Android 中的 X509TrustManager 接口可以作为 X509 证书的信任管理器,来为安全的 sockets 执行身份验证.开发人员可以实现 X509TrustManager 接口,重写证书验证过程来代替库中的实现.在分析第三方 SDK 的过程中,我们发现这些方法的例程是空的.这意味着它们什么也不做,即使在出现非法证书的情况下也不会抛出异常.此外,虽然一些 SDK 执行证书验证,但即便是证书过期或被吊销,它们也没有抛出异常.此类漏洞在第三方 SDK(例如广告平台和推送式消息平台)中相当普遍.根据分析结果,有 20 个 SDK 包含与 SSL/TlS 相关的问题.

3.3 V3:滥用敏感权限

通常情况下,Android 应用程序会请求比所需要的更多的权限.它们使用额外的权限来窥探用户的隐私信息,甚至植入恶意背景的插件.我们的分析显示,16 个 SDK 有上述恶意行为.当应用程序开发人员将第三方 SDK 加入到应用程序中时,会将某些权限、组件、数据等信息添加到 manifest 文件中.

Umeng 是一个推送消息 SDK,可以请求用来发送 SMS、读取 SMS 和接收 SMS 的权限.在对其他推送消息 SDK 分析之后,我们认为这些权限对于核心功能来说并不是必要的.

另外,第三方 SDK 可以与主机应用程序共享 manifest 文件中的权限,也就是说,即使 SDK 在开发文档中没有声明需要某些权限,如果 manifest 文件声明,那么它也可以使用这些权限.这些 SDK 利用代码来检查宿主应用程序是否请求了某个权限(执行此检查的代码示例如图 9 所示).

```
PackageManager pm = getPackageManager();
boolean permission =
    (PackageManager.PERMISSION_GRANTED ==
pm.checkPermission(
 "android.permission.RECORD_AUDIO", "packageName"));
if (permission) {
......
}
else {
......
}
```

Fig.9 Permission check in Android application

图 9 Android 应用中的权限检查

3.4 V4:身份识别

推送消息 SDK 是第三方 SDK 中的一个比较常见的类型,它能够帮助移动应用程序开发商向在用户设备上运行的 APP 传递消息和通知.推送消息 SDK 的结构如图 10 所示.找到这个服务的结构并不困难,但是因为该服务需要协调开发人员与应用之间的交互,这使得它容易出错.

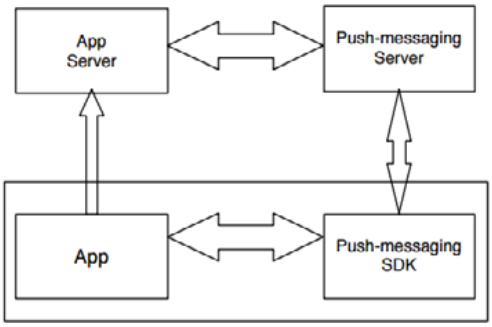

Fig.10 Structure of Push service

图 10 推送服务的架构

由 Google 提供的 Google Cloud Messaging(GCM)SDK 被许多应用程序订阅,包括 Facebook、Oracle、Skype 等,它的运行机制类似于 Apple Push Notification Service.据报道,一些网络犯罪分子使用 GCM 来控制恶意软

件[41,42].除了 Google 和苹果之外,还有许多其他第三方推送消息服务提供商都为应用程序开发人员提供 SDK.

某 Push-messaging SDK:Push-messaging SDK 的云服务器通常利用注册 ID 与相应的应用程序进行通信.然而,我们的分析显示,注册 ID 会泄露用户的敏感信息.经过分析,我们发现此 Push-messaging SDK 所使用的注册 ID 是基于其国际移动设备标识(IMEI)、国际移动用户身份(IMSI)、MAC 地址和包名称在设备上确定生成的.因此,当我们卸载带有推送消息 SDK 的应用程序,然后再在同一个设备上重装后,该应用程序的注册 ID 不会改变.如果设备的敏感信息暴露,攻击者就可以根据它计算注册 ID.由于 SDK 通过注册 ID 来识别特定设备上的应用,所以,如果攻击设备计算出 ID 并且具有受害者设备,那么它就有权从远程服务器接收推送消息.为了防御这种攻击,我们应该增加随机性,例如引入盐值,使得注册 ID 可以不确定地产生.

另外,该 Push-messaging SDK 生成的注册 ID 经对称加密算法(AES)加密后,存储在共享配置中.然而加密密钥是在本地生成,而不是通过与远程服务器的密钥协商来生成(如在 HTTPS 协议中).攻击者可以通过逆向工程获得关键字以及偏移向量,这有助于解密存储在共享配置中的注册 ID.

