Secure Android Application

Defend Customer Data

(In development)

baolq@hpt.vn

@BaoLQOnSecurity

VIETNAM SECURITY DAY 2015

Presenter: **LE QUOC BAO**

WHO AM I

- Web/Mobile Pentester at HPT
- CEH, eMAPT certified
- Guitar, Coffee, Traveling, Panda and ...

hpt
knowing IT

Agenda

- Imagine a World without Mobile?
- Android Overview
- Defend Customer Data
- Q & A

HPT VIETNAM CORPORATION

hpt
knowing IT

Imagine a World without Mobile?

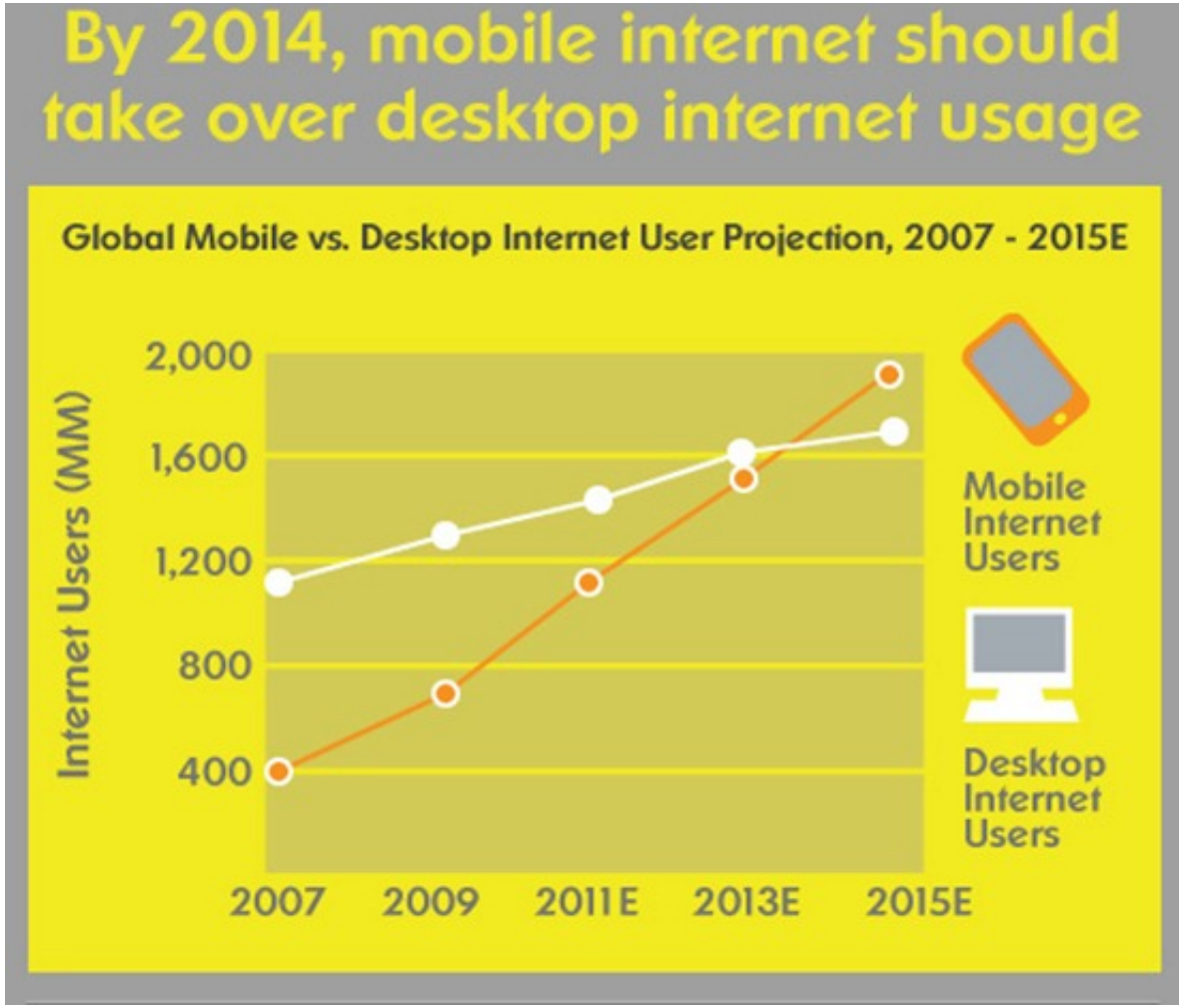
knowing IT

“Awesome” Figures

It’s my super hero! It changed my life!

- The age of Ultron (Mobile Devices)!

http://www.smartinsights.com/



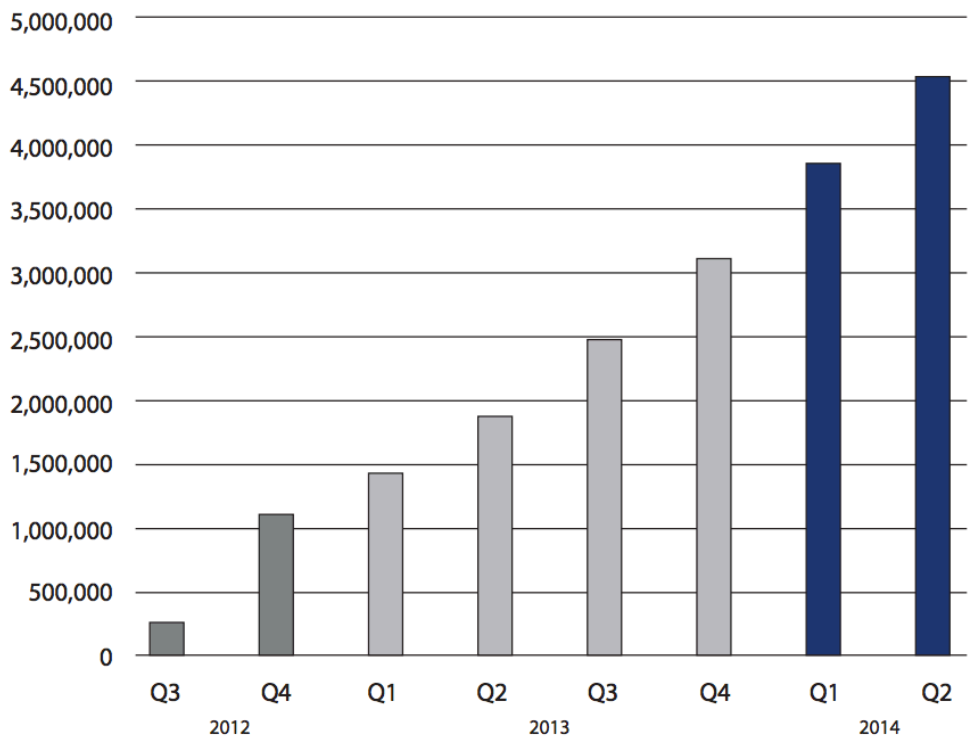
"Awesome" Figures

McAfee Mobile Malware Report

Total count of McAfee's mobile malware increased by 17% in Q2 2014

Significant number of these malware exploits were designed to exploit digital wallet service & popular messaging app. Mobile malware samples grew by 167 percent between Q1 2013 and Q1 2014.

Total count of McAfee's mobile malware increased by 17% in Q2 2014.



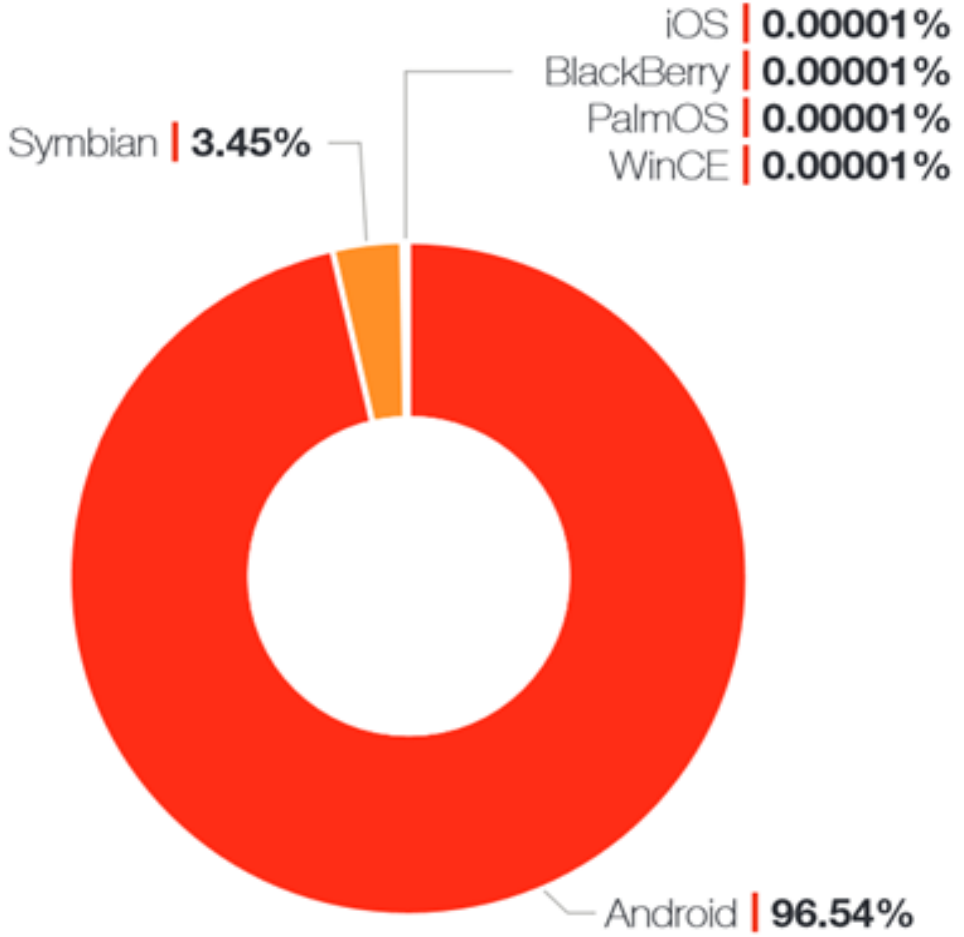
167%

2013 - 2014

"Awesome" Figures

Think about a Device you bring everyday?

- The more popular – The most targeted

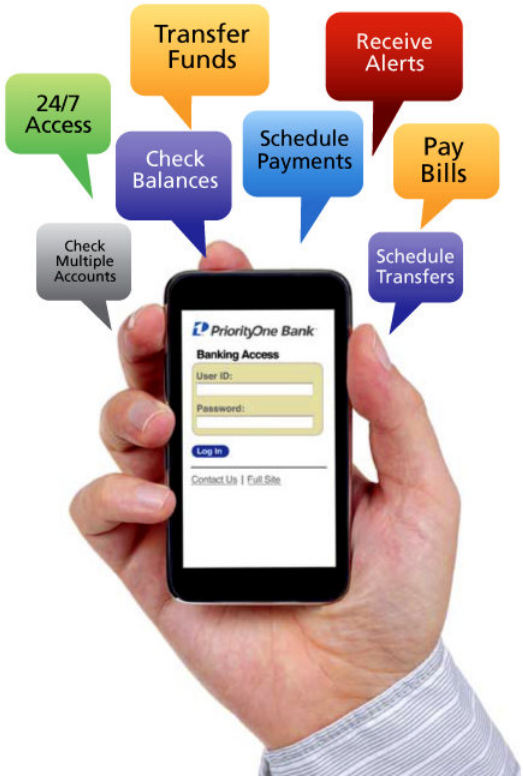


Mobile Malware - Fortinet results (2014 report)

"Awesome" Figures

So scary, leave my Money alone?

- How secure your Bank's app dev can ensure?



Security Report of
Top 100 Mobile Banking Apps

Appvigil™

70%

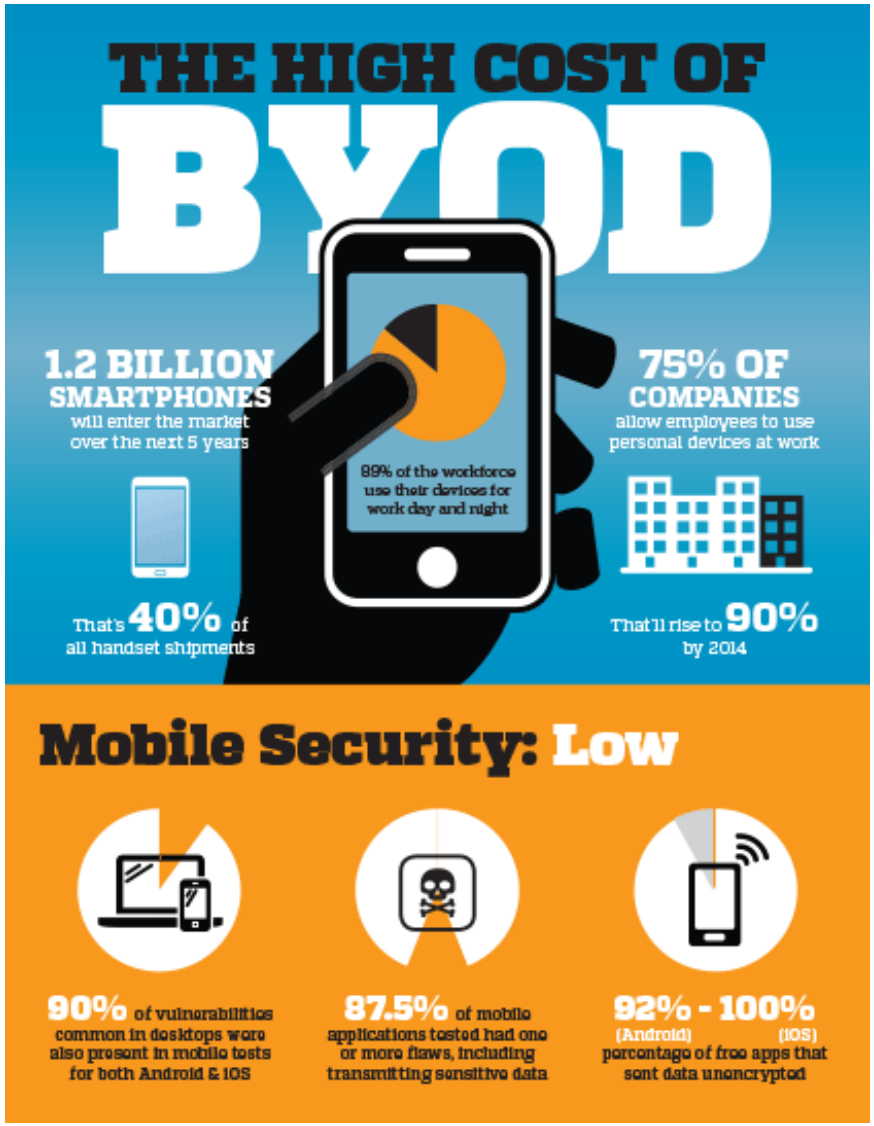
Mobile Banking Apps of Top 100 Banks in India & APAC are Vulnerable to Security Attacks & Data Leaks

BYOD – The New Trends

What's the effective Enterprise solution?

