Group 1 report:

Group ones primary obfuscation technique was to obfuscate class names and field variables with a binary number counting system where 1 and 0 were replaced with capital I and l. The purpose of this was to try an obscure their meanings and make it harder for humans to differentiate between variables. Our first step in the obfuscation was to use a Python script to convert the names to a binary number and then convert the binary number to a decimal number preceded by the $ character. Whilst this did not immediately add more meaning it did however make the code easier to read. We also used some common sense naming where it was clear what the variable represented. For instance:

**public** Bitmap IIIIIIIII;

**public** String IIIIIIlIl;

**public** **char** IIIIlIIII;

**public** String IIIllIIlI;

**public** RandomPointGenerator IIlIIlIIl;

**public** StringBuilder IIllIIIIl;

**public** File IlIIIIIll;

**public** **char** IlIIIllII;

**public** String lIIIIlIIl;

**public** **int** lIIIlIlll;

**public** File lIIlIIIII;

**public** **int** lIIlIIlII;

**public** Point lIIlIllIl;

**public** Bitmap lIIllIIII;

**public** File lIIlllIII;

**public** OutputStream lIlIlIlIl;

**public** Uri llIIIIlII;

**public** **int** lllIIIIlI;

**public** Intent lllIllIlI;

**public** **int** llllIIIIl;

**public** **int** lllllIIII;

**public** Uri lllllIllI;

**public** String llllllllI;

**public** Bitmap $0;

**public** String fileName;

**public** **char** $16;

**public** String $50;

**public** RandomPointGenerator randomPointGen;

**public** StringBuilder $97;

**public** File inFile;

**public** **char** $140;

**public** String date;

**public** **int** $279;

**public** File outFile;

**public** **int** $292;

**public** Point $301;

**public** Bitmap $304;

**public** File $312;

**public** OutputStream $341;

**public** Uri $388;

**public** **int** $450;

**public** Intent $474;

**public** **int** $481;

**public** **int** $496;

**public** Uri $502;

**public** String $510;

Group one secondary obfuscation technique was to break up their main logic into single calls to static methods in other classes. For instance:

**protected** **void** onHandleIntent(**final** Intent intent) {

llllllllll.*I*(**this**, intent);

}

**public** **class** llllllllll

{

**public** llllllllll() {

**super**();

}

**public** **static** **void** I(**final** EncoderService encoderService, **final** Intent intent) {

**if** (intent.getStringExtra("imageURI") != **null**) {

IIllllllll.*II*(encoderService, intent);

}

IIIIllIIlI.*IIII*(encoderService, intent);

}

}

As you can see this static method then make another call to another static method in another class and so on. Along the way small parts of the logic are completed and by following these function calls the original functions can be reproduced. For instance:

**protected** **void** onHandleIntent(**final** Intent intent) {

**if** (intent.getStringExtra("imageURI") != **null**) {

**this**.$388 = Uri.parse(intent.getStringExtra("imageURI"));

**try** {

**this**.$304 = MediaStore.Images.Media.getBitmap(**this**.getContentResolver(), **this**.$388);

}

**catch** (Exception ex) {}

}

**if** (**this**.$304 != **null**) {

**this**.$0 = **this**.$304.copy(Bitmap.Config.ARGB\_8888, **true**);

**this**.$50 = intent.getStringExtra("message");

**if** (**this**.$50.equals("") || **this**.$50.equals(**null**)) {

**this**.$50 = "Enter a message in !!!!";

}

**this**.hideText(**this**.$0, **this**.$50);

}

}

**public** **void** hideText(**final** Bitmap bitmap, **final** String s) {

lllIllIIll.*l*(**this**, bitmap, s);

}

Our process:

Through the use of tools such as BytecodeViewer.2.9.8 and jd-gui-1.4.0 reverse engineering this app to obtain the obfuscated source code was relatively easy. Next we used a Python script (see appendix) to assign more human readable names to variables and classes. Then through the use of an IDE like eclipse it was a trivial but laborious task to follow the static function calls through the various classes and replace them with the original logic.

Evaluation: