

안드로이드 앱 모의해킹

201812745 김종원

2021.08.05

목 차

1. 취약점 분석

2. InsecureBankv2

3. 모의해킹

취약점 분석

- 분석 방법

- 정적 분석

- 프로그램을 실행하지 않고 검사/분석 하는 방법
- 코드전체 분석 가능
- 정확한 코드 위치 파악
- 실행해야만 알 수 있는 정보 파악 불가

- 동적 분석

- 프로그램을 직접 실행해서 분석하는 작업
- 의심되는 어떠한 행위에 대해서 브레이크포인트를 주면서 분석

취약점 분석

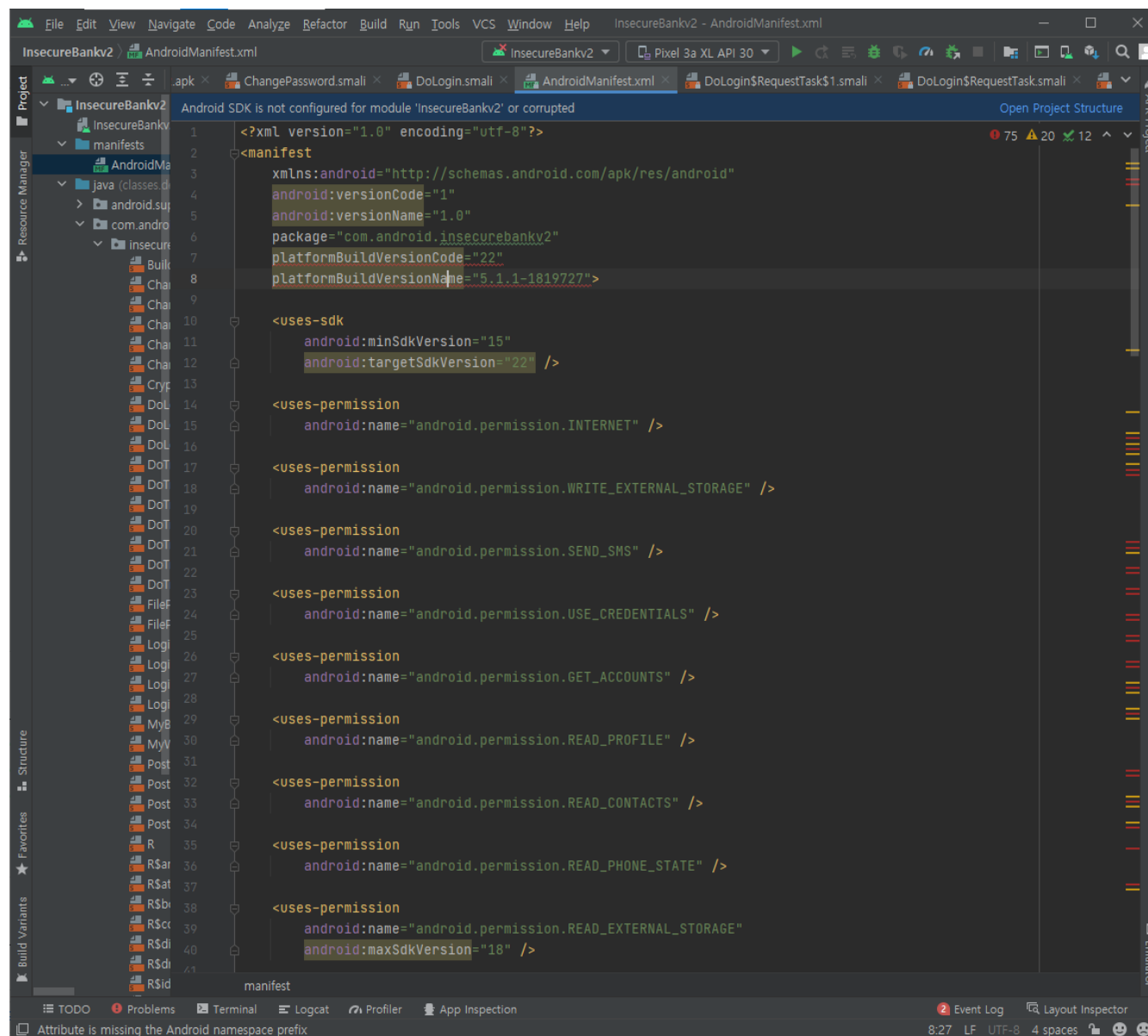
- **Android**

- Linux 기반 운영체제
- JAVA 라이브러리와 달빅 가상머신으로 구성
- 각 애플리케이션마다 한 개의 가상머신 사용
- 액티비티, 서비스, 콘텐츠 프로바이더, 브로드캐스트 리시버 등으로 구성
 - Activity : 애플리케이션과 사용자의 상호작용(UI구성요소)
 - Services : 백그라운드로 실행(음원 스트리밍)
 - Content Provider : 애플리케이션 공유 공간(연락처 앱에서 정보 가져오기)
 - BroadCast Receiver : 실시간으로 시스템 상태를 확인(배터리 충전)
- **분석 시 Manifest, Classes 중요**

취약점 분석

- Android - Manifest

안드로이드 시스템이 앱의 코드 실행 전에 확보해야 하는 앱에 대한 필수 정보를 시스템에 제공하는 목록



```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:versionCode="1"
    android:versionName="1.0"
    package="com.android.insecurebankv2"
    platformBuildVersionCode="22"
    platformBuildVersionName="5.1.1-1819727">

    <uses-sdk
        android:minSdkVersion="15"
        android:targetSdkVersion="22" />

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

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

    <uses-permission
        android:name="android.permission.SEND_SMS" />

    <uses-permission
        android:name="android.permission.USE_CREDENTIALS" />

    <uses-permission
        android:name="android.permission.GET_ACCOUNTS" />

    <uses-permission
        android:name="android.permission.READ_PROFILE" />

    <uses-permission
        android:name="android.permission.READ_CONTACTS" />

    <uses-permission
        android:name="android.permission.READ_PHONE_STATE" />

    <uses-permission
        android:name="android.permission.READ_EXTERNAL_STORAGE"
        android:maxSdkVersion="18" />

</manifest>
```