- Employee Devices

- Security



hpt
knowing IT

Android Overview

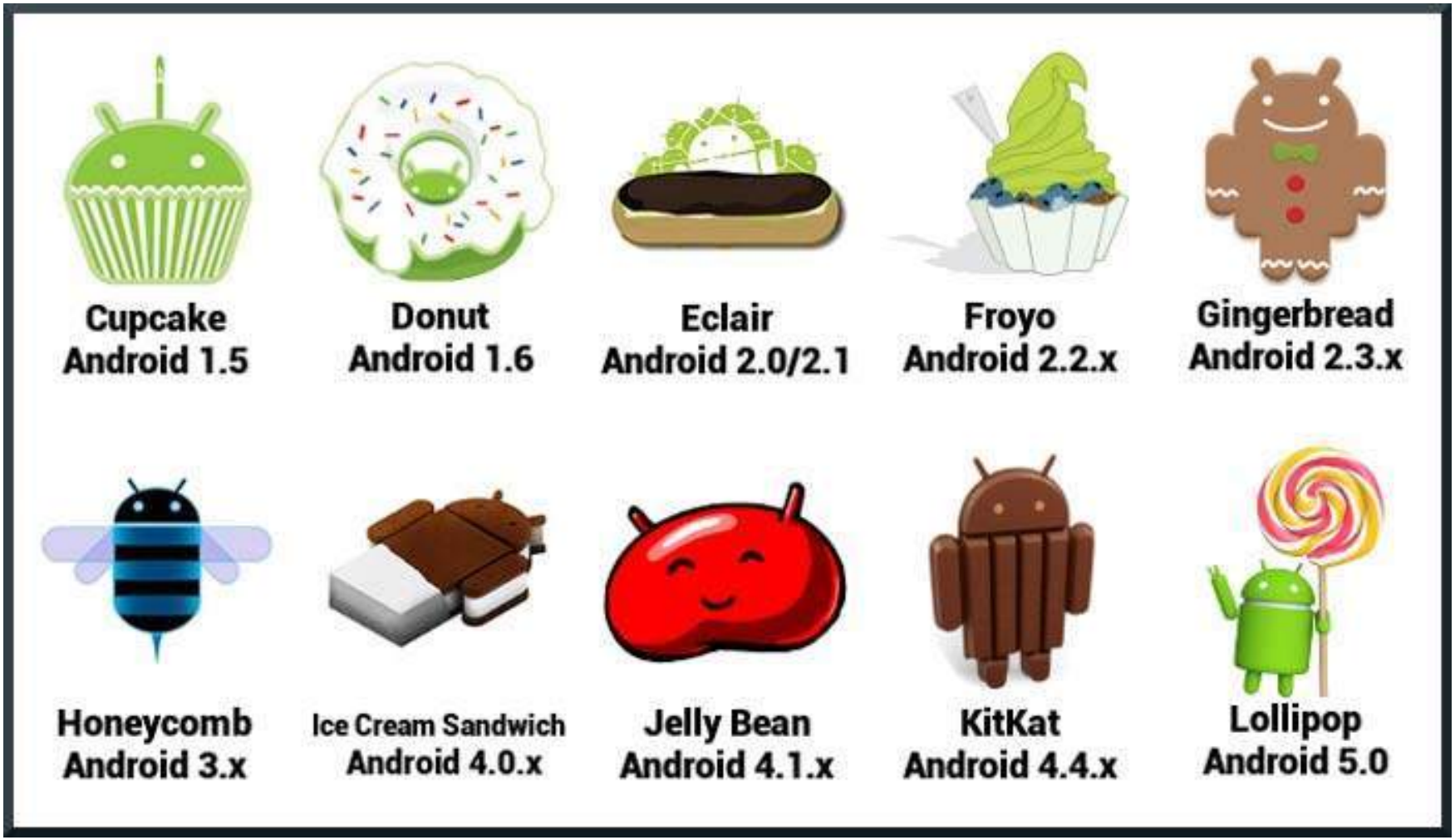
knowing IT

Android Overview

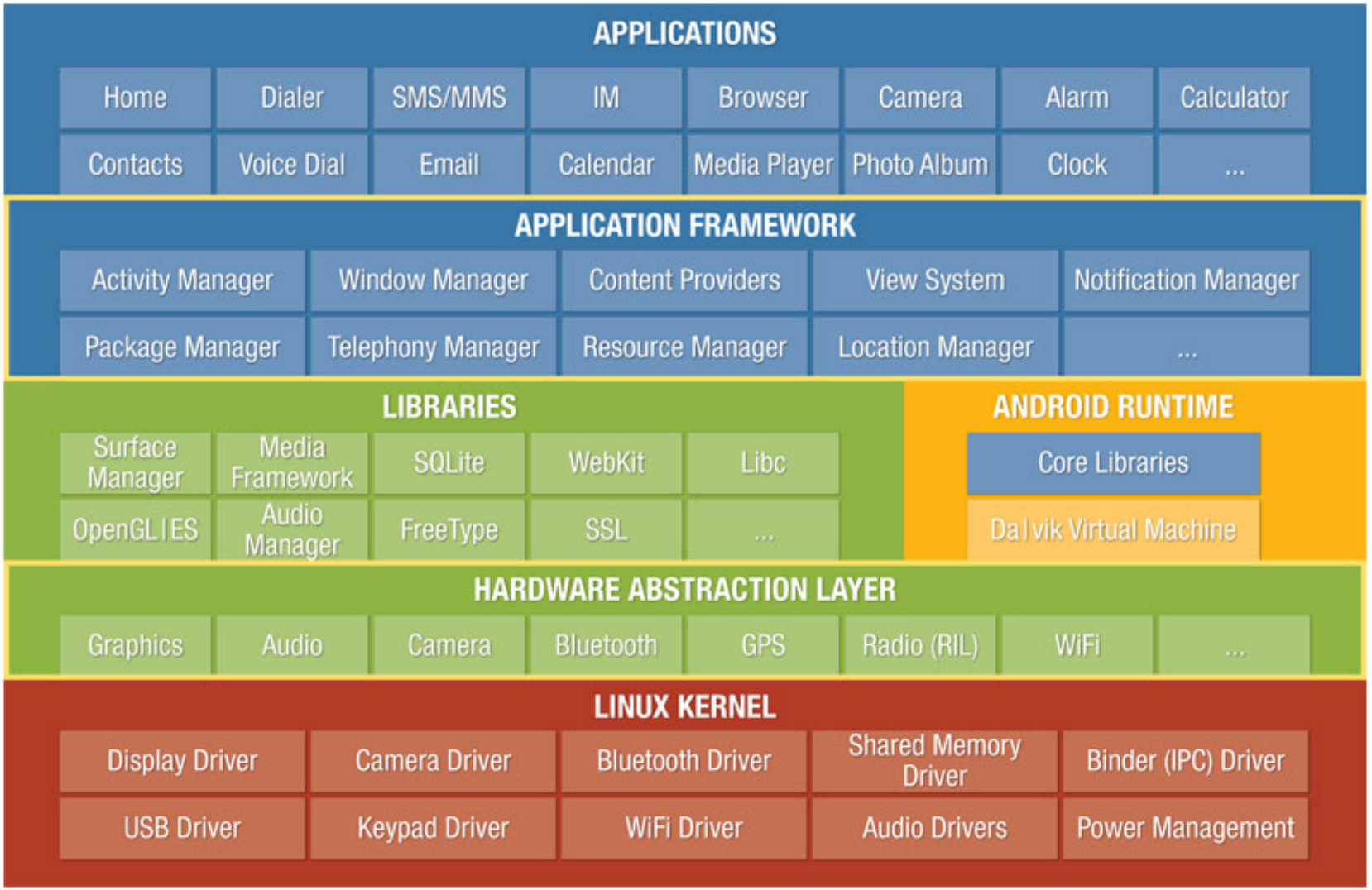
- Android OS & Architecture Model
- App Directory
- App Components
- Access Controls

Android History

They are too delicious! aren't they?



Architecture Model



App Directory

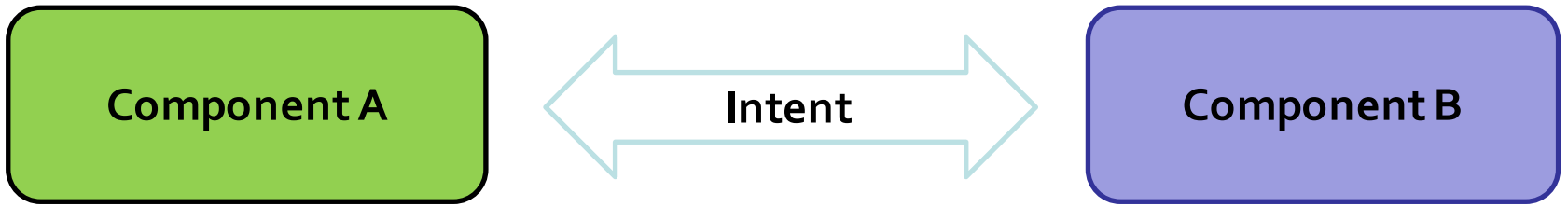
/data/data/<package_name>/

- **AndroidManifest.xml**: declare components, permissions, version, package, ...
- Lib folder: native library .so, .dll (C/C++, C#)
- **classes.dex**: like exe
- Res folder: contains resource not compile
- Assets folder: retrieved by AssessManager

App Components

We use Intent to communicate each other

- **Activity:** GUI
- **Service:** Background process
- **Content Provider:** SQLite database
- **Broadcast Receiver:** responds to system-wide broadcast announcements



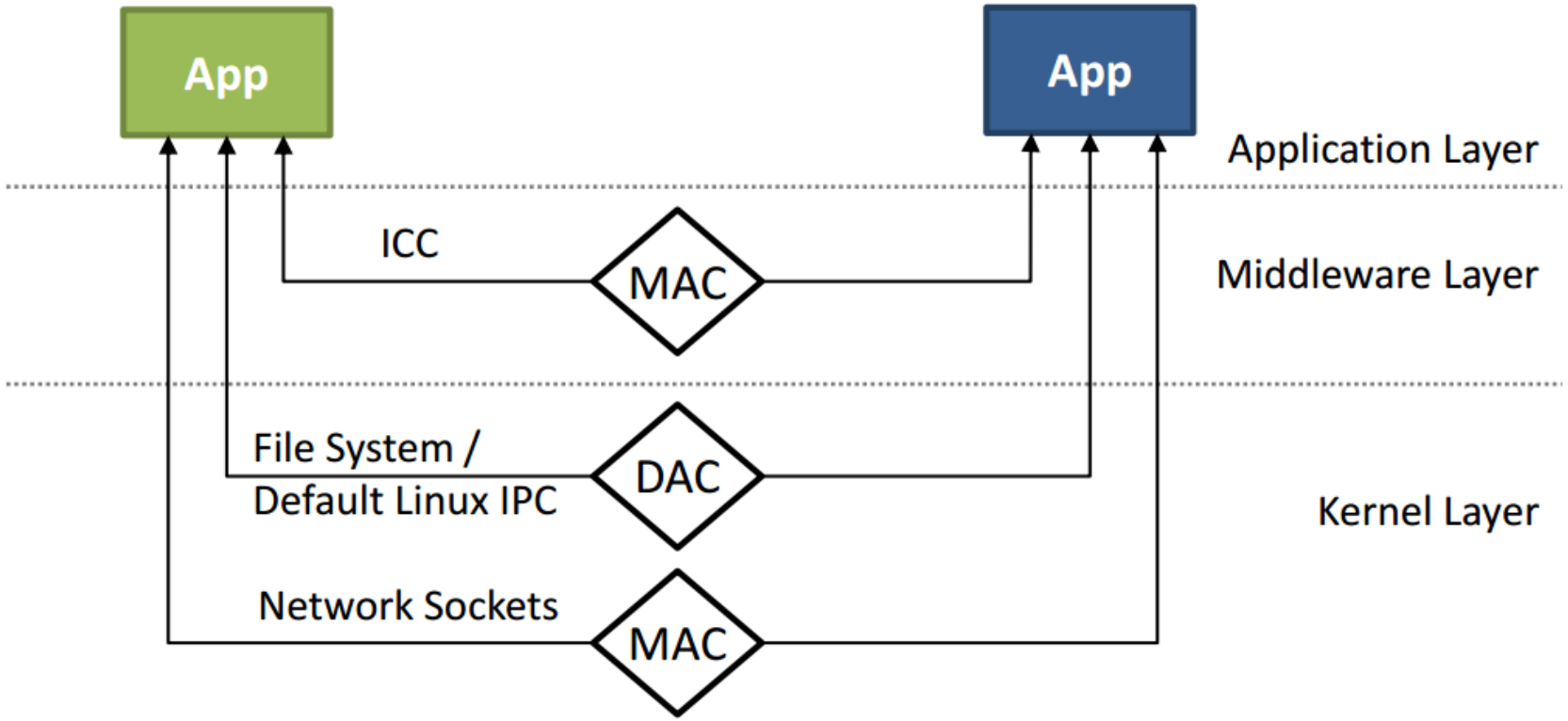
Access Controls

Linux-based OS

- **Discretionary Access Control (DAC)** – used to isolate apps (Linux)
- **Mandatory Access Control (MAC)** – used to mediate the establishment of inter-component/application communication

Access Controls

Diff between 2 concepts



ICC – Inter-Component Communication (or IPC)
DAC – Discretionary Access Control
MAC – Mandatory Access Control

hpt
knowing IT

Security Features

knowing IT

hpt knowing IT

Security Features

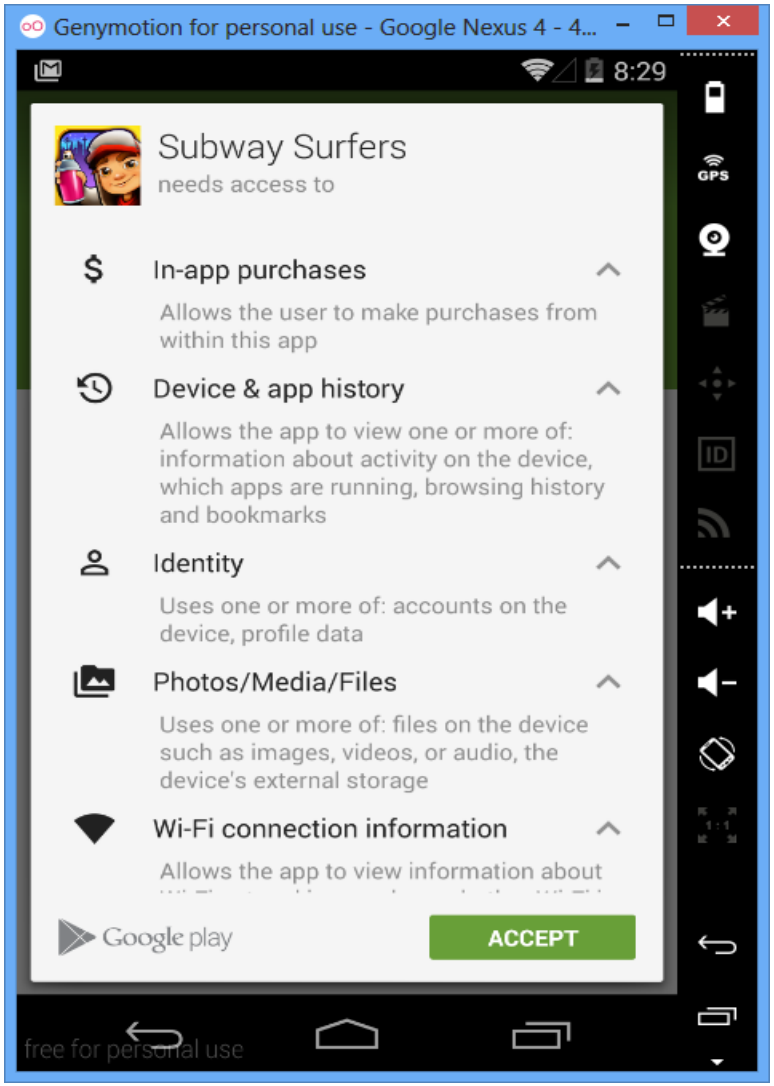
- Permissions
- App Signing
- App Verification
- App Sandbox

Permissions

User-granted Permission

Protected APIs (OS layer):

- Camera functions
- Location data (GPS)
- Bluetooth functions
- Telephony functions
- SMS/MMS functions
- Network/data connections



Permissions

Application-defined Permission

```
<permission android:name="com.hpt.MY_PERMISSION"
   android:description="@string/des_permission"
   android:label="mycustom"
</permission> // define permission
<uses-permission android:name="com.hpt.MY_PERMISSION"/> // for
external app usage
<activity android:permission="com.hpt.MY_PERMISSION"
        android:name=".YourActivity"
        android:label="@string/activity_label" /> // apply to activity
```

Permissions

Protection Level

Normal	Dangerous	Signature	SignatureOrSystem
• System automatically grants at installation • Don't warn to user	• Need user confirmation when install • Don't need to match signature	• System automatically grants the permission • Should be signed by the same Certificate	• Same certificate • Grants only to Applications in the Android system image

Permissions

List of “Dangerous” Permissions

- **CALENDAR** : READ_CALENDAR, WRITE_CALENDAR
- **CAMERA** : CAMERA CONTACTS : READ_CONTACTS, WRITE_CONTACTS, GET_ACCOUNTS
- **LOCATION** : ACCESS_FINE_LOCATION, ACCESS_COARSE_LOCATION
- **MICROPHONE** : RECORD_AUDIO
- **PHONE** : READ_PHONE_STATE, CALL_PHONE, READ_CALL_LOG, WRITE_CALL_LOG, ADD_VOICEMAIL, USE_SIP, PROCESS_OUTGOING_CALLS
- **SENSORS** : BODY_SENSORS
- **SMS** : SEND_SMS, RECEIVE_SMS, READ_SMS, RECEIVE_WAP_PUSH, RECEIVE_MMS
- **STORAGE** : READ_EXTERNAL_STORAGE, WRITE_EXTERNAL_STORAGE

hpt knowing IT

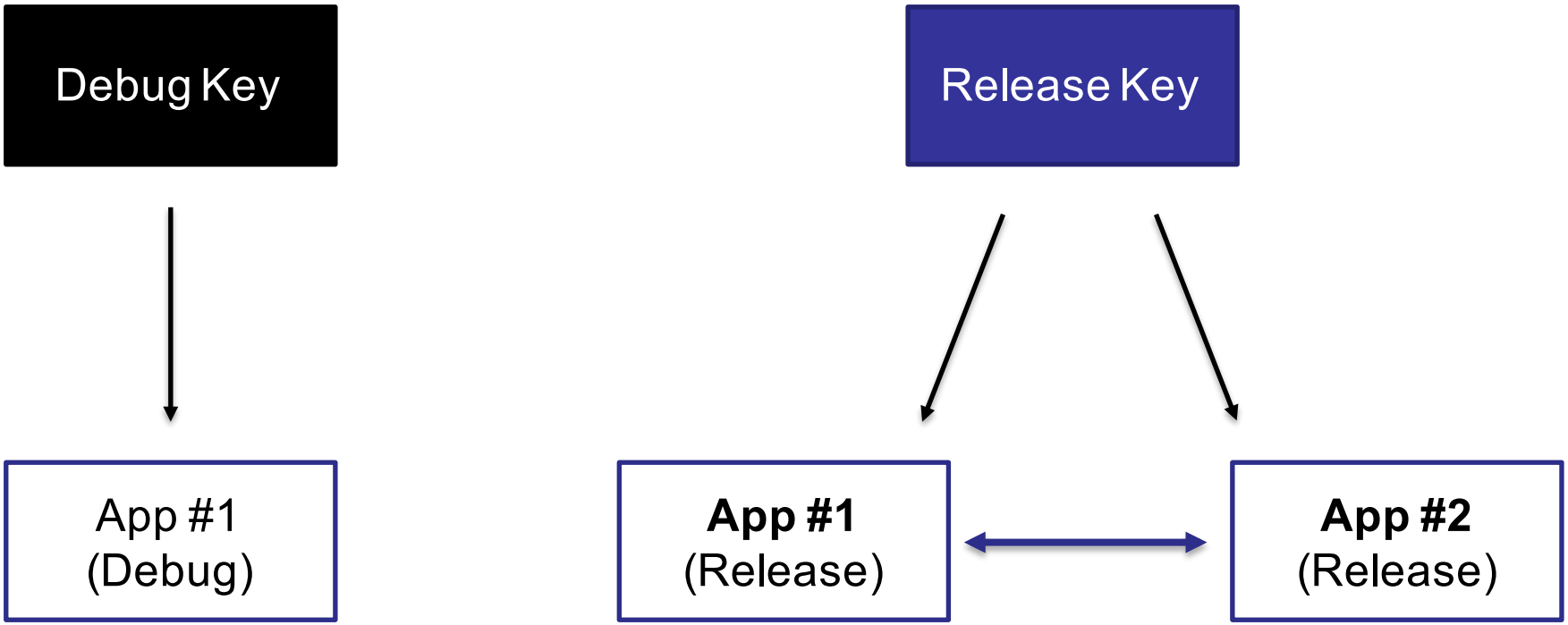
App Signing

Your apk must be signed to install

- Ensure the authenticity of the author on updates
- Establish trust relationship among apps signed with same key (Share Permission, UID, Process, Sandbox)

App Signing

Signing Type

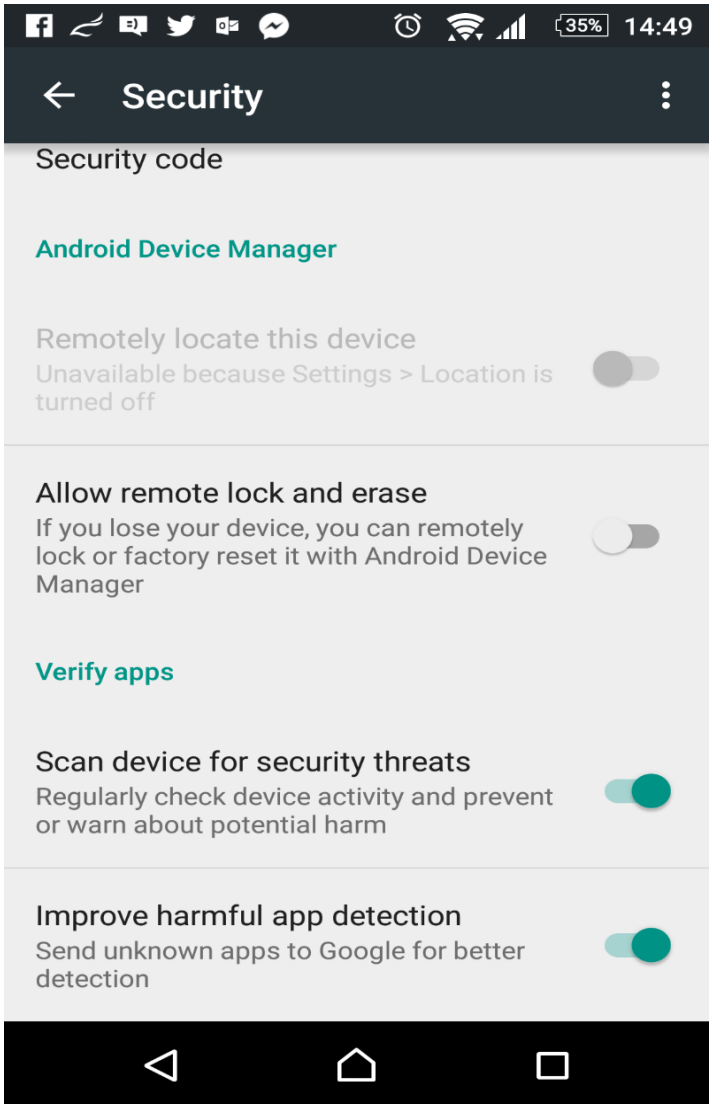


Same signing key means
permissionLevel="**signature**" works!

App Verification

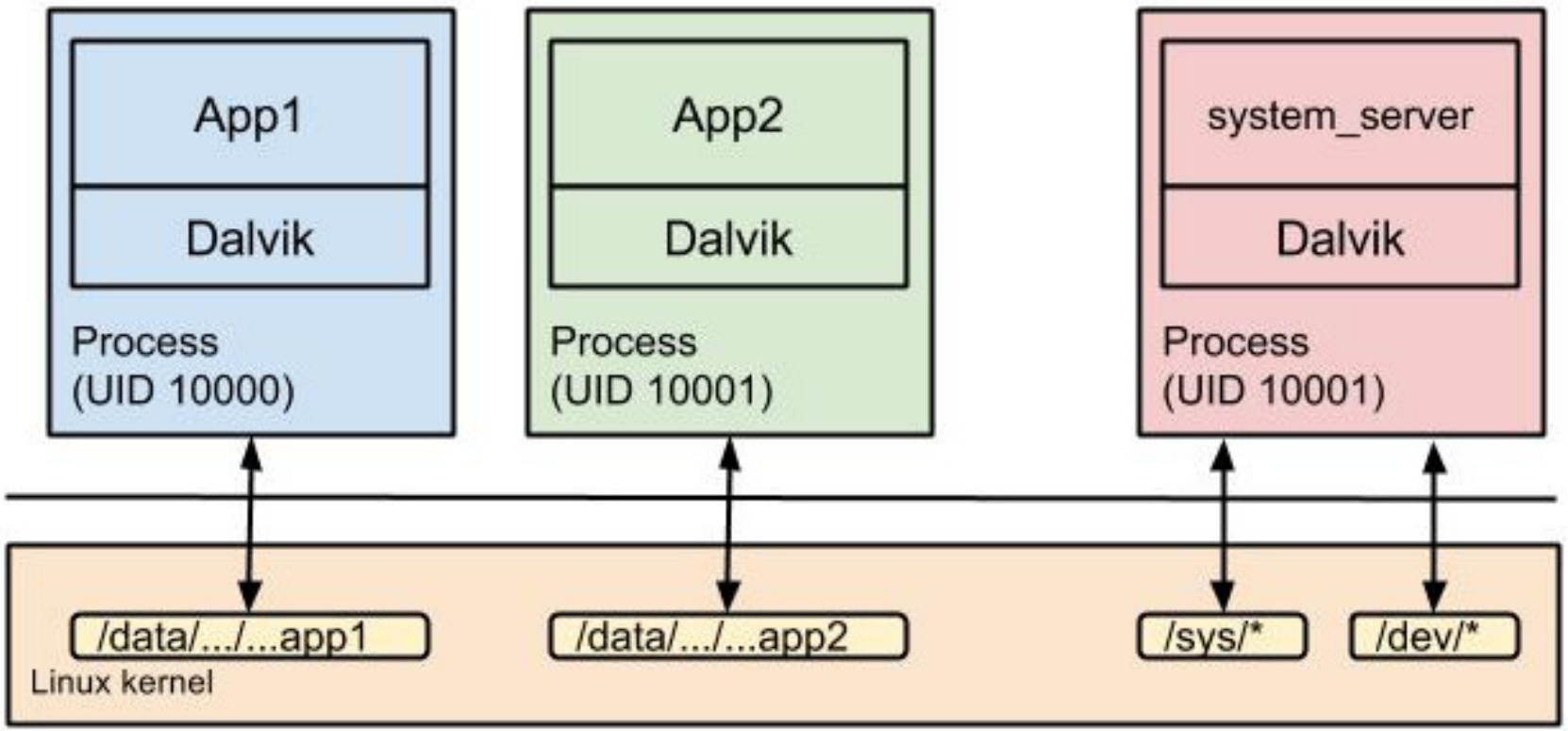
Risk mitigation for your Device

- Android 4.2 and later.
- Enable by default (Google Settings app).
- Alert when install harmful app or Block installation.



App Sandbox

Isolated File Storage



hpt
knowing IT

Defend Customer Data (development tips)

knowing IT

Defend Customer Data

- Secure Client Storage
- Secure Transport Layer
- Reduce Information Leak
- Storing Secret Key
- Data Validation
- Secure Inter-process Communication
- Binary Protection

hpt knowing IT

Secure Client Storage

10 Million Customers At Risk

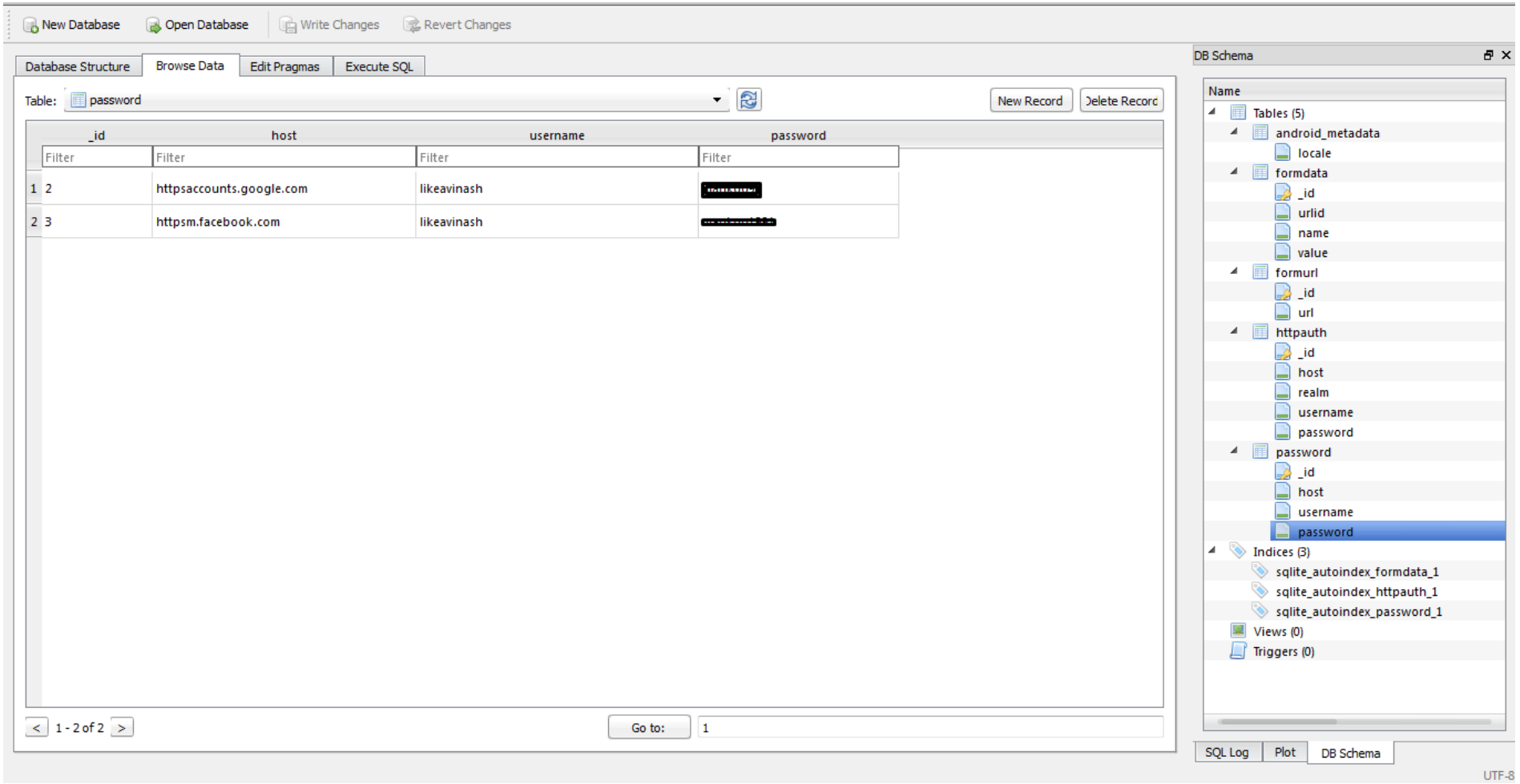
[CVE-2014-0647] Insecure Data Storage of User Data Elements in Starbucks v2.6.1 iOS mobile application

From: Daniel Wood <daniel.wood () owasp org>
Date: Mon, 13 Jan 2014 22:28:33 -0600

```
Title: [CVE-2014-0647] Insecure Data Storage of User Data Elements in Starbucks v2.6.1 iOS mobile application
Published: January 13, 2014
Reported to Vendor: December 2013 (no direct response)
CVE Reference: CVE-2014-0647
Credit: This issue was discovered by Daniel E. Wood
http://www.linkedin.com/in/danielewood

Product: Starbucks iOS mobile application
Version: 2.6.1 (May 02, 2013)
Vendor: Starbucks Coffee Company
URL: https://itunes.apple.com/us/app/starbucks/id331177714

Issue:  Username, email address, and password elements are being stored in clear-text in the session.clslog crashlytics
log file.
Location:       /Library/Caches/com.crashlytics.data/com.starbucks.mystarbucks/session.clslog

Within session.clslog there are multiple instances of the storage of clear-text credentials that can be recovered and
leveraged for unauthorized usage of a users account on the malicious users' own device or online at
https://www.starbucks.com/account/signin.  It contains the HTML of the mobile application page that performs the
account login or account reset.  session.clslog also contains the OAuth token (signed with HMAC-SHA1) and OAuth
signature for the users account/device to the Starbucks service.
```

