网络直播平台主播地理位置泄露漏洞的分析与利用

乐洪舟[1)] 张玉清[1),2)]

[1)](西安电子科技大学综合业务网理论及关键技术国家重点实验室 西安 710071)

[2)](中国科学院大学国家计算机网络入侵防范中心 北京 101408)

摘 要 2016 年被誉为“中国网络直播元年”,截至 2017 年 5 月,全国在线网络直播平台的数量超过 200 家,网络直播成为当前中国互联网领域最为火爆的事物之一.目前许多网络直播类移动应用(App)都有搜索附近的主播的功能,在某一地点搜索附近的主播时,服务器会返回附近的主播信息以及主播与当前位置的距离.我们从排名前 15 的直播类 App 中找到了 7 个具有这种功能的 App.经过研究发现,这 7 个 App 在“附近的主播”的功能实现上都存在泄露主播地理位置的漏洞.该文首先对 App 开发者常用的距离计算方法进行了分析和总结,将现有的距离计算方法分为球面距离计算方法和平面距离计算方法两种类型,然后针对这两种距离计算方法提出了相应的主播位置定位方法,并针对一些 App 采用的距离模糊措施提出了相应的破解方案.根据 App 在网络通信过程中是否采用加密,提出了绕过加密的漏洞利用方法,并从漏洞利用的简便性和高效性出发,提出了三套漏洞利用方案.通过实验验证了所提出的位置定位方法和漏洞利用方案的有效性,并分析了影响定位准确性和漏洞利用时间的因素,最后,我们给出了漏洞防御方案.

关键词 网络直播;附近的主播;地理位置隐私;隐私泄露;移动应用

中图法分类号 TP393 **DOI 号** 10.11897/SP.J.1016.2019.01095

Vulnerability Analysis and Exploitation of Location Privacy Leakage in Webcasting Platforms

YUE Hong-Zhou[1)] ZHANG Yu-Qing[1),2)]

[1)](*State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an* 710071)

[2)](*National Computer Network Intrusion Protection Center, University of Chinese Academy of Sciences, Beijing* 101408)

Abstract 2016 is known as “The Webcasting Era of China”. By May 2017, there were more than 200 webcasting platforms in China and webcasting is becoming one of the hottest topics in China's Internet. Now many webcasting mobile apps (applications) have the function of searching webcasters nearby. When an audience searches nearby webcasters somewhere, server will return webcasters' information nearby, along with the distance between webcaster and audience. 7 of the top 15 most popular webcasting apps are found to have this function. However, it is found through experiments that all of them have vulnerabilities of leaking webcasters' location privacy. This paper analyzes and sums up developers' commonly used distance calculation methods, and the existing distance calculation methods are divided into two types—spherical distance calculation and plane distance calculation method. The corresponding localization methods are proposed based on these two distance calculation methods, and some crack methods are proposed to deal with the distance ambiguity measures taken by some apps. According to whether app uses encryption in the process of network communication, the vulnerability exploitation methods based on encryption-bypassing are proposed, and to exploit the vulnerability simply and efficiently,

收稿日期:2017-05-03;在线出版日期:2017-11-29.本课题得到国家自然科学基金项目(61272481,61572460)、国家重点研发计划(2016YFB0800703)资助.**乐洪舟**,博士研究生,主要研究方向为移动互联网安全.E-mail:yuehz@xynu.edu.cn.**张玉清**(通信作者),博士,教授,中国计算机学会(CCF)会员,主要研究领域为信息安全.E-mail:zhangyq@ucas.ac.cn.

three vulnerability exploitation frameworks are constructed. The validity of localization methods and vulnerability exploitation frameworks is verified by experiments, and the factors that affect localization accuracy and vulnerability exploitation time are analyzed. At last, defensive measures for the vulnerability are proposed.

Keywords webcasting; webcaster nearby; location privacy; privacy leakage; mobile application

1 引 言

地理位置隐私一直是学术界关注的重要问题，移动应用(以下简称 App)泄漏用户地理位置隐私的事件也屡见不鲜. 常见的地理位置隐私泄露分为两种类型:第一种是开发者在未经过用户同意的情况下,对用户位置的恶意采集和使用[1-4]. 另一种是在开发者未恶意采集用户位置信息的情况下,由于程序中存在漏洞,造成用户的地理位置隐私信息被恶意攻击者所获取[5-6]. 一些研究者发现,在一些社交类 App 中,由于 App 设计的缺陷,攻击者仅仅根据服务器返回的信息便可以计算出用户的地理位置[7-10]. 经过我们最近的研究发现,这种开发者无意中泄露用户地理位置隐私的问题,如今也出现在许多网络直播平台上,并且可能导致更加严重的后果.

