

移动安全 - APP 渗透进阶之 AppCan 本地文件解密

本篇文章由团队大佬 pyth0n 总结编写

前言

之前渗透 app 的时候，发现好几款 app，拖到 jadx 里搜不到相关代码，后来发现在 assets\widget 目录下有对应的 js 和 html 文件，原来都是 html 混合开发的



每个资源文件都是被加密过的，且每个文件的最后一行都有 "3G2WIN Safe Guard" 标识。通过 assets\widget\js\appcan.js 文件名搜索发现用的是 Appcan 这个移动平台

http://newdocx.appcan.cn/newdocx/docx?type=1247_1234

说明：AppCan IDE 为开发者提供了应用加密功能，支持全包（.html 文件、.css 文件、.js 文件）加密及部分（可选文件）加密，以保证您的代码安全。

为了进一步渗透分析，只能解密资源文件了。

正文

一 分析

既然是网页实现肯定会用到 webview 之类的框架，那么 app 对资源的加载流程可能为：

1)WEBVIEW -> 加载页面 -> 拦截 / 查找本地文件 有 -> 解密 / 写回数据

2)WEBVIEW -> 加载页面 -> 拦截 / 查找本地文件 无 -> 请求网络文件

这里有个共同的点都是需要 拦截, 而 WebView 只有一个实现这个功能的接口:

WebViewClient.shouldInterceptRequest

下面是 shouldInterceptRequest API 的介绍:

API

shouldInterceptRequest 方法

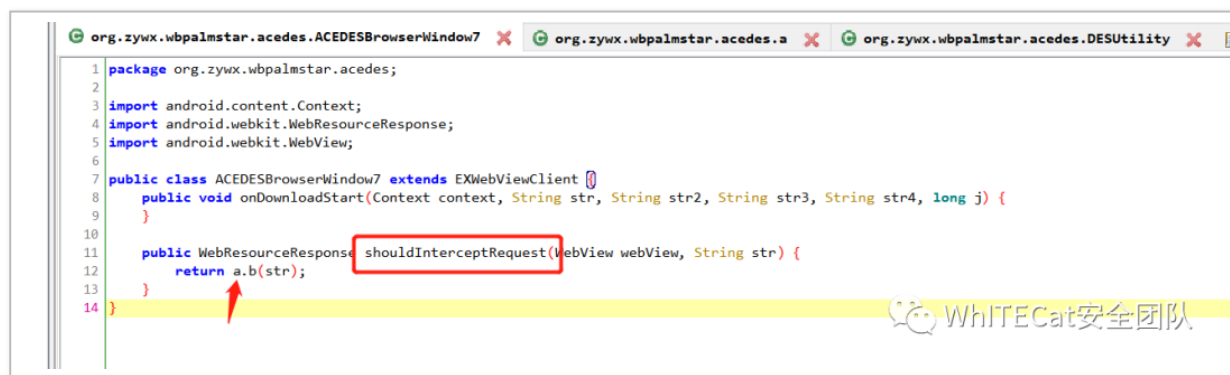
- **WebResourceResponse shouldInterceptRequest**(WebView view, WebResourceRequest request) Notify the host application of a resource request and allow the application to return the data. 通知资源请求的主机应用程序, 并允许应用程序返回数据。
 - If the return value is null, the WebView will continue to load the resource as usual. Otherwise, the return response and data will be used. NOTE: This method is called on a thread **other than** the UI thread so clients should **exercise caution** when accessing private data or the view system.
 - 该函数会在请求资源前调用, 且无论任何资源, 比如超链接、JS文件、图片等, 在每一次请求资源时都会回调。我们可以通过返回一个自定义的WebResourceResponse来让WebView加载指定的资源。比如, 如果我们需要改变网页的背景, 替换网页中图片等, 都可以在这个回调时处理。但是必须注意的是, 此回调是在非UI线程中执行的。
 - 参数 request: Object containing the details of the request. 包含请求的详细信息对象
 - 返回值: A WebResourceResponse containing the response information or null if the WebView should load the resource itself.

