Done By: SAPTARSHI CHATTERJEE(Security Researcher CYBER KINGS INDIA)
Date: 02/April/2020
TARGET: FRONT APP
Email: sapchatterjee1998@gmail.com
Contact:+919748858085

At first I want to tell I have done this for noble purpose I didn't have any bad motive so I will be writing certain Critical and some not that Critical bugs so I will be Highlighting the CRITICAL ones with RED colour.

Platform: Android
Package Name: com.frontapp.mobile
Package Version Name: 3.3.6
Package Version Code: 300306
Min Sdk: 21
Target Sdk: 29
MD5 : e4960738c41d5cb577b411a1b4686c38
SHA1 : 0d517f6091c0a843122b1ede5037599bc26e89c6
SHA256: 623e01122a1f37ce7ccf6288ed5aea73721523d6c25832181f7a38ca322709ed
SHA512:
3b7f3f806062829b5bd88216bc2b8cf7c3c42049d7f492fb0e4ba66bd8cc99646940f2ca75e50427f701383a0bd708ecdb025049b06a005ed3f4aa24bc8fed17
Analyze Signature:
73b5cea737371bbefb74106e964d9241346dedde8b9c930d6c1a7d6856dbef4e6b10fe2ef8b97636838eca4d8d8cf14b199204f67b110d32bca6e1b169e4c5d4

[Critical] <#BID 64208, CVE-2013-6271#> Fragment Vulnerability Checking:

'Fragment' or 'Fragment for ActionbarSherlock' has a severe vulnerability prior to Android 4.4 (API 19).

Please check:

(1)http://developer.android.com/reference/android/os/Build.VERSION_CODES.html#KITKAT

You MUST override 'isValidFragment' method in every "PreferenceActivity" class to avoid Exception throwing in Android 4.4:

Lcom/frontapp/mobile/activity/a;

All of the potential vulnerable "fragment":

Landroidx/lifecycle/ReportFragment;
Lcom/bumptech/glide/m/k;
Lcom/google/android/gms/common/api/internal/zza;

[Critical] <Implicit_Intent> Implicit Service Checking:

To ensure your app is secure, always use an explicit intent when starting a Service and DO NOT declare intent filters for your services. Using an implicit intent to start a service is a security hazard because you cannot be certain what service will respond to the intent, and the user cannot see which service starts.

[Critical] <SSL_Security> SSL Connection Checking:

URLs that are NOT under SSL (Total:3):

http://acs.amazonaws.com/groups/global/AllUsers
=> Lcom/amazonaws/services/s3/model/GroupGrantee;-><clinit>()V
http://acs.amazonaws.com/groups/global/AuthenticatedUsers
=> Lcom/amazonaws/services/s3/model/GroupGrantee;-><clinit>()V
http://acs.amazonaws.com/groups/s3/LogDelivery
=> Lcom/amazonaws/services/s3/model/GroupGrantee;-><clinit>()V

[Critical] <WebView><Remote Code Execution><#CVE-2013-4710#> WebView RCE Vulnerability Checking:

Found a critical WebView "addJavascriptInterface" vulnerability. This method can be used to allow JavaScript to control the host application.

This is a powerful feature, but also presents a security risk for applications targeted to API level JELLY_BEAN(4.2) or below,because JavaScript could use reflection to access an injected object's public fields. Use of this method in a WebView containing untrusted content could allow an attacker to manipulate the host application in unintended ways, executing Java code with the permissions of the host application.

Please modify the below code:
=>
Lcom/frontapp/mobile/richcontenteditor/EmailComposerEditText;-><init>(Landroid/content/Context;
)V (0x118) --->
Landroid/webkit/WebView;->addJavascriptInterface(Ljava/lang/Object;
Ljava/lang/String;)V
=>
Lcom/frontapp/mobile/richcontenteditor/EmailComposerEditText;-><init>(Landroid/content/Context
;
Landroid/util/AttributeSet;)V (0x118) --->
Landroid/webkit/WebView;->addJavascriptInterface(Ljava/lang/Object;
Ljava/lang/String;)V
=> Lcom/frontapp/mobile/view/v1;-><init>(Landroid/content/Context;)V (0x10e)
--->
Landroid/webkit/WebView;->addJavascriptInterface(Ljava/lang/Object;
Ljava/lang/String;)V
=> Le/a/a/u/b/e;-><init>(Landroid/content/Context;)V (0xac) --->
Landroid/webkit/WebView;->addJavascriptInterface(Ljava/lang/Object;
Ljava/lang/String;)V
=>
Lcom/frontapp/mobile/login/fragment/LoginFragment;->onViewCreated(Landroid/view/View;
Landroid/os/Bundle;)V (0x2fa) --->
Landroid/webkit/WebView;->addJavascriptInterface(Ljava/lang/Object;
Ljava/lang/String;)V
[Warning] External Storage Accessing:
External storage access found (Remember DO NOT write important files to external
storages):
=>
Landroidx/core/content/FileProvider;->parsePathStrategy(Landroid/content/Context;
Ljava/lang/String;)Landroidx/core/content/FileProvider$PathStrategy; (0xcc)
--->
Landroid/os/Environment;->getExternalStorageDirectory()Ljava/io/File;
=>
Landroidx/core/os/EnvironmentCompat;->getStorageState(Ljava/io/File;)Ljava/lang/String; (0x1e) --->
Landroid/os/Environment;->getExternalStorageDirectory()Ljava/io/File;
[Warning] AndroidManifest Exported Components Checking:
Found "exported" components(except for Launcher) for receiving outside applications'
actions (AndroidManifest.xml).
These components can be initilized by other apps. You should add or modify the
attribute to [exported="false"] if you don't want
to.
You can also protect it with a customized permission with "signature" or higher
protectionLevel and specify in
"android:permission" attribute.
activity => com.pusher.pushnotifications.reporting.OpenNotificationActivity
service => com.pusher.pushnotifications.fcm.EmptyMessagingService
service => io.smooch.core.FcmService
[Warning] <WebView> WebView Local File Access Attacks Checking:

Found "setAllowFileAccess(true)" or not set(enabled by default) in WebView. The attackers could inject malicious script into
WebView and exploit the opportunity to access local resources. This can be mitigated or prevented by disabling local file system
access. (It is enabled by default)
Note that this enables or disables file system access only. Assets and resources are still accessible using file:///android_asset
and file:///android_res.
The attackers can use "mWebView.loadUrl("file:///data/data/[Your_Package_Name]/[File]");" to access app's local file.
Reference:
(1)https://labs.mwrinfosecurity.com/blog/2012/04/23/adventures-with-android-webviews/

(2)http://developer.android.com/reference/android/webkit/WebSettings.html#setAllowFileAccess(boolean)
Please add or modify "yourWebView.getSettings().setAllowFileAccess(false)" to your WebView:

Landroidx/webkit/WebSettingsCompat;->getDisabledActionModeMenuItems(Landroid/webkit/WebSettings;)I

Landroidx/webkit/WebSettingsCompat;->getForceDark(Landroid/webkit/WebSettings;)I

Landroidx/webkit/WebSettingsCompat;->getOffscreenPreRaster(Landroid/webkit/WebSettings;)Z

Landroidx/webkit/WebSettingsCompat;->getSafeBrowsingEnabled(Landroid/webkit/WebSettings;)Z

Landroidx/webkit/WebSettingsCompat;->setDisabledActionModeMenuItems(Landroid/webkit/WebSettings; I)V

Landroidx/webkit/WebSettingsCompat;->setForceDark(Landroid/webkit/WebSettings; I)V

Landroidx/webkit/WebSettingsCompat;->setOffscreenPreRaster(Landroid/webkit/WebSettings; Z)V

Landroidx/webkit/WebSettingsCompat;->setSafeBrowsingEnabled(Landroid/webkit/WebSettings; Z)V

Lcom/frontapp/mobile/login/fragment/LoginFragment;->onViewCreated(Landroid/view/View; Landroid/os/Bundle;)V
Lcom/frontapp/mobile/richcontenteditor/j;-><init>(Landroid/content/Context; Landroid/util/AttributeSet; I)V
Lcom/frontapp/mobile/view/v1$f;->a(Lcom/frontapp/mobile/network/status/b;)V
Lcom/frontapp/mobile/view/v1;-><init>(Landroid/content/Context;)V
Lcom/frontapp/mobile/view/v1;->e()V
Lcom/frontapp/mobile/view/v1;->f()V
Lcom/frontapp/mobile/view/v1;->g()V
Lcom/frontapp/mobile/view/v1;->h()Z
Le/a/a/u/b/e;-><init>(Landroid/content/Context;)V
[Warning] <WebView> WebView Potential XSS Attacks Checking:

=> Lcom/frontapp/mobile/richcontenteditor/j;-><init>(Landroid/content/Context; Landroid/util/AttributeSet; I)V (0x1e) --->
Landroid/webkit/WebSettings;->setJavaScriptEnabled(Z)V
=> Lcom/frontapp/mobile/view/v1;-><init>(Landroid/content/Context;)V (0xd0) --->
Landroid/webkit/WebSettings;->setJavaScriptEnabled(Z)V
=> Le/a/a/u/b/e;-><init>(Landroid/content/Context;)V (0x9a) ---> Landroid/webkit/WebSettings;->setJavaScriptEnabled(Z)V

=>
Lcom/frontapp/mobile/login/fragment/LoginFragment;->onViewCreated(Landroid/view/View;
Landroid/os/Bundle;)V (0x2dc) --->
Landroid/webkit/WebSettings;->setJavaScriptEnabled(Z)V
[Notice] <Database><#CVE-2011-3901#> Android SQLite Databases Vulnerability Checking:
This app is using Android SQLite databases but it's "NOT" suffering from SQLite Journal
Information Disclosure Vulnerability.
[Notice] File Unsafe Delete Checking:
Everything you delete may be recovered by any user or attacker, especially rooted
devices.
Please make sure do not use "file.delete()" to delete essential files.
Check this video: https://www.youtube.com/watch?v=tGw1fxUD-uY
=> Landroidx/core/content/FileProvider;->delete(Landroid/net/Uri;
Ljava/lang/String; [Ljava/lang/String;)I (0xc) --->
Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromInputStream(Landroid/content/Con
text;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x1c) --->
Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromInputStream(Landroid/content/Con
text;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x34) --->
Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromInputStream(Landroid/content/Con
text;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x3e) --->
Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromInputStream(Landroid/content/Con
text;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x46) --->
Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromResourcesFontFile(Landroid/conten
t/Context;
Landroid/content/res/Resources; I Ljava/lang/String;
I)Landroid/graphics/Typeface; (0x1c) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromResourcesFontFile(Landroid/conten
t/Context;
Landroid/content/res/Resources; I Ljava/lang/String;
I)Landroid/graphics/Typeface; (0x34) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromResourcesFontFile(Landroid/conten
t/Context;
Landroid/content/res/Resources; I Ljava/lang/String;
I)Landroid/graphics/Typeface; (0x3e) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatBaseImpl;->createFromResourcesFontFile(Landroid/conten
t/Context;
Landroid/content/res/Resources; I Ljava/lang/String;
I)Landroid/graphics/Typeface; (0x46) ---> Ljava/io/File;->delete()Z