2016 年被誉为“中国网络直播元年”①,截至 2017 年 5 月,全国在线直播平台数量超过 200 家,网络直播是当前中国互联网领域最为火爆的事物之一,用户规模超过三亿②. 当前许多手机网络直播类 App 都有搜索“附近的主播”的功能,在某一地点查询附近的主播时,会返回附近的主播信息以及主播与当前位置的距离. 我们从排名前 15 的直播类 App 中找到了 7 个具有这种功能的 App. 并且经过我们的研究发现,尽管一些直播类 App 采用了一些距离模糊手段,防止主播的地理位置被计算出来,但我们依然可以通过运用一些地理和数学运算,计算出这些 App 中主播的地理位置.

因此,本文对网络直播类 App 存在的主播地理位置泄露漏洞进行了研究,主要工作和创新点包括:

(1)对网络直播平台“附近的主播”进行了研究,并发现其中存在的主播地理位置泄露的问题. 我们选择了国内排行前 15 的直播类 App,发现了有 7 个 App 存在“附近的主播”的功能. 并且通过测试发现,这 7 个 App 都存在泄露主播地理位置的漏洞.

(2)分析和总结了厂商常用的距离计算方法,将现有的距离计算方法分为球面距离计算方法和平面距离计算方法两种类型,并给出了针对这些距离计算方法的通用性位置定位方案. 针对一些 App 采用的距离模糊方法如低精度距离、观众方坐标旋转偏移等提出了针对性的应对方案.

(3)根据 App 在网络通信过程中是否采用加密,提出了绕过加密的漏洞利用方法,并从漏洞利用的简便性和高效性出发,提出了三套漏洞利用方案.

(4)针对现有的直播类 App 存在的主播地理位置泄露漏洞的情况,提出了相应的漏洞防御方法.

本文第 2 节对背景知识进行介绍;第 3 节给出厂商常用的距离计算方案;第 4 节针对这些距离计算方案提出相应的位置定位方法;第 5 节讨论漏洞利用的技术方案;第 6 节对直播类 App 进行具体的实验;第 7 节对漏洞的风险和防御方法进行讨论;第 8 节给出关于本文的缺陷和未来工作的讨论;第 9 节介绍相关工作;第 10 节对本文做出总结.

2 背景介绍

2.1 “附近的主播”功能模型

直播类 App 的“附近的主播”功能模型如图 1 所示,图中假设两位主播(A 和 B)和观众 C 在同一个区域内(同一个城市或同一个地区). 主播在开播时,会通过网络向主播信息服务器传送自己的地理位置坐标(Coordinate,包括经度和纬度值),主播信息服务器收到信息后将主播的地理位置保存在本地数据库中. 观众 C 打开 App,并使用“附近的主播”功能搜索附近的主播,App 会向主播信息服务器传送 C 的地理位置坐标,主播信息服务器收到 C 的请求后,将本地数据库中保存的主播坐标按照一定规则取出,并计算各主播与 C 的距离,当算出 A 和 B

① 中国人才网. 2016 年中国网络直播元年. http://gongwen.cnrencai.com/dashiji/123101.html

② 中国产业信息网. 2016 年中国在线直播行业市场发展概况分析. http://www.chyxx.com/industry/201605/414008.html

在 C 附近时，便会向 C 传送 A 和 B 的信息，在这些信息中，可能会包括 A、B 的位置信息，或 C 到 A、B 的距离信息等. 获得这些信息后，C 便可以选择并进入 A 或 B 的直播间观看直播.

2.2　问题提出

为了便于介绍，下文分别用主播方和观众方来表示主播和观众两种对象，以主播方坐标和观众方坐标分别表示主播和观众的地理位置坐标.

