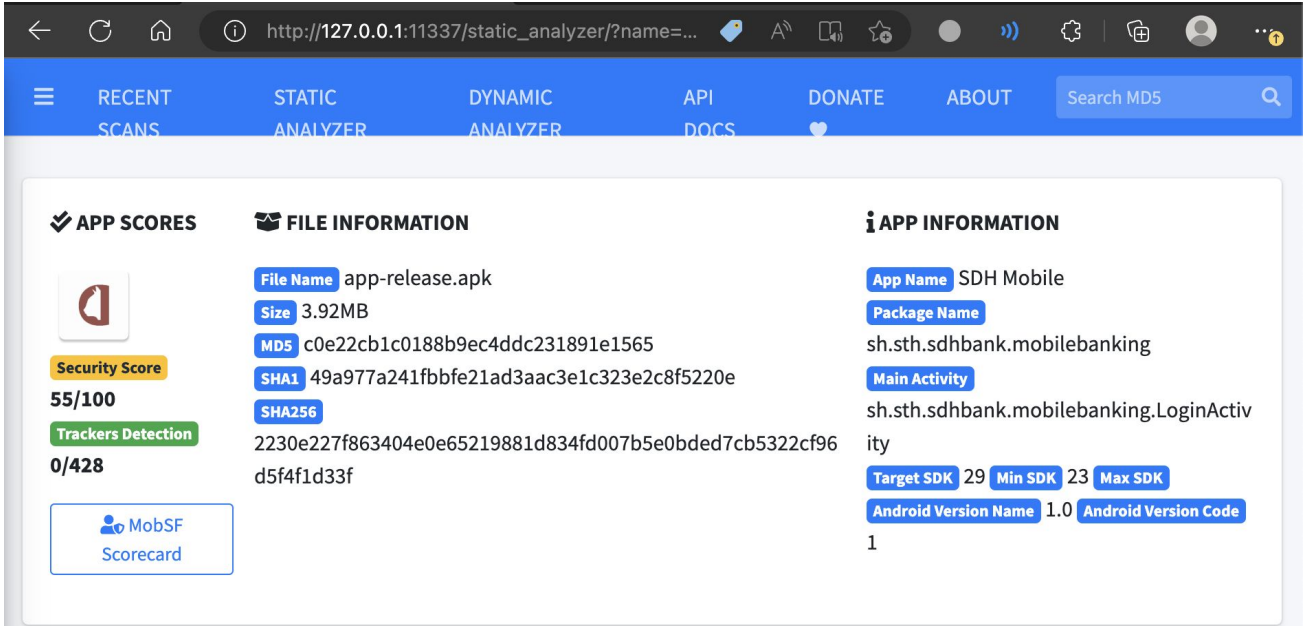
Etc.
- APKEditor



https://link.springer.com/chapter/10.1007/978-1-4842-6914-5_12

Mobile Security Framework (mobsf)

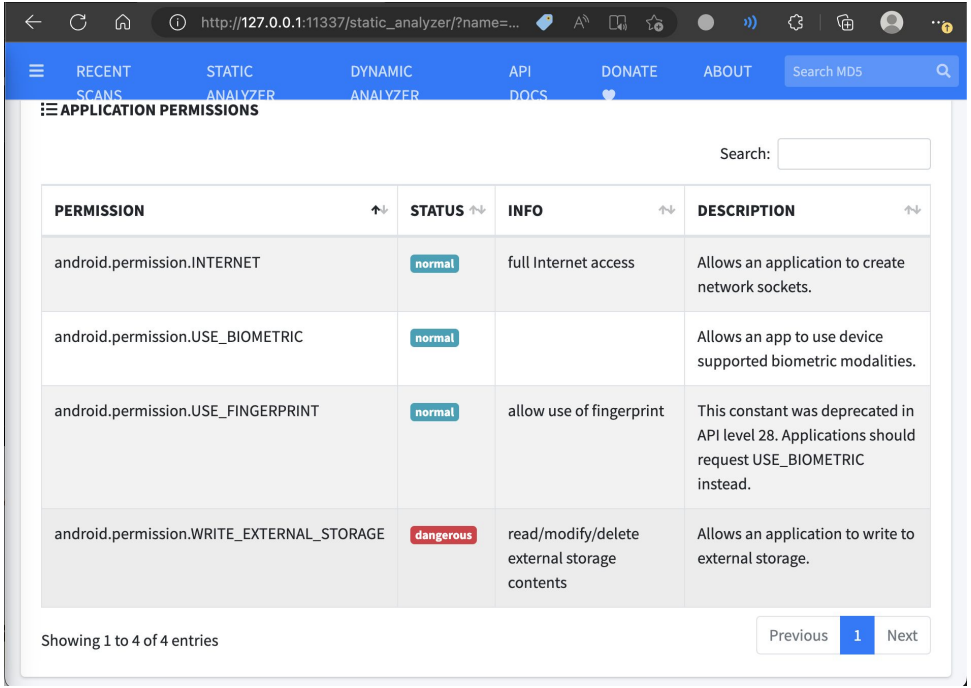
https://github.com/MobSF/Mobile-Security-Framework-MobSF
รายละเอียดแอปเบื้องต้น



Mobile Security Framework (mobsf)

ตัวอย่างการวิเคราะห์ผล mobsf #1 - การขอสิทธิ์ที่มีความเสี่ยงทางด้านความปลอดภัย

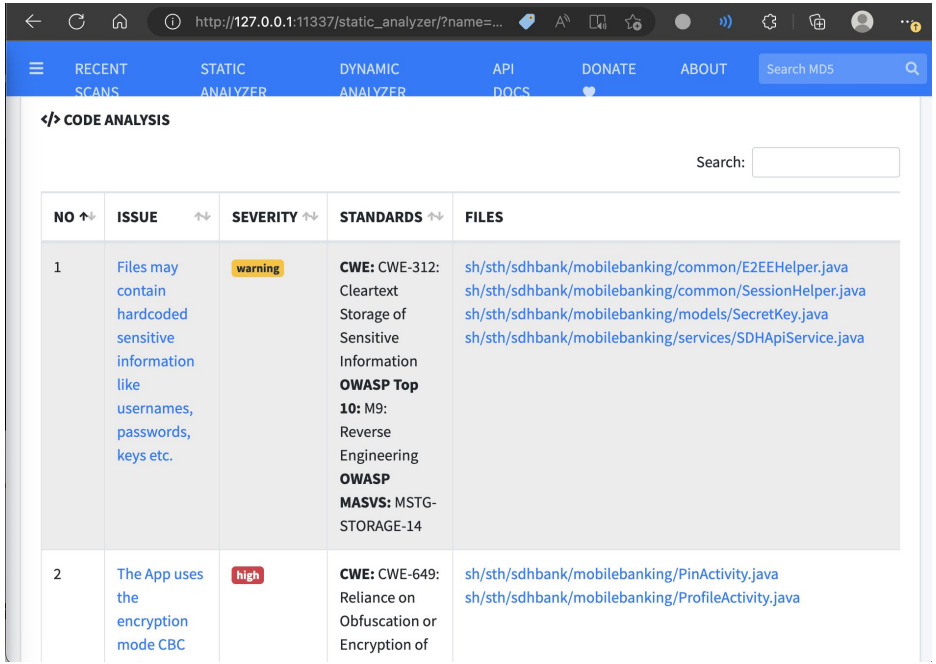
- มีการขอสิทธิ์ WRITE_EXTERNAL_STORAGE
- ใช้เขียนไฟล์ลง External Storage ได้
- มีความเสี่ยงเพราะ ถ้าแอปอื่นบนเครื่องโทรศัพท์มีสิทธิ์ อ่าน External Storage อาจถูกขโมยไฟล์ได้

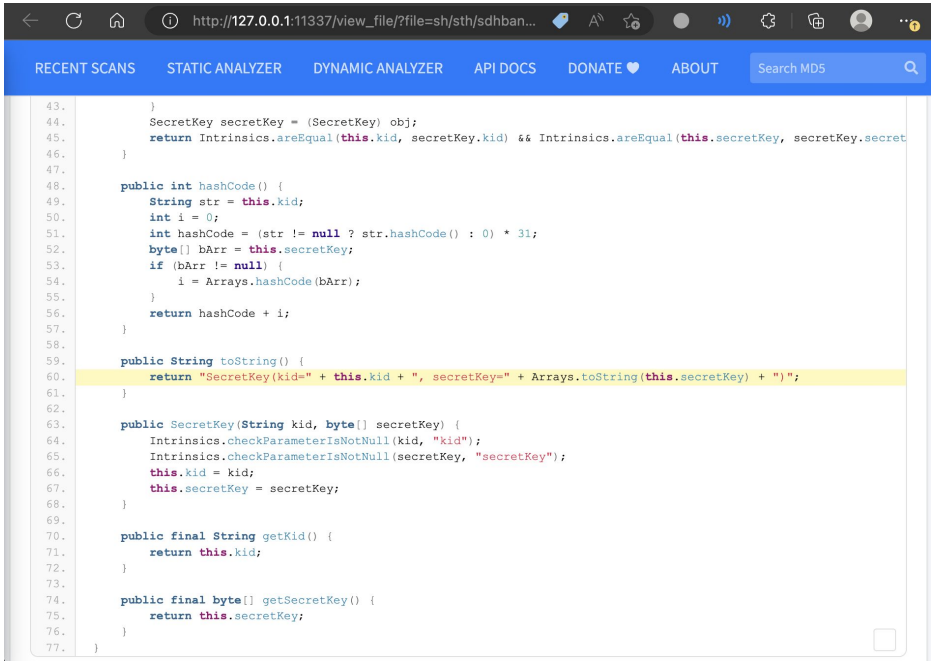


Mobile Security Framework (mobsf)

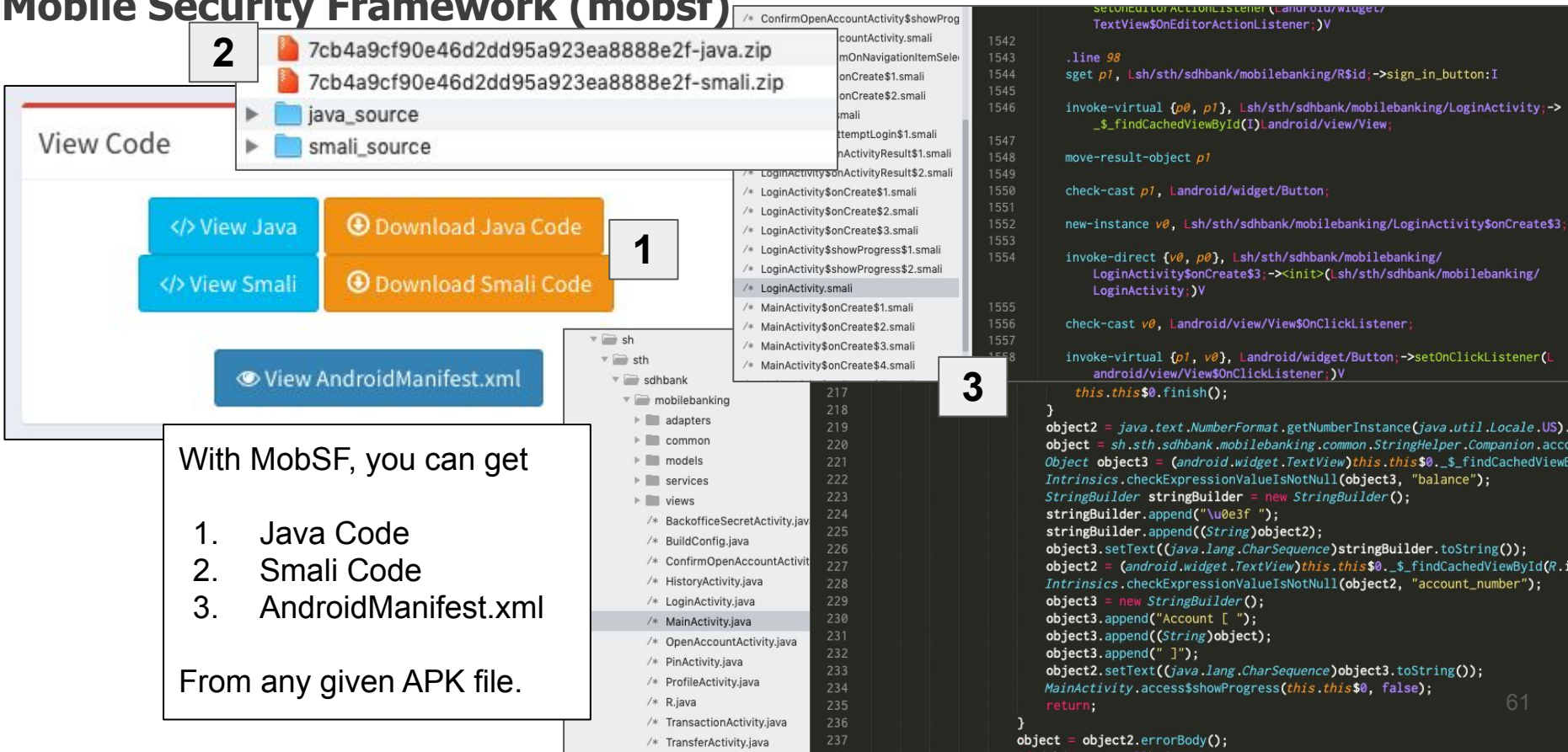
ตัวอย่างการวิเคราะห์ผล mobsf #2 - ตรวจสอบโค้ดที่มีความเสี่ยง

จากตัวอย่าง มีการพบการใช้งานการเข้ารหัสในรูปแบบ CBC

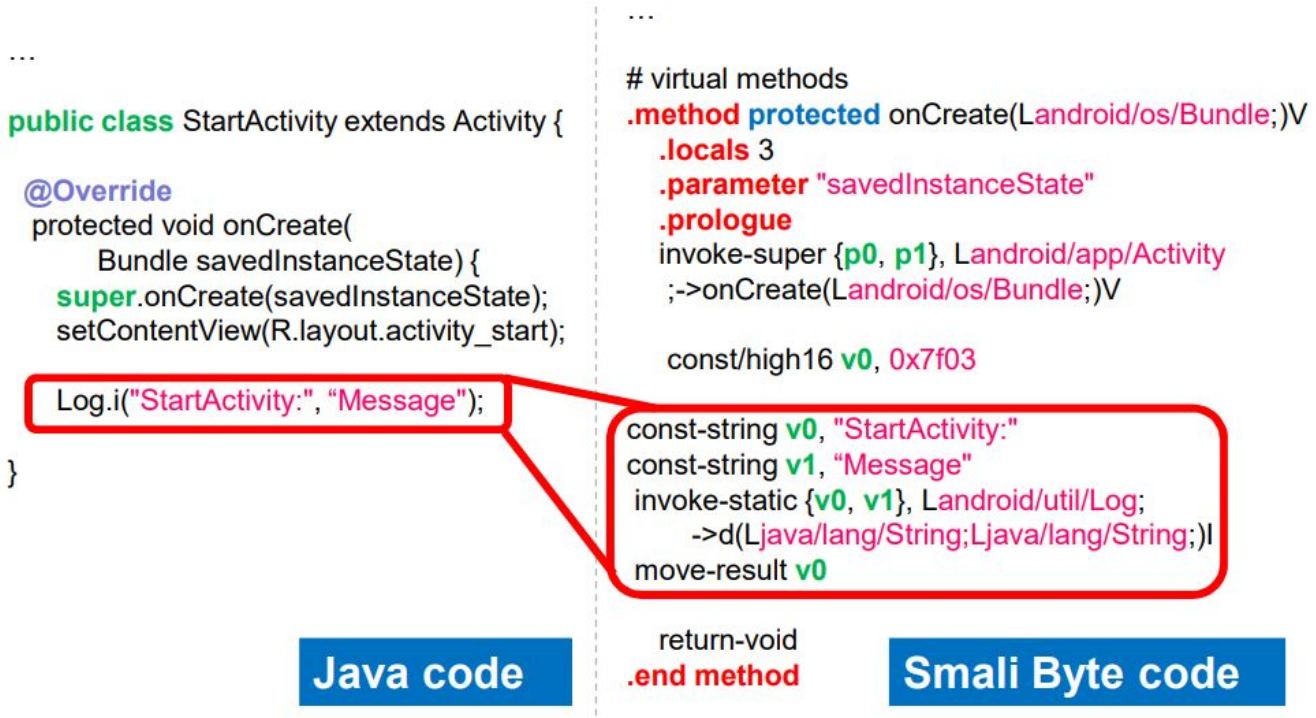




Mobile Security Framework (mobsf)



Smali


…
public class StartActivity extends Activity {
@Override
protected void onCreate(
Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_start);
Log.i("StartActivity:", “Message");
}
Java code
…
virtual methods
.method protected onCreate(Landroid/os/Bundle;)V
.locals 3
.parameter "savedInstanceState"
.prologue
invoke-super {p0, p1}, Landroid/app/Activity
;->onCreate(Landroid/os/Bundle;)V
const/high16 v0, 0x7f03
const-string v0, "StartActivity:"
const-string v1, “Message"
invoke-static {v0, v1}, Landroid/util/Log;
->d(Ljava/lang/String;Ljava/lang/String;)I
move-result v0
return-void
.end method
Smali Byte code

Smali

```
.method private doSomething()V
```

V	void
Z	boolean
B	byte
S	short
C	char
F	float
I	int
J	long
D	double
[	array

Smali - Example Bypass Root Detection

```
if-nez v0, :cond_1

invoke-virtual {p0}, Lcom/scottyab/rootbeer/b;->f()Z

move-result v0

if-nez v0, :cond_1

invoke-virtual {p0}, Lcom/scottyab/rootbeer/b;->d()Z

move-result v0

if-eqz v0, :cond_0

goto :goto_0

:cond_0
const/4 v0, 0x0  // false

goto :goto_1

:cond_1
:goto_0
const/4 v0, 0x1
```