例如, 替换所有的图片为自定义的图片

```
if (request.getUrl().toString().endsWith(".jpg")) {  
    try {  
        return new WebResourceResponse("text/html", "UTF-8", view.getContext().getAssets().open("icon.jpg"));  
    } catch (IOException e) {  
        e.printStackTrace();  
    }  
}  
return null;
```

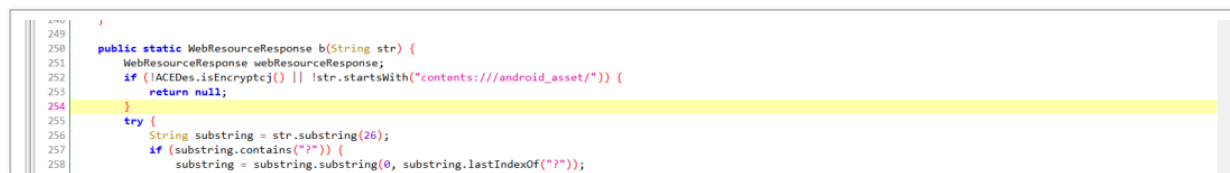
WhITeCat安全团队

在 Jadx 搜索这个方法, 如图



WhITeCat安全团队

发现 a.b 方法, 继续跟进



```

259     }
260     InputStream open = ACEDes.getContext().getAssets().open(substring);
261     if (str.endsWith(".css") || str.endsWith(".js")) {
262         ByteArrayInputStream byteArrayInputStream = new ByteArrayInputStream(DESUtility.a(DESUtility.a(open, open.available()), DESUtility.a(str)).getBytes());
263         webResourceResponse = str.endsWith(".css") ? new WebResourceResponse("text/css", "UTF-8", byteArrayInputStream) : str.endsWith(".js") ? new WebResourceResponse("text/js", "UTF-8", byteArrayInputStream) : null;
264     } else {
265         webResourceResponse = new WebResourceResponse((String) null, "UTF-8", open);
266     }
267     return webResourceResponse;
268 } catch (IOException e) {
269     e.printStackTrace();
270     return null;
271 }
272 }

```

WhITeCat安全团队

把这段代码粘贴下来分析：

```

public static WebResourceResponse b(String str) {
    WebResourceResponse webResourceResponse;
    if (!ACEDes.isEncryptcj() || !str.startsWith("contents:///android_asset/")) { //
如果不是加密过的文件或者是不是以contents:///android_asset/ 开头的文件就退出
        return null;
    }
    try {
        String substring = str.substring(26);
        if (substring.contains("?")) {
            substring = substring.substring(0, substring.lastIndexOf("?"));
        }
        InputStream open = ACEDes.getContext().getAssets().open(substring);
        if (str.endsWith(".css") || str.endsWith(".js")) { //判断是不是js和css文件
            ByteArrayInputStream byteArrayInputStream = new ByteArrayInputStream(DESUtility.a(DESUtility.a(open, open.available()), DESUtility.a(str)).getBytes()); //返回真正的资源文件
            webResourceResponse = str.endsWith(".css") ? new WebResourceResponse("text/css", "UTF-8", byteArrayInputStream) : str.endsWith(".js") ? new WebResourceResponse("text/js", "UTF-8", byteArrayInputStream) : null;
        } else {
            webResourceResponse = new WebResourceResponse((String) null, "UTF-8", open);
        }
        return webResourceResponse;
    } catch (IOException e) {
        e.printStackTrace();
        return null;
    }
}

```

这里就是 Appcan 的 sdk 了。关注 DESUtility.a 这个方法就行，继续跟进

```
22
23
24 public static String a(String str) {
25     String str2 = null;
26     int lastIndexOf = str.lastIndexOf(47);
27     if (lastIndexOf > 0) {
28         str2 = str.substring(lastIndexOf + 1, str.length());
29     }
30     int lastIndexOf2 = str2.lastIndexOf(46);
31     return lastIndexOf2 > 0 ? str2.substring(0, lastIndexOf2) : str2;
32 }
33
34 public static String a(byte[] bArr, String str) {
35     if (bArr == null || bArr.length == 0) {
36         return "";
37     }
38     int length = bArr.length;
39     int length2 = "3G2MIN_Safe_Guard".length();
40     if (length <= length2) {
41         return new String(bArr);
42     }
43     if (!"3G2MIN_Safe_Guard".equals(new String(bArr, length - length2, length2))) {
44         return new String(bArr);
45     }
46     int i = length2 + 256;
47     if (length <= i) {
48         return new String(bArr);
49     }
50     int i2 = length - i;
51     Resources resources = ACDes.getContext().getResources();
52     return nativeHtmlDecode(bArr, str, Integer.toString(i2), a.a(resources.getString(resources.getIdentifier(a.f, EUExUtil.s
53     )
54     public static byte[] a(InputStream inputStream, int i) {
55     byte[] bArr = null;
```