- 앱의 패키지 이름

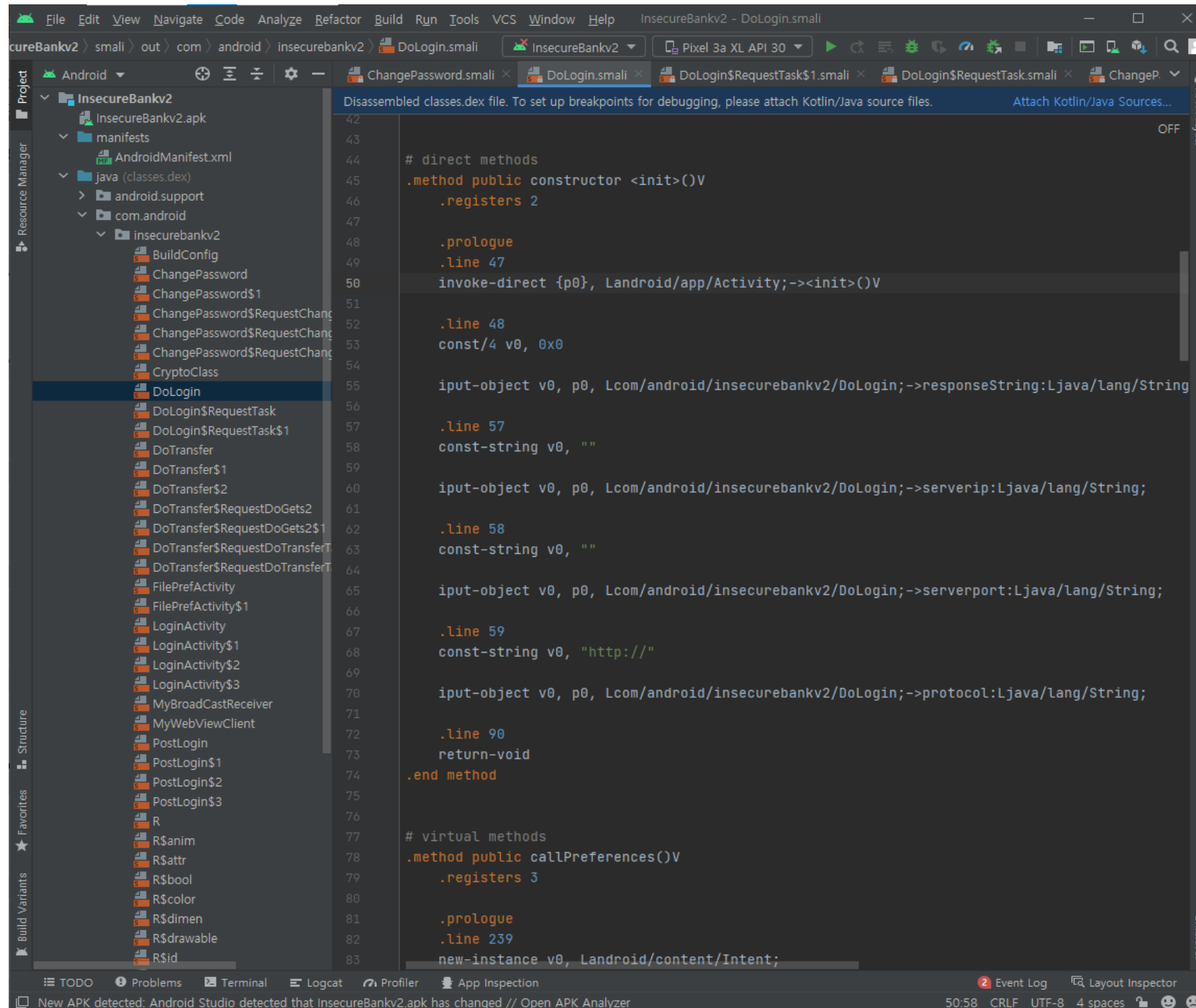
- 앱에서 사용되는 컴포넌트

- 권한

- 앱에서 요구하는 하드웨어와 소프트웨어 특징

취약점 분석

- Android - classes



취약점 분석

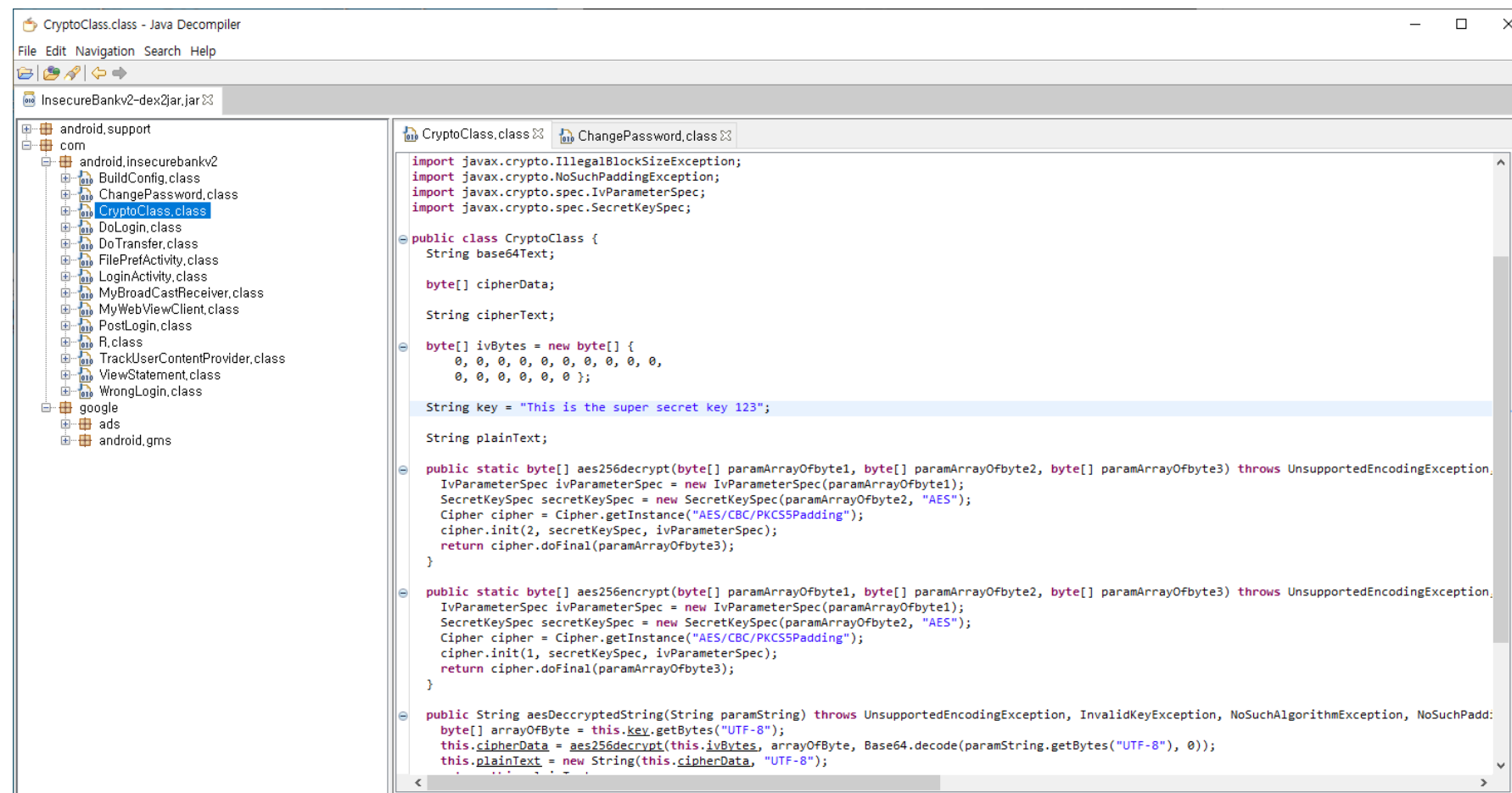
- apk 디컴파일 – dex2jar

```
C:\Users\JB\Desktop\dex2jar-2.0>d2j-dex2jar.bat C:\Users\JB\Desktop\InsecureBankv2.apk  
dex2jar C:\Users\JB\Desktop\InsecureBankv2.apk -> .\InsecureBankv2-dex2jar.jar
```



InsecureBankv2-dex2jar.jar

2021-08




jd-gui에서 실행한 화면

취약점 분석


- SUPER Android Analyzer

```
C:\Users\JB\Downloads\super>super-analyzer.exe -v C:\Users\JB\Desktop\mobile\app\insecureBankv2.apk
```

```
formation being disclosed.  
Possible high criticality vulnerability found!: This applications is performing checks for rooted device. This could be  
use to execute specific code if the device is rooted to take control of it.  
Possible high criticality vulnerability found!: The application could execute system command.  
  
The source code was analyzed correctly!  
  
Starting report generation.  
First we'll create the results folder.  
Results folder created. Time to create the reports.  
The application HTML results exist. But no more...  
Starting HTML report generation. First we create the file.  
The report file has been created. Now it's time to fill it.  
HTML report generated.  
Everything went smoothly, you can now check all the results.  
  
I will now analyze myself for vulnerabilities...  
Nah, just kidding, I've been developed in Rust!  
  
C:\Users\JB\Downloads\super>_
```

<< super > results > com.android.insecurebankv2 >		▼	🔄	🔍 com.android.insecurebankv2 검색	
이름	수정한 날짜	유형	크기		
css	2021-08-03 오후 7:01	파일 폴더			
img	2021-08-03 오후 7:01	파일 폴더			
js	2021-08-03 오후 7:01	파일 폴더			
src	2021-08-03 오후 7:01	파일 폴더			
 index.html	2021-08-03 오후 7:01	Chrome HTML D...	101KB		

취약점 분석



SUPER Android Analyzer Report

This is the vulnerability report for the android application *com.android.insecurebankv2*.
Report generated on Tue, 3 Aug 2021 19:01:46 +0900 with SUPER Android Analyzer 0.5.1.