0x0

ตัวอย่าง Bypass Root Detection ด้วย Smali อย่างง่าย

```
42      public boolean isRooted() {
43
44          return detectRootManagementApps() || detectPotentiallyDangerousApps() || checkForBinary(BINARY_SU)
45                  || checkForDangerousProps() || checkForRWPaths()
46                  || detectTestKeys() || checkSuExists() || checkForRootNative() || checkForMagiskBinary();
47      }
```

Code Obfuscation - BEFORE vs AFTER

BEFORE

SDHApiService.java — mobsf_files

SDHApiService.java

```
public final SDHApiService
    Object object = new OkHttpClient.Builder();
    if (BUILD_TYPE.equals(BUILD_TYPE)) {
        object.certificatePinner(new CertificatePinner.Builder().add("
            sdhbank.p7z.pw", "
            sha256/v6GDcb6WbRBSUpUzM3I04Pc5k4zHiukcvyX9scPdhUc=").build
            ());
    }
    object = object.addInterceptor(new Interceptor(){

        @NotNull
        public okhttp3.Response intercept(@NotNull okhttp3.
            Interceptor$Chain object) {
            Intrinsics.checkParameterIsNotNull(object, "chain");
            Object object2 = sh.sth.sdhbank.mobilebanking.common.
                StringHelper.Companion.getRandomString(8);
            java.lang.CharSequence charSequence = new java.lang.
                StringBuilder();
            charSequence.append("SDHtest!");
            charSequence.append((String)object2);
            charSequence = charSequence.toString();
            Object object3 = object.request();
            if (object3.body() != null) {
                Object object4 = Factory.INSTANCE;
                Intrinsics.checkExpressionValueIsNotNull(object3, "
                    originalRequest");
                object4 = Factory.access$bodyToString((Factory)object4,
                    (Request)object3);
                java.lang.StringBuilder stringBuilder = new java.lang.
                    StringBuilder();
                stringBuilder.append("Request Body Encrypted: ");
```

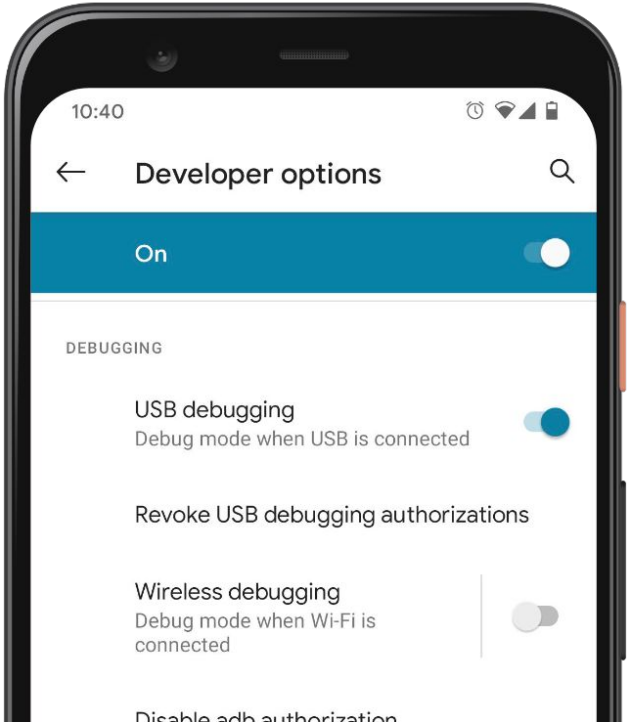
AFTER

b.java — mobsf_proguard_files

```
public final b a() {
    int n2;
    Object object2;
    Object object;
    Object object3 = new F.a();
    if (b.equals(b)) {
        object2 = new ArrayList();
        object = new String[]{"
            sha256/v6GDcb6WbRBSUpUzM3I04Pc5k4zHiukcvyX9scPdhUc="};
        int n3 = ((String[])object).length;
        for (n2 = 0; n2 < n3; ++n2) {
            object2.add(new h.a("sdhbank.p7z.pw", object[n2]));
        }
        object3.p = new e.h(new LinkedHashSet<h.a>((Collection<h.a>)
            object2), null);
    }
    object2 = new h.a.a.a.d.a();
    object3.e.add((B)object2);
    object3 = new F((F.a)object3);
    object2 = g.B.a;
    object = new ArrayList();
    ArrayList object42 = new ArrayList();
    Object object4 = new g.a.a.a(new q(r.a, c.d.a.j.a, Collections.<Type,
        s<?>>emptyMap(), false, false, false, true, false, false, false,
        c.d.a.H.a, null, 2, 2, Collections.<K>emptyList(), Collections.<
        K>emptyList(), Collections.<K>emptyList()));
```

Dynamic Analysis

เปิด Android Debug Bridge (adb)



adb

adb devices
adb shell pm path package_name
adb pull <remote> [<locaDestination>]
jadx -d [path-output-folder] [path-apk-or-dex-file]

```
C:\Users\[redacted]\Mobile\pull>adb pull /data/app/com.shortstack.hackertracker-1/base.apk
/data/app/com.shortstack.hackertracker-1/base.apk: 1 file pulled. 4.8 MB/s (2987176 bytes in 0.596s)

C:\Users\[redacted]\Mobile\pull>dir
 Volume in drive C has no label.
 Volume Serial Number is 1058-651E

 Directory of C:\[redacted]\Mobile\pull

10/28/2019  03:12 PM    <DIR>          .
10/28/2019  03:12 PM    <DIR>          ..
10/28/2019  03:12 PM         2,987,176 base.apk
               1 File(s)      2,987,176 bytes
               2 Dir(s)  217,974,800,384 bytes free
```

รวมคำสั่ง adb ที่ใช้งานบ่อย

```
$ adb shell
bullhead:/ $ su
bullhead:/ # id
uid=0(root) gid=0(root) groups=0(root) context=u:r:su:s0
```

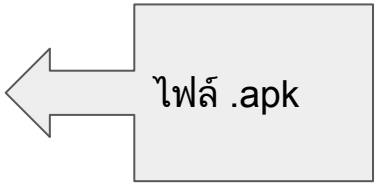
```
// รีสตาร์ท Device
$ adb shell reboot

// โยนไฟล์ไปเครื่อง Android
$ adb push <computer.file.path> /sdcard/Download/

// ดึงไฟล์จากเครื่อง Android
$ adb pull /sdcard/Download/<filename>

// เช็คว่ามีเครื่อง Android ต่ออยู่
$ adb devices

// เรียกดูรายชื่อ packages ของแอปพลิเคชัน
$ adb shell pm list packages -f
```

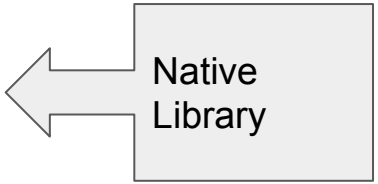
```
// รีเซ็ตการตั้งค่า Proxy
$ adb shell settings put global http_proxy :0

// ตั้งค่า Proxy
$ adb shell settings put global http_proxy 127.0.0.1:8080

// เพิ่มการตั้งค่า reverse
route จาก port 8080 ของมือถือ ไปยัง port 8080 ของคอมพิวเตอร์ที่เปิด Burp อยู่
$ adb reverse tcp:8080 tcp:8080

// ลบการตั้งค่า reverse
$ adb reverse --remove-all
```

/data/app/[RANDOM]/[App Package Name][RANDOM]

ที่เก็บไฟล์ APK

```
.:
total 40M
drwxrwxr-x 4 system system  4.0K 2022-04-24 01:53 .
drwxrwxr-x 3 system system  4.0K 2022-04-24 01:53 ..
-rw-r--r-- 1 system system   48M 2022-04-24 01:52 base.apk
drwxr-xr-x 3 system system  4.0K 2022-04-24 01:53 lib
drwxrwx--x 3 system install 4.0K 2022-04-24 01:53 oat
-rw-r--r-- 1 system system   17M 2022-04-24 01:53 split_config.armeabi_v7a.apk
-rw-r--r-- 1 system system   44K 2022-04-24 01:53 split_config.en.apk
-rw-r--r-- 1 system system   15M 2022-04-24 01:53 split_config.xxhdpi.apk

./lib:
total 12K
drwxr-xr-x 3 system system 4.0K 2022-04-24 01:53 .
drwxrwxr-x 4 system system 4.0K 2022-04-24 01:53 ..
drwxr-xr-x 2 system system 4.0K 2022-04-24 01:53 arm

./lib/arm:
total 17M
drwxr-xr-x 2 system system 4.0K 2022-04-24 01:53 .
drwxr-xr-x 3 system system 4.0K 2022-04-24 01:53 ..
-rwxr-xr-x 1 system system 662K 1981-01-01 01:01 libc++_shared.so
-rwxr-xr-x 1 system system  26K 1981-01-01 01:01 libchecks.so
-rwxr-xr-x 1 system system  18K 1981-01-01 01:01 libcustomaudio.so
-rwxr-xr-x 1 system system 341K 1981-01-01 01:01 libdexprotector.so
-rwxr-xr-x 1 system system 369K 1981-01-01 01:01 libdexprotector_h.so
-rwxr-xr-x 1 system system 865K 1981-01-01 01:01 libiconv.so
```





/data/data/[App Package Name]

ที่เก็บไฟล์ ข้อมูลแต่ละแอป

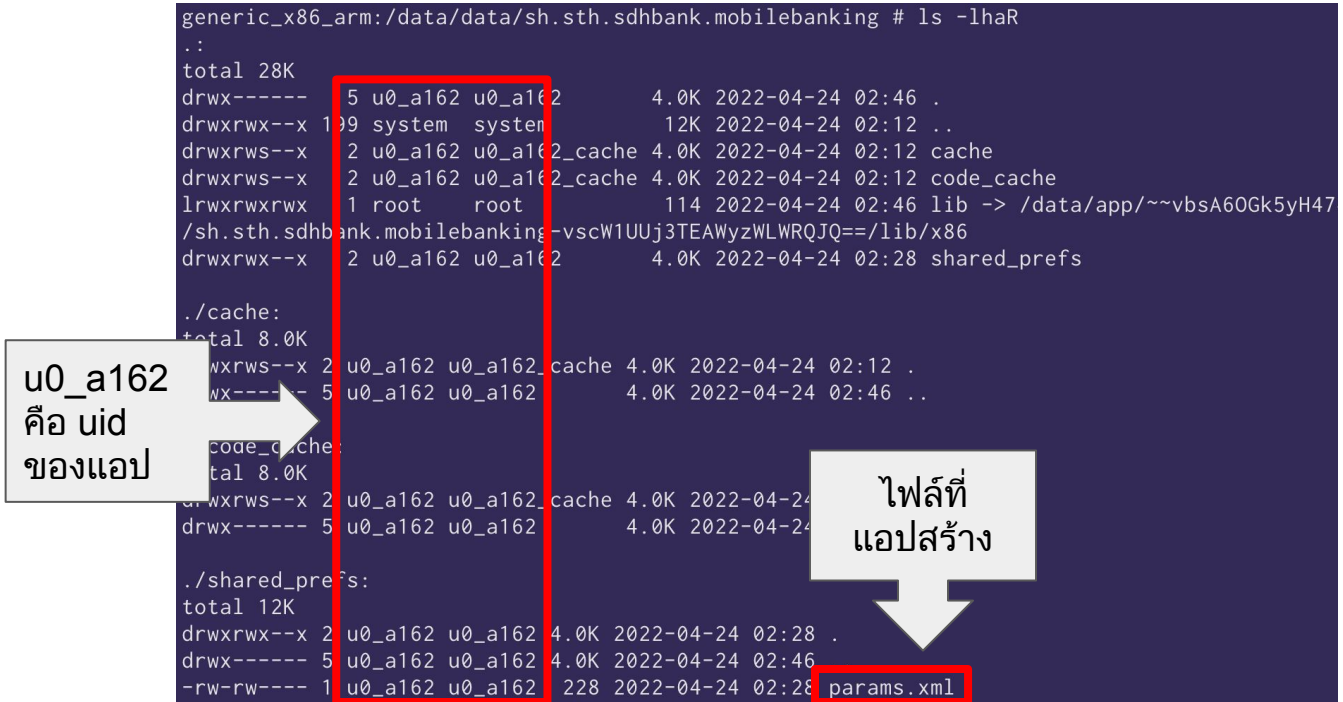
```
generic_x86_arm:/ # ls /data/data/
android
android.auto_generated_rro_product__
com.android.backupconfirm
com.android.bips
com.android.bips.auto_generated_rro_product__
drwx------   9 u0_a118       u0_a118       4.0K 2022-04-24 02:46 com.google.android.videos
drwx------   7 u0_a113       u0_a113       4.0K 2022-04-24 02:46 com.google.android.webview
drwx---      9 u0_a148       u0_a148       4.0K 2022-04-24 01:38 com.google.android.wifi.resources
drwx---      9 u0_a123       u0_a123       4.0K 2022-04-24 02:45 com.google.android.youtube
drwx---      9 u0_a122       u0_a122       4.0K 2022-04-24 01:38 com.google.mainline.telemetry
drwx---      9 u0_a155       u0_a155       4.0K 2022-04-24 01:47 com.joeykrim.rootcheck
drwx------   4 u0_a156       u0_a156       4.0K 2022-04-24 02:46 com.kasikorn.retail.mbanking.wap
drwx------   4 u0_a159       u0_a159       4.0K 2022-04-24 02:46 com.krungsri.uchoose
drwx------   4 u0_a158       u0_a158       4.0K 2022-04-24 02:46 com.scb.phone
drwx------   4 u0_a161       u0_a161       4.0K 2022-04-24 02:46 com.wireguard.android
drwx------   4 u0_a154       u0_a154       4.0K 2022-04-24 02:46 io.github.vvb2060.magisk
drwx------   4 u0_a112       u0_a112       4.0K 2022-04-24 01:38 org.chromium.webview_shell
drwx------   5 u0_a162       u0_a162       4.0K 2022-04-24 02:46 sh.sth.sdhbank.mobilebanking
```

ที่เก็บแอปบน Android

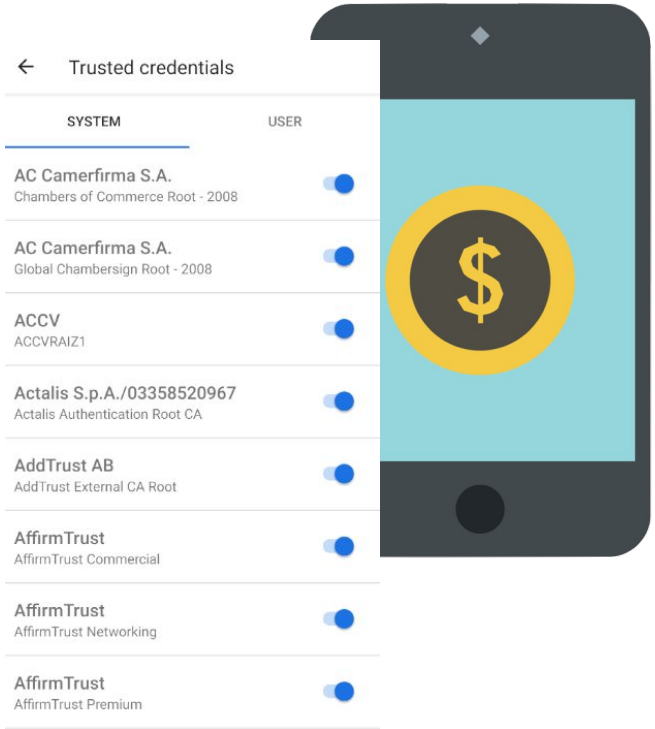
สิทธิ์ของแต่ละแอป (uid)

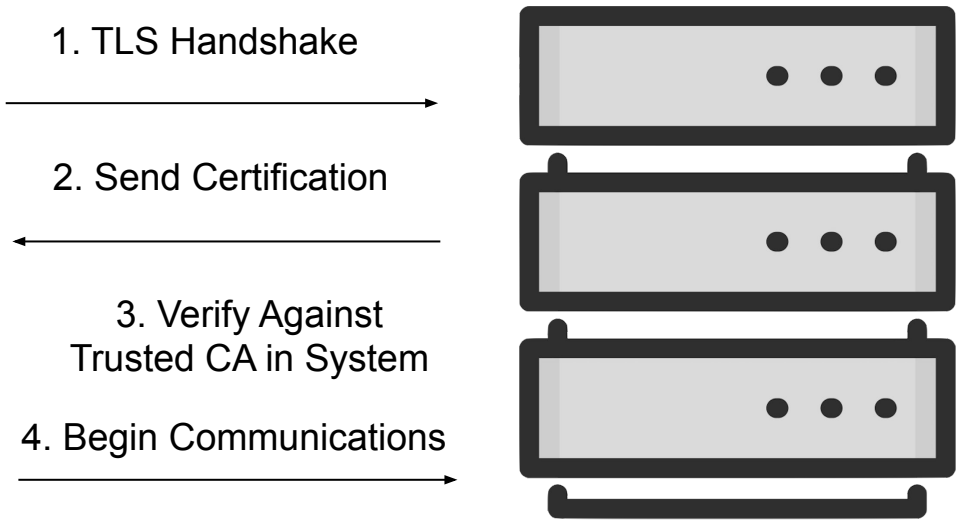
/data/data/[App Package Name]