=>
Landroidx/core/graphics/TypefaceCompatUtil;->copyToDirectBuffer(Landroid/content/Context;
Landroid/content/res/Resources;
I)Ljava/nio/ByteBuffer; (0x1c) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatUtil;->copyToDirectBuffer(Landroid/content/Context;
Landroid/content/res/Resources;
I)Ljava/nio/ByteBuffer; (0x2c) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatUtil;->copyToDirectBuffer(Landroid/content/Context;
Landroid/content/res/Resources;
I)Ljava/nio/ByteBuffer; (0x36) ---> Ljava/io/File;->delete()Z
=> Landroidx/core/util/AtomicFile;->delete()V (0x4) ---> Ljava/io/File;->delete()Z
=> Landroidx/core/util/AtomicFile;->delete()V (0xe) ---> Ljava/io/File;->delete()Z
=> Landroidx/core/util/AtomicFile;->failWrite(Ljava/io/FileOutputStream;)V (0x14)
---> Ljava/io/File;->delete()Z
=> Landroidx/core/util/AtomicFile;->finishWrite(Ljava/io/FileOutputStream;)V
(0x14) ---> Ljava/io/File;->delete()Z
=> Landroidx/core/util/AtomicFile;->openRead()Ljava/io/FileInputStream; (0x14)
---> Ljava/io/File;->delete()Z
=> Landroidx/core/util/AtomicFile;->startWrite()Ljava/io/FileOutputStream; (0x7e)
---> Ljava/io/File;->delete()Z
=> Landroidx/exifinterface/media/ExifInterface;->saveAttributes()V (0x9c) --->
Ljava/io/File;->delete()Z
=> Landroidx/exifinterface/media/ExifInterface;->saveAttributes()V (0xca) --->
Ljava/io/File;->delete()Z
=> Landroidx/multidex/MultiDex;->clearOldDexDir(Landroid/content/Context;)V
(0xf8) ---> Ljava/io/File;->delete()Z
=> Landroidx/multidex/MultiDex;->clearOldDexDir(Landroid/content/Context;)V
(0x16c) ---> Ljava/io/File;->delete()Z
=> Landroidx/multidex/MultiDexExtractor;->clearDexDir()V (0xb2) --->
Ljava/io/File;->delete()Z
=> Landroidx/multidex/MultiDexExtractor;->extract(Ljava/util/zip/ZipFile;
Ljava/util/zip/ZipEntry; Ljava/io/File;
Ljava/lang/String;)V (0x134) ---> Ljava/io/File;->delete()Z
=> Landroidx/multidex/MultiDexExtractor;->extract(Ljava/util/zip/ZipFile;
Ljava/util/zip/ZipEntry; Ljava/io/File;
Ljava/lang/String;)V (0x1fe) ---> Ljava/io/File;->delete()Z
=> Landroidx/multidex/MultiDexExtractor;->performExtractions()Ljava/util/List;
(0x1d8) ---> Ljava/io/File;->delete()Z
=>
Landroidx/sqlite/db/SupportSQLiteOpenHelper$Callback;->deleteDatabaseFile(Ljava/lang/String;)V
(0x7a) --->
Ljava/io/File;->delete()Z
=> Lcom/bumptech/glide/k/a;->J(Ljava/io/File; I I J)Lcom/bumptech/glide/k/a;
(0x44) ---> Ljava/io/File;->delete()Z
=> Lcom/bumptech/glide/k/a;->U()V (0x198) ---> Ljava/io/File;->delete()Z
=> Lcom/bumptech/glide/k/a;->v(Ljava/io/File;)V (0xc) ---> Ljava/io/File;->delete()Z
=> Lcom/bumptech/glide/k/a;->V(Ljava/lang/String;)Z (0x4a) --->
Ljava/io/File;->delete()Z
=> Lcom/bumptech/glide/k/c;->b(Ljava/io/File;)V (0x2a) --->
Ljava/io/File;->delete()Z
=>
Lcom/crashlytics/android/core/CrashlyticsController;->deleteSessionPartFilesFor(Ljava/lang/String;)V
(0x14) --->
Ljava/io/File;->delete()Z

=>
Lcom/crashlytics/android/core/CrashlyticsController;->recursiveDelete(Ljava/io/File;)V (0x2c) --->
Ljava/io/File;->delete()Z
=>
Lcom/crashlytics/android/core/CrashlyticsController;->retainSessions([Ljava/io/File; Ljava/util/Set;)V
(0x60) --->
Ljava/io/File;->delete()Z
=>
Lcom/crashlytics/android/core/CrashlyticsController;->retainSessions([Ljava/io/File; Ljava/util/Set;)V
(0xae) --->
Ljava/io/File;->delete()Z
=>
Lcom/crashlytics/android/core/CrashlyticsController;->doCleanInvalidTempFiles([Ljava/io/File;)V
(0x122) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/CrashlyticsFileMarker;->remove()Z (0x8) --->
Ljava/io/File;->delete()Z
=>
Lcom/crashlytics/android/core/LogFileManager;->discardOldLogFiles(Ljava/util/Set;)V (0x38) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/Utils;->capFileCount(Ljava/io/File;
Ljava/io/FilenameFilter; I Ljava/util/Comparator;)I
(0x28) ---> Ljava/io/File;->delete()Z
=> Lcom/frontapp/mobile/service/c;->c(Ljava/lang/String;)V (0x2c) --->
Ljava/io/File;->delete()Z
=> Lcom/getkeepsafe/relinker/c;->b(Landroid/content/Context; Ljava/lang/String;
Ljava/lang/String;)V (0x64) --->
Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatApi21Impl;->createFromFontFamilyFilesResourceEntry(Lan
droid/content/Context;