Application data:

Package: com.android.insecurebankv2
Version: 1.0
Version number: 1
Minimum SDK version: 15 (Android 4.0.3 *Ice Cream Sandwich MR1*)
Target SDK: None
Fingerprints:
MD5: 5ee4829065640f9c936ac861d1650ffc
SHA-1: 80b53f80a3c9e6bfd98311f5b26ccddcd1bf0a98
SHA-256: b18af2a0e44d7634bbcdf93664d9c78a2695e050393fcfbb5e8b91f902d194a4
[Check source code](#)

Total vulnerabilities found: 11

- Critical: 1 ⇒
- High: 7 ⇒
- Medium: 1 ⇒
- Low: 2 ⇒
- Warnings: 29 ⇒

Vulnerabilities:

Critical vulnerabilities: 1

C01: +

Label: Manifest Debug

High criticality vulnerabilities: 1

취약점 분석

C01: -

Label: Manifest Debug

Description: The application is in debug mode. This allows any malicious person to inject arbitrary code in the application. This option should only be used while in development.

File: [AndroidManifest.xml](#)

Line : 17

Affected code:

```
13 <uses-permission android:name="android.permission.READ_CALL_LOG" />
14 <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
15 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
16 <uses-feature android:glEsVersion="131072" android:required="true" />
17 <application android:allowBackup="true" android:label="@com.android.insecurebankv2:string/app_name"
18     <activity android:name="com.android.insecurebankv2.LoginActivity" android:label="@com.android.insecurebankv2:string/app_name"
19         <intent-filter>
20             <action android:name="android.intent.action.MAIN" />
21             <category android:name="android.intent.category.LAUNCHER" />
22         </intent-filter>
23     </activity>
24 </application>
25 </manifest>
```

```
16 <uses-feature android:glEsVersion="131072" android:required="true" />
17 <application android:allowBackup="true" android:label="@com.android.insecurebankv2:string/app_name" android:debuggable="true"
18     <activity android:name="com.android.insecurebankv2.LoginActivity" android:label="@com.android.insecurebankv2:string/app_name"
19         <intent-filter>
20             <action android:name="android.intent.action.MAIN" />
21             <category android:name="android.intent.category.LAUNCHER" />
22         </intent-filter>
23     </activity>
24 </application>
25 </manifest>
```

취약점 분석

M01: -

Label: Allows Backup

Description: This option allows backups of the application data via adb. Malicious people with physical access could use adb to get private data of your app into their PC.

File: AndroidManifest.xml

Line : 17

Affected code:

```
13 <uses-permission android:name="android.permission.READ_CALL_LOG" />
14 <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
15 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
16 <uses-feature android:glEsVersion="131072" android:required="true" />
17 <application android:allowBackup="true" android:label="@com.android.insecurebankv2:string/app_name"
18     <activity android:name="com.android.insecurebankv2.LoginActivity" android:label="@com.android.insecurebankv2:string/app_name"
19         <intent-filter>
20             <action android:name="android.intent.action.MAIN" />
21             <category android:name="android.intent.category.LAUNCHER" />
```

취약점 분석

L02: -

Label: Unchecked output in Logs

Description: Sensitive information should never be logged since it can lead to that information being disclosed.

File: classes\com\android\winsecurebankv2\DoLogin.java

Line : 250

Affected code:

```
246     {
247         if (result.indexOf("Correct Credentials") == -1) {
248             break label426;
249         }
250         Log.d("Successful Login:", ", account=" + username + ":" + password);
251         saveCreds(username, password);
252         trackUserLogins();
253         paramString = new Intent(getApplicationContext(), PostLogin.class);
254         paramString.putExtra("uname", username);
```

InsecureBankv2

- 안드로이드 모의 해킹을 위해 만들어진 앱.

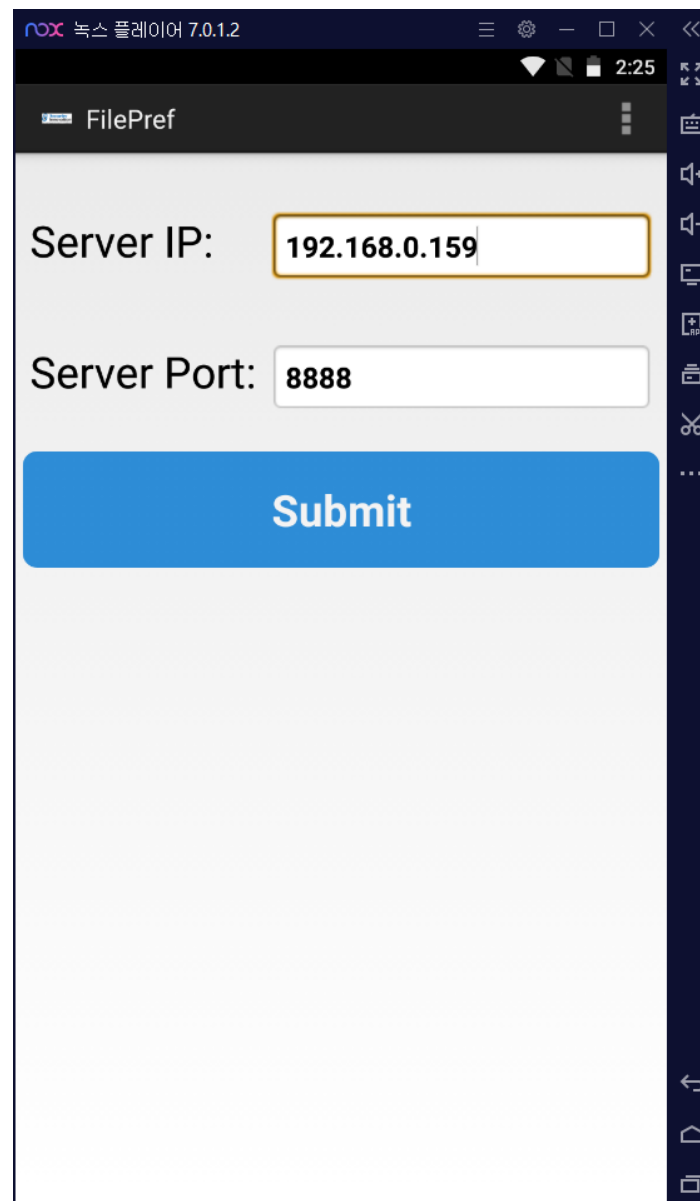
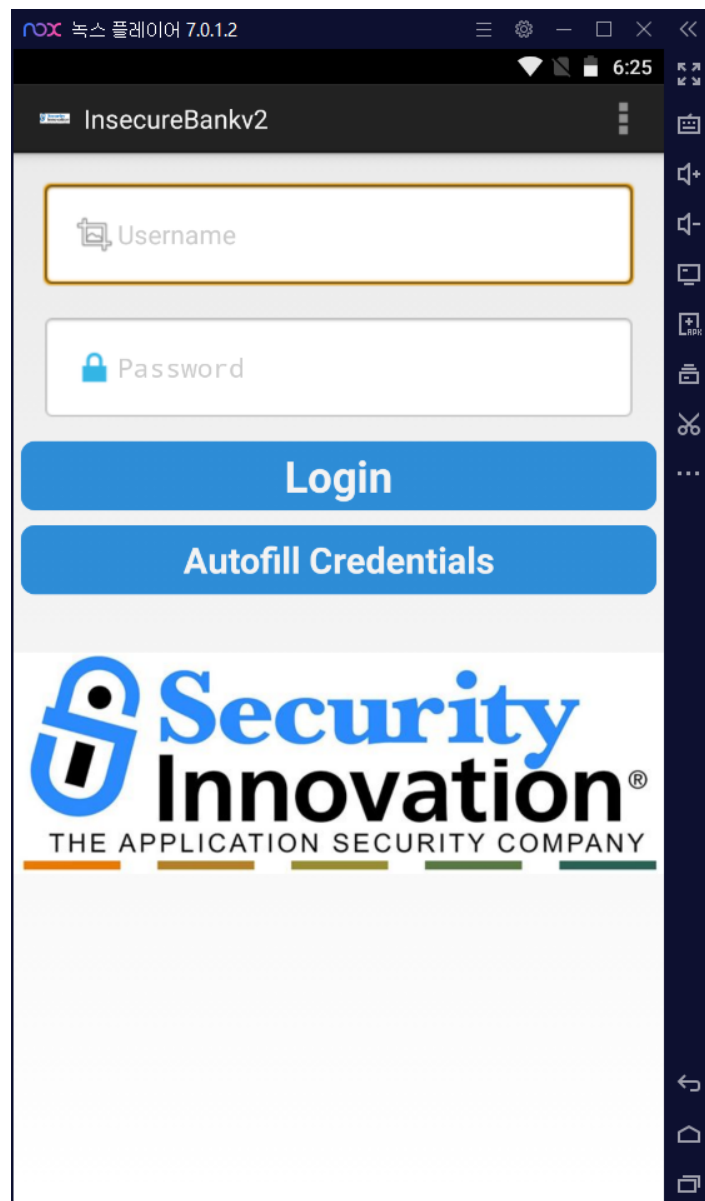
- Flawed Broadcast Receivers
- Intent Sniffing and Injection
- Weak Authorization mechanism
- Local Encryption issues
- Vulnerable Activity Components
- Root Detection and Bypass
- Emulator Detection and Bypass
- Insecure Content Provider access
- Insecure Webview implementation
- Weak Cryptography implementation
- Application Patching
- Sensitive Information in Memory
- Insecure Logging mechanism
- Android Pasteboard vulnerability
- Application Debuggable
- Android keyboard cache issues
- Android Backup vulnerability
- Runtime Manipulation
- Insecure SDCard storage
- Insecure HTTP connections
- Parameter Manipulation
- Hardcoded secrets
- Username Enumeration issue
- Developer Backdoors

