G4 analysis:

The obfuscation techniques they used:

1) AES plus base64 encryption for all strings including sql query

2) some control flow obfuscation method that messed up the original control flow with try catch and goto, this method is very successful that confused all the decompilation tools I tried, I have to analyse the original byte code to understand the code

3) class/method renaming, to remove the meaning represented by the original name

4) Inserting some middle classes to increase the difficulty of analysis

De-obfuscation:

1. the attached plaintext table got from exact the same code:



create another android app,

import class b and c,

decrypt the cithpertext with some code similar to this:

Log.d("test",c.a("A77uC1DdVSiO6G0LdEx5gQ=="));

2) no good method, I can only combine the bytecode with the not very accurate decompiled source code to understand it.

3) no significant method to the class renaming, since the original names have been removed permanently.

4) no good method, just try to write down the functionalities of every class/method.

But they have broken the rule which limit the source code by 1000 LOC:

Regarding point 3, they invoked google gms and android support packages which makes these two packages were included into the source code. But they didn’t prevent these standard packages from being obfuscated. This significantly increased the lines of code from a thousand to multiple tens of thousands, because all the google code (around 2000 files 4M bytes) is modified and cannot be easily understand, see below dir output:



Also they changed a standard google gms call into something like this:

this.m.add(new **f**().a(Integer.toString(n2)).a(d2, d3, 200.0f).a(86400000).a(3).a());

Here f returns standard class SupportPlaceAutocompleteFragment

static /\* synthetic \*/ SupportPlaceAutocompleteFragment **f**(m2m5pcv0 m2m5pcv02) {

All the function ‘a’ following f should be some standard gms classes, but now people have to read carefully function by function through the obfuscated gms and android source code to understand G4’s app. It is unfair, since it greatly exceed the 1000 line limitation. And also these packages are used in many files:

*G10edb67.java:import com.google.android.gms.maps.model.LatLng;*

*I.java:import com.google.android.gms.maps.model.LatLng;*

*J.java:import com.google.android.gms.common.api.Status;*

*J.java:import com.google.android.gms.location.places.a;*

*J.java:import com.google.android.gms.location.places.ui.c;*

*M2m5pcv0.java:import com.google.android.gms.common.ConnectionResult;*

*M2m5pcv0.java:import com.google.android.gms.common.api.Status;*

*M2m5pcv0.java:import com.google.android.gms.common.api.n;*

*M2m5pcv0.java:import com.google.android.gms.common.api.o;*

*M2m5pcv0.java:import com.google.android.gms.common.api.q;*

*M2m5pcv0.java:import com.google.android.gms.common.api.r;*

*M2m5pcv0.java:import com.google.android.gms.common.api.t;*

*M2m5pcv0.java:import com.google.android.gms.common.api.w;*

*M2m5pcv0.java:import com.google.android.gms.common.api.x;*

*M2m5pcv0.java:import com.google.android.gms.location.GeofencingRequest;*

*M2m5pcv0.java:import com.google.android.gms.location.e;*

*M2m5pcv0.java:import com.google.android.gms.location.f;*

*M2m5pcv0.java:import com.google.android.gms.location.m;*

*M2m5pcv0.java:import com.google.android.gms.location.places.ui.SupportPlaceAutocompleteFragment;*

*M2m5pcv0.java: private com.google.android.gms.location.places.a v;*

*M2m5pcv0.java: static /\* synthetic \*/ com.google.android.gms.location.places.a a(m2m5pcv0 m2m5pcv02, com.google.android.gms.location.places.a a2) {*

*M2m5pcv0.java: static /\* synthetic \*/ com.google.android.gms.location.places.a d(m2m5pcv0 m2m5pcv02) {*

*M2m5pcv0.java: com.google.android.gms.location.j j2 = new com.google.android.gms.location.j();*

*Z1flnjcn.java:import com.google.android.gms.location.e;*

*Z1flnjcn.java:import com.google.android.gms.location.h;*

And

*G10edb67.java:import android.support.v4.app.bs;*

*G10edb67.java:import android.support.v4.app.dh;*

*G10edb67.java:import android.support.v7.a.ax;*

*Lpsdt5r.java:import android.support.design.widget.FloatingActionButton;*

*Lpsdt5r.java:import android.support.v7.a.u;*

*Lpsdt5r.java:import android.support.v7.widget.Toolbar;*

*Lpsdt5r.java: if (android.support.v4.b.c.a((Context)this, "android.permission.ACCESS\_FINE\_LOCATION") == 0) return; // if dont permit to get location return*

*Lpsdt5r.java: android.support.v4.app.a.a(this, new String[]{"android.permission.ACCESS\_FINE\_LOCATION"}, 9);*

*Lpsdt5r.java: android.support.v4.app.a.a(this, new String[]{"android.permission.ACCESS\_FINE\_LOCATION"}, 9);*

*M2m5pcv0.java:import android.support.v4.app.ah;*

*M2m5pcv0.java:import android.support.v7.a.u;*

*Z1flnjcn.java:import android.support.v4.app.bs;*

*Z1flnjcn.java:import android.support.v4.app.dh;*

*Z1flnjcn.java:import android.support.v7.a.ax;*

So I just stopped working on this project because I just could not read so many obfuscated source code, I should do some meaningful things

Attachment 1: the understanding of code against this project until I run into the breach, see comments in line:



Attachment2: the obfuscation exceptions report by the decompiler (only 3 in their own code out of all the 82 exceptions):