3.5 V5:本地服务器带来的漏洞

如第 2 节所述,具有本地服务器的第三方 SDK 可以收集设备信息,进而获得设备的控制权.如果本地服务器不适当地执行访问控制,攻击者就可以访问它,检索敏感数据,甚至操纵设备.第 2 节提到,moplus SDK 无法验证请求 URL,因此它可能会被网络犯罪分子触发,这会给包含此 SDK 的所有应用程序带来巨大的威胁.

Baidu Map 是一个著名的移动应用,可以提供在线和离线地图搜索服务.由于中国用户无法访问 Google 地图服务,因此百度地图在中国被广泛应用.第 2 节描述了可能的攻击细节.此外,百度及其他公司开发的约 3 000 款应用程序的某些版本中包含该 SDK,其中有电话助手、输入、云端、浏览器和视频等.虽然其中一些应用的最新版本已删除此 SDK,但仍有大量用户在使用易受攻击的版本.实际上,我们使用 Nmap 扫描了网络上许多设备的 TCP 端口 40310,发现其中许多设备仍然打开此端口.

事实上,还有一些其他具有本地服务器的第三方 SDK 以及许多大众应用程序包含这些 SDK,如 Gaode Map、腾讯和 360.每个相关的应用程序都拥有大量的用户,而且这些应用的旧版本仍然在很多设备上运行.

3.6 V6:未关闭日志造成的信息泄漏

Android 日志系统为开发人员提供了记录应用程序和设备运行状态的接口.日志消息被写入设备的内部存储中.开发人员通常使用 android.util.log 打印调试信息.但是,如果他们在应用上线前未关闭日志,则会成为安全风险.在开发中,开发人员通常使用 debug 属性.该代码确定是否输出日志(如图 11 所示),这使得我们很容易修改调试属性.在 Android 4.1 版本之前,具有 READ_LOGS 权限的 Android 应用程序能够读取设备上所有应用程序的日志文件.因此,将敏感数据写入日志会导致敏感数据泄露.在分析中,我们发现 mapbar SDK(专业的电子地图提供商)会将个人身份信息,如 IMEI 通过日志进行记录.在我们分析的 129 个第三方 SDK 中,有 12 个包含此漏洞.

```
<application android:allowBackup="true"
             android:debuggable="true"
             android:icon="@drawable/ic_launcher"
             android:label="@string/app_name"
             android:theme="@style/AppTheme
 ......
```

Fig.11 Attributes of Log in AndroidManifest.xml

图 11 AndroidManifest.xml 中的有关 Log 的属性

3.7 应用程序开发人员的失误

(1) uid 误用

一些社交平台如 Facebook、Twitter、新浪微博等提供了 SDK 用于第三方登录,这可以帮助用户快速完成

登录或注册过程,无需为当前访问的应用程序注册新帐户.这些 SDK 使用 OAuth 2.0 协议对用户的账户进行身份验证.如果用户通过认证,SDK 的服务器将返回访问令牌和 uid(用户在该平台上的唯一标识)到当前应用程序的服务器.之后,应用程序可以使用访问令牌和 uid 访问用户授权的资源.然而,一些应用程序开发人员只使用 uid 作为用户的凭证,在这种情况下,攻击者可以拦截 uid,并将其篡改为指定 uid 进行登录.

(2) 使用不安全的 API

当第三方 SDK 在 WebView 中使用 JavaScriptInterface 时,远程 Web 页面可以通过这个接口执行本地命令.当 WebView 显示页面时,会在 JavaScript 代码中调用本地代码.远程网页可以利用反射机制来执行自己的命令(如图 12 所示).

```
function execute()
{
  return aboj.getClass()
              .forName("java.lang.Runtime")
            .getMethod("getRuntime",null)
              .invoke(null,null)
            .exec(cmdArgs);
}
```

Fig.12 JavaScript code that executes local commands

图 12 JavaScript 代码执行本地命令

4 讨 论

(1) 第三方 SDK 漏洞的影响

第三方 SDK 越流行,如果存在漏洞,它所造成的安全威胁就越大.例如,Moplus SDK 影响大约 3 000 个~4 000 个下载次数达数百万的应用程序.由于第三方 SDK 是由第三方服务商各自维护,安全水平层次不齐.所以,开发者应该注意引入第三方 SDK 时可能存在的安全风险.

(2) 证书验证

我们的分析结果表明,SSL/TLS 漏洞在第三方 SDK 中很常见.

(3) 第三方 SDK 的保护机制

开发人员可以通过以下方式保护 SDK 的代码:代码混淆;使用 Java 反射机制;将关键代码放在.so 文件中;SSL/TLS 应正确配置;请勿 root 您的设备.