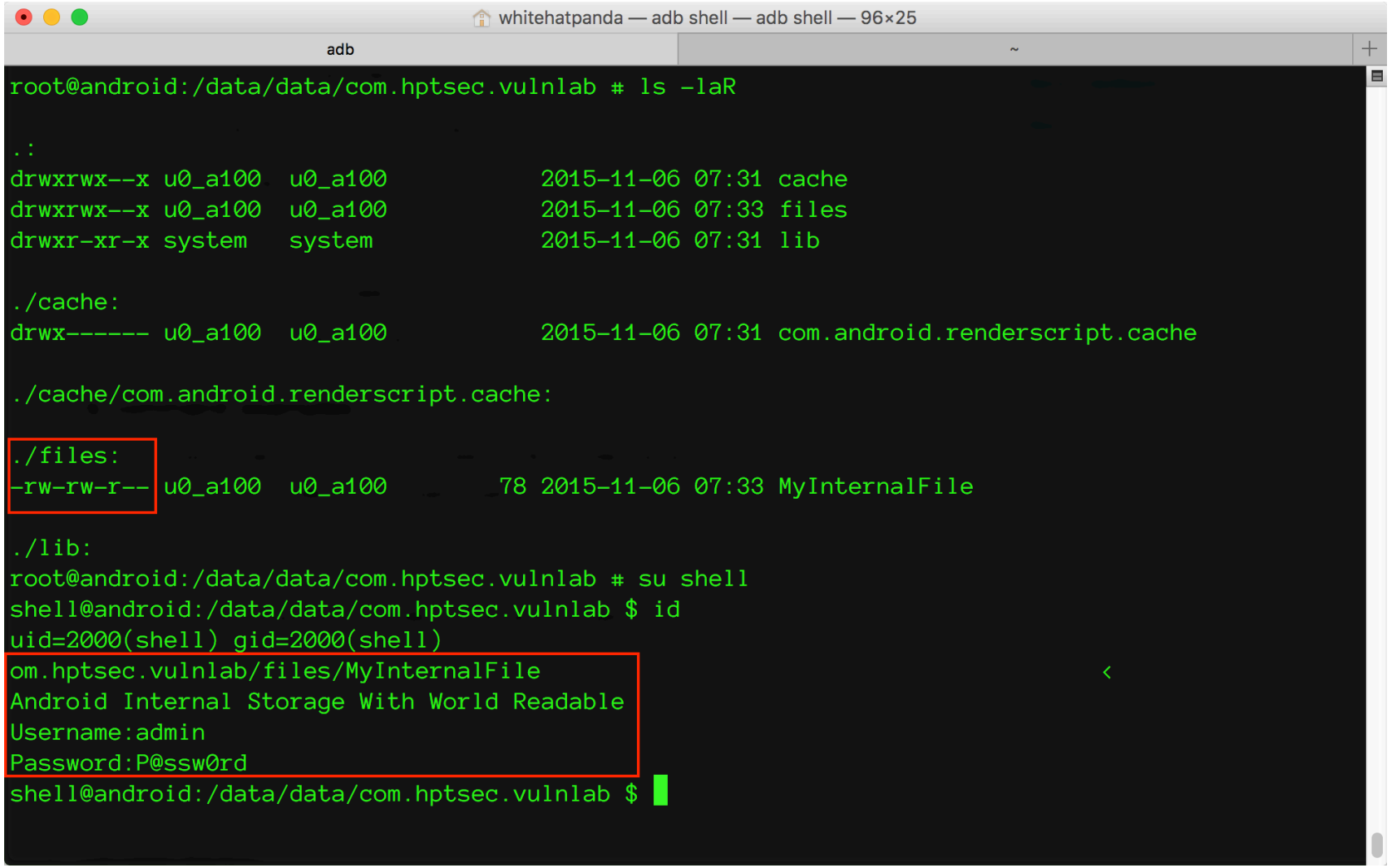
Secure Client Storage

Wine App webview.db plaintext



Secure Client Storage

MODE_WORLD_READABLE



Development Tips

- Store data in app sandbox
- Encrypt high sensitive data in shared_prefs, sqlite db, etc
- Don’t use MODE_WORLD_READ/WRITEABLE.
- API < 19, SD Card is readable by default

Secure Client Storage

Where is the Encryption Key?