모의해킹

- Anaconda
- NOX Player

ID : jack

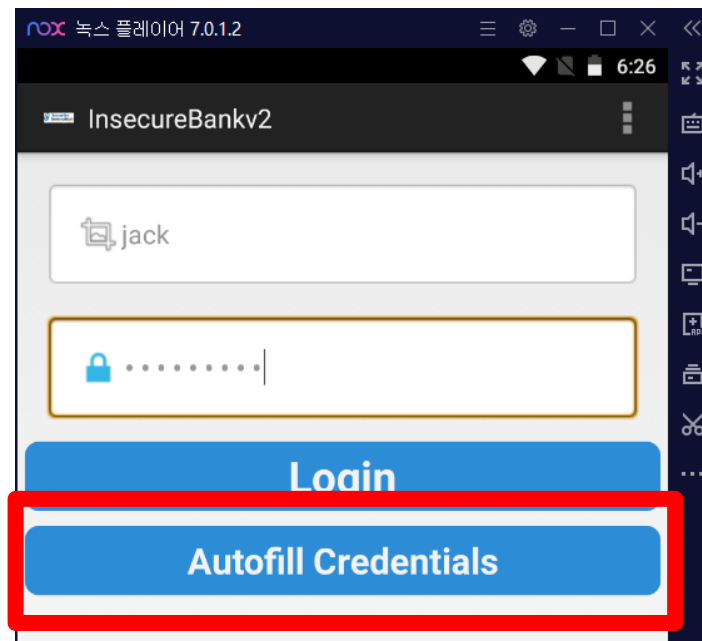
PW: Jack@123\$



모의해킹

- Local Encryption issue

Local Encryption issue란 중요정보를 단말기에 저장할 때 취약한 암호화 알고리즘을 사용하거나 평문으로 저장해서 발생하는 이슈

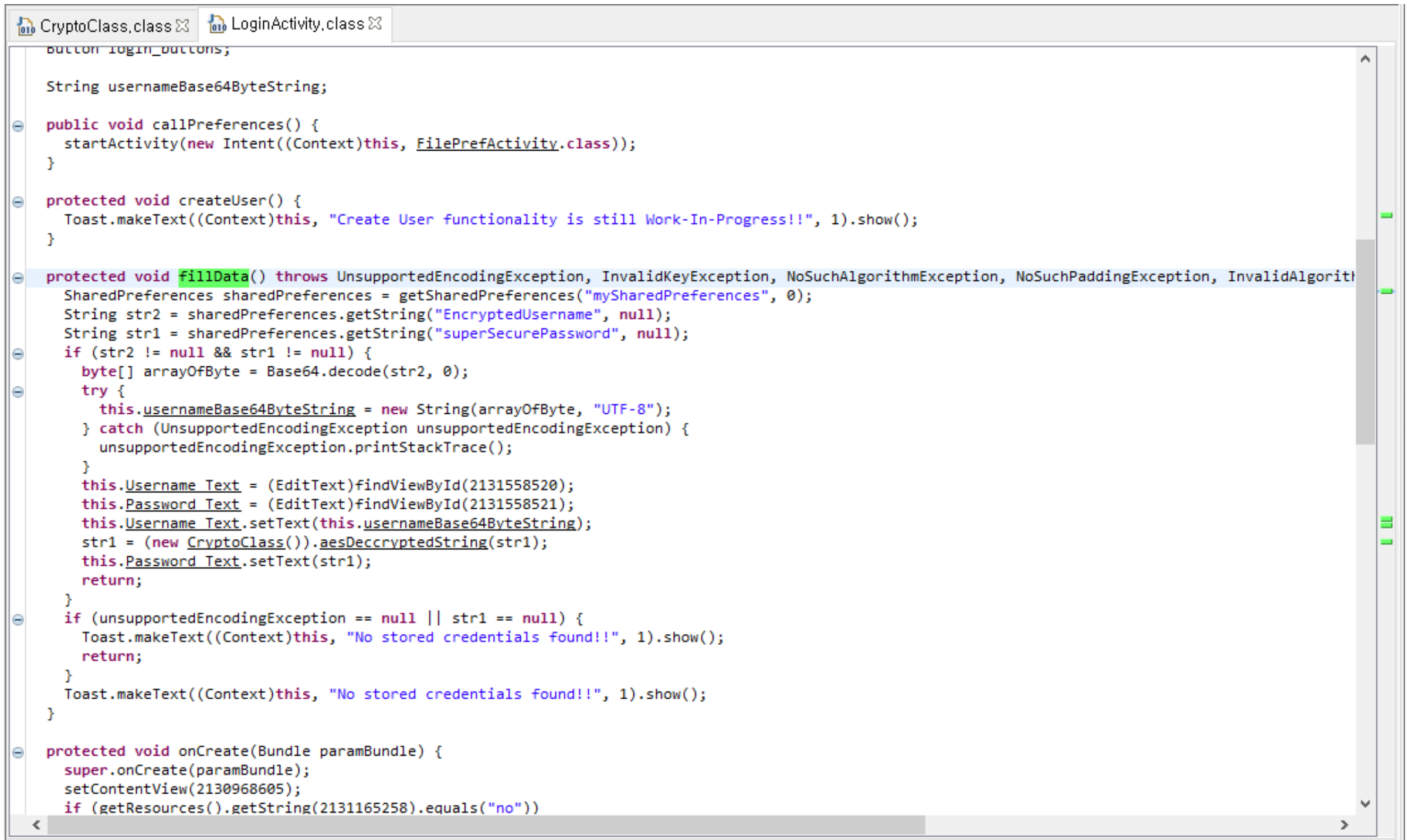


```
star2lte:/data/data/com.android.insecurebankv2/shared_prefs # cat mySharedPreferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="EncryptedUsername">amFjaw==&#13;&#10;    </string>
  <string name="superSecurePassword">v/sJpihDCo2ckDmLW5Uwiw==&#10;    </string>
</map>
star2lte:/data/data/com.android.insecurebankv2/shared_prefs #
```