Landroidx/core/content/res/FontResourcesParserCompat$FontFamilyFilesResourceEntry;
Landroid/content/res/Resources;
I)Landroid/graphics/Typeface; (0x40) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatApi21Impl;->createFromFontFamilyFilesResourceEntry(Lan
droid/content/Context;

Landroidx/core/content/res/FontResourcesParserCompat$FontFamilyFilesResourceEntry;
Landroid/content/res/Resources;
I)Landroid/graphics/Typeface; (0x6c) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatApi21Impl;->createFromFontFamilyFilesResourceEntry(Lan
droid/content/Context;

Landroidx/core/content/res/FontResourcesParserCompat$FontFamilyFilesResourceEntry;
Landroid/content/res/Resources;
I)Landroid/graphics/Typeface; (0x74) ---> Ljava/io/File;->delete()Z
=>
Landroidx/core/graphics/TypefaceCompatApi21Impl;->createFromFontFamilyFilesResourceEntry(Lan
droid/content/Context;

Landroidx/core/content/res/FontResourcesParserCompat$FontFamilyFilesResourceEntry;
Landroid/content/res/Resources;
I)Landroid/graphics/Typeface; (0x82) ---> Ljava/io/File;->delete()Z

=>
Landroidx/core/graphics/TypefaceCompatApi21Impl;->createFromFontFamilyFilesResourceEntry(Landroid/content/Context;

Landroidx/core/content/res/FontResourcesParserCompat$FontFamilyFilesResourceEntry;
Landroid/content/res/Resources;
I)Landroid/graphics/Typeface; (0x8a) ---> Ljava/io/File;->delete()Z
=>
Landroidx/documentfile/provider/RawDocumentFile;->deleteContents(Ljava/io/File;)Z (0x32) --->
Ljava/io/File;->delete()Z
=> Landroidx/documentfile/provider/RawDocumentFile;->delete()Z (0xe) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/InvalidSessionReport;->remove()V (0x4c) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/NativeSessionReport;->remove()V (0x50) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/NativeSessionReport;->remove()V (0x94) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/QueueFileLogStore;->deleteLogFile()V (0xa) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/SessionReport;->remove()V (0x44) --->
Ljava/io/File;->delete()Z
=>
Lcom/frontapp/mobile/upload/FileUploadWorker$h;->a(Landroidx/work/ListenableWorker$Result;)V
(0x4) --->
Ljava/io/File;->delete()Z
=>
Lcom/google/android/gms/common/data/BitmapTeleporter;->zabz()Ljava/io/FileOutputStream;
(0x32) --->
Ljava/io/File;->delete()Z
=>
Lcom/frontapp/mobile/incoming/b$a;->invokeSuspend(Ljava/lang/Object;)Ljava/lang/Object; (0x7e)
--->
Ljava/io/File;->delete()Z
=>
Lcom/frontapp/mobile/settings/AdvancedPreferencesFragment$startRealmUpload$1;->invokeSuspend(Ljava/lang/Object;)Ljava/lang
/Object; (0x80) ---> Ljava/io/File;->delete()Z
=>
Lcom/frontapp/mobile/settings/AdvancedPreferencesFragment$startRealmUpload$1;->invokeSuspend(Ljava/lang/Object;)Ljava/lang
/Object; (0xdc) ---> Ljava/io/File;->delete()Z
[Notice] <Debug><Hacker> Codes for Checking Android Debug Mode:
Found codes for checking "ApplicationInfo.FLAG_DEBUGGABLE" in
AndroidManifest.xml:
=> Lcom/google/android/gms/common/GoogleSignatureVerifier;->zza
(Ljava/lang/String; I)Lcom/google/android/gms/common/zzm;
=> Lcom/google/android/gms/common/GoogleSignatureVerifier;->zzc
(Ljava/lang/String;)Lcom/google/android/gms/common/zzm;
[Notice] <Signature><Hacker> Getting Signature Code Checking:
This app has code checking the package signature in the code. It might be used to check
for whether the app is hacked by the
attackers.
=>
Landroidx/core/provider/FontsContractCompat;->getProvider(Landroid/content/pm/PackageManager;