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
<string
name="sdk_phone_number">iHCgspXVW8Xic4ZyRZ77RoGA=
=</string>
<string name="sdk_is_active">HAWA7OUMK1QI=</string>
<string
name="sdk_fullname">WX1f0crHrpQxKxq7Uk2L4WtEQ==</str
ing><string </map>
</xml>
```

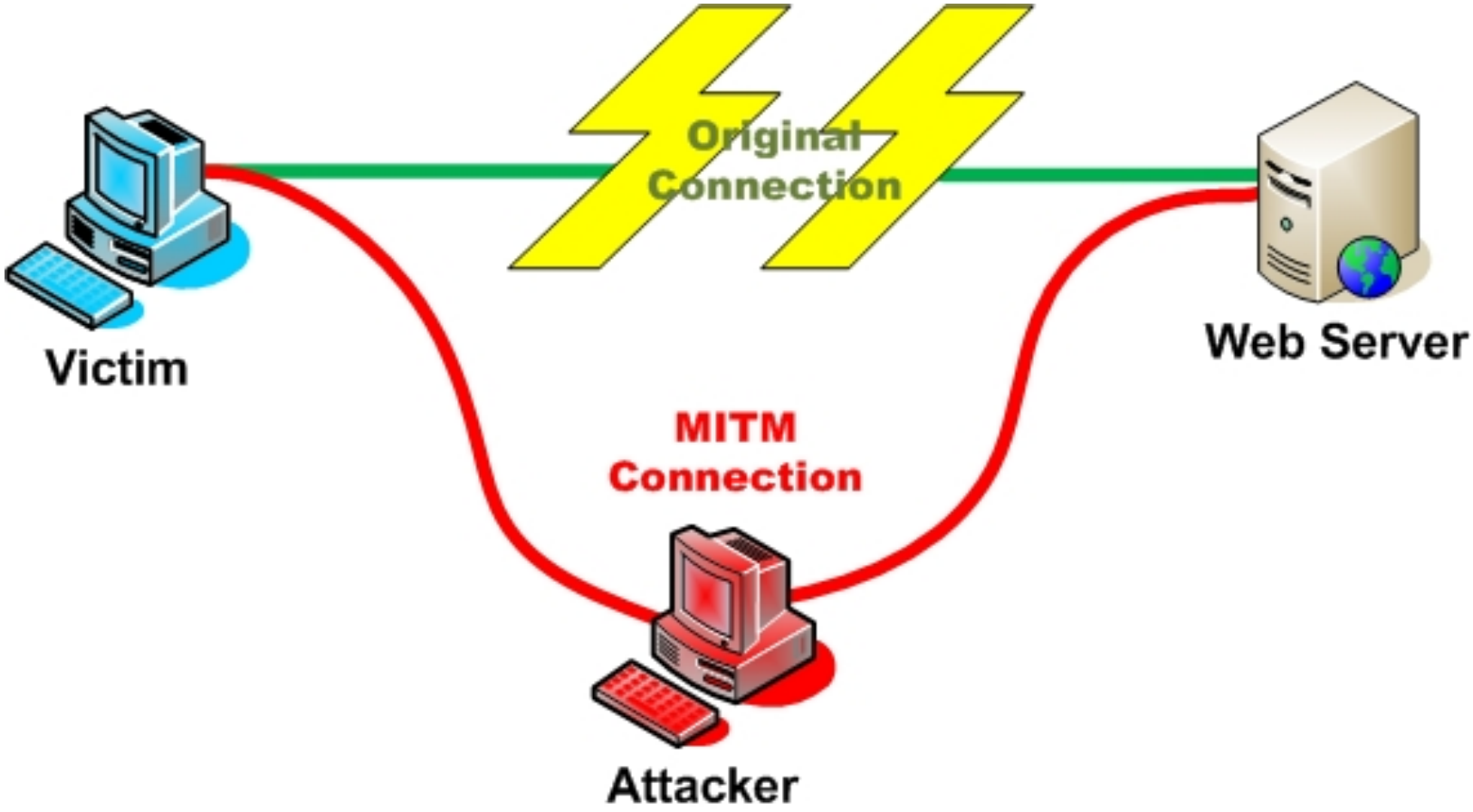
Secure Transport Layer

There’re too many un-trusted network ‘round me

- Authenticating user through HTTPS
- Certificate Pinning
- Review 3rd-party data transfer over HTTP
- Do not Trust all Certificates

Secure Transport Layer

Man-in-The-Middle



Secure Transport Layer

Various app fails to validate SSL

Android apps that fail to validate SSL

File Edit View Insert Format Data Tools Add-ons Help

whitehat.panda@gmail.com

Share

View only

App

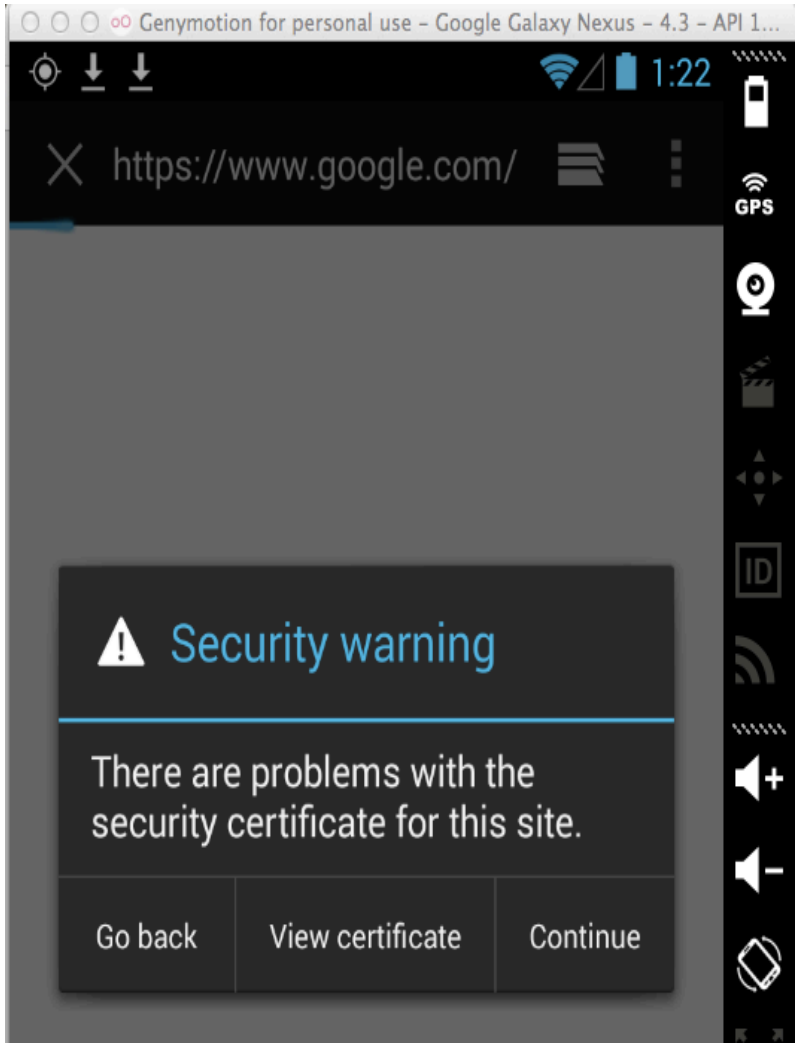
Temporary filter 1 **Range:** A1:AA23668

	A	B	C	D	E	F	G	H	I	J	K
1	App	Link	Genre	Count	Date added	Version tested	nallodroid broker	Library traffic observed	Non-library traffic observed	Verified	'ulnerabl
19470	Pence Wealth Management	com.mobileapploader.a	Finance	50+	2014-10-21	219796	TRUE	TRUE	TRUE	Yes	Yes
20858	Huntington Mobile	com.huntington.m	Finance	500,000+	2014-09-03	2.1.222	Maybe		TRUE	Yes	Yes
20860	State Bank Anywhere	com.sbi.SBIFreedomPl	Finance	500,000+	2014-09-13	2.0.1	FALSE		TRUE	Yes	Yes
20862	Finansbank Cep Şubesi	com.finansbank.mobile	Finance	500,000+	2014-09-16	1.1.5	TRUE		TRUE	Yes	Yes
20863	Альфа-Банк (Alfa-Bank)	ru.alfabank.mobile.andr	Finance	500,000+	2014-09-25	5.5.1.1	TRUE		TRUE	Yes	Yes
20864	Westpac Mobile Banking	org.westpac.bank	Finance	500,000+	2014-09-25	5.21	TRUE		TRUE	Yes	Yes
20866	Virtual Wallet by PNC	com.pnc.ecommerce.m	Finance	500,000+	2014-10-01	2.1.1	TRUE		TRUE	Yes	Yes
20867	Supermóvil	mx.bancosantander.su	Finance	500,000+	2014-10-21	2.0	TRUE		TRUE	Yes	Yes
20965	Kmart	com.kmart.android	Shopping	500,000+	2014-09-10	6.2.6	TRUE		TRUE	Yes	Yes
20966	Sears	com.sears.android	Shopping	500,000+	2014-09-10	6.2.6	TRUE		TRUE	Yes	Yes
20971	MiniInTheBox Online Shopping	com.miniinthebox.andr	Shopping	500,000+	2014-09-15	2.0.0	Maybe	TRUE	TRUE	Yes	Yes
21708	United Educational CU	com.metova.cuae.uecu	Finance	500+	2014-09-27	1.0.27	TRUE		TRUE	Yes	Yes
21709	United Advantage NW Federal Cr	com.myappengine.uanv	Finance	500+	2014-09-27	1.7	TRUE		TRUE	Yes	Yes
21711	Smart 어음정보	kr.or.knote.android	Finance	500+	2014-10-02	1.0.3	FALSE		TRUE	Yes	Yes
21712	Quest Federal CU Mobile	com.metova.cuae.ques	Finance	500+	2014-10-09	1.0.27	TRUE		TRUE	Yes	Yes
21713	Forest Area FCU Mobile	com.metova.cuae.fafcu	Finance	500+	2014-10-10	1.0.29	TRUE		TRUE	Yes	Yes
22609	LocalSense	com.LocalSense	Shopping	500+	2014-09-28	1.2.1	FALSE		TRUE	Yes	Yes

Secure Transport Layer

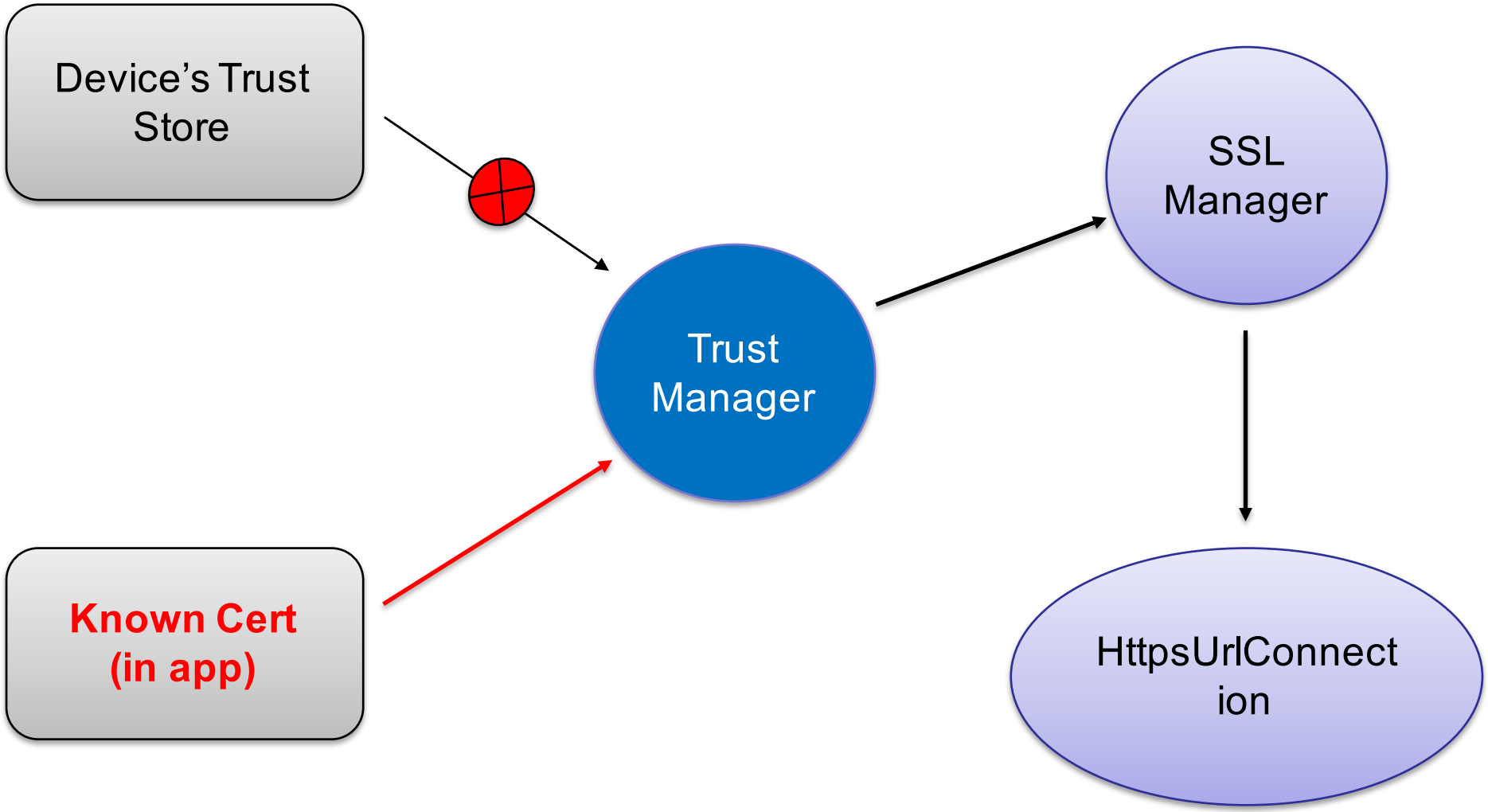
Common Mistakes

- Do not check Server certificates
- Do not verify hostname
- Ignore SSL errors



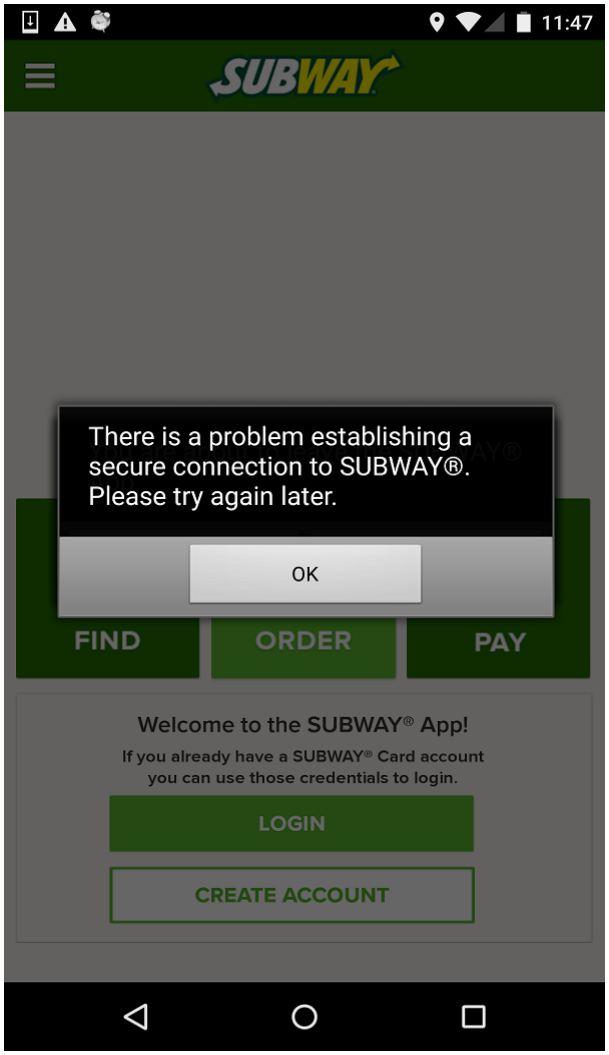
Secure Transport Layer

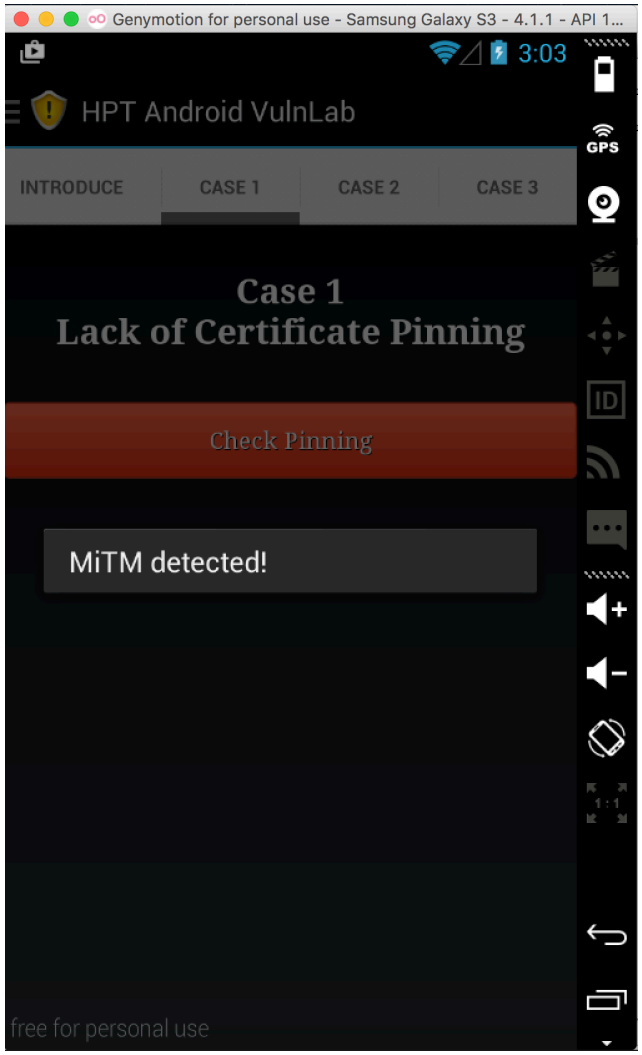
Certificate Pinning – Only my Cert



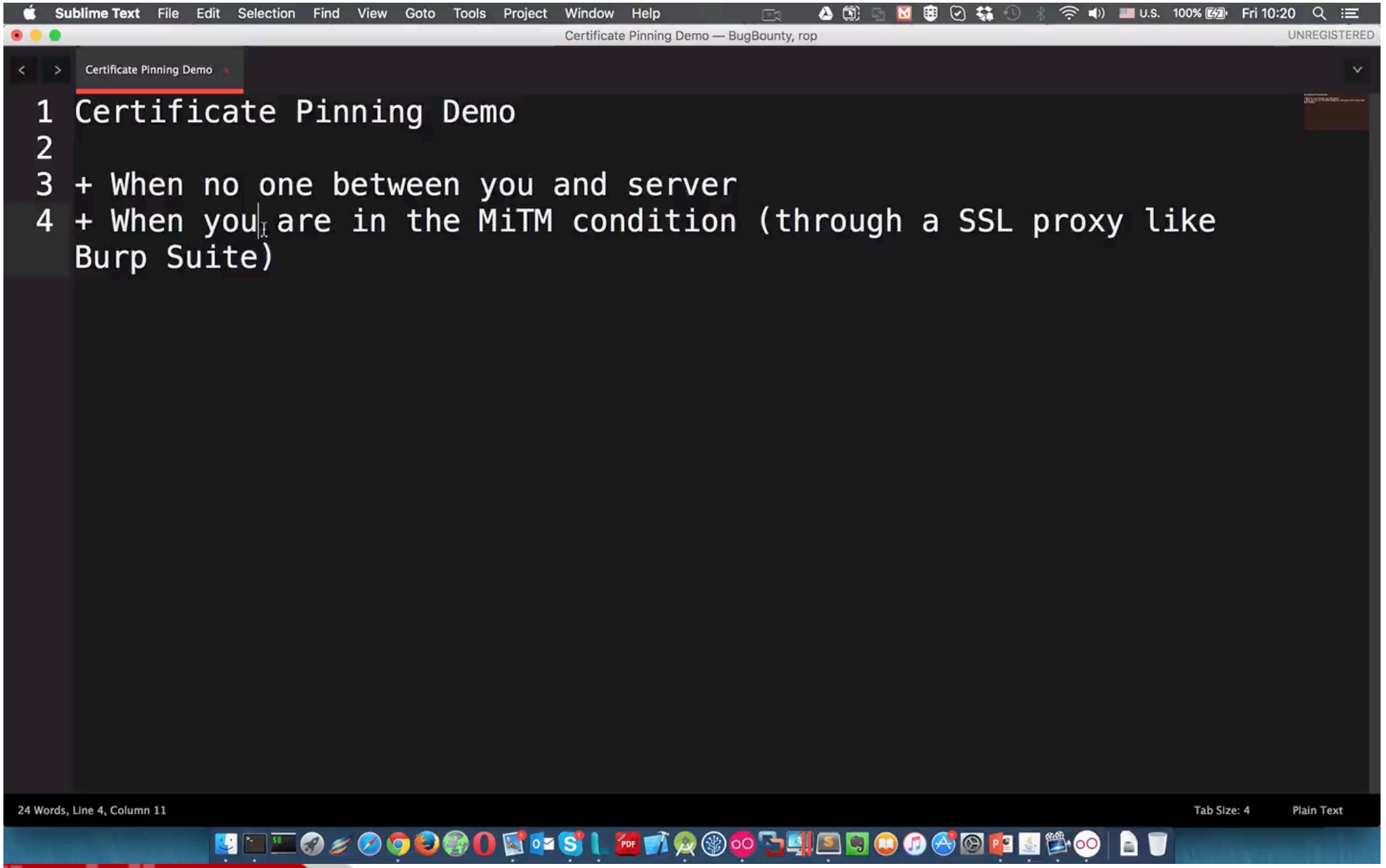
Secure Transport Layer

Customers have more belief in Application





Certificate Pinning Demo



Reduce Information Leak

Android Logcat – System Log

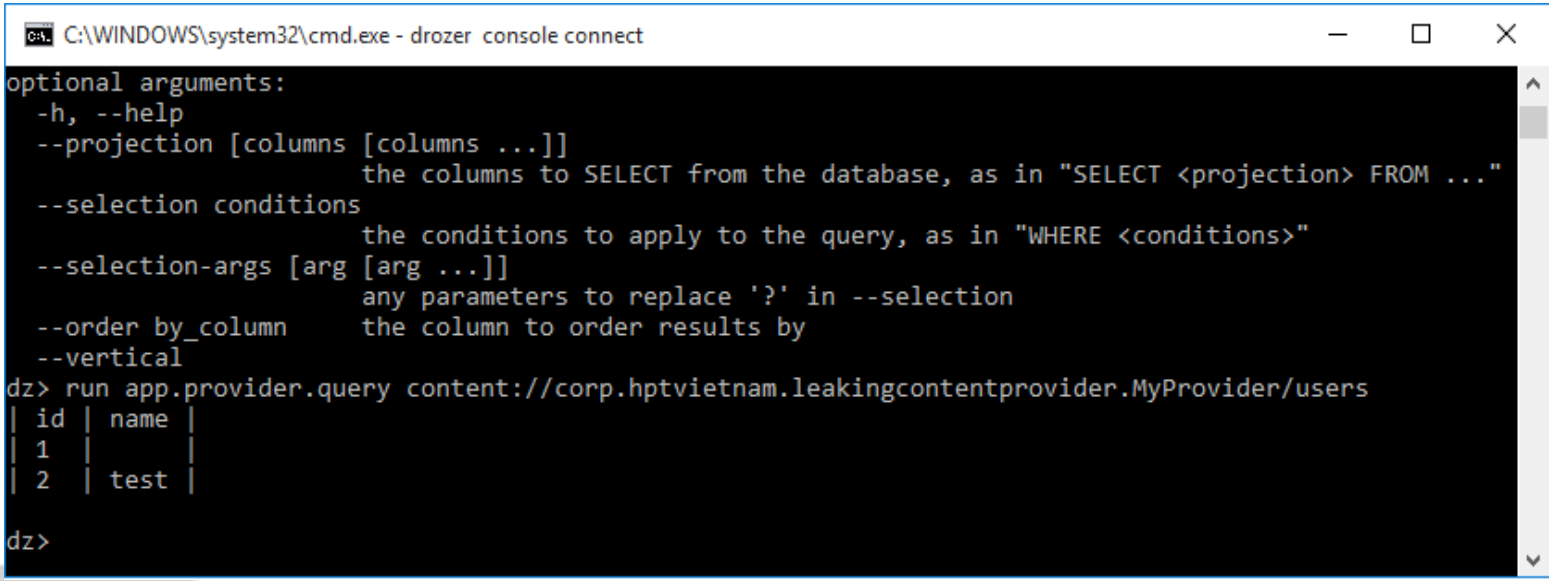
```
public void login(String userid, String password, OnLoginCompleted callback) throws
Exception {
      StringBuffer v1 = new StringBuffer("");
      v1.append("{");
      v1.append(String.format("\"app_key\":\"%s\"", "xxxxxxx"));
      v1.append(String.format(",\"user_email\":\"%s\"", userid));
      v1.append(String.format(",\"password\":\"%s\"", password));
      v1.append("}");
      HttpRequest vo = new HttpRequest();
      vo.setUrl(new URL("http://xxx.xxx/login"));
      vo.setBody(v1.toString());
      Log.e("LOGIN_DATA", v1.toString());
      new APICallTask(this.client.getHttputil(), new OnAPICallCompleted() {
         ...[SNIP]...
      }).execute(new Object[]{vo});
   }
```

Reduce Information Leak

Secure Content Provider

```
<provider
        android:name=".MyProvider"
        android:authorities="corp.hptvietnam.leakingcontentprovider.MyProvider"
        android:exported="true" >
</provider>
```

(API <= 16: exported true by default)



Storing Secret Key

Hardcoded Key is Bad

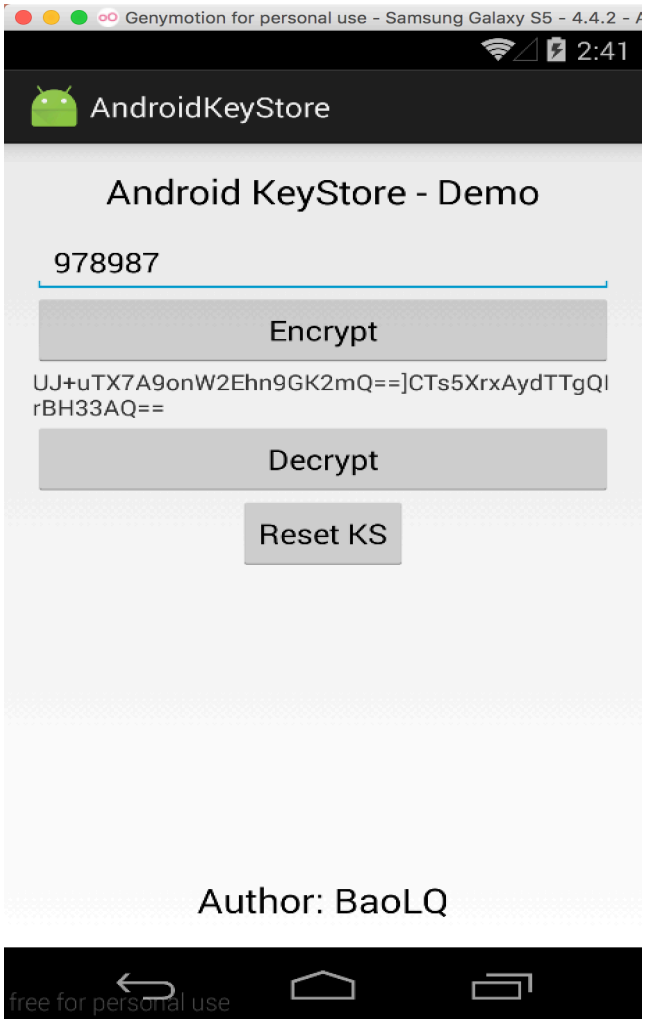
- It is very easy to figure out

```
static {
        MyAESKey.value = new byte[]{84, 104, 101, 66,
101, 115, 116, 83, 101, 99, 114, 101, 116, 75, 101, 121};
        }
```

Storing Secret Key

Are there any way to store EncKey safety?

- Android Keystore is the best way
- Support AES – CBC, RSA (private/public key)...
- More complex than the old one



https://github.com/nelenkov/android-keystore

Android Keystore Demo

Data Validation

Parameterized Statement

- The oldest bug still alive in SQLite DB

```
query = "SELECT * FROM " + TABLE_NOTE + " WHERE " +
KEY_NOTEDATA + " LIKE '%" + mySearch + "%' AND "+
KEY_PRIVATENOTE + " = 1";
```