/data/data/sh.sth.sdhbank.mobilebanking/

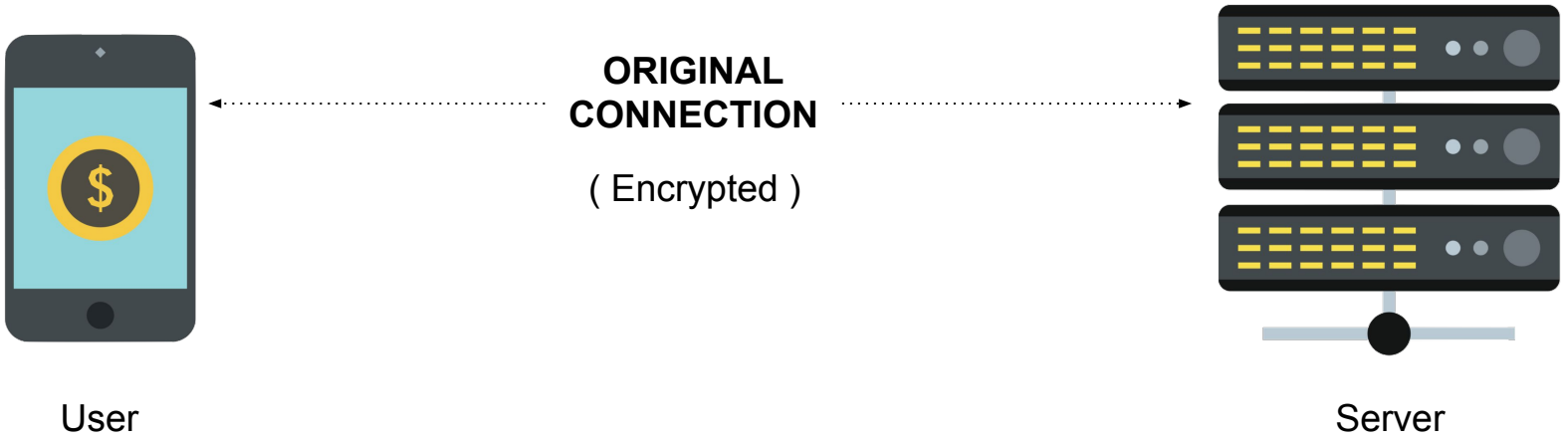


Standard HTTPS (without Burp Suite)

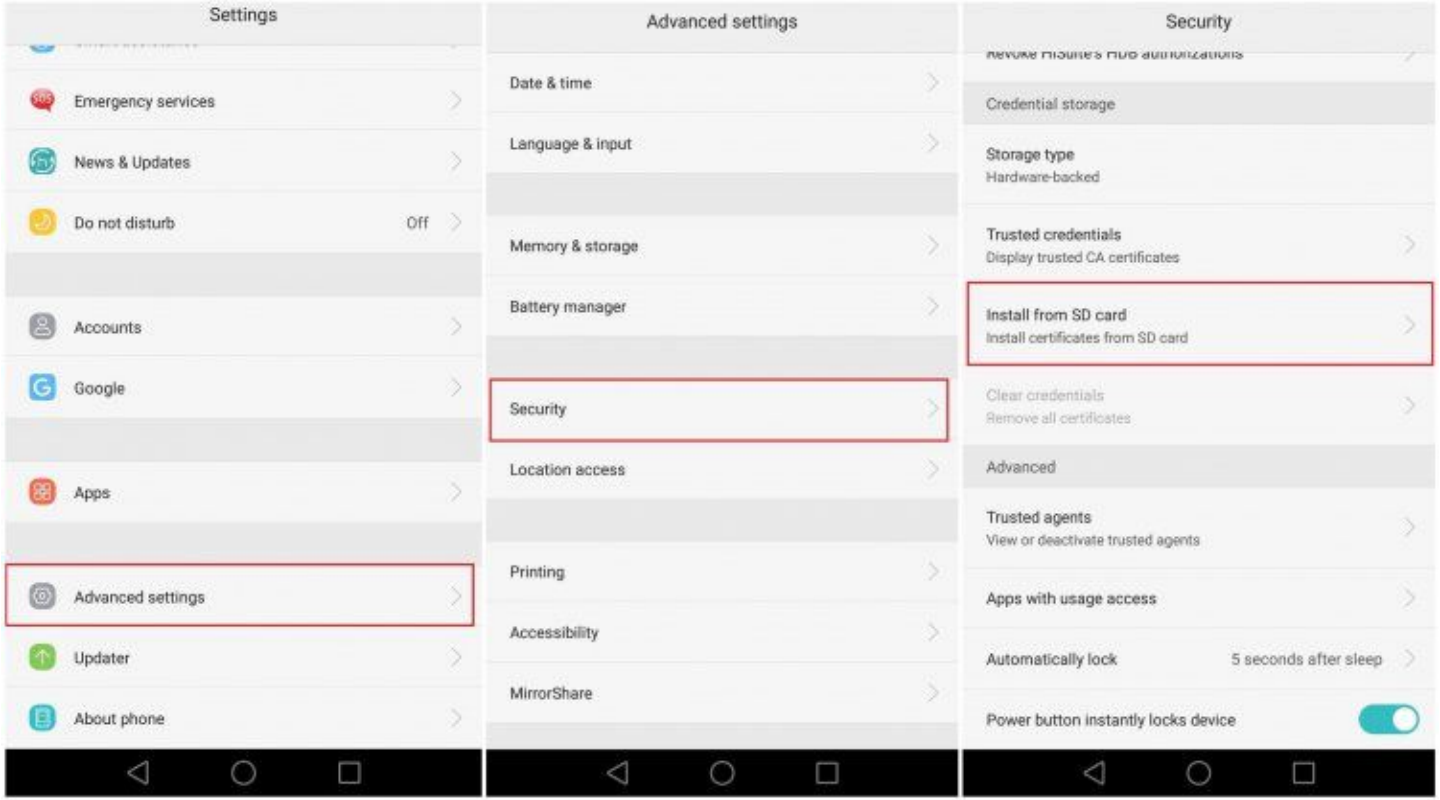




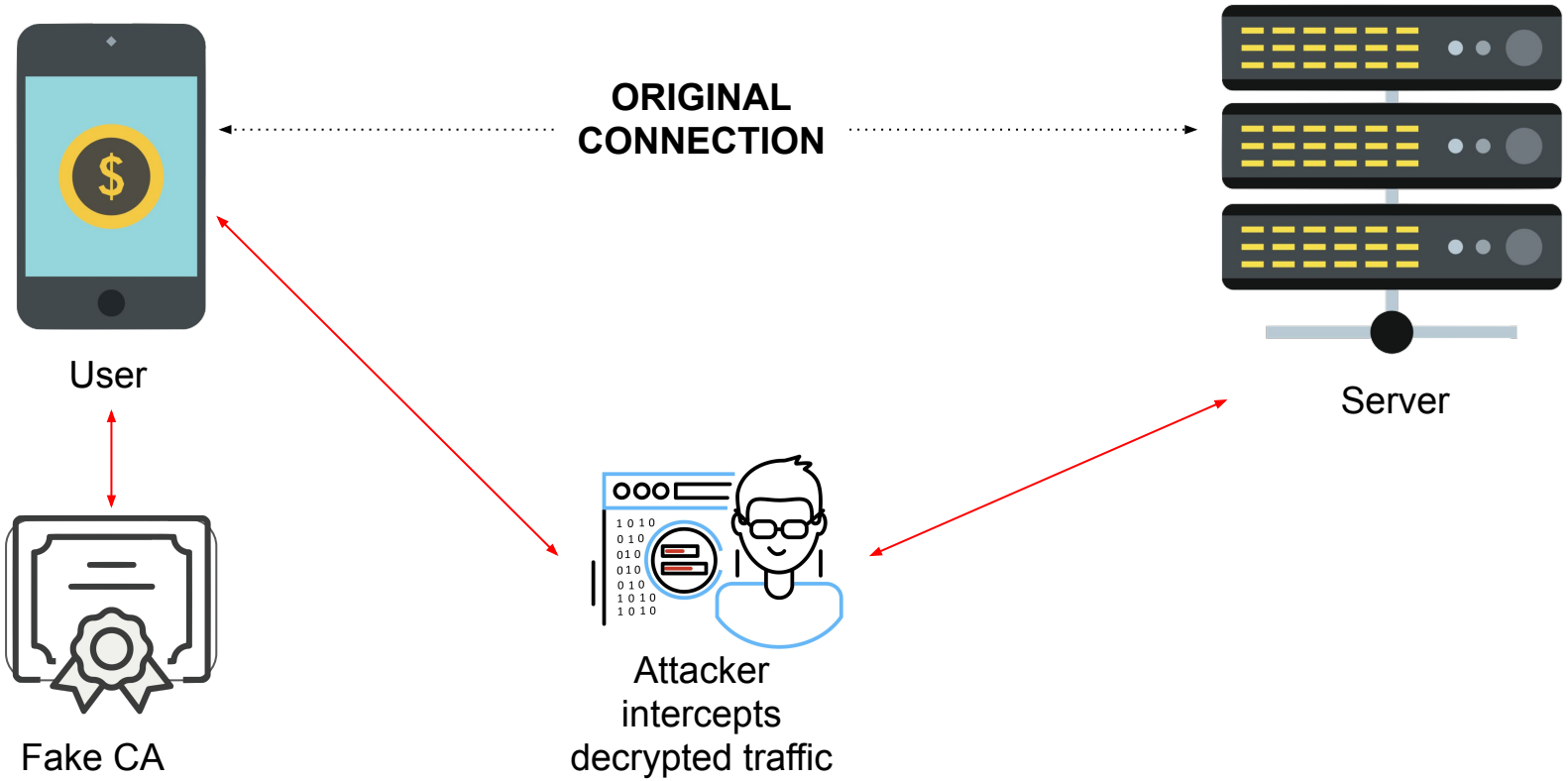
Standard HTTPS (without Burp Suite)



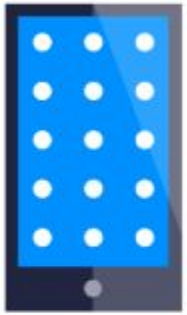
Standard HTTPS (with Burp Suite)



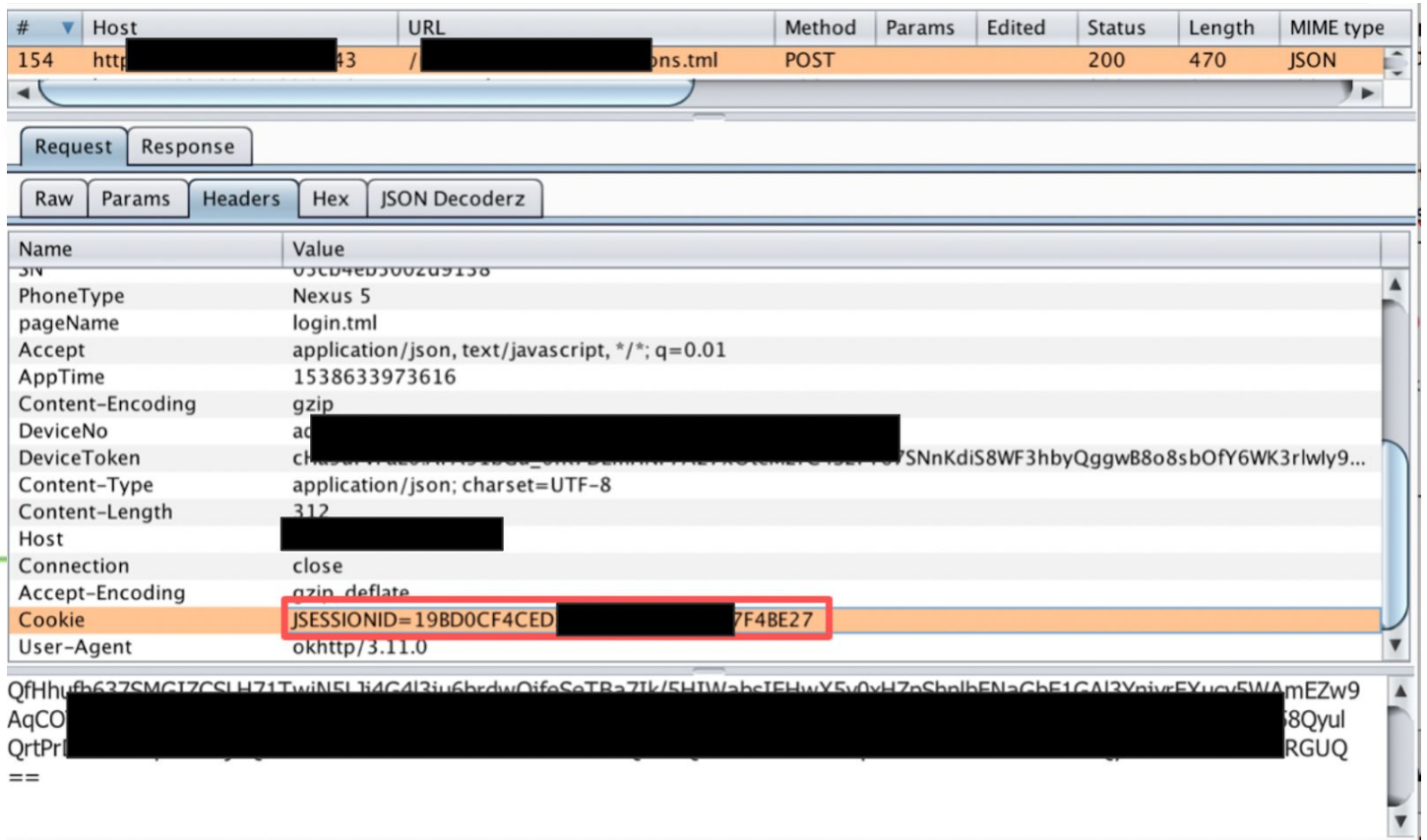
Standard HTTPS (with Burp Suite)



Standard HTTPS (with Burp Suite)



User



ลองทำแลป !

- Lab 1: วิเคราะห์ Local Storage
- Lab 2: วิเคราะห์ PIN API

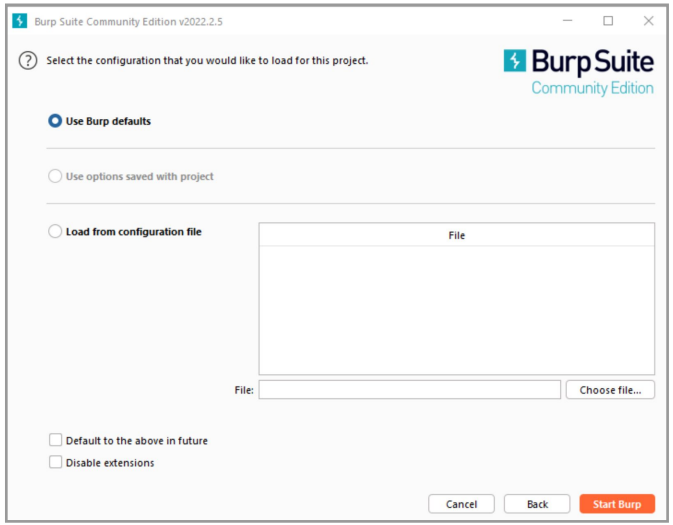
Thank you !!

Appendix

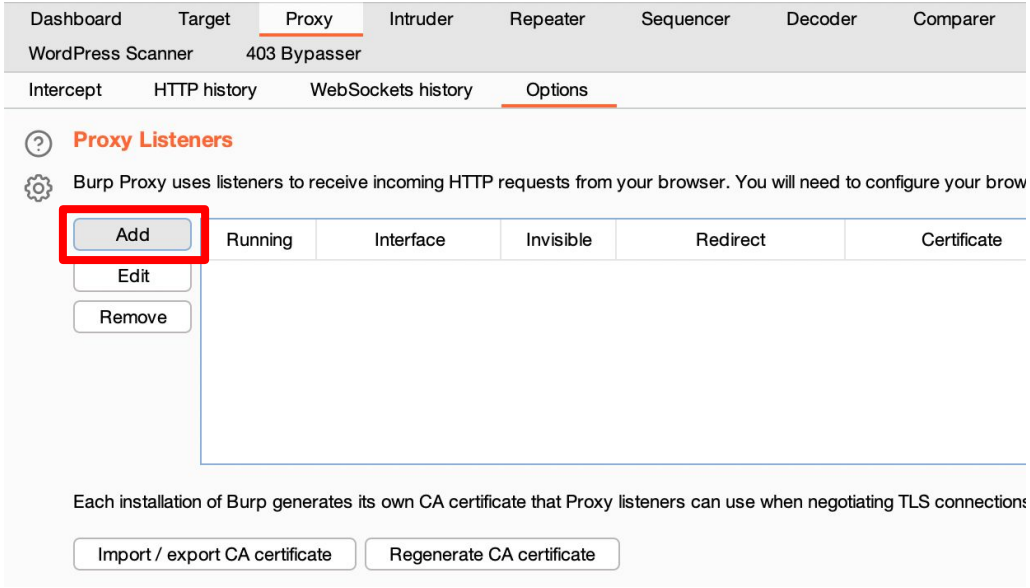
การติดตั้ง Burp Suite และ Root CA

1 ดาวน์โหลดและติดตั้ง Burp Suite Community

https://portswigger-cdn.net/burp/releases/download?product=community&version=2022.2.5&type=WindowsX64



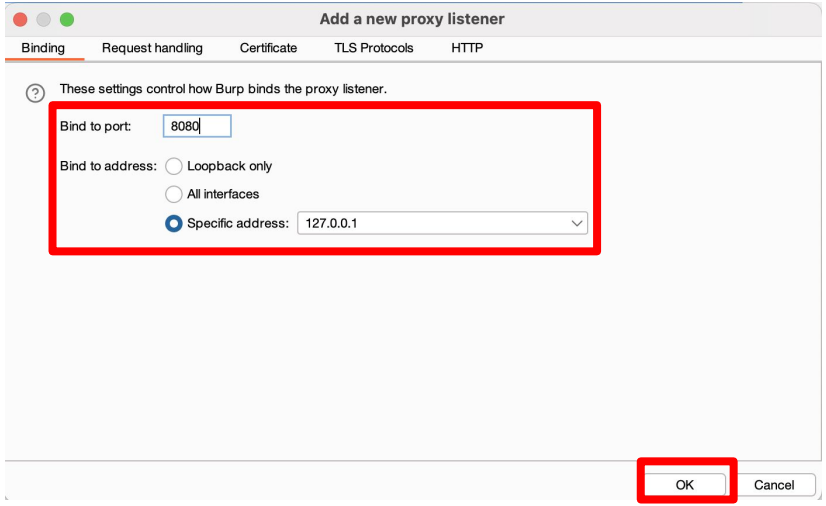
2 ตั้ง Proxy Listener โดยไปที่หน้า Proxy แล้วกด Add