SharedPreferences : DB나 서버에 저장하기에는 가벼운 정보(초기 설정값, 간단한 문자열)를 저장

모의해킹

- Local Encryption issue



```
button_login_buttons;

String usernameBase64ByteString;

public void callPreferences() {
    startActivity(new Intent((Context)this, FilePrefActivity.class));
}

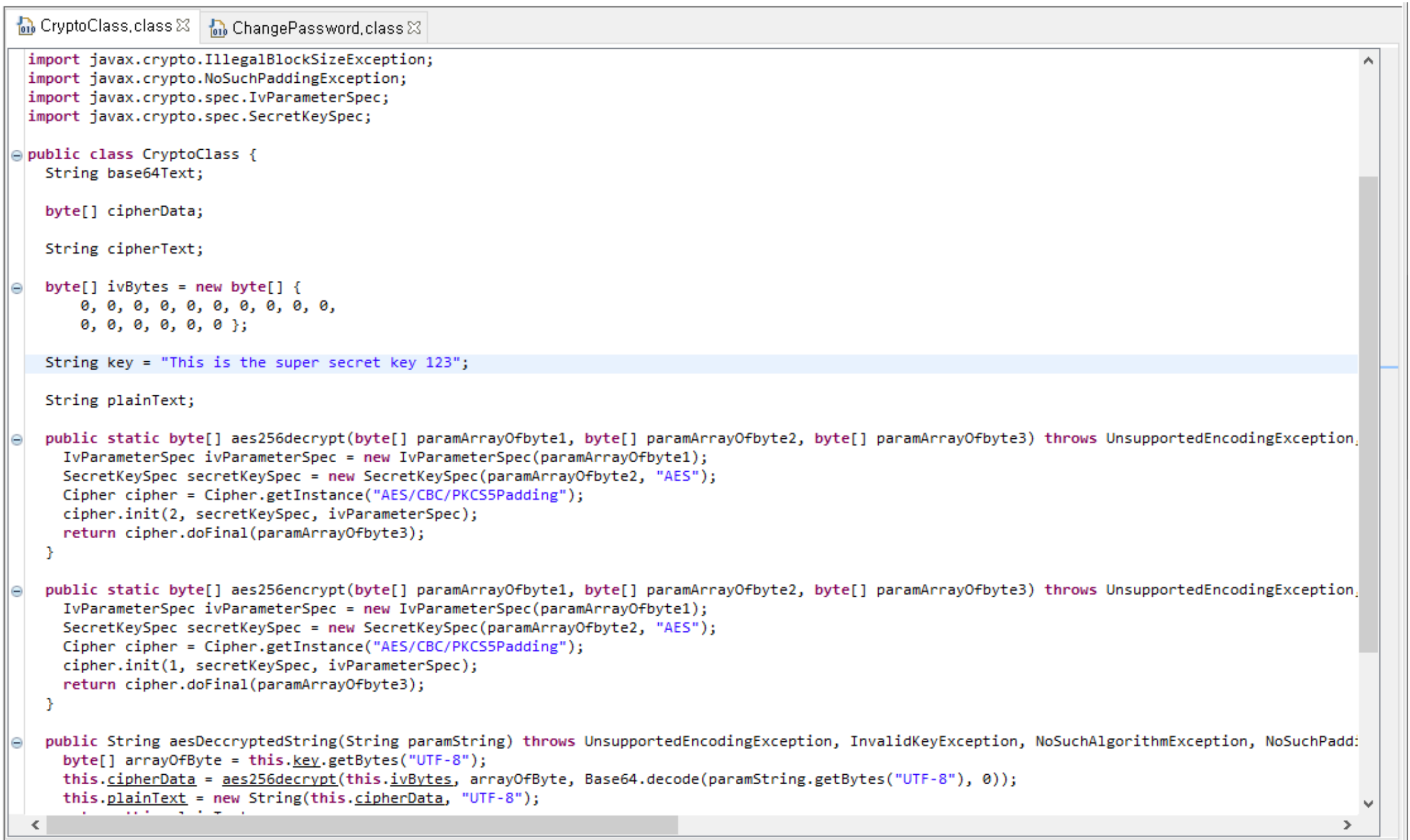
protected void createUser() {
    Toast.makeText((Context)this, "Create User functionality is still Work-In-Progress!!", 1).show();
}

protected void fillData() throws UnsupportedOperationException, InvalidKeyException, NoSuchAlgorithmException, NoSuchPaddingException, InvalidAlgorithmParameterException {
    SharedPreferences sharedPreferences = getSharedPreferences("mySharedPreferences", 0);
    String str2 = sharedPreferences.getString("EncryptedUsername", null);
    String str1 = sharedPreferences.getString("superSecurePassword", null);
    if (str2 != null && str1 != null) {
        byte[] arrayOfByte = Base64.decode(str2, 0);
        try {
            this.usernameBase64ByteString = new String(arrayOfByte, "UTF-8");
        } catch (UnsupportedEncodingException unsupportedEncodingException) {
            unsupportedEncodingException.printStackTrace();
        }
        this.Username_Text = (EditText)findViewById(2131558520);
        this.Password_Text = (EditText)findViewById(2131558521);
        this.Username_Text.setText(this.usernameBase64ByteString);
        str1 = (new CryptoClass()).aesDecryptedString(str1);
        this.Password_Text.setText(str1);
        return;
    }
    if (unsupportedEncodingException == null || str1 == null) {
        Toast.makeText((Context)this, "No stored credentials found!!", 1).show();
        return;
    }
    Toast.makeText((Context)this, "No stored credentials found!!", 1).show();
}

protected void onCreate(Bundle paramBundle) {
    super.onCreate(paramBundle);
    setContentView(2130968605);
    if (getResources().getString(2131165258).equals("no"))
```


모의해킹

- Local Encryption issue



```
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

public class CryptoClass {
    String base64Text;

    byte[] cipherData;

    String cipherText;

    byte[] ivBytes = new byte[] {
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0 };

    String key = "This is the super secret key 123";

    String plainText;

    public static byte[] aes256decrypt(byte[] paramArrayOfbyte1, byte[] paramArrayOfbyte2, byte[] paramArrayOfbyte3) throws UnsupportedEncodingException,
        IvParameterSpec ivParameterSpec = new IvParameterSpec(paramArrayOfbyte1);
        SecretKeySpec secretKeySpec = new SecretKeySpec(paramArrayOfbyte2, "AES");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(2, secretKeySpec, ivParameterSpec);
        return cipher.doFinal(paramArrayOfbyte3);
    }

    public static byte[] aes256encrypt(byte[] paramArrayOfbyte1, byte[] paramArrayOfbyte2, byte[] paramArrayOfbyte3) throws UnsupportedEncodingException,
        IvParameterSpec ivParameterSpec = new IvParameterSpec(paramArrayOfbyte1);
        SecretKeySpec secretKeySpec = new SecretKeySpec(paramArrayOfbyte2, "AES");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(1, secretKeySpec, ivParameterSpec);
        return cipher.doFinal(paramArrayOfbyte3);
    }

    public String aesDecryptedString(String paramString) throws UnsupportedEncodingException, InvalidKeyException, NoSuchAlgorithmException, NoSuchPaddingException {
        byte[] arrayOfByte = this.key.getBytes("UTF-8");
        this.cipherData = aes256decrypt(this.ivBytes, arrayOfByte, Base64.decode(paramString.getBytes("UTF-8"), 0));
        this.plainText = new String(this.cipherData, "UTF-8");
    }
}
```