代码混淆、使用 Java 的反射机制以及将关键代码放在.so 文件,使得静态分析第三方 SDK 变得更困难.SSL/TLS 正确配置能够确保通信安全.已 root 与未 root 的设备相比,所面临的安全风险大为增加.

5 结 论

在本文中,我们分析了在 Android 生态系统中具有网络通信功能的第三方 SDK 存在的常见安全隐患.结果显示:在选取的这些 SDK 中,超过 60%含有各种漏洞(例如 HTTP 的误用、SSL/TLS 的不正确配置、敏感权限滥用、身份识别、本地服务、通过日志造成信息泄露、开发人员的失误).我们希望我们的工作可以引起第三方服务商的关注,专注于其 SDK 的安全问题.

References:

[1] IDC. Smartphone OS market share. 2015. http://www.idc.com/prodserv/smartphone-os-market-share.jsp

[2] Wang R, Zhou Y, Chen S, Qadeer S, Evans D, Gurevich Y. Explicating SDKs: Uncovering assumptions underlying secure authentication and authorization. In: Proc. of the USENIX Security. 2013. 399−414.

[3] Google maps API. https://developers.google.com/maps/

[4] PayPal. https://developer.paypal.com/

[5] Permission mapping for Android 2.2.3-4.1.1. http://pscout.csl.toronto.edu/

[6] Amazon Webservices. http://aws.amazon.com/

[7] Book T, Pridgen A, Wallach DS. Longitudinal analysis of android ad library permissions. In: Proc. of the MoST 2013. IEEE, 2013.

[8] Enck W, Octeau D, McDaniel P, Swarat C. A study of android application security. In: Proc. of the USENIX Security 2011. USENIX, 2011.

[9] Grace M, Zhou W, Jiang X, Sadeghi AR. Unsafe exposure analysis of mobile in-app advertisements. In: Proc. of the WISEC 2012. ACM Press, 2012. [doi: 10.1145/2185448.2185464]

[10] Seo J, Kim D, Cho D, Kim T, Shin I. FlexDroid: Enforcing in-app privilege separation in Android. In: Proc. of the NDSS 2016. 2016.

[11] Stevens R, Gibler C, Crussell J, Erickson J, Chen H. Investigating user privacy in Android ad libraries. In: Proc. of the MoST 2012. IEEE, 2012.

[12] The Hacker News. Warning: 18 000 Android apps contains code that spy on your text messages. 2016. http://thehackernews.com/2015/10/android-appssteal-sms.html

[13] The Hacker News. Backdoor in Baidu Android SDK puts 100 million devices at risk. 2016. http://thehackernews.com/2015/11/androidmalware-backdoor.html

[14] Parse Blog. Discovering a major security hole in facebooks Android SDK. 2016. http://blog.parse.com/learn/engineering/discoveringa-majorsecurity-hole-in-facebooks-android-sdk

[15] Poeplau S, Fratantonio Y, Bianchi A, Kruegel C, Vigna G. Execute this! Analyzing unsafe and malicious dynamic code loading in Android applications. In: Proc. of the NDSS 2014. San Diego, 2014. [doi: 10.14722/ndss.2014.23328]

[16] Support V. Security vulnerability in Android SDKs prior to 3.3.0. 2016. https://support.vungle.com/hc/en-us/articles/205142650-Security-Vulnerability-in-AndroidSDKs-prior-to-3-3-0

[17] The Hacker News. Facebook SDK vulnerability puts millions of smartphone users accounts at risk. 2016. http://thehackernews.com/2014/07/facebook-sdkvulnerabilityputs.html

[18] Dropbox Blog. Security bug resolved in the dropbox SDKs for android. 2016. https://blogs.dropbox.com/developers/2015/03/securitybug-resolved-in-thedropbox-sdks-for-android

[19] Shekhar S, Dietz M, Wallach DS. Adsplit: Separating smartphone advertising from applications. In: Proc. of the USENIX Security 2012. USENIX, 2012.

[20] Pearce P, Porter Felt A, Nunez G, Wagner D. AdDroid: Privilege separation for applications and advertisers in Android. In: Proc. of the ASIACCS 2012. ACM Press, 2012. [doi: 10.1145/2414456.2414498]

[21] Yang W, Li J, Zhang Y, Li Y, Shu J, Gu D. Apklancet: Tumor payload diagnosis and purification for android applications. In: Proc. of the ASIACCS 2014. ACM Press, 2014. [doi: 10.1145/2590296.2590314]

[22] Setting the record straight on moplus SDK and the wormhole vulnerability. http://blog.trendmicro.com/trendlabs-securityintelligence/setting-the-recordstraight-on-moplus-sdk-and-thewormhole-vulnerability/