在图 1 所示的模型中，观众方通过“附近的主播”功能找到附近的主播，并能够获取其与主播的距离. 虽然这种距离信息没有直接泄露主播的地理位置，然而，早已经有研究表明，在球面（地球）上，已知三点的坐标（经纬度）和到目标点的距离，便可以用三点定位的方法计算出目标点的位置[11]. 这便引出本文要解决的一系列安全问题：

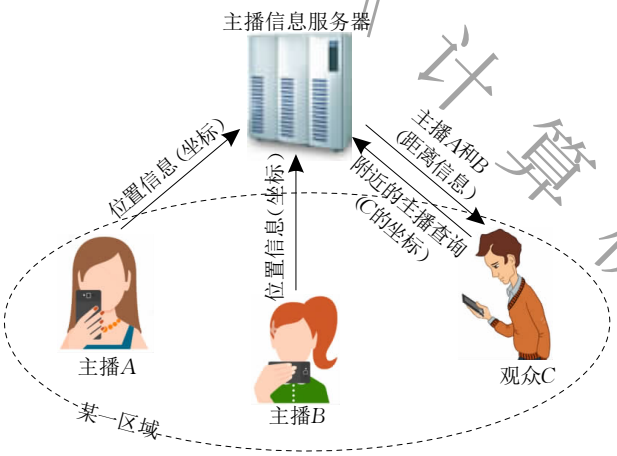

图 1　“附近的主播”功能模型

（1）如果观众方在三个或三个以上不同的坐标位置获得了相应的与主播方的距离，那么主播的地理位置是否可以被计算出来？或者根本不需要计算，主播方坐标本身就隐藏在服务器返回给观众方的信息中，只是没有显示给用户看？

（2）由于主播身份的特殊性，一些 App 开发者会采取一定的措施，防止主播的地理位置被计算出来. 那么现有的直播类 App 所采用的防止主播位置被破解的措施是什么，是否有效？

（3）如果 App 开发者没有采用防御措施或所采用的防御措施无效，那么攻击者如何以最简单、有效的方法对主播方坐标进行破解，效果又如何？

接下来的章节我们将围绕这些问题而展开，依次讨论现有的直播类 App 常用的距离计算方法和基于这些距离计算方法的主播位置定位方法，以及可以采取的漏洞利用技术方案.

3　距离计算

我们研究发现，当前直播类 App 主要采用两种距离计算方法计算观众方与主播方的距离——球面距离计算方法和平面距离计算方法. 本节我们将依次对这两种距离计算方法进行介绍.

3.1　球面距离计算方法

球面距离计算方法考虑到了地球的球面特性，计算两点间的球面距离，是一种正确的、常见的距离计算方法. 我们通过图 2 所示的例子介绍这种距离计算方法.

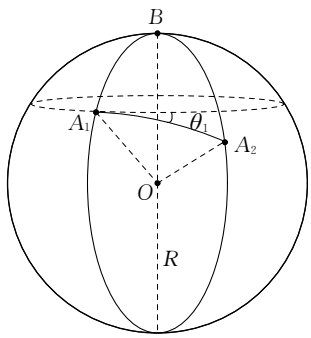

图 2　球面距离计算方法

图 2 中 B 点代表地球的极点，不妨令其代表北极点，$A_1(m_1,n_1)$，$A_2(m_2,n_2)$ 是球面上的两点，其中 m 和 n 分别代表纬度和经度. O 点代表地心，地球的半径用 R 表示. 根据三面角余弦公式，可以得到

$$\angle A_1OA_2=\arcsin(\sin(m_2)\times\sin(m_1)+\cos(m_2)\times\cos(m_1)\times\cos(n_2-n_1)) \tag{1}$$

于是 A_1 与 A_2 之间的距离 l_{12} 可以表示为

$$l_{12}=\frac{\angle A_1OA_2}{180}\times\pi\times R \tag{2}$$

3.2　平面距离计算方法

这种方法不考虑地球的球面特性，把两点当作平面上的两点，计算两点间的平面距离，是一种不标准的计算方法. 虽然不标准，但也起到了距离模糊的效果，被一些 App 所采用. 这种距离计算方法如下：

设每经度的距离增量为 a，每纬度的距离增量为 b，由于地球的各经度线长度一样，因此，对于地球上的任何位置来说，都有

$$b=\frac{2\pi R}{360} \tag{3}$$

a 的值随纬度的变化而变化，赤道的纬度周长最长，因此 a 的值也最大. 而从赤道到两极，由于纬度周长变小，a 的值也会随之变小. 对于地面上某一

点 $A_i(m_i,n_i)$来说,有

$$a=\frac{2\pi R\times\cos m_i}{360} \tag{4}$$

在计算 A_1与 A_2之间的距离 l_{12}时,开发者通常取 A_1与 A_2的纬度之一,不同的开发者选择不同.