```
query = "SELECT * FROM TABLE_NOTE WHERE
KEY_NOTEDATE LIKE ?";
PreparedStatement prepStmt =
con.prepareStatement(selectStatement);
prepStmt.setString(1, "%" + mySearch + "%");
```

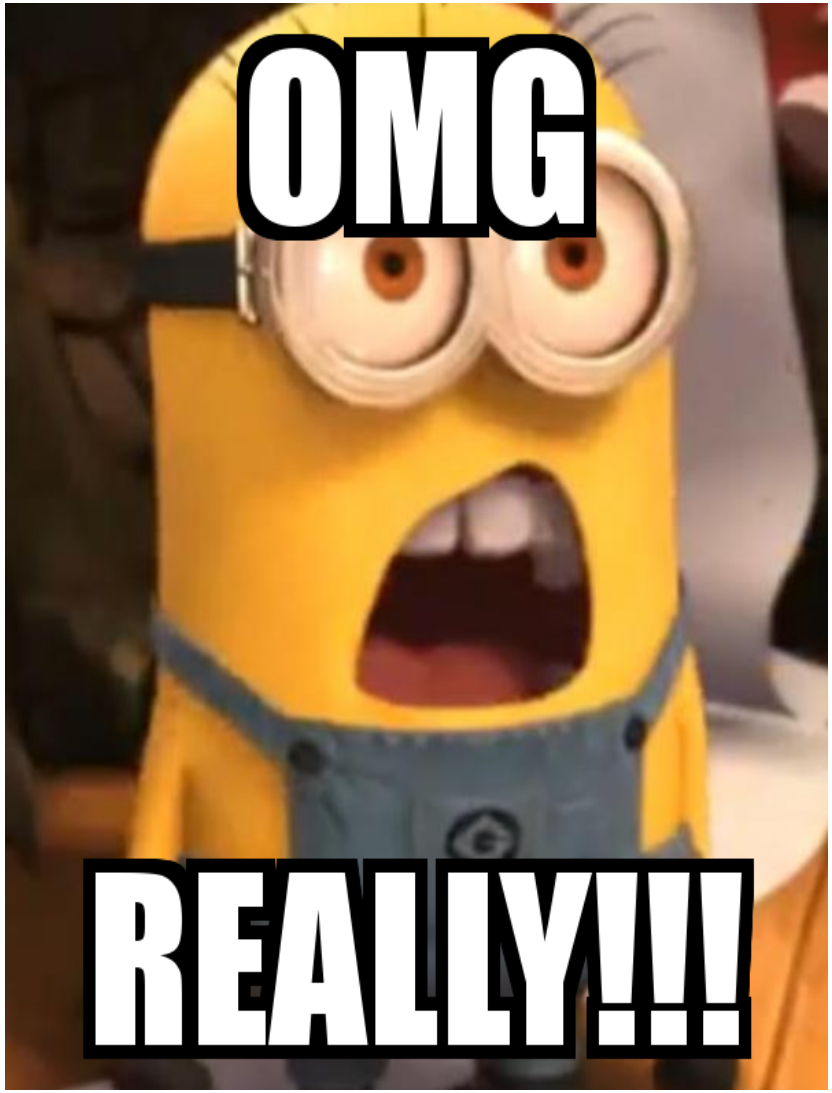
Data Validation

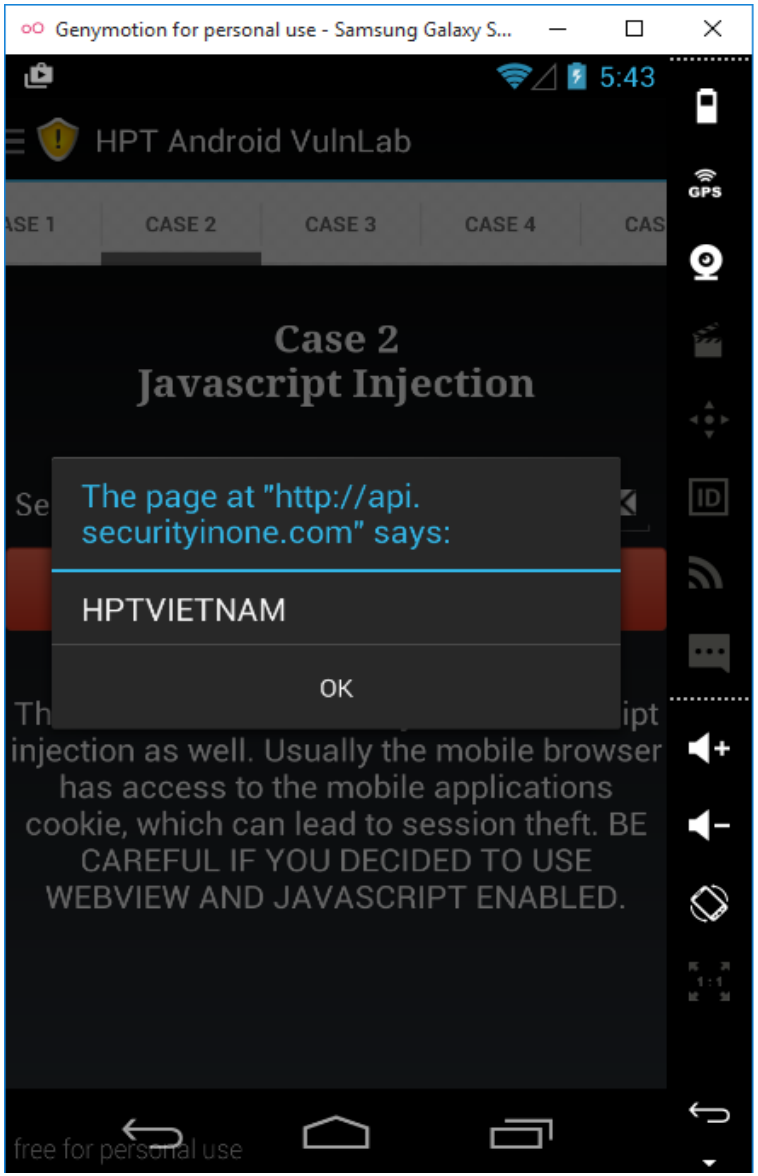
Disable WebView Javascript/FileAccess

- Disable JavaScript or validate the input on server

```
WebSettings webSetting = webView.getSettings();
webSetting.setJavaScriptEnabled(false);
webSetting.setAllowFileAccess(false);
webView.setWebChromeClient(new  WebChromeClient());
webView.loadUrl(url);
```

Data Validation





Data Validation

Sanitize input URI on client

- Exported content provider
- Don't sanitize any untrusted URI input

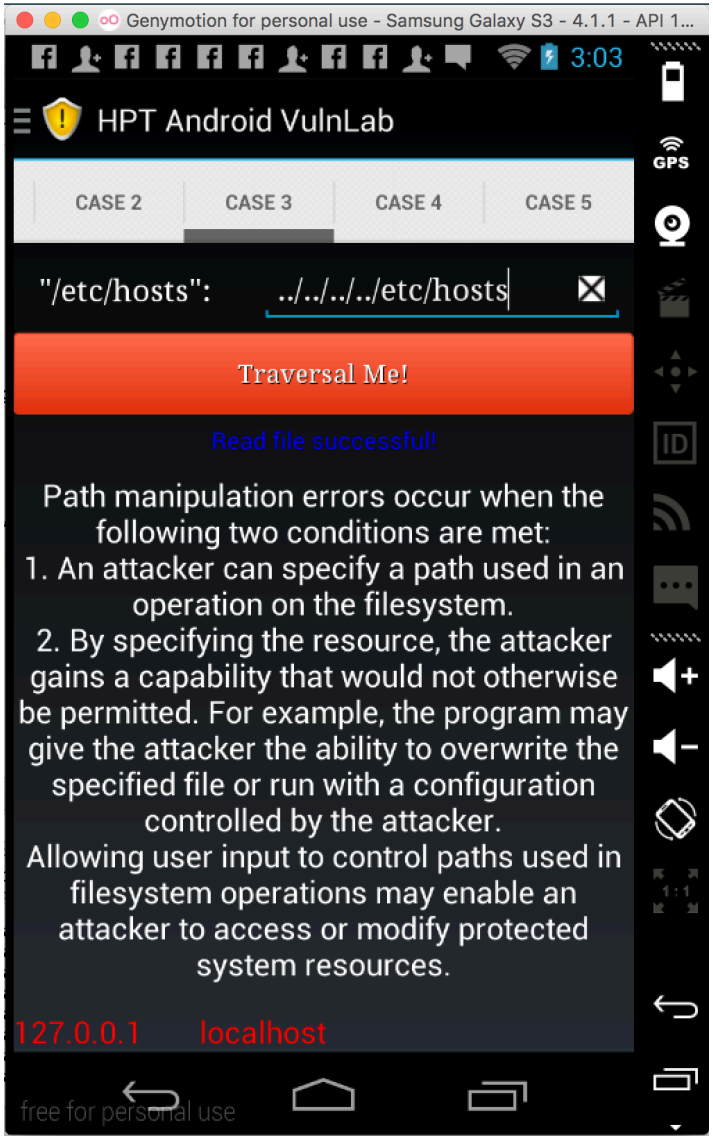
```
# Pass malicious path to FileHandle
ParcelFileDescriptor  myFilePointer =
getActivity().getContentResolver().openFileDescriptor(Uri.parse
(FileHandleProvider.PROVIDER_URI.toString()  + myUri), "r");
```

```
# FileHandle  -> Open File
File f = new File(getContext().getFilesDir(),  fullUri.getPath());
if (f.exists()) { return ParcelFileDescriptor.open(f,
ParcelFileDescriptor.MODE_READ_ONLY);  }
```

Data Validation

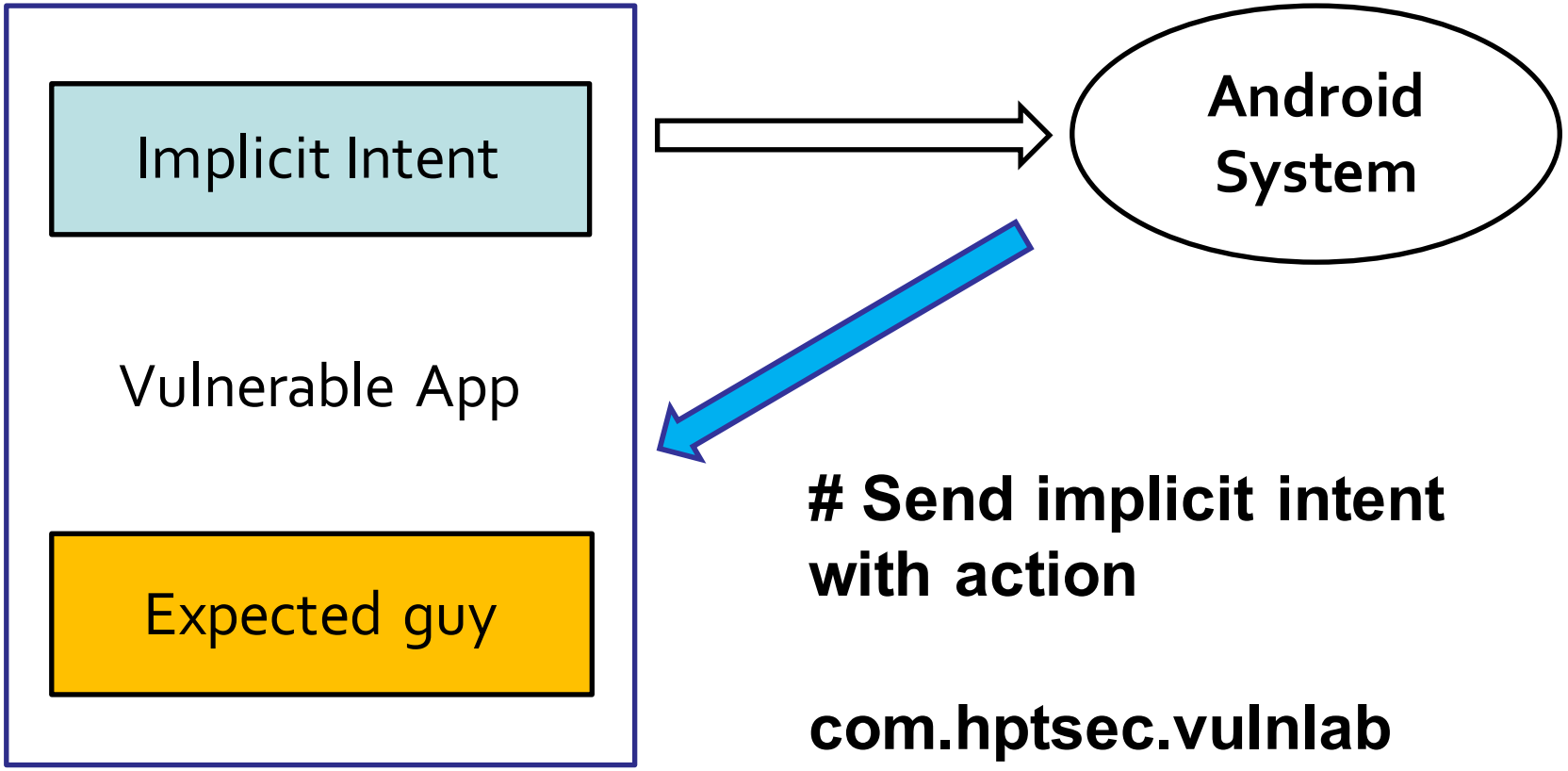
Path Traversal - PoC

Aww! Noo. Pray for him!



When implicit Intent call

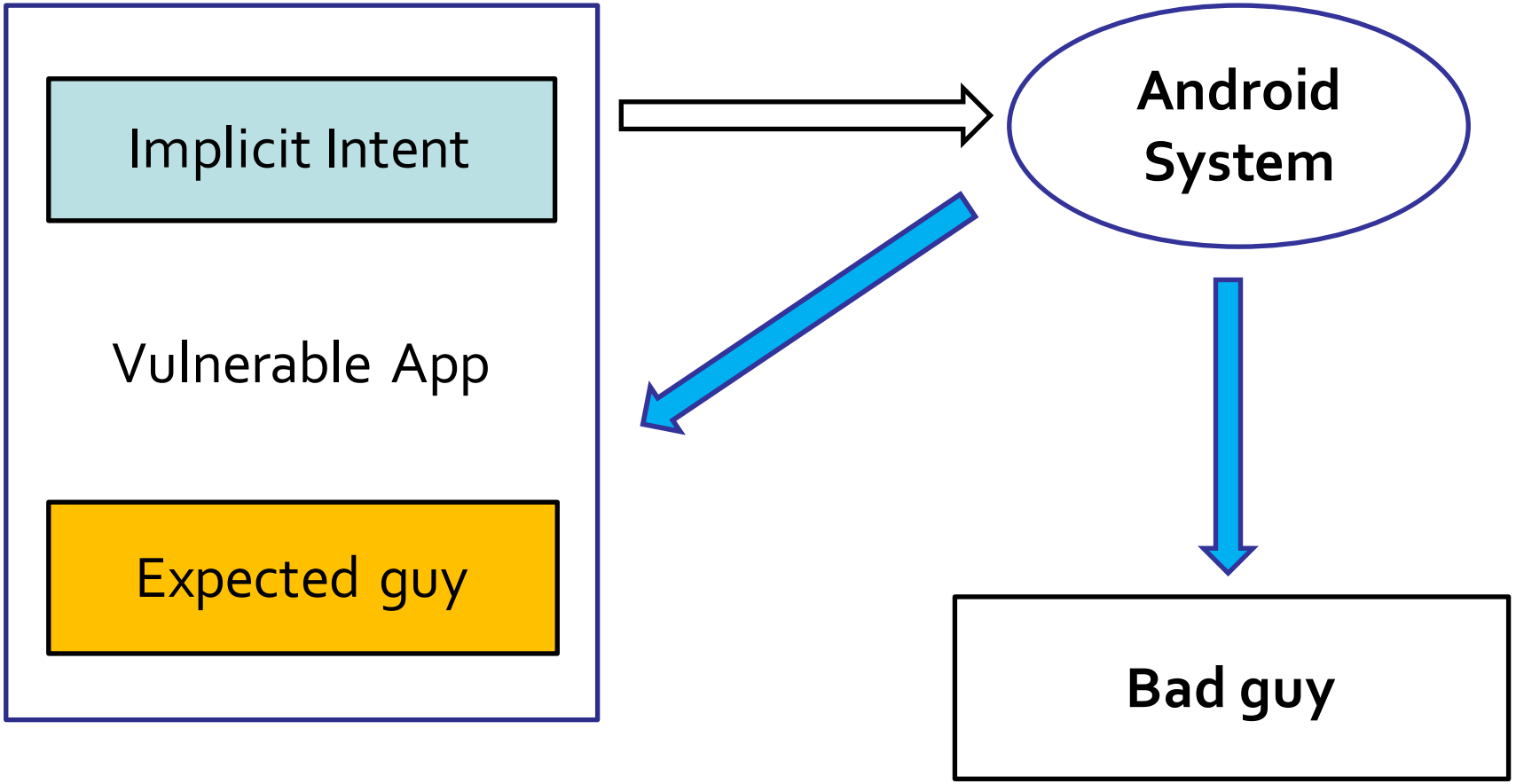
- Case #1



Secure Inter-process Communication

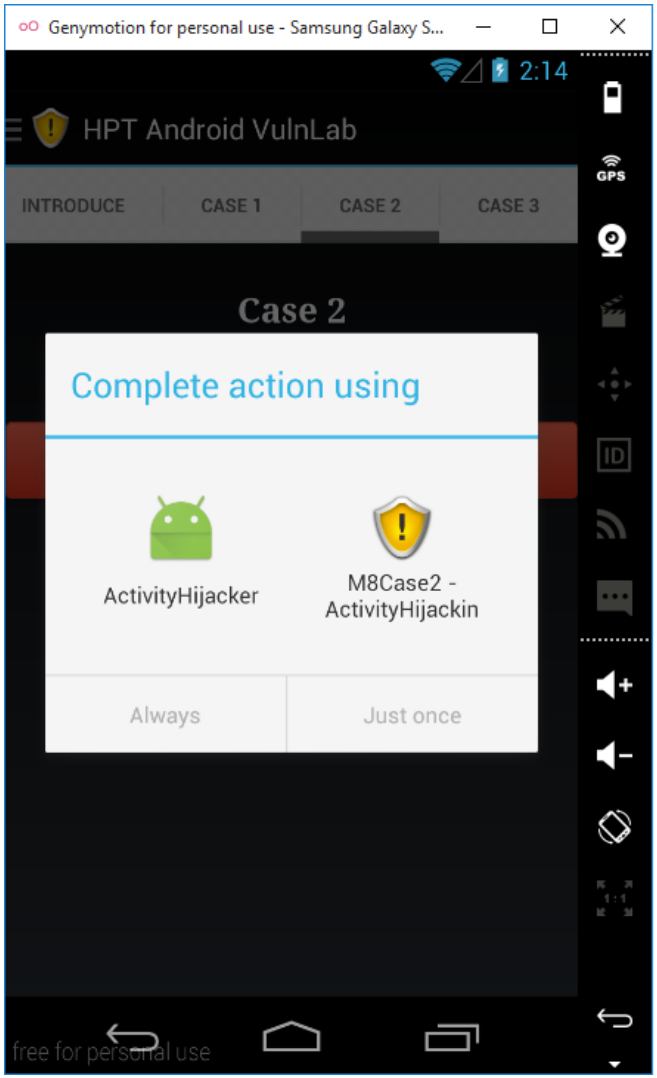
End-user doesn't know about this!

- Case #2



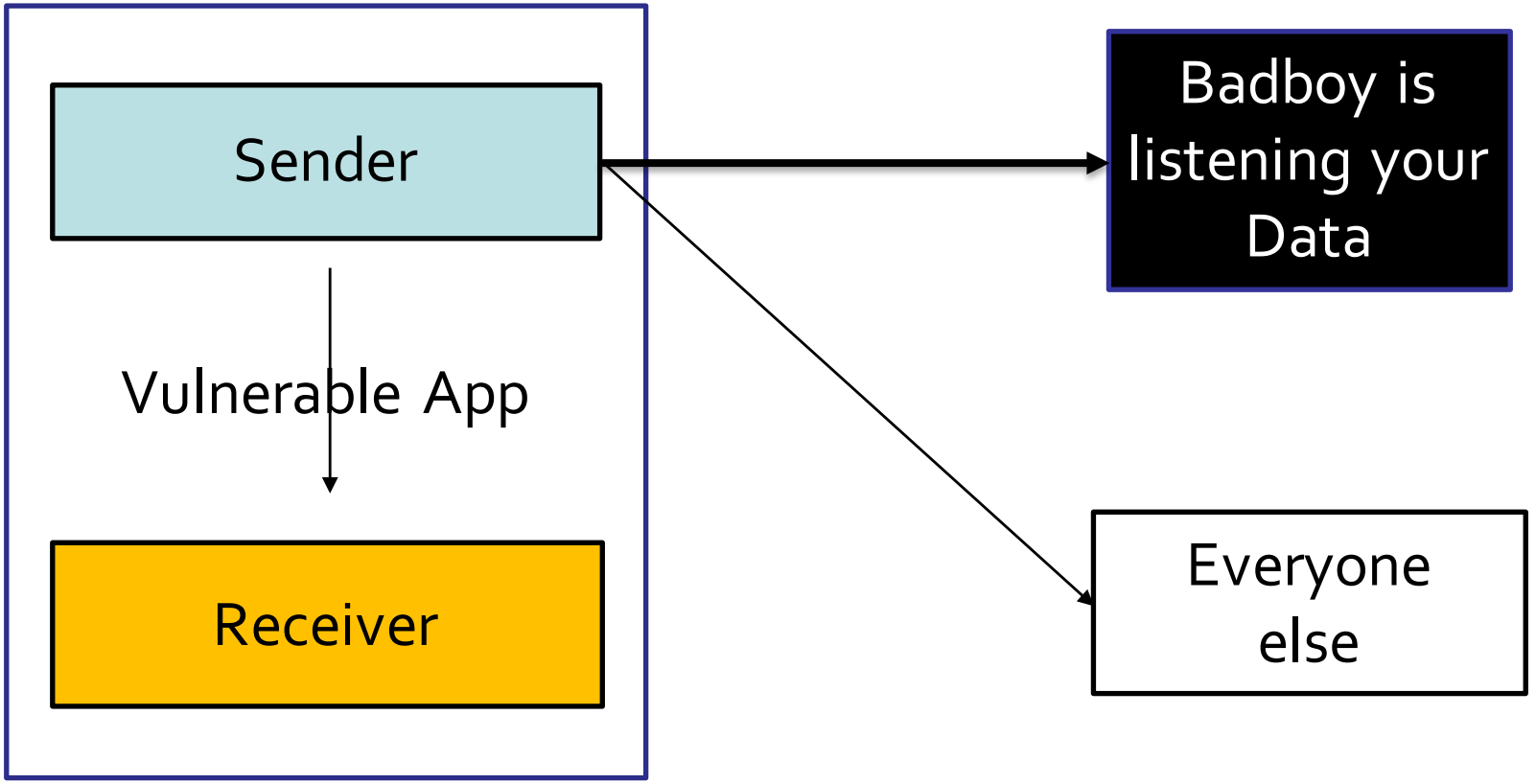
Secure Inter-process Communication

At the street corner?

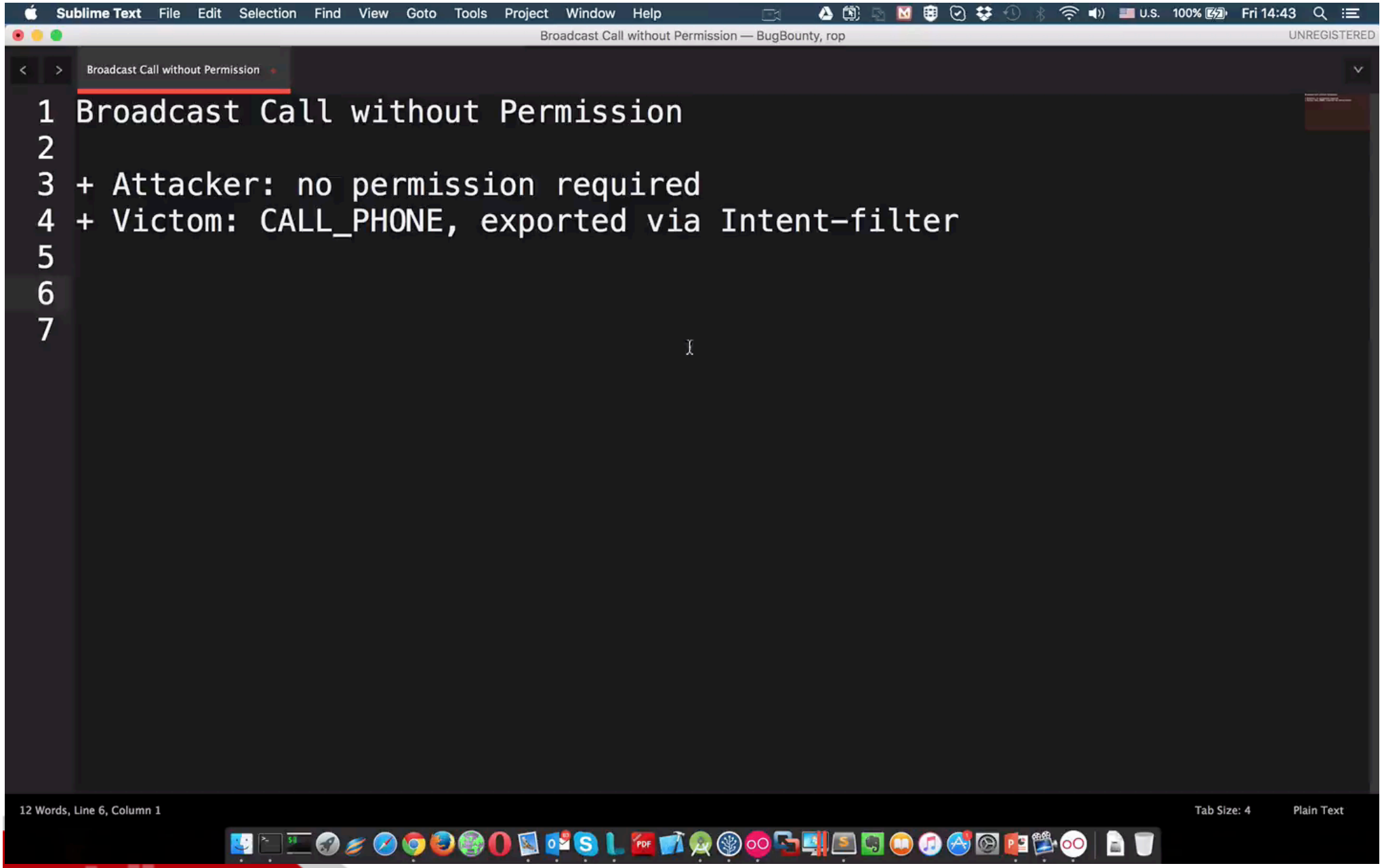


Secure Inter-process Communication

Don’t put sensitive info in broadcast intent

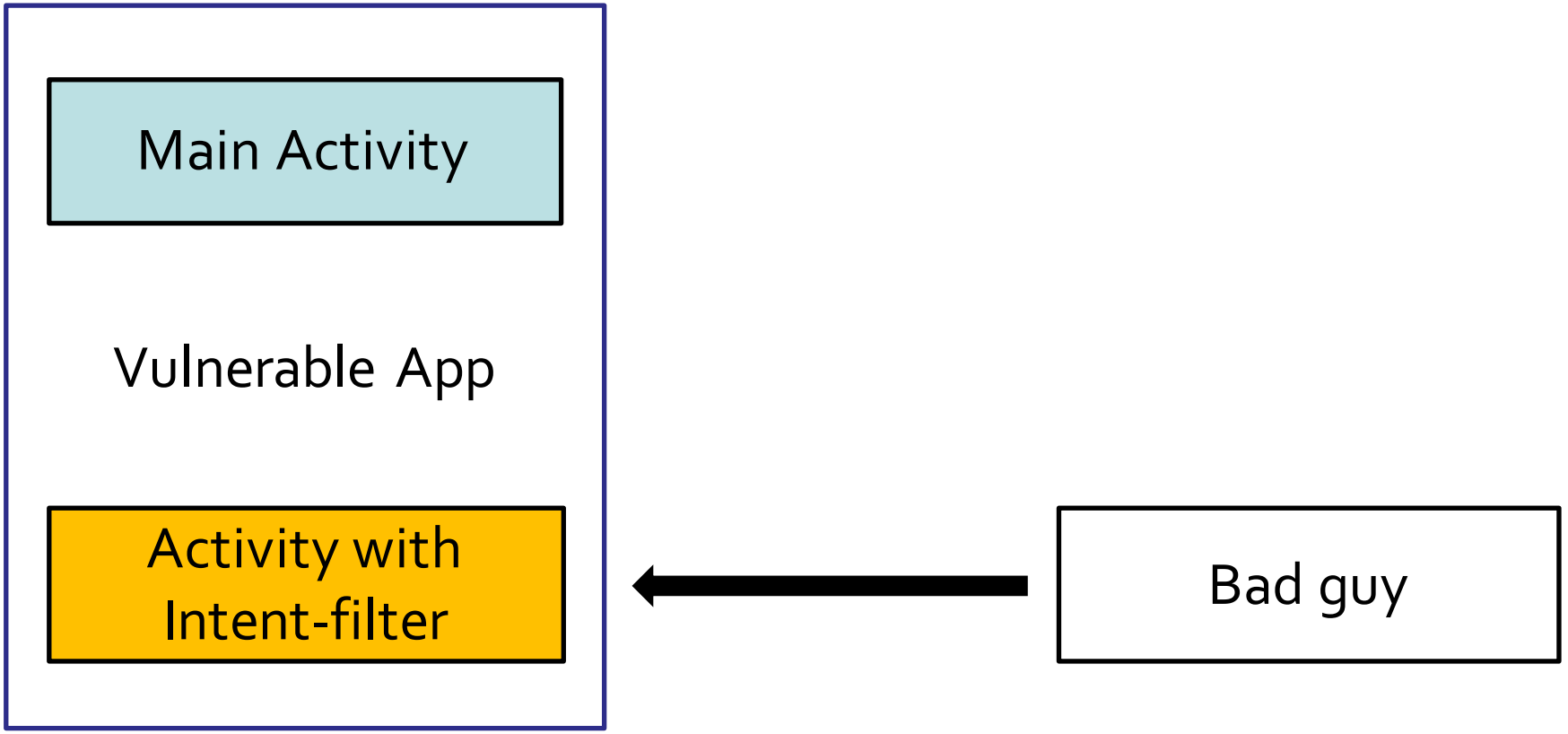


Unprotected Broadcast Receiver PoC



Secure Activity/Service

- Intent-filter will public every components use it



Best Practices for IPC

- Define your own custom permission with "**Signature**" level
- Apply it to each component in your AndroidManifest
- **Intent-filter** is not security boundary

➢ The day hacker went away

Secure Inter-process Communication

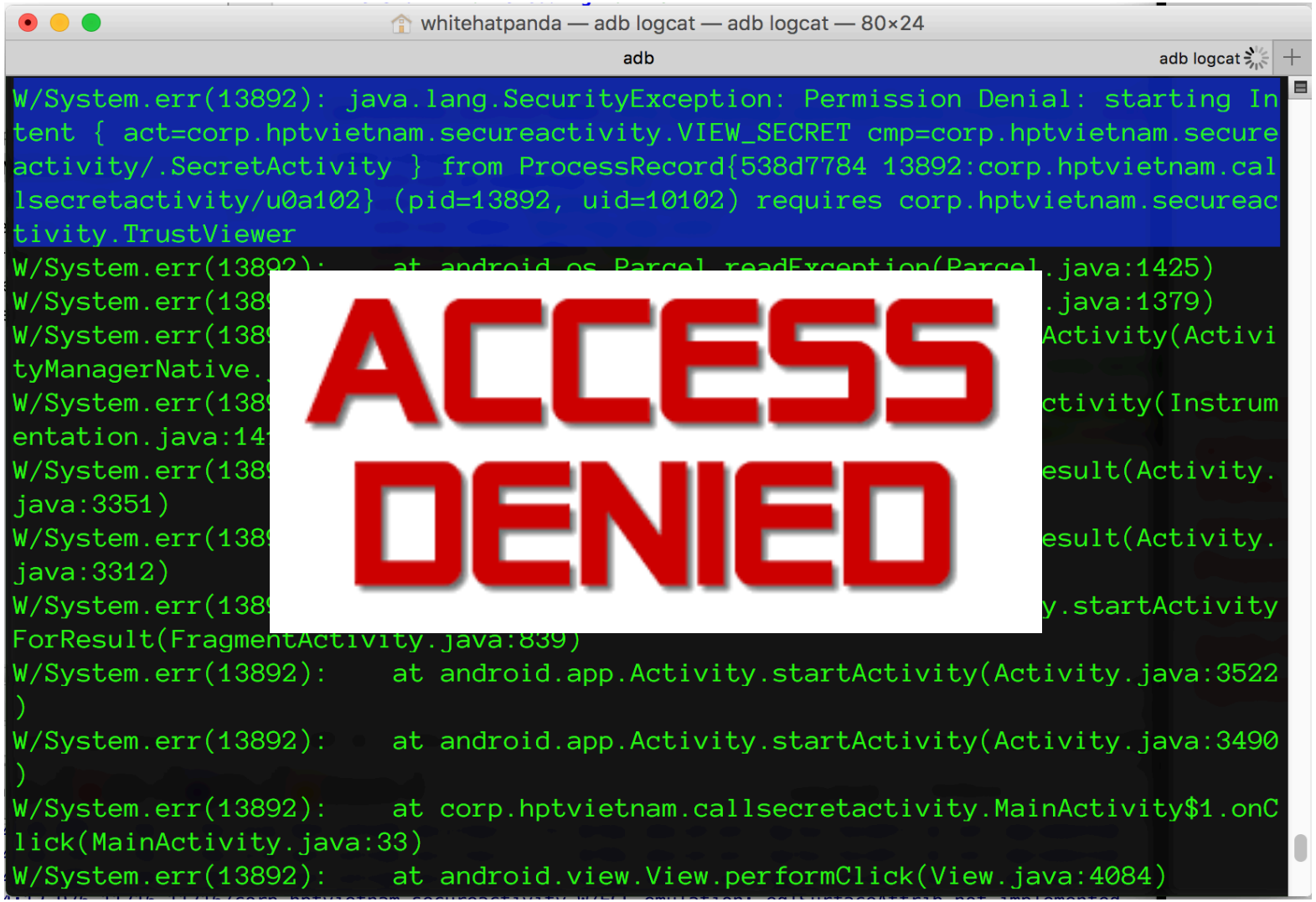
Protection Level = “**signature**”

activity_main.xml | AndroidManifest.xml | SecureActivity | app | activity_secret.xml | SecretActivity.java | MainActivity.java

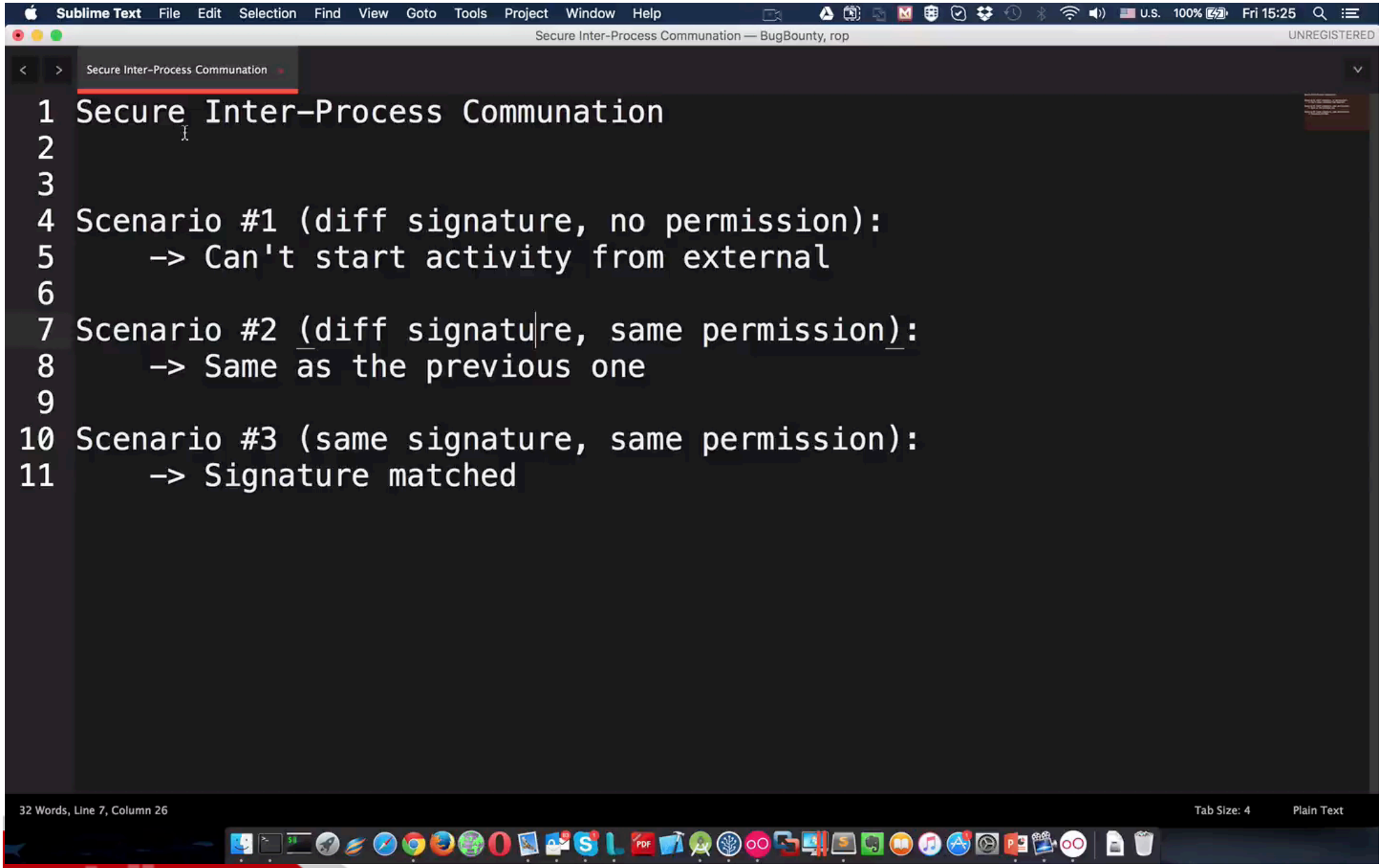
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="corp.hptvietnam.secureactivity" >

    <!-- Step 1: Define Permission with <permission> tag with protectionLevel="signature" -->
    <permission android:name="corp.hptvietnam.secureactivity.TrustViewer" android:protectionLevel="signature" ></permission>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity"...>

        <!-- Define secret activity with intent-filter (implicit exported), but protected with android:permission -->
        <activity android:name=".SecretActivity" android:permission="corp.hptvietnam.secureactivity.TrustViewer"> <!--android:exported="true" -->
            <intent-filter>
                <action android:name="corp.hptvietnam.secureactivity.VIEW_SECRET"></action>
                <category android:name="ANDROID.INTENT.CATEGORY.DEFAULT"></category>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Secure Inter-process Communication

No permission then go way



Secure IPC Demo



IPC Protection Summary

Type	Concepts
Private	• Exported=false • Explicit intent call (by class name) • Validate data on send & receiver context • No Intent-Filter (if exists, enforce permission)
Public	• Exported=true • Validate every Intent's Extra on receive • Do not perform critical action based on these info
In-house	• Exported=true or via Intent-Filter • Use custom permission with protectionLevel="Signature" • Enforce permission for each component

Binary Protection

- Code Obfuscation
- Integrity Checksum
- Release Config

Binary Protection

Code obfuscation

- Enable proguard in release mode

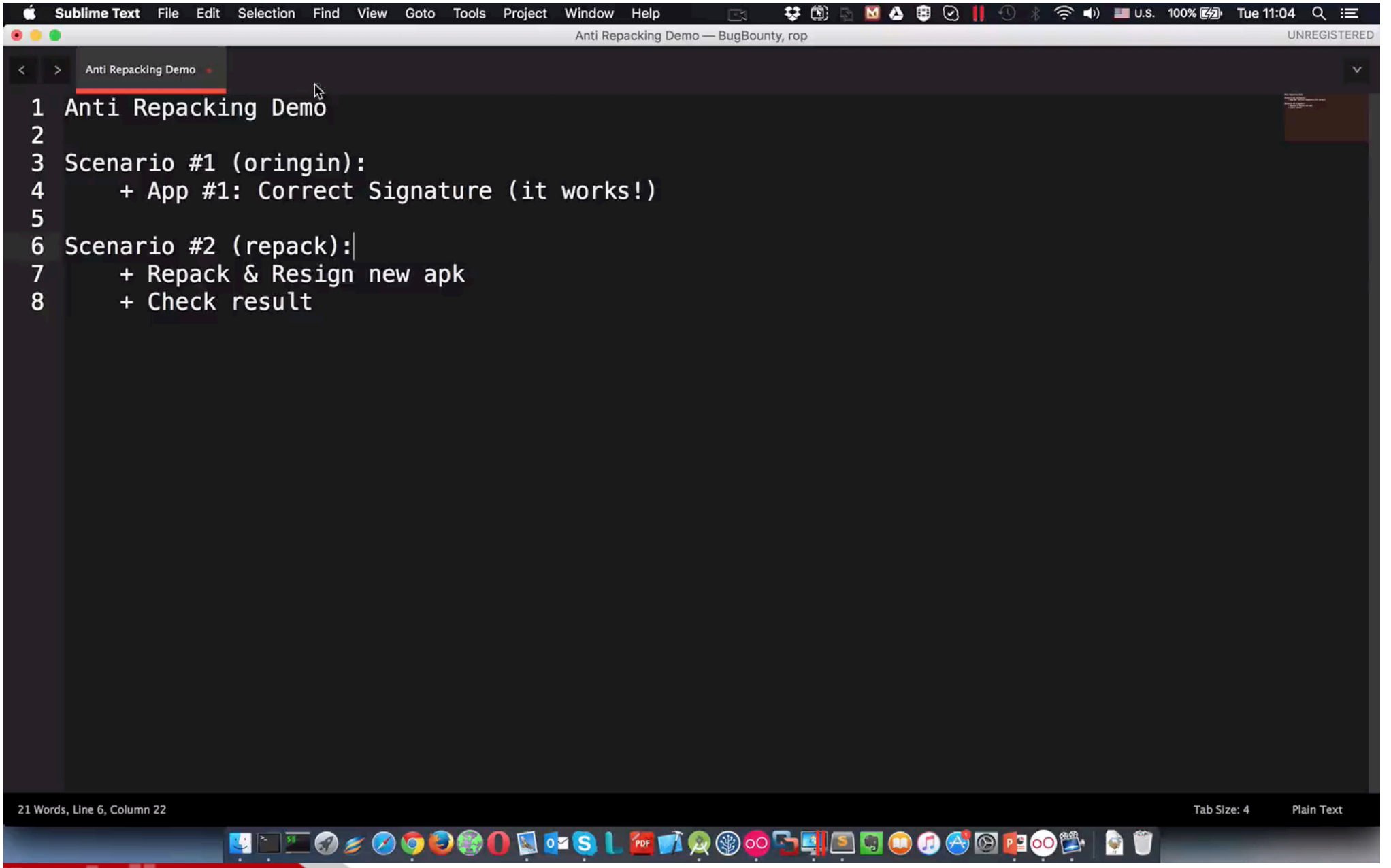
```
signingConfigs {
        releaseConfig {
                storeFile file("C:\\.android\\your.keystore")
                storePassword "android"
                keyAlias "androiddebugkey"
                keyPassword "android"
} }
buildTypes {
        release {
                minifyEnabled true
                proguardFile 'proguard-android.txt'
                signingConfig signingConfigs.releaseConfig
} }
```

Integrity Checksum

- Almost application doesn't know it was repacked or not