모의해킹

- Local Encryption issue

Decode from Base64 format

Simply enter your data then push the decode button.

amFjaw==

< **DECODE** >

Decodes your data into the area t

jack

Input Text Format: ☒Base64 ☐Hex

Select Mode

ECB

Key Size in Bits

256

Enter Secret Key

This is the super secret key

Decrypt

AES Decrypted Output (Base64):

SmFja0AxMjMk

Decode to Plain Text

Jack@123\$

모의해킹

- Local Encryption issue

- ID 같은 경우 base64로 취약한 암호화를 사용하고 있기 때문에 패스워드처럼 AES같은 암호화 알고리즘을 적용 해야할 것.
- PASSWORD같은 경우는 AES를 이용해 암호화 했지만 jar 분석 결과, 대칭키가 그대로 드러나 있음. -> Key를 안전하게 저장하는 방법이 필요함.
- Ex) Key Store api 이용

모의해킹

- Insecure Logging

L02: -

Label: Unchecked output in Logs

Description: Sensitive information should never be logged since it can lead to that information being disclosed.

File: `classes\com\android\insecurebankv2\DoLogin.java`

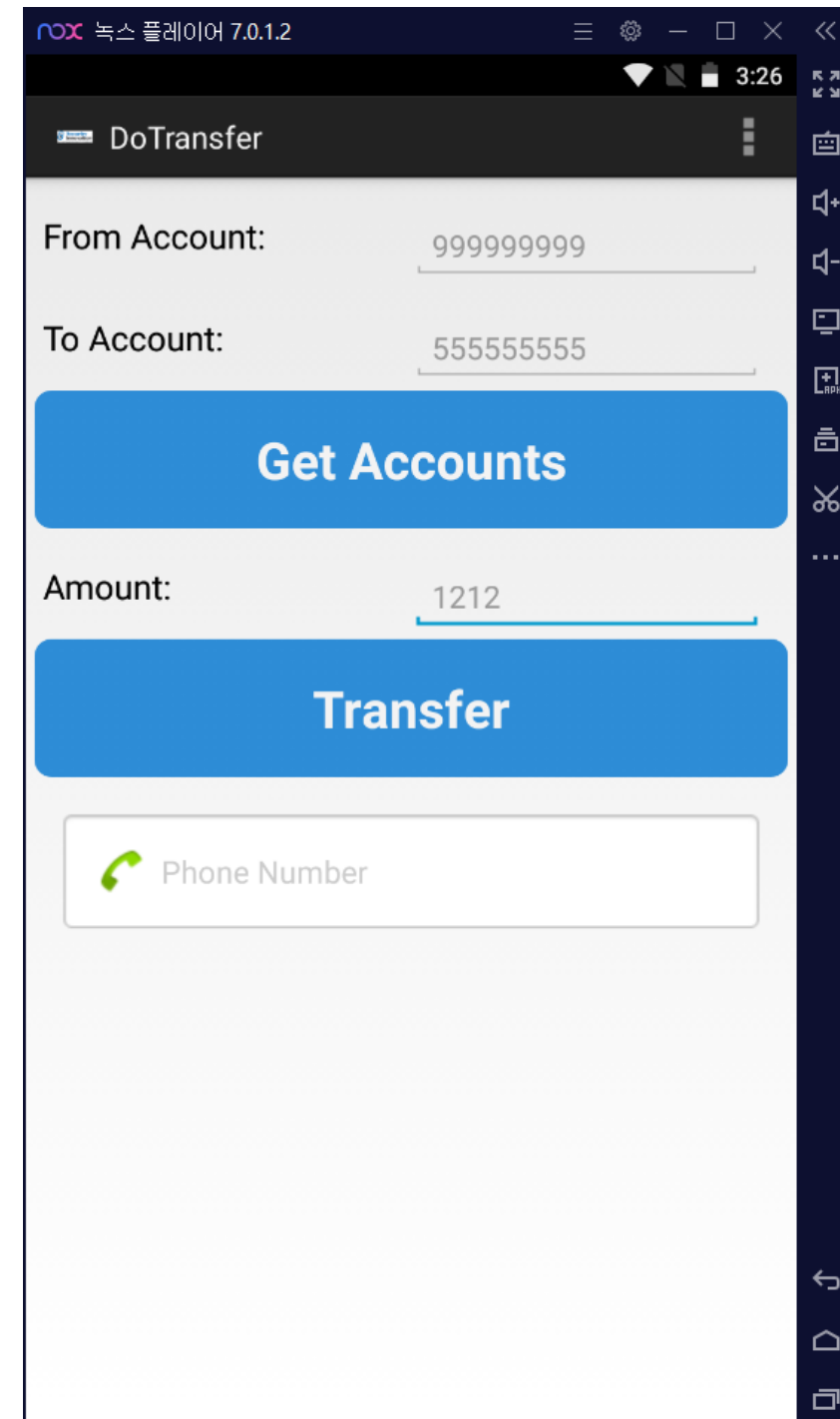
Line : 250

Affected code:

```
246     {
247         if (result.indexOf("Correct Credentials") == -1) {
248             break label426;
249         }
250         Log.d("Successful Login:", "", account=" + username + ":" + password);
251         saveCredentials(username, password);
252         trackUserLogins();
253         paramString = new Intent(getApplicationContext(), PostLogin.class);
254         paramString.putExtra("uname", username);
```

모의해킹

- Insecure Logging



모의해킹

- Insecure Logging

```
08-04 14:28:30.456 9347 20987 D Successful Login:: , account=jack:Jack@123$  
08-04 14:29:36.258 9347 9347 I System.out: Message:Success From:999999999 To:555555555 Amount:1212
```

- 계정정보나 계좌이체 정보 같이 중요 정보를 평문으로 로그에 남길 시 악성코드 감염이나 단말기 분실로 로그 정보가 공격자에게 로그정보가 유출 될 수 있기 때문에 위험.
- 따라서 개발 중에 로그 확인을 위해 사용했더라도 배포 할 때는 해당 코드를 삭제 후 배포.

모의해킹

- Android Backup

M01: -

Label: Allows Backup

Description: This option allows backups of the application data via adb. Malicious people with physical access could use adb to get private data of your app into their PC.