把这段代码粘贴下来分析：

```
public static WebResourceResponse b(String str) {
    WebResourceResponse webResourceResponse;
    if (!ACDes.isEncryptcj() || !str.startsWith("contents:///android_asset/")) { //
如果不是加密过的文件或者是不是以contents:///android_asset/ 开头的文件就退出
        return null;
    }
    try {
        String substring = str.substring(26);
        if (substring.contains("?")) {
            substring = substring.substring(0, substring.lastIndexOf("?"));
        }
        InputStream open = ACDes.getContext().getAssets().open(substring);
        if (str.endsWith(".css") || str.endsWith(".js")) { //判断是不是js和css文件
            ByteArrayInputStream byteArrayInputStream = new ByteArrayInputStream(DESU
tility.a(DESUutility.a(open, open.available()), DESUutility.a(str)).getBytes()); //返回真正
的资源文件
```

```

的源文件
        webResourceResponse = str.endsWith(".css") ? new WebResourceResponse("text/css", "UTF-8", byteArrayInputStream) : str.endsWith(".js") ? new WebResourceResponse("text/js", "UTF-8", byteArrayInputStream) : null;
    } else {
        webResourceResponse = new WebResourceResponse((String) null, "UTF-8", open);
    }
    return webResourceResponse;
} catch (IOException e) {
    e.printStackTrace();
    return null;
}
}
}

```

这里就是 Appcan 的 sdk 了。关注 DESUtility.a 这个方法就行，继续跟进

```

22
23
24 public static String a(String str) {
25     String str2 = null;
26     int lastIndexOf = str.lastIndexOf(47);
27     if (lastIndexOf > 0) {
28         str2 = str.substring(lastIndexOf + 1, str.length());
29     }
30     int lastIndexOf2 = str2.lastIndexOf(46);
31     return lastIndexOf2 > 0 ? str2.substring(0, lastIndexOf2) : str2;
32 }
33
34 public static String a(byte[] bArr, String str) {
35     if (bArr == null || bArr.length == 0) {
36         return "";
37     }
38     int length = bArr.length;
39     int length2 = "3G2WIN Safe Guard".length();
40     if (length <= length2) {
41         return new String(bArr);
42     }
43     if (!"3G2WIN Safe Guard".equals(new String(bArr, length - length2, length2))) {
44         return new String(bArr);
45     }
46     int i = length2 + 256;
47     if (length <= i) {
48         return new String(bArr);
49     }
50     int i2 = length - i;
51     Resources resources = ACDes.getContext().getResources();
52     return nativeHtmlDecode(bArr, str, Integer.toString(i2), a.a(resources.getString(resources.getIdentifier(a.f, EUEUtil.s, ACDes.getContext().getPackageName(), "C"));
53 }
54
55 public static byte[] a(InputStream inputStream, int i) {
56     byte[] bArr = null;
57 }

```

分析后我们发现了“3G2WIN Safe Guard”字符，以及关键方法 nativeHtmlDecode 这是一个 native 方法

```

93     }
94
95     public static native String nativeHtmlDecode(byte[] bArr, String str, String s, String s2);
96 }

```

在代码上面我们发现了 这里 load 了一个 so 文件，"BDebug.TAG=appcan"


```

82 v36,
83 v37,
84 v38,
85 v39,
86 v40,
87 v41,
88 v42);
89 j_mmemset(&v46, 0, 256);
90 j_memcpy(&v45, &unk_2EA4, 256);
91 v28 = j_strlen(&v43);
92 v7 = 0;
93 do

```

WhITeCat安全团队

代码太长了，下面还有很多，就不贴了，看着着实让人头大。目前想到的有两个方法

- 1, 死磕 so 文件的解密方法，然后写一个工具，直接解密
 - 2, 直接用 Xposed hook DESUtility.a 方法把 返回的结果也就是明文的资源 保存起来即可。
- 显然我选第二个

二 编码

xposed 开发规范就不说了，这里直接贴代码

```

public void handleLoadPackage(XC_LoadPackage.LoadPackageParam lpparam) throws Throwable {
    if ("app包名".equals(lpparam.packageName)) {
        Log.d("appcan", "load app " + lpparam.packageName);
        XposedHelpers.findAndHookMethod("android.net.Uri$HierarchicalUri", lpparam.classLoader, "getPath", new XC_MethodHook() {
            @Override
            protected void afterHookedMethod(MethodHookParam param) throws Throwable
            {
                super.afterHookedMethod(param);
                Log.d("Appcan--contentProvider", param.getResult().toString()); //获取 /android_asset/widget/js/swiper-3.4.1.min.js 文件路径
                filename=param.getResult().toString(); //将文件路径 保存到filename里
            }
        });
        Class clazz = lpparam.classLoader.loadClass("org.zywx.wbpalmstar.acedes.DESUtility");
        XposedHelpers.findAndHookMethod(clazz, "a", byte[].class, String.class, new XC_MethodHook() {
            @Override
            protected void afterHookedMethod(MethodHookParam param) throws Throwable
            {
                super.afterHookedMethod(param);
                //Log.d("Appcan--result--xposed", String.valueOf(param.getResult()));
                saveFile(param.getResult().toString(), filename); // hook org.zywx.wbp
            }
        });
    }
}

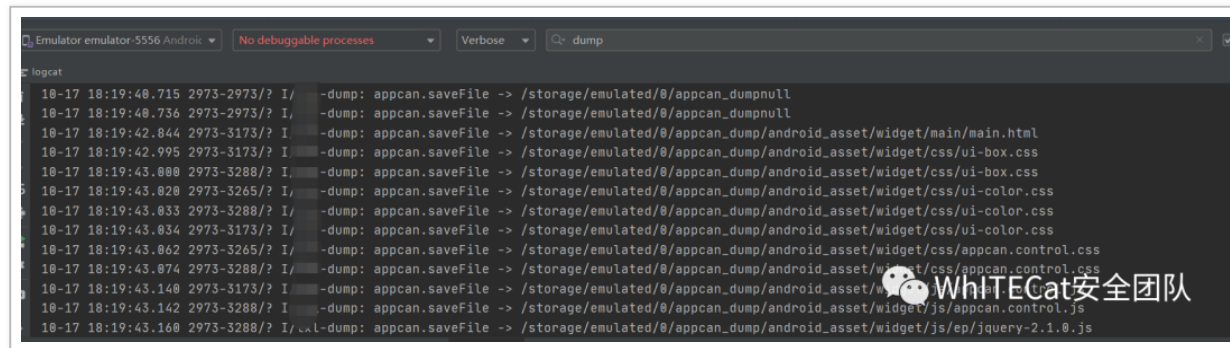
```

almstar.acedes.DESUtility.a方法 返回值 为解密后的源码

```
    }  
    });  
}  
}  
public static void saveFile(String content, String fileName) { //保存文件的函数  
    String basePath = Environment.getExternalStorageDirectory().getPath() + "/appcan_  
dump";  
  
    String filePath = basePath + fileName;  
    Log.i("appcan-dump", "appcan.saveFile -> " + filePath);  
    try {  
        File file = new File(filePath);  
        File parentFile = file.getParentFile();  
        if (parentFile.isFile()) {  
            parentFile.delete();  
        }  
        parentFile.mkdirs();  
        file.createNewFile();  
        IOUtils.write(content, (OutputStream) new FileOutputStream(file), "utf-8");  
    } catch (IOException e) {  
        e.printStackTrace();  
    }  
}
```

这段代码就是 hook org.zywx.wbpalmstar.acedes.DESUtility.a 方法，然后调用 saveFile 方法保存明文资源。

然后安装到模拟器上，在 xposed 激活，重启模拟器后，再次打开 app，可以看到保存文件的日志

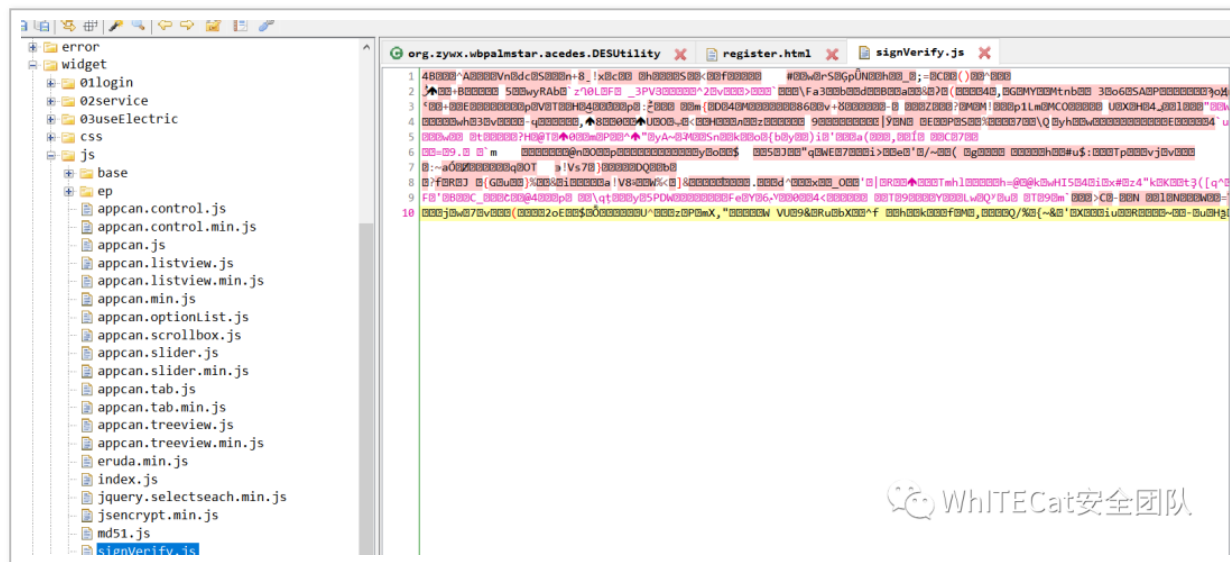


我们打开 sdcard 目录下面的 appcan_dump 目录



和 app 内的一样的结构目录文件都创建好了，如果文件 dump 不全就把每个功能都点一下，就会把文件保存下来，我们打开一个文件来看看

signVerify.js 解密之前：



解密之后：

```
scanme.xml ajsShooter.py shell.jsp 1.jsp test1.js signVerify.js
1 //在此js前面需要引用jseencrypt.min.js和md51.js
2 var publicKey =
  "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCyGffCqoC1vCDLeBvjfuHdw4johGvub0pQjEhhPzW1PbLSRKsNBLgj+eDG0iZE9BwmEwqy16sM0q0kMlhewTQlRrL
  JNlw3L0iogs9WTiGm3e1lSuZLyMnMksnV0NCsuq538cPMNppZRwARb7NXmpmh0KM79fJ/1xqnp0itgRcv4wIDAQAB";
3 var PrivateKey = "MIICdgIBADANBgkqhkiG9w0BAQEFAASCAmAwggJcAgEAAoGBAKbNojYr8KLqKD/y" +
  "C0d7QXu3e4TsrHd4sz3XgDYWEZzgyYqIjVdpcnLztwomgjHj9xSxdpyCc85G06a0" +
  "lva1fNZp66KXYS1xuFa9G7FRbaACoCL31TRv8t4TNkfQhQ7e2S7ZktqyUePWYLz" +
  "u8hx5jXdrIeRiX1jWK1q1NeEd3NagMBAECgYAws70b+4JeBLfRy9pbs/ovpCf1" +
  "bKECLQRILyZBJHpoHKZPzt7k6D4bRFT4irvTMLoQmawXEG09o3UOT8YQLHdRLitW"
4 + "1CYKLy8k8ycyNpB/1L2vP+kHDzmM6Pr0IvkFgnbIFQmXeS5N8V+x0dLAYzuPFkCy" +
  "fUSOKdmt3F/PbF9EhQJBANrF5Uaxmk7q6XfRV7tCT+f27eAWtYi2h/gJenLrmtke" +
  "Hg7SkGdiYHERJ0ns85va4cnhaAzAI1eSIHVaxh3JGxcCQQDDL9ns78LNDr/QuHN9" +
  "pmeDdLQfikeDKzW8dMcUIqGVX4WQJMpvtviZuf3cMvgm9+hDTLVSePdTLA9YSCF4" +
  "VNPbAkEAvbe54XlpCK8IX7iilRkPdGiV1qu614j7FqUZLAkvKrpMeywuQygNXHZ+"
5 + "HuGWTIUfItQfSfdjDrEBBuPMFGZtdwJAV5N3xyyIjfMjM4AFKYhpN333HrOvhHX1" +
  "xVnsHOew8L6KnvHy96x11+xPISN/QYMa24dQo50Am@TOXwbsF73MwJAHzaKZPs" +
  "EN08JunW0Ks3ZS+92maJIm1YGdYf5ip88/Bm3wElNJsCiAeRqYKmAHLcZ5x+Z" + "AsuC1sjcp2r7xw=";
6 var signStr=buildKey();
7
8 var encrypt = new JSEncrypt();
9
10 function addSign(str){
11     return hex_md5(str + signStr);
12 }
13
14 function verSign(str,ver){
15     if(hex_md5(str + signStr)==ver){
16         return true;
17     }else{
18         return false;
19     }
20 }
21
22 function RSASign(str){
23     encrypt.setPublicKey(publicKey);
24     return encrypt.encrypt(str);
25 }
26
27 function RSADecrypt(str){
28     encrypt.setPrivateKey(PrivateKey);
29     return encrypt.decrypt(str);
30 }
31
32 function encodeMine(str){
33     return encodeURIComponent(str).replace("!", "%21");
```

WhITeCat安全团队

ok, 这样我们就可以分析加密参数等一系列操作了

三 结尾

下一步, 就是把 hook 代码改一下, 实现通用的, 不然换一个 app 还得改一下代码。

Over!