在小范围内,由勾股定理,可以得到

$$l_{12}=\sqrt{a^2(n_1-n_2)^2+b^2(m_1-m_2)^2} \tag{5}$$

采用式(3)和(4)的 a、b 的取值计算两点间的距离虽然有误差,却不至于太大.然而我们研究发现,有些采用平面距离计算方法的 App 并不采用式(3)和(4)的 a、b 的取值,而是对 a、b 的取值有所改变,这样计算出来的距离是错误的,但可以达到距离模糊的效果."映客直播"便是采用这种方法,我们将在 6.3 节中具体介绍.

有一些直播 App 采用对观众方坐标进行旋转偏移的办法计算观众与主播的距离,这种方式计算出的距离是不准确的,但却也达到了距离模糊的效果,增加了破解的难度.例如,"一直播"便采用了这种距离模糊方法,我们也将在 6.3 节中对其进行介绍.我们以 $D(m_d,n_d)$、$A(m,n)$分别表示主播方和观众方的地理位置坐标,取 $x=n-n_d$,$y=m-m_d$,以 x 和 y 为坐标轴建立直角坐标系,如图 3 所示,对于坐标系中的任意一点 $C(x,y)$,将 C 围绕原点进行旋转,得到 $C'(x',y')$,以 θ 表示沿逆时针方向旋转的角度,可以得到

$$\begin{cases}x'=\cos\theta\cdot x-\sin\theta\cdot y\\ y'=\cos\theta\cdot y+\sin\theta\cdot x\end{cases} \tag{6}$$

A 与 D 点的距离 l 通过 x',y'求出:

$$l=\sqrt{a^2\cdot[\cos\theta\cdot(n-n_d)-\sin\theta\cdot(m-m_d)]^2+b^2\cdot[\cos\theta\cdot(m-m_d)+\sin\theta\cdot(n-n_d)]^2} \tag{7}$$

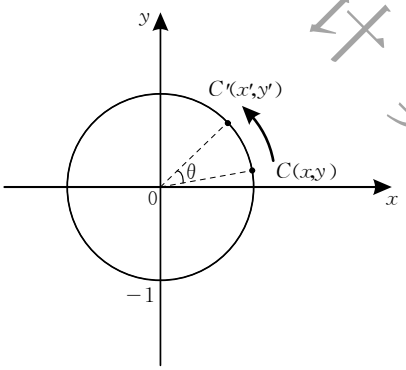

图 3 观众方坐标旋转偏移

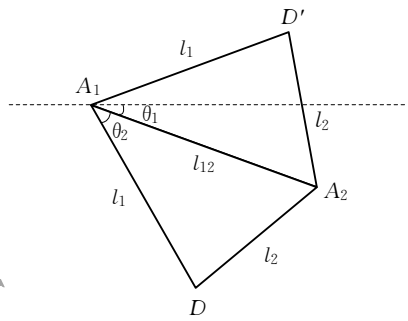

图 4 基于球面距离计算方法的位置定位

4 位置定位

本节分别根据第 3 节中提到的距离计算方法,介绍我们所采用的位置定位算法.

4.1 基于球面距离计算方法的位置定位

对于球面距离计算方法而言,已知三点的坐标和到目标点的距离,便可以用三点定位[11]的方法计算出目标点的位置.我们以 D 代表目标点,三点的坐标表示为:$A_1(m_1,n_1,l_1)$,$A_2(m_2,n_2,l_2)$,$A_3(m_3,n_3,l_3)$,其中 l 表示到 D 的距离.

实际上,使用三点中的两个点就可以将目标点锁定在两个位置.如图 4 所示,由对称性可以确定 D 点可能的两个位置 $D(m_d,n_d)$和 $D'(m'_d,n'_d)$.因此可以先通过 A_1和 A_2算出 D 和 D',然后通过 A_3对 D 和 D'进行验证和取舍.