File: [AndroidManifest.xml](#)

Line : 17

Affected code:

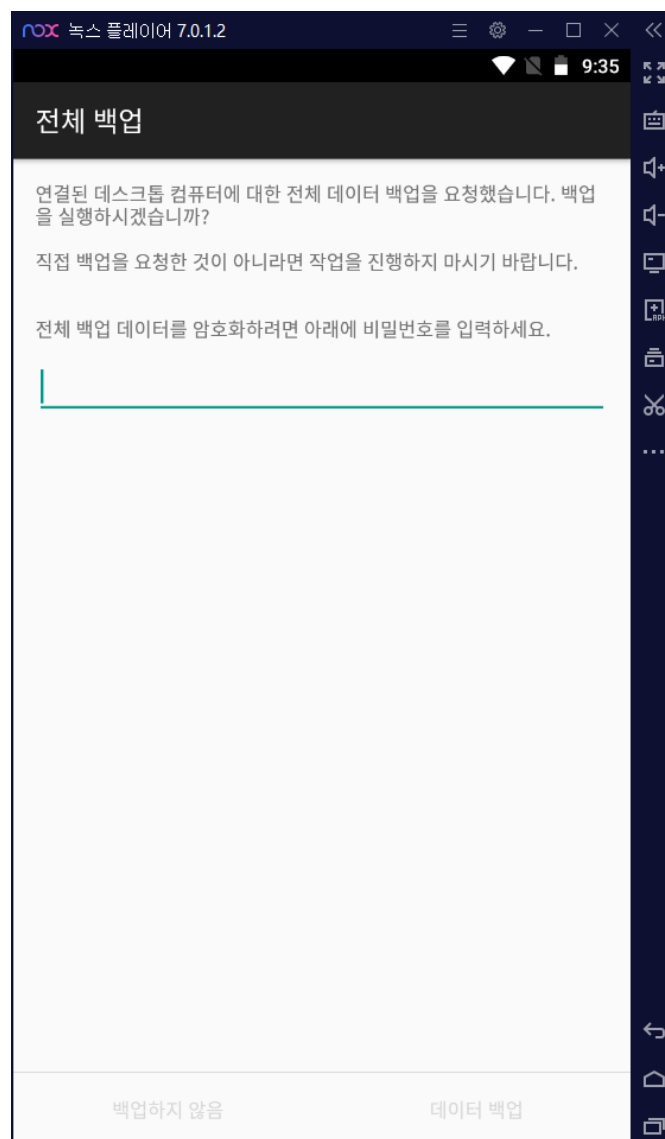
```
13 <uses-permission android:name="android.permission.READ_CALL_LOG" />
14 <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
15 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
16 <uses-feature android:glEsVersion="131072" android:required="true" />
17 <application android:allowBackup="true" android:label="@com.android.insecurebankv2:string/app_name"
18     <activity android:name="com.android.insecurebankv2.LoginActivity" android:label="@com.android.insecurebankv2:string/app_name"
19         <intent-filter>
20             <action android:name="android.intent.action.MAIN" />
21             <category android:name="android.intent.category.LAUNCHER" />
```

모의해킹

- Android Backup

```
C:\Program Files (x86)\Nox\bin>adb backup com.android.insecurebankv2 -f insecurebankv2_backup.ab
Now unlock your device and confirm the backup operation...

C:\Program Files (x86)\Nox\bin>_
```

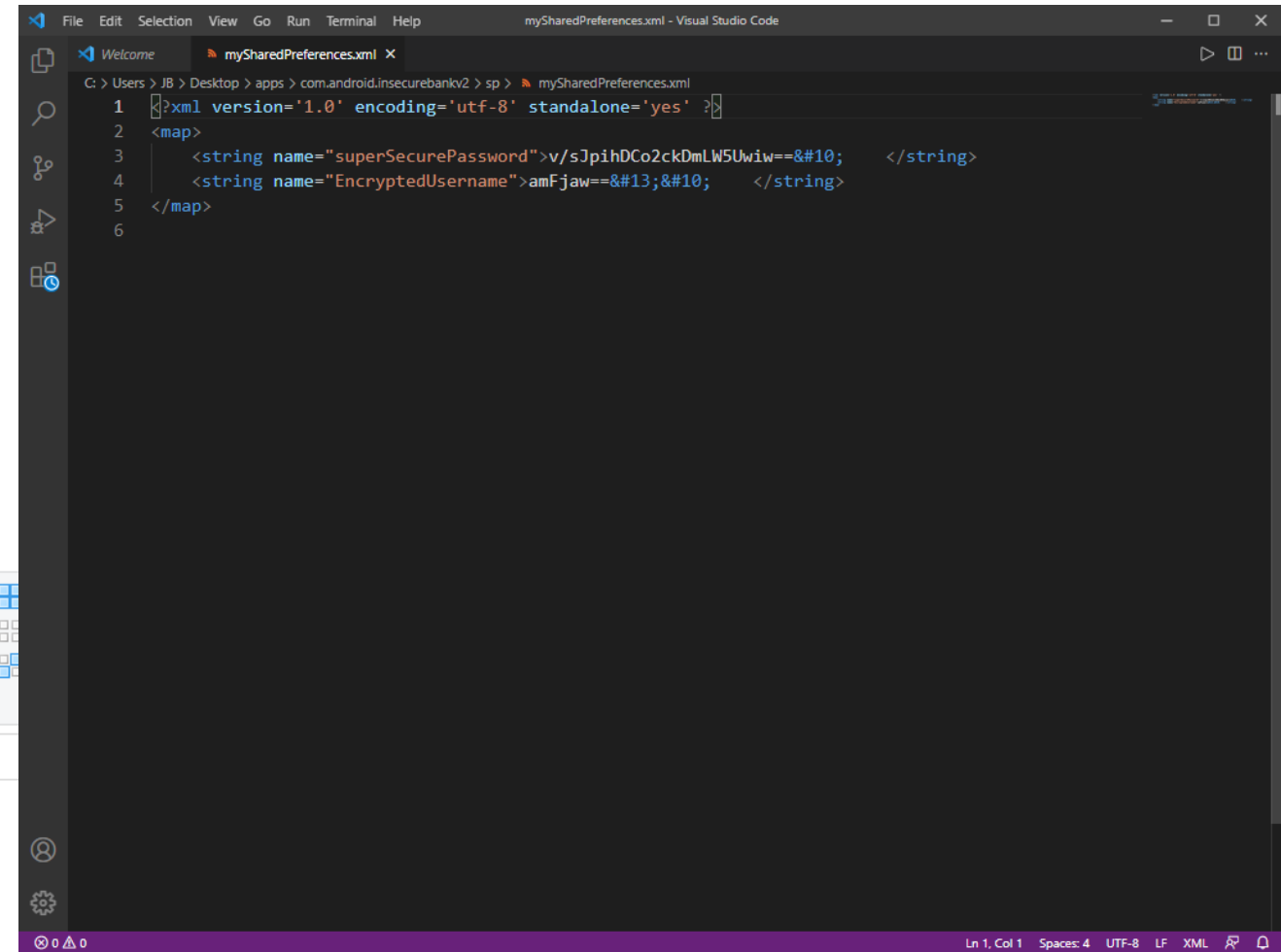
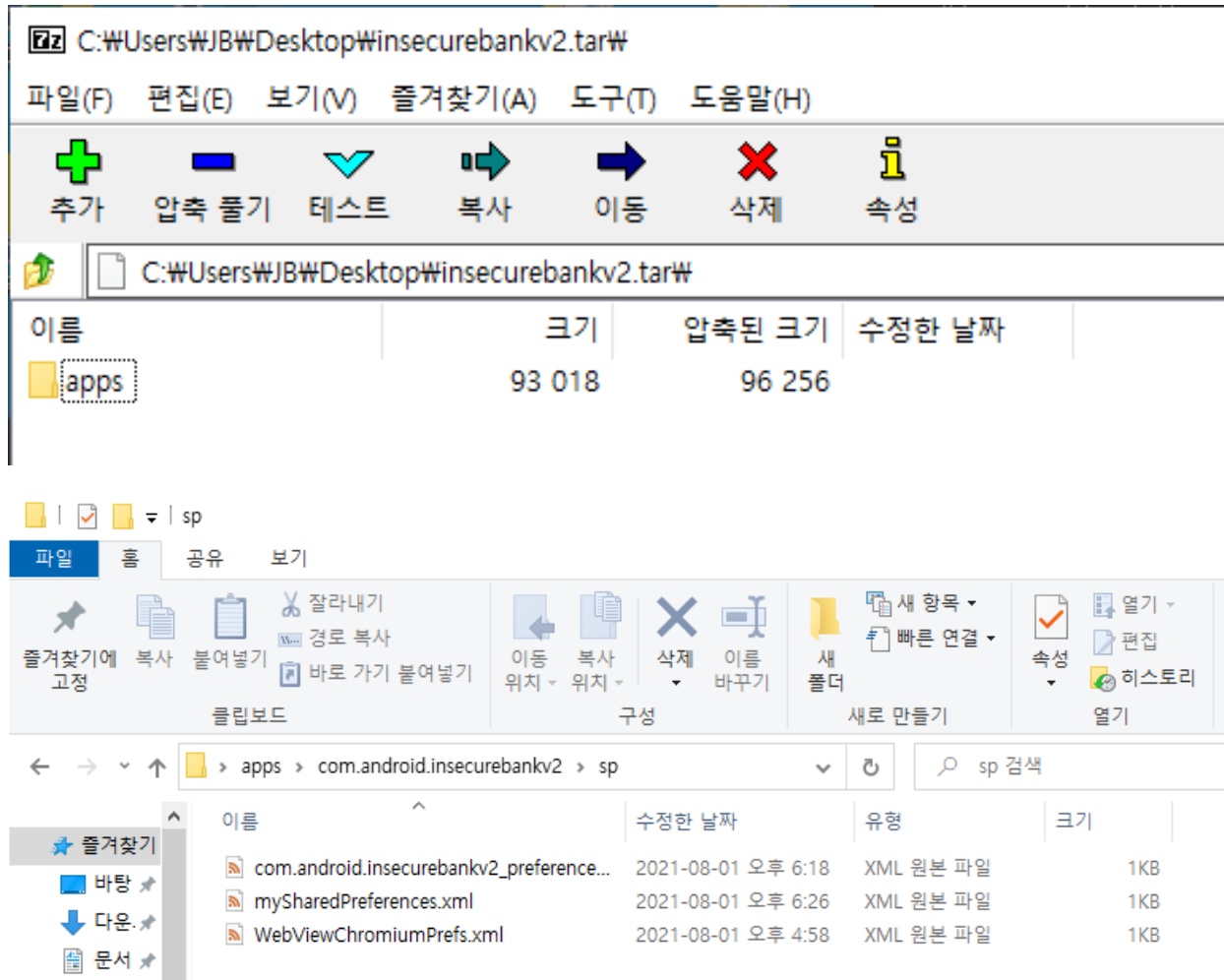


