“应用克隆”漏洞

1. 漏洞背景

2018 年 1 月腾讯玄武实验室对外披露了名为“应用克隆”的移动应用攻击方法。受害者应用在加载恶意链接后，关键文件被盗取，包括存储于本地的认证 token、隐私等数据，进一步使得攻击者能够伪造用户登录，实现更严重的攻击。检测过程中，多个大厂应用中招。

2. 漏洞原理

多数厂家采用 Hybrid App（Native+Webview）方式进行开发，能够在 APP 中打开 http 网络页面，实现功能的动态更新。一个简单的 Webview 例子如下：

（1）layout 页面布局

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    。。。>
    <WebView
        android:id="@+id/mwebview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"></WebView>
</RelativeLayout>
```

（2）Java 加载代码

```
WebView webView = (WebView) findViewById(R.id.mwebview);
webView.loadUrl("http://www.evil.com"); //webview 加载 www.evil.com 网址内容
```

除了通过加载 http://域网址，webview 还可通过 file://域加载本地文件数据，如修改以上 java 加载代码为

（3）Java 加载代码

```
WebView webView = (WebView) findViewById(R.id.mwebview);
webView.loadUrl("file:///sdcard/data");  //加载 sdcard 下的 data 文件数据
```

理解以上 webview 工作原理后，假设受害者 APP 应用在 webview 打开一个恶意链接后，恶意链接执行 js 脚本尝试通过 file:// 域访问本地的数据，比如/sdcard(外存数据)、/data/data/app/(应用私有目录数据)，一旦成功即可盗取数据传回服务器。

假设包名为 com.xx 的应用打开以下攻击代码，则其私有目录/data/data/com.xx 下的文件可能被盗取。

（4）攻击脚本

```
<script type="text/javascript">
    var url="file:///data/data/com.xx/..."; //具体需要盗取的本地文件
    var xmlhttp;
    if (window.XMLHttpRequest) {
        xmlhttp = new XMLHttpRequest();
    }
    xmlhttp.onreadystatechange = function(){
        if (xmlhttp.readyState == 4) {
            alert(xmlhttp.responseText);
        }
    }
    xmlhttp.open("GET",url);
    xmlhttp.send(null);
</script>
```

根据攻击原理，实施攻击需要满足以下几个条件：

- APP 允许 webview 中打开了恶意链接

- APP 允许 webview 中执行 Js 脚本
- APP 允许 webview 中外部链接跨域通过 file:// 访问本地文件

3. 漏洞检测与修复

（1）白名单检查外部链接

针对第一个攻击条件，需要判断 APP 是否允许加载外部链接，通常攻击入口主要包括：APP 功能（如二维码扫描、外部分享入口）、可信网页中引用外部网页、用户输入、网络攻击（中间人攻击）等。为了避免遭受此类攻击，可对加载网页进行白名单限制。具体可通过 mWebview.setWebViewClient(...)重写 WebViewClient 接口监控 URL 的加载。

（2）限制 webview 中加载网页的 Js 代码执行能力

webView.getSettings().setJavaScriptEnabled(…)接口限制加载网页是否能够具有 Js 执行代码能力。系统默认为 false，不支持 Js 代码执行。

（3）禁止在 webview 中通过 file:// 访问或者跨域访问本地文件

webView.getSettings().setAllowFileAccessFromFileURLs(…)接口可设置是否禁用通过 file://访问本地文件。参数为 true 是表示可访问，false 为不可访问。该接口属于兼容接口，Android 4.1 后参数默认为 false，而 4.0.3 及以下默认为 true。所以为了避免该类漏洞，应显式的设置参数为 false。（注：如果设置了 setAllowUniversalAccessFromFileURLs 接

口为 true 时，该接口设置忽略无效，具体见下一段)。

webView.getSettings().setAllowUniversalAccessFromFileURLs(…)接口设置是否能够跨域通过 file://访问本地文件，参数设置为 true 表示任意网页可以访问本地文件，存在安全漏洞；false 为不可访问。该接口属于兼容接口，Android 4.1 后参数默认为 false，而 4.0.3 及以下默认为 true。所以为了避免该类漏洞，应显式的设置参数为 false。

4. 其他

（1）测试代码复现过程中，js 调用 alert 无法弹出对话框问题，可使用 webView.setWebChromeClient(new WebChromeClient())设置。