图 4 中的虚线代表经过 A_1的纬度线,θ_1表示 A_2 相对于 A_1在纬度线上的偏角,θ_2表示 A_1A_2与 A_1D 的夹角.计算步骤如下:

(1)计算 θ_1的值.为了便于表述,我们不妨认为图 4 中的 A_1,A_2和 θ_1对应于图 2 中的 A_1、A_2和 θ_1.根据球面正弦公式,可以得到

$$\frac{\sin\angle A_2OB}{\sin\angle A_2A_1B}=\frac{\sin\angle A_1OB}{\sin\angle A_1A_2B}=\frac{\sin\angle A_1OA_2}{\sin\angle A_1BA_2} \tag{8}$$

在 3.1 节中,已经获得了$\angle A_1OA_2$的值,那么可以利用$\angle A_1OA_2$求出 θ_1的值:

$$\angle A_2A_1B=\arcsin\left(\frac{\cos(m_2)\times\sin(n_2-n_1)}{\sin\angle A_1OA_2}\right) \tag{9}$$

$$\theta_1=|\angle A_2A_1B-90| \tag{10}$$

(2)计算 θ_2的值.根据 3.1 节中的介绍,已知 A_1、A_2的坐标,可以求得 A_1与 A_2之间的球面距离 l_{12}.然后,可以通过余弦定理求得 θ_2的值.

$$\theta_2=\arccos\left(\frac{l_1^2+l_{12}^2-l_2^2}{2l_1l_{12}}\right) \tag{11}$$

(3)计算 $D(m_d,n_d)$和 $D'(m'_d,n'_d)$.

$$\begin{cases}\Delta m=\dfrac{l_1\cos(\theta_1+\theta_2)\cdot 180}{R\cos\left(m_1\cdot\dfrac{\pi}{180}\right)\cdot\pi}\\ \Delta n=\dfrac{L_1\sin(\theta_1+\theta_2)\cdot 180}{R\cdot\pi}\end{cases}\tag{12}$$

$$\begin{cases}\Delta m'=\dfrac{L_1\cos(\theta_2-\theta_1)\cdot 180}{R\cos\left(m_1\cdot\dfrac{\pi}{180}\right)\cdot\pi}\\ \Delta n'=\dfrac{L_1\sin(\theta_2-\theta_1)\cdot 180}{R\cdot\pi}\end{cases}\tag{13}$$

$$\begin{cases}m_d=m_1+\Delta m\\ n_d=n_1+\Delta n\end{cases}\tag{14}$$

$$\begin{cases}m'_d=m_1+\Delta m'\\ n'_d=n_1+\Delta n'\end{cases}\tag{15}$$

(4) 对 D 和 D' 进行取舍. 以 $A_3(m_3,n_3,l_3)$ 作为参考点，根据 3.1 节的球面距离计算方法，D 和 D' 中与 A_3 的距离更为接近 l_3 的点被保留.

4.2　基于平面距离计算方法的位置定位

对于采用平面距离算法的 App，我们依然可以通过三点 $A_1(m_1,n_1,l_1)$，$A_2(m_2,n_2,l_2)$，$A_3(m_3,n_3,l_3)$ 定位目标点 $D(m_d,n_d)$ 的位置. 具体来说，对于 A_1、A_2、A_3 满足以下方程组：

$$\begin{cases}a^2(n_1-n_d)^2+b^2(m_1-m_d)^2=l_1^2\\ a^2(n_2-n_d)^2+b^2(m_2-m_d)^2=l_2^2\\ a^2(n_3-n_d)^2+b^2(m_3-m_d)^2=l_3^2\end{cases}\tag{16}$$

在分母不为零的情况下，解方程可得到 m_d 和 n_d 的值，结果如式(17)、(18)所示.

$$m_d=\frac{\begin{vmatrix}l_1^2-l_2^2+b^2(m_2^2-m_1^2)+a^2(n_2^2-n_1^2) & 2b^2(n_2-n_1)\\ l_1^2-l_3^2+b^2(m_3^2-m_1^2)+a^2(n_3^2-n_1^2) & 2b^2(n_3-n_1)\end{vmatrix}}{\begin{vmatrix}2b^2(m_2-m_1) & 2a^2(n_2-n_1)\\ 2b^2(m_3-m_1) & 2a^2(n_3-n_1)\end{vmatrix}}\tag{17}$$

$$n_d=\frac{\begin{vmatrix}2b^2(m_2-m_1) & l_1^2-l_2^2+b^2(m_2^2-m_1^2)+a^2(n_2^2-n_1^2)\\ 2b^2(m_3-m_1) & l_1^2-l_3^2+b^2(m_3^2-m_1^2)+a^2(n_3^2-n_1^2)\end{vmatrix}}{\begin{vmatrix}2b^2(m_2-m_1) & 2a^2(n_2-n_1)\\ 2b^2(m_3-m_1) & 2a^2(n_3-n_1)\end{vmatrix}}\tag{18}$$