icuuc57.dll	2021-06-02 오후 5:36	응용 프로그램 확장	2,316KB
InsecureBankv2.apk	2021-08-01 오후 2:02	Nox.apk	3,382KB
insecurebankv2_backup.ab	2021-08-01 오후 9:45	AB 파일	5KB
kntd32.exe	2021-06-02 오후 5:36	응용 프로그램	44KB
kntd64.exe	2021-06-02 오후 5:36	응용 프로그램	67KB

ab파일을 abe.jar unpack을 이용해 tar 파일로 변환

모의해킹

- Android Backup



모의해킹

- **Android Backup**

- AndroidManifest의 allowBackup 값이 true로 설정되어 있어 전체 백업이 가능했음.
- 이처럼 전체 백업이 가능한 경우 백업된 파일을 통해 특정 정보 확인이 가능함.
- 따라서 allowBackup 값을 false로 변경하거나 전체 백업을 가능하게 하더라도 중요 정보는 암호화를 하는 것이 필요함.

모의해킹

- Activity vulnerable

```
<activity
    android:label="@ref/0x7f070057"
    android:name="com.android.insecurebankv2.FilePrefActivity"
    android:windowSoftInputMode="0x34" />

<activity
    android:label="@ref/0x7f070054"
    android:name="com.android.insecurebankv2.DoLogin" />

<activity
    android:label="@ref/0x7f07005b"
    android:name="com.android.insecurebankv2.PostLogin"
    android:exported="true" />

<activity
    android:label="@ref/0x7f07005e"
    android:name="com.android.insecurebankv2.WrongLogin" />

<activity
    android:label="@ref/0x7f070055"
    android:name="com.android.insecurebankv2.DoTransfer"
    android:exported="true" />

<activity
    android:label="@ref/0x7f07005d"
    android:name="com.android.insecurebankv2.ViewStatement"
    android:exported="true" />

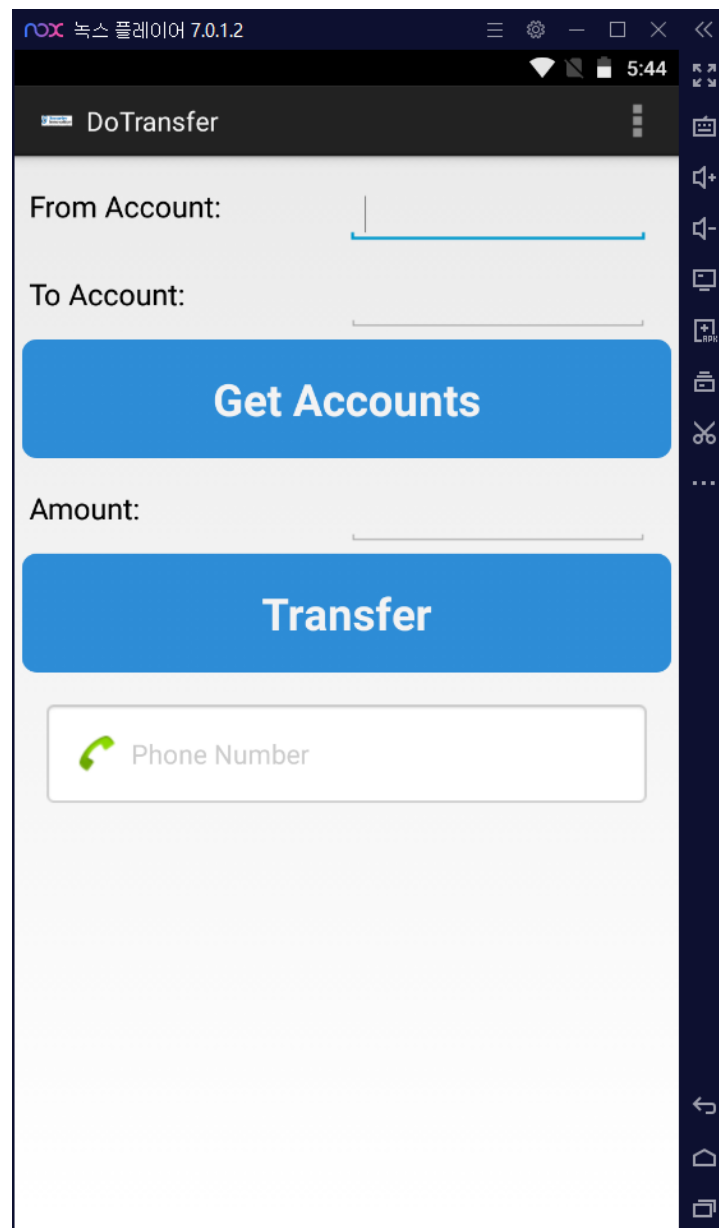
<provider
    android:name="com.android.insecurebankv2.TrackUserContentProvider"
    android:exported="true"
    android:authorities="com.android.insecurebankv2.TrackUserContentProvider" />

<receiver
    android:name="com.android.insecurebankv2.MyBroadCastReceiver"
    android:exported="true">

<intent-filter>
```

모의해킹

- Activity vulnerable



```
C:\Program Files (x86)\Nox\bin>adb shell
*7*[r*[999;999H*[6n*8star2lte:/ #
star2lte:/ #
star2lte:/ # am start -n com.android.insecurebankv2/.DoTransfer
Starting: Intent { cmp=com.android.insecurebankv2/.DoTransfer }
star2lte:/ #
```

모의해킹

- **Activity vulnerable**
 - AndroidManifest.xml Activity의 exported 값이 true로 설정되어 있어 권한 없이 해당 Activity를 호출 가능함. -> 권한이 없는 기능 사용 가능.
 - 따라서 AndroidManifest.xml Activity의 exported 을 false로 변경하는 것이 필요함.