[23] GuardSquare. Proguard java obfuscator. http://proguard.sourceforge.net

[24] Gibler C, Crussell J, Erickson J, Chen H. Androidleaks: Automatically detecting potential privacy leaks in Android applications on a large scale. In: Proc. of the TRUST 2012. Springer-Verlag, 2012. [doi: 10.1007/978-3-642-30921-2_17]

[25] Arzt S, Rasthofer S, Fritz C, Bodden E, Bartel A, Klein J, le Traon Y, Octeau D, McDaniel P. Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for Android apps. In: Proc. of the PLDI 2014. 2014. [doi: 10.1145/2594291.2594299]

[26] Wei FG, Roy S, Ou XM, Robby. Amandroid: A precise and general inter-component data flow analysis framework for security vetting of Android apps. In: Proc. of the CCS 2014. ACM Press, 2014. [doi: 10.1145/2660267.2660357]

[27] Gordon MI, Kim D, Perkins J, Gilham L, Nguyen N, Rinard M. Information-Flow analysis of Android applications in DroidSafe. In: Proc. of the NDSS 2015. 2015. [doi: 10.14722/ndss.2015.23089]

[28] Backes M, Bugiel S, Derr E, Gerling S, Hammer C. RDroid: Leveraging Android app analysis with static slice optimization. In: Proc. of the ASIACCS 2016. ACM Press, 2016. [doi: 10.1145/2897845.2897927]

[29] Wijesekera P, Baokar A, Hosseini A, Egelman S, Wagner D, Beznosov K. Android permissions remystified: A field study on contextual integrity. In: Proc. of the USENIX Security 2015. USENIX, 2015.

[30] Oltrogge M, Acar Y, Dechand S, Smith M, Fahl S. To pin or not to pinhelping app developers bullet proof their TLS connections. In: Proc. of the USENIX Security 2015. USENIX, 2015.

[31] Fahl S, Harbach M, Muders T, Baumgärtner L, Freisleben B, Smith M. Why eve and mallory love Android: An analysis of Android ssl (in)security. In: Proc. of the CCS 2012. ACM Press, 2012. [doi: 10.1145/2382196.2382205]

[32] Egele M, Brumley D, Fratantonio Y, Kruegel C. An empirical study of cryptographic misuse in Android applications. In: Proc. of the CCS 2013. ACM Press, 2013. [doi: 10.1145/2508859.2516693]

[33] AdMob. https://developers.google.com/admob/

[34] Permission mapping for Android 4.1.1-5.1.1. http://pscout.csl.toronto.edu/

[35] SOOT. https://sable.github.io/soot/

[36] Au KWY, Zhou YF, Huang Z, Lie D. Pscout: Analyzing the Android permission specification. In: Proc. of the 2012 ACM Conf. on Computer and Communications Security. ACM Press, 2012. 217−228. [doi: 10.1145/2382196.2382222]

[37] Cui X, Wang J, Hui LC, Xie Z, Zeng T, Yiu S. Wechecker: Efficient and precise detection of privilege escalation vulnerabilities in Android apps. In: Proc. of the 8th ACM Conf. on Security & Privacy in Wireless and Mobile Networks. ACM Press, 2015. [doi: 10.1145/2766498.2766509]

[38] Greenwood DSJSG, Khan ZLL. Smv-Hunter: Large scale, automated detection of SSL/TLS man-in-the-middle vulnerabilities in Android apps. 2014. [doi: 10.14722/ndss.2014.23205]

[39] Wei X, Gomez L, Neamtiu I, Faloutsos M. Permission evolution in the Android ecosystem. In: Proc. of the 28th Annual Computer Security Applications Conf. ACM Press, 2012. 31−40. [doi: 10.1145/2420950.2420956]

[40] Lu L, Li Z, Wu Z, Lee W, Jiang G. Chex: Statically vetting android apps for component hijacking vulnerabilities. In: Proc. of the 2012 ACM Conf. on Computer and Communications Security. ACM Press, 2012. 229−240. [doi: 10.1145/2382196.2382223]

[41] Li T, Zhou X, Xing L, Lee Y, Naveed M, Wang X, Han X. Mayhem in the push clouds: Understanding and mitigating security hazards in mobile pushmessaging services. In: Proc. of the 2014 ACM SIGSAC Conf. on Computer and Communications Security. ACM Press, 2014. 978−989. [doi: 10.1145/2660267.2660302]

[42] Cybercriminals use Google cloud messaging to control malware on Android devices. http://www.pcworld.com/article/2046642/cybercriminals-usegoogle-cloud-messagingservice-tocontrol-malware-on-android-devices.html

马凯(1992－),男,山东德州人,学士,主要研究领域为网络空间安全.

郭山清(1976－),男,博士,副教授,博士生导师,CCF 专业会员,主要研究领域为网络空间安全.