Landroidx/core/provider/FontRequest;
Landroid/content/res/Resources;)Landroid/content/pm/ProviderInfo; (0x36) --->
Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
=>
Lcom/google/android/gms/common/GooglePlayServicesUtilLight;->zza(Landroid/content/Context; Z
I)I (0x50) --->
Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
=>
Lcom/google/android/gms/common/util/UidVerifier;->isGooglePlayServicesUid(Landroid/content/Co
ntext; I)Z (0x20) --->
Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
=>
Lcom/google/android/gms/common/wrappers/PackageManagerWrapper;->zza(Ljava/lang/String; I
I)Landroid/content/pm/PackageInfo; (0x10) --->
Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
[Notice] Native Library Loading Checking:
Native library loading codes(System.loadLibrary(...)) found:
[libcrashlytics.so]
=> Lcom/crashlytics/android/ndk/JniNativeApi;-><clinit>()V (0x4) --->
Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V
[Notice] AndroidManifest Exported Components Checking 2:
Found "exported" components(except for Launcher) for receiving Google's "Android"
actions (AndroidManifest.xml):
activity => com.frontapp.mobile.incoming.IncomingIntentActivity
[Info] AndroidManifest Adb Backup Checking:
This app has disabled Adb Backup.
[Info] <Command> Runtime Command Checking:
This app is not using critical function 'Runtime.getRuntime().exec("...")'.
[Info] <Command> Executing "root" or System Privilege Checking:
[Info] <Database> SQLiteDatabase Transaction Deprecated Checking:
Ignore checking "SQLiteDatabase:beginTransactionNonExclusive" because your set
minSdk >= 11.
[Info] <Database> Android SQLite Databases Encryption (SQLite Encryption Extension (SEE)):
This app is "NOT" using SQLite Encryption Extension (SEE) on Android
(http://www.sqlite.org/android) to encrypt or decrpyt
databases.
[Info] <Database> Android SQLite Databases Encryption (SQLCipher):
This app is "NOT" using SQLCipher(http://sqlcipher.net/) to encrypt or decrpyt
databases.
[Info] <Debug> Android Debug Mode Checking:
DEBUG mode is OFF(android:debuggable="false") in AndroidManifest.xml.
[Info] Dynamic Code Loading:
No dynamic code loading(DexClassLoader) found.
[Info] <Framework> Framework - MonoDroid:
This app is NOT using MonoDroid Framework (http://xamarin.com/android).
[Info] <Hacker> Base64 String Encryption:
No encoded Base64 String or Urls found.
[Info] <Database><Hacker> Key for Android SQLite Databases Encryption:
Did not find using the symmetric key(PRAGMA key) to encrypt the SQLite databases (It's
still possible that it might use but we
did not find out).
[Info] <Hacker> APK Installing Source Checking:
Did not detect this app checks for APK installer sources.