```
pkInfo =
mContext.getPackageManager().getPackageInfo(mContext.getPackage
Name(), PackageManager.GET_SIGNATURES);
for (Signature signature : pkInfo.signatures) {
    MessageDigest md = MessageDigest.getInstance("SHA");
        md.update(signature.toByteArray());
        result = Base64.encodeToString(md.digest(), Base64.DEFAULT);
}
```

Anti Repack Demo



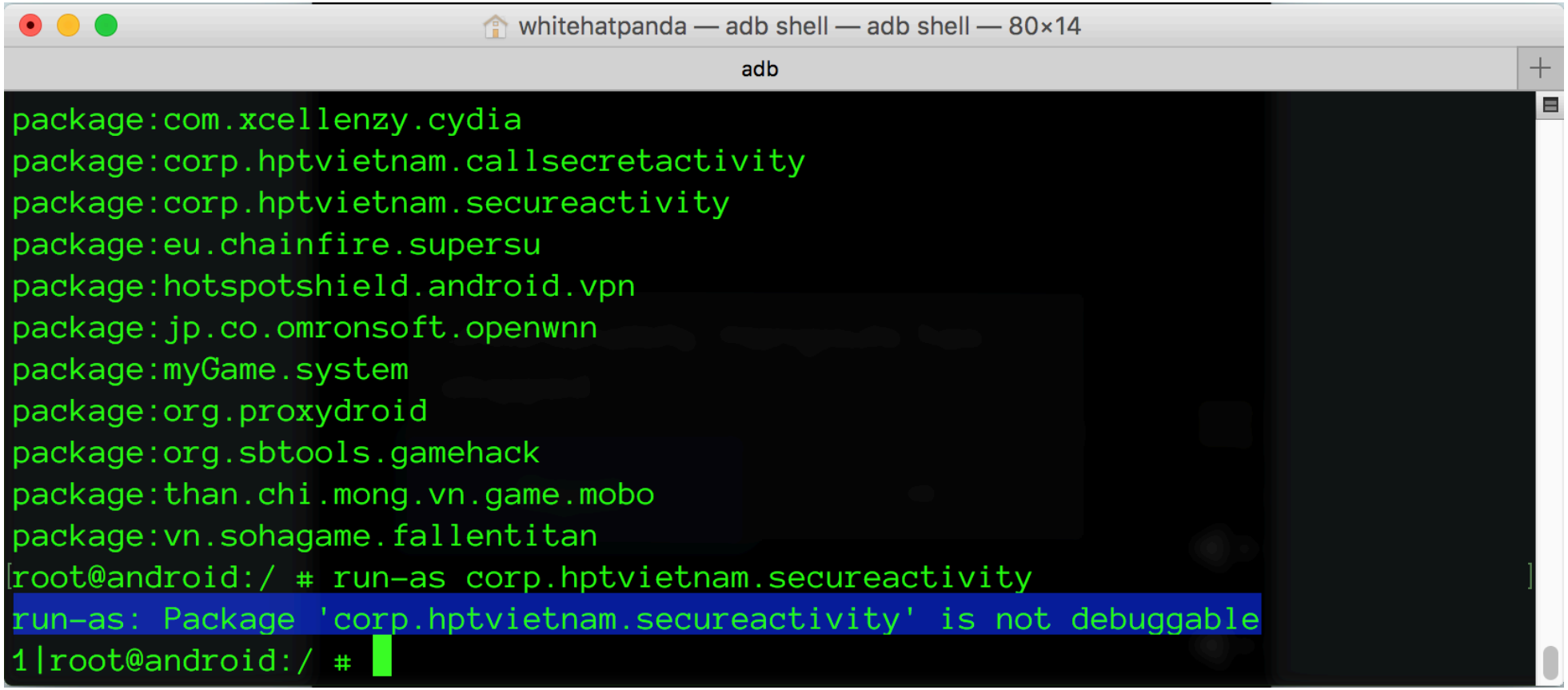
Binary Protection

Release Config - Make it harder to Exploit!

- Something to do before publish your app
  - Android:debuggable="false"
  - Android:allowBackup="false"

Binary Protection

When debuggable = False



References

Developer documentation on security

- **Presentation & Documents:**

https://github.com/whitehatpanda/VN-SecurityDay-2015

  - ✓ For references

- **Android Secure Coding - EN:**

http://www.jssec.org/dl/android_securecoding_en.pdf

  - ✓ Describes how various secure coding way to prevent hacker's eyes

- **Security and Permissions:**

http://developer.android.com/guide/topics/security/permissions.html

  - ✓ SDK documentation on the Android permission system

hpt
knowing IT

Q & A

knowing IT


Thank You
hpt
knowing IT