การติดตั้ง Burp Suite และ Root CA

3 ตั้งค่า Proxy ตามรูปแล้วกด OK

https://portswigger-cdn.net/burp/releases/download?product=community&version=2022.2.5&type=WindowsX64

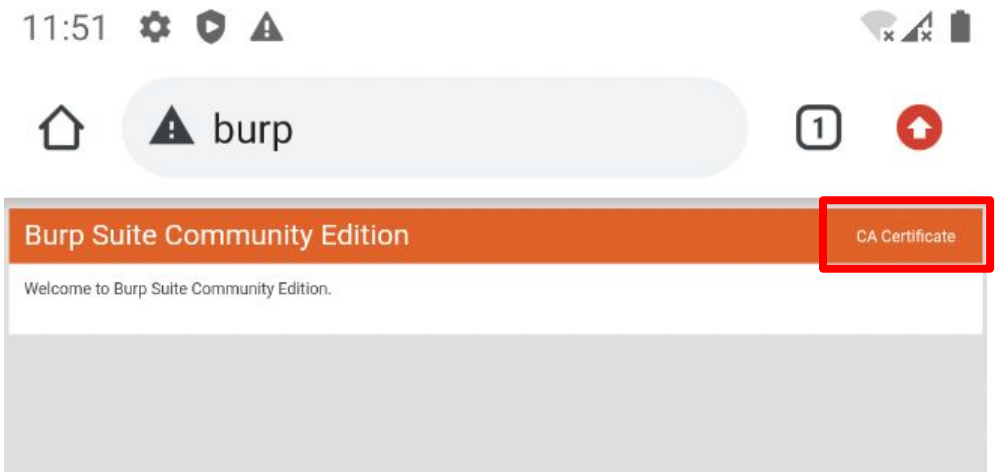


4 เปิด CMD และใช้คำสั่งต่อไปนี้

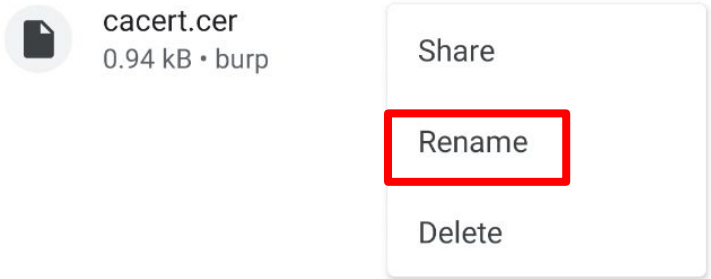
```
$ adb reverse tcp:8080 tcp:8080
$ adb shell settings put global http_proxy 127.0.0.1:8080
```

การติดตั้ง Burp Suite และ Root CA

5 เปิด Android Emulator แล้วเข้าไปที่เว็บ http://burp
จากนั้นกด CA Certificate

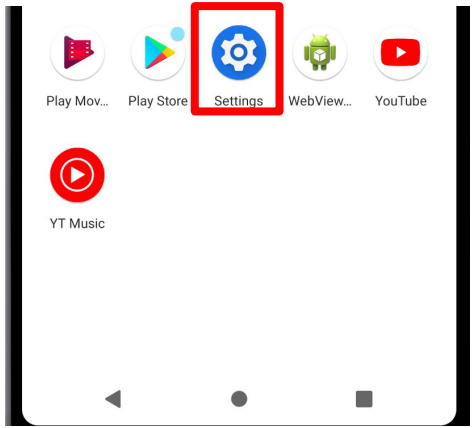


6 Rename ไฟล์จาก .der เป็น .cer

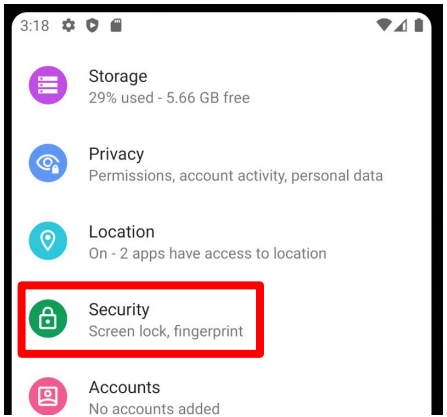


การติดตั้ง Burp Suite และ Root CA

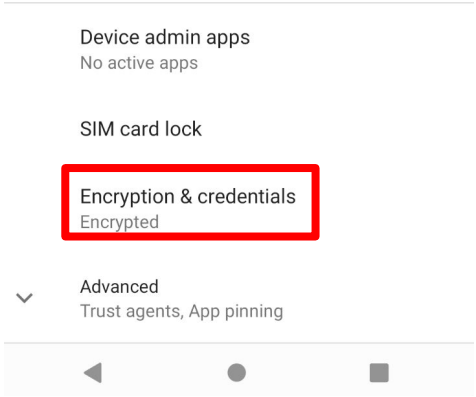
7 เปิด Settings ใน Emulator



8 เลือก 'Security'

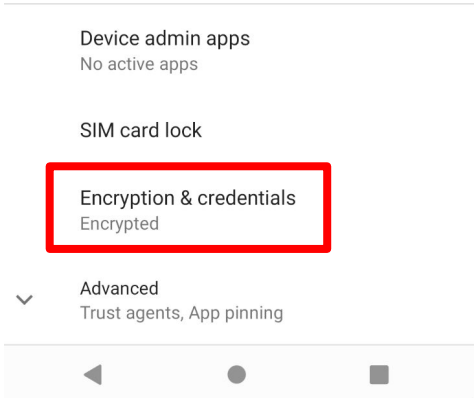


9 เลือก 'Encryption & Credentials'

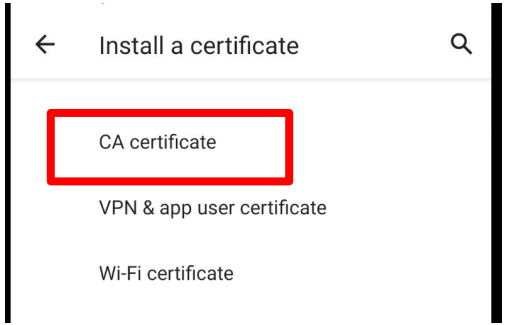


การติดตั้ง Burp Suite และ Root CA

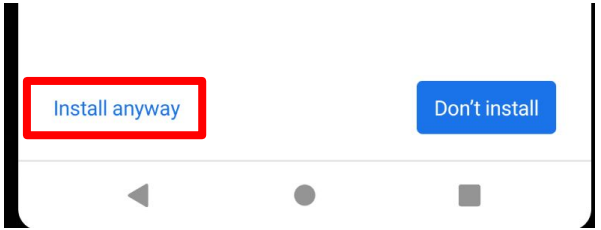
10 เลือก 'Install a certificate’



11 เลือก 'CA Certificate'.

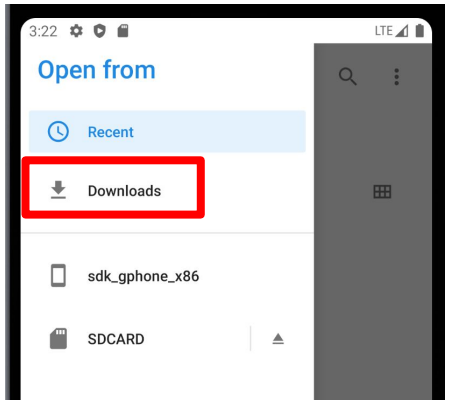


12 เลือก Install anyway

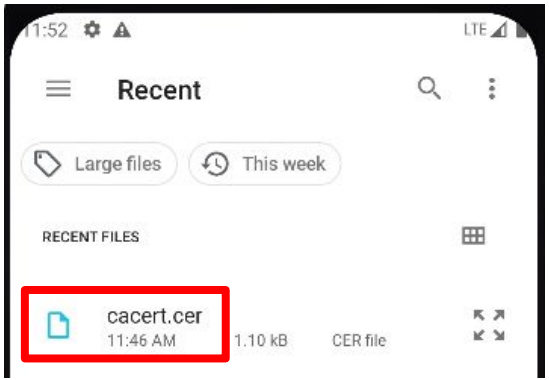


การติดตั้ง Burp Suite และ Root CA

13 เลือก Downloads



14 เลือก cacert.cer



การติดตั้ง Burp Suite และ Root CA

15 วิธีการติดตั้ง Certificate ของ Portswigger ให้เป็น System Trusted Credentials
ดาว์นโหลดไฟล์ตามที่ URL

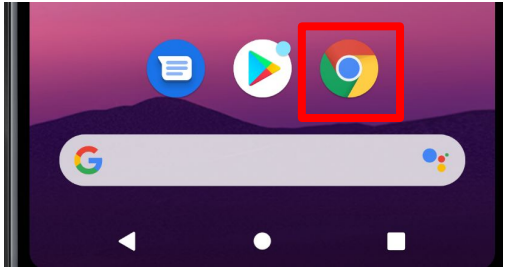
https://github.com/NVISOsecurity/MagiskTrustUserCerts/releases/download/v0.4.1/AlwaysTrustUserCerts.zip

16 เอาไฟล์ไปไว้บนเครื่อง Android ด้วยคำสั่ง

```
$ adb push AlwaysTrustUserCerts.zip /sdcard/Download
```