[Info] <KeyStore><Hacker> KeyStore File Location:
Did not find any possible BKS keystores or certificate keystore file (Notice: It does not mean this app does not use keysotre):
[Info] <KeyStore><Hacker> KeyStore Protection Checking:
Ignore checking KeyStore protected by password or not because you're not using KeyStore.
[Info] <Hacker> Code Setting Preventing Screenshot Capturing:
Did not detect this app has code setting preventing screenshot capturing.
[Info] HttpURLConnection Android Bug Checking:
Ignore checking "http.keepAlive" because you're not using "HttpURLConnection" and min_Sdk > 8.
[Info] <KeyStore> KeyStore Type Checking:
KeyStore 'BKS' type check OK
[Info] Google Cloud Messaging Suggestion:
Nothing to suggest.
[Info] <#CVE-2013-4787#> Master Key Type I Vulnerability:
No Master Key Type I Vulnerability in this APK.
[Info] App Sandbox Permission Checking:
No security issues "MODE_WORLD_READABLE" or "MODE_WORLD_WRITEABLE" found on 'openOrCreateDatabase' or 'openOrCreateDatabase2' or
'getDir' or 'getSharedPreferences' or 'openFileOutput'
[Info] AndroidManifest Dangerous ProtectionLevel of Permission Checking:
No "dangerous" protection level customized permission found (AndroidManifest.xml).
[Info] AndroidManifest PermissionGroup Checking:
PermissionGroup in permission tag of AndroidManifest sets correctly.
[Info] AndroidManifest "intent-filter" Settings Checking:
"intent-filter" of AndroidManifest.xml check OK.
[Info] AndroidManifest Normal ProtectionLevel of Permission Checking:
No default or "normal" protection level customized permission found (AndroidManifest.xml).
[Info] <#CVE-2013-6272#> AndroidManifest Exported Lost Prefix Checking:
No exported components that forgot to add "android:" prefix.
[Info] AndroidManifest ContentProvider Exported Checking:
No exported "ContentProvider" found (AndroidManifest.xml).
[Info] <Sensitive_Information> Getting IMEI and Device ID:
Did not detect this app is getting the "device id(IMEI)" by "TelephonyManager.getDeviceId()" approach.
[Info] <Sensitive_Information> Getting ANDROID_ID:
Did not detect this app is getting the 64-bit number "Settings.Secure.ANDROID_ID".
[Info] Codes for Sending SMS:
Did not detect this app has code for sending SMS messages (sendDataMessage, sendMultipartTextMessage or sendTextMessage).
[Info] <System> AndroidManifest sharedUserId Checking:
This app does not use "android.uid.system" sharedUserId.
[Info] <SSL_Security> SSL Implementation Checking (Verifying Host Name in Custom Classes):
Self-defined HOSTNAME VERIFIER checking OK.
[Info] <SSL_Security> SSL Implementation Checking (Verifying Host Name in Fields):
Critical vulnerability "ALLOW_ALL_HOSTNAME_VERIFIER" field setting or "AllowAllHostnameVerifier" class instance not found.
[Info] <SSL_Security> SSL Implementation Checking (Insecure component):
Did not detect SSLSocketFactory by insecure method "getInsecure".
[Info] <SSL_Security> SSL Implementation Checking (HttpHost):
DEFAULT_SCHEME_NAME for HttpHost check: OK
[Info] <SSL_Security> SSL Implementation Checking (WebViewClient for WebView):
Did not detect critical usage of "WebViewClient"(MITM Vulnerability).
[Info] <SSL_Security> SSL Certificate Verification Checking:
Did not find vulnerable X509Certificate code.

[Info] Unnecessary Permission Checking:
Permission 'android.permission.ACCESS_MOCK_LOCATION' sets correctly.
[Info] Accessing the Internet Checking:
This app is using the Internet via HTTP protocol.
[Info] AndroidManifest System Use Permission Checking:
No system-level critical use-permission found.

CONCULTION

At last I want to conclude my report by saying the THREAT level is less but there are some issues which have been ignored which if used professionally can hamper some of them I will like to mention about the app but before that I will like to see MORE permissions taken by the app which u can change in manifest file and I expect a Sign Up module to be added as well.

1. 'Fragment' or 'Fragment for ActionbarSherlock' has a severe vulnerability prior to Android 4.4 (API 19).
2. You MUST override 'isValidFragment' method in every "PreferenceActivity" class to avoid Exception throwing in Android 4.4:
3. To ensure your app is secure, always use an explicit intent when starting a Service and DO NOT declare intent filters for your services. Using an implicit intent to start a service is a security hazard because you cannot be certain what service will respond to the intent, and the user cannot see which service starts.
4. Found a critical WebView "addJavascriptInterface" vulnerability. This method can be used to allow JavaScript to control the host application.
This is a powerful feature, but also presents a security risk for applications targeted to API level JELLY_BEAN(4.2) or below,because JavaScript could use reflection to access an injected object's public fields. Use of this method in a WebView containing untrusted content could allow an attacker to manipulate the host application in unintended ways, executing Java code with the permissions of the host application.
5. Found "setJavaScriptEnabled(true)" in WebView, which could exposed to potential XSS attacks. Please check the web page code carefully and sanitize the output:
6. Did not find codes checking "root" permission(su) or getting system permission.
7. Missing Sign up Page(Recommended).