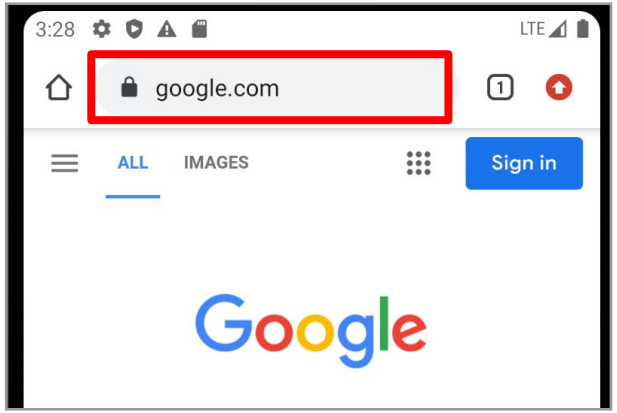
17 ในเครื่อง Android ให้เราเปิดแอพ Magisk แล้วเลือก Modules (แท็บด้านล่าง) แล้วกด install from storage แล้วเลือกไฟล์ AlwaysTrustUserCerts.zip ที่อยู่ในโฟลเดอร์ Download แล้วหลังนั้นให้ Reboot

18 ลองเข้าแอป Chrome

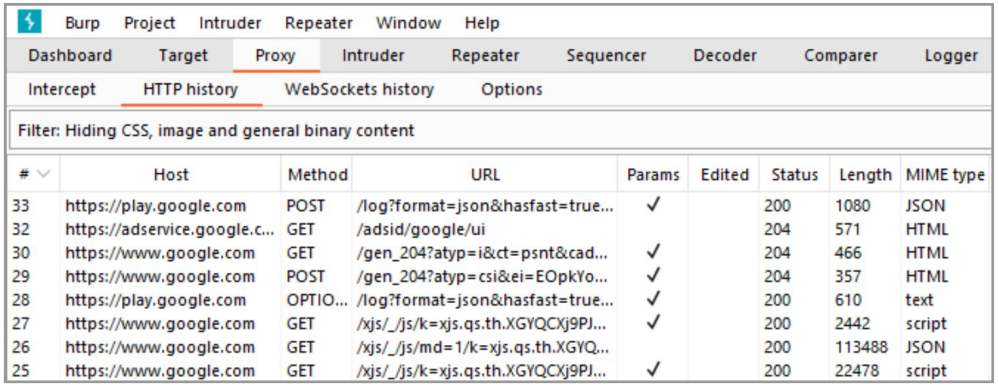


การติดตั้ง Burp Suite และ Root CA

19 ลองเข้า google.com



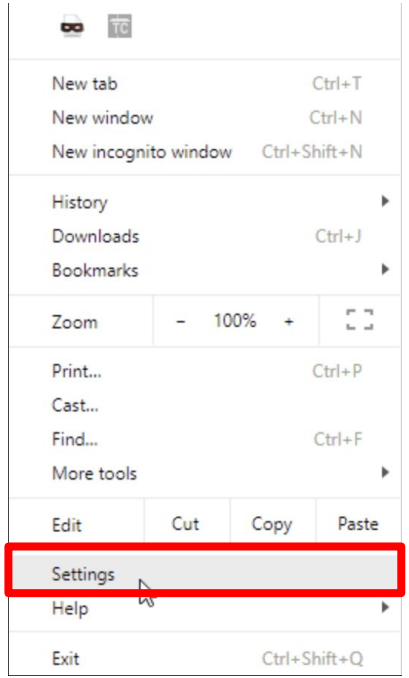
20 ดูใน Burp Suite จะดัก HTTP Request และ HTTP Response ของเว็บที่เป็น HTTPS ได้



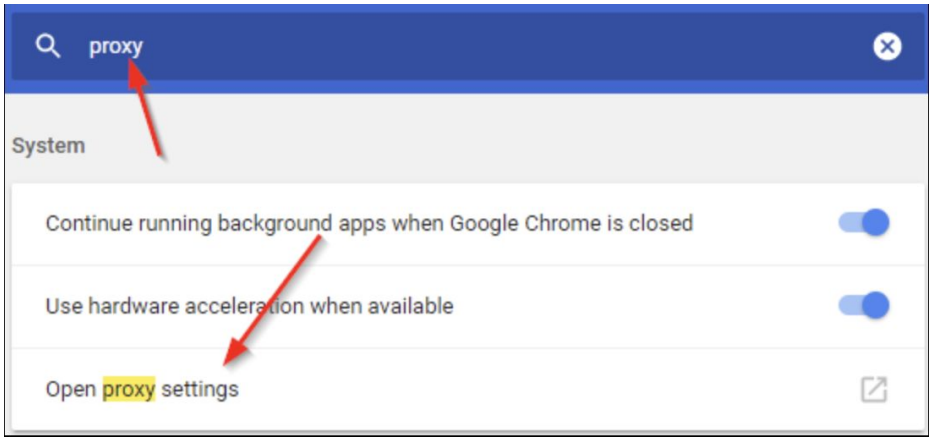
#	Host	Method	URL	Params	Edited	Status	Length	MIME type
33	https://play.google.com	POST	/log?format=json&hasfast=true...	✓		200	1080	JSON
32	https://adservice.google.c...	GET	/adsid/google/ui			204	571	HTML
30	https://www.google.com	GET	/gen_204?atyp=i&ct=psnt&cad...	✓		204	466	HTML
29	https://www.google.com	POST	/gen_204?atyp=csi&ei=EOpkYo...	✓		204	357	HTML
28	https://play.google.com	OPTIO...	/log?format=json&hasfast=true...	✓		200	610	text
27	https://www.google.com	GET	/xjs/_/js/k=xjs.qs.th.XGYQCXj9PJ...	✓		200	2442	script
26	https://www.google.com	GET	/xjs/_/js/md=1/k=xjs.qs.th.XGYQ...			200	113488	JSON
25	https://www.google.com	GET	/xjs/_/js/k=xjs.qs.th.XGYQCXj9PJ...	✓		200	22478	script

ตั้งค่า Proxy ให้ Google Chrome ผ่าน Burp Suite

1 เปิด Google Chrome แล้วเข้าไปที่ Settings ที่มุมขวาบนของหน้า Browser

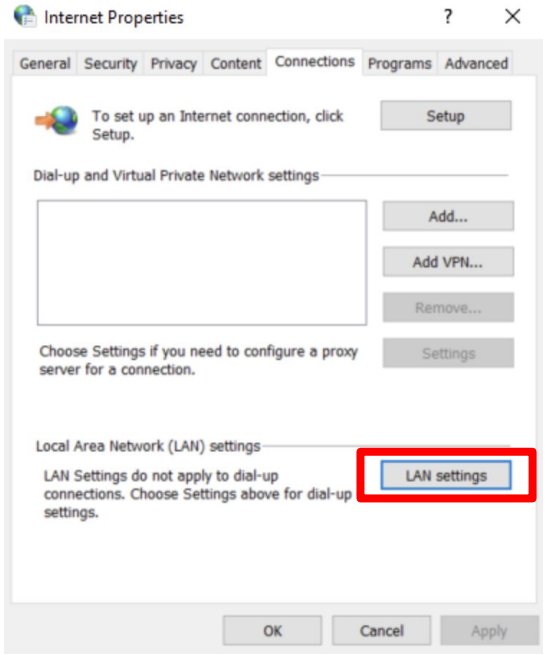


2 ค้นหาคำว่า “proxy” ในหน้า Settings แล้วกดเข้าไปใน Open proxy settings

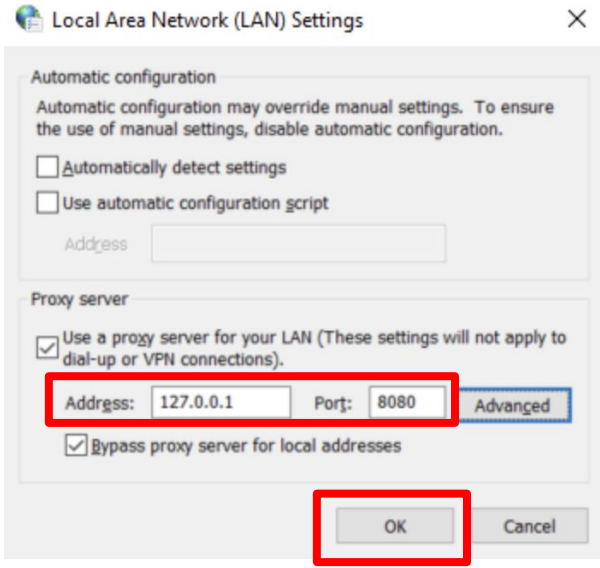


ตั้งค่า Proxy ให้ Google Chrome ผ่าน Burp Suite

3 จะมีหน้าต่าง Internet Properties เด้งขึ้นมาจากนั้นให้กดตรง LAN Settings



4 ให้ใส่ค่า 127.0.0.0 และ 8080 ตามลำดับดังรูปภาพที่แสดงข้างล่างนี้ แล้วกด OK



5 จากนั้นลองเข้า Google.com ดูใน Burp Suite จะดัก HTTP Request และ HTTP Response ได้