对于 3.2 节介绍的观众方坐标旋转偏移的平面距离计算方法，我们也可以通过解方程定位目标点 $D(m_d,n_d)$ 的位置：

$$\begin{cases}x'_i=\cos\theta\cdot(n_i-n_d)-\sin\theta\cdot(m_i-m_d)\\ y'_i=\cos\theta\cdot(m_i-m_d)+\sin\theta\cdot(n_i-n_d)\end{cases}\tag{19}$$

$$\begin{cases}a^2(x'_1)^2+b^2(y'_1)^2=l_1^2\\ a^2(x'_2)^2+b^2(y'_2)^2=l_2^2\\ a^2(x'_3)^2+b^2(y'_3)^2=l_3^2\end{cases}\tag{20}$$

在分母不为零的情况下，解方程可得到 m_d 和 n_d 的值，结果如式(21)、(22)所示.

$$m_d=\frac{\begin{vmatrix}l_1^2-l_2^2-\alpha(n_1^2-n_2^2)-\beta(m_1^2-m_2^2)-\gamma(m_1n_1-m_2n_2) & \gamma(m_2-m_1)+2\alpha(n_2-n_1)\\ l_1^2-l_3^2-\alpha(n_1^2-n_3^2)-\beta(m_1^2-m_3^2)-\gamma(m_1n_1-m_3n_3) & \gamma(m_3-m_1)+2\alpha(n_3-n_1)\end{vmatrix}}{\begin{vmatrix}\gamma(n_2-n_1)+2\beta(m_2-m_1) & \gamma(m_2-m_1)+2\alpha(n_2-n_1)\\ \gamma(n_3-n_1)+2\beta(m_3-m_1) & \gamma(m_3-m_1)+2\alpha(n_3-n_1)\end{vmatrix}}\tag{21}$$

$$n_d=\frac{\begin{vmatrix}\gamma(n_2-n_1)+2\beta(m_2-m_1) & l_1^2-l_2^2-\alpha(n_1^2-n_2^2)-\beta(m_1^2-m_2^2)-\gamma(m_1n_1-m_2n_2)\\ \gamma(n_3-n_1)+2\beta(m_3-m_1) & l_1^2-l_3^2-\alpha(n_1^2-n_3^2)-\beta(m_1^2-m_3^2)-\gamma(m_1n_1-m_3n_3)\end{vmatrix}}{\begin{vmatrix}\gamma(n_2-n_1)+2\beta(m_2-m_1) & \gamma(m_2-m_1)+2\alpha(n_2-n_1)\\ \gamma(n_3-n_1)+2\beta(m_3-m_1) & \gamma(m_3-m_1)+2\alpha(n_3-n_1)\end{vmatrix}}\tag{22}$$

其中，$\begin{cases}\alpha=a^2\cos^2\theta+b^2\sin^2\theta\\ \beta=a^2\sin^2\theta+b^2\cos^2\theta\\ \gamma=2(b^2-a^2)\sin\theta\cos\theta\end{cases}$.

4.3　平面距离计算方法的相关参数的确定

我们在 3.2 节中介绍了平面距离算法的两个重要参数 a 和 b，若采用式(3)和(4)对 a 和 b 进行赋值，则距离计算的误差并不大. 然而，有些 App 并没有这样赋值，例如映客直播就对 a 和 b 采取了不同的赋值，以达到距离模糊的效果，因此，计算出来的距离存在较大偏差.

为了确定 a 和 b 的值，我们假设已知主播的位置 $D(m_d,n_d)$(通过各种方式，如穷举探索或自己当主播等)，以 $A(m,n,l)$ 代表观众方的坐标(纬度 m，经度 n)和其与 D 点的距离(l)，$x=n-n_d$，$y=m-m_d$，则 x,y,l 满足以下方程：

$$l=\sqrt{a^2x^2+b^2y^2}\tag{23}$$

我们以原点为中心不断改变 A 点位置，使 x,y 满足 $x^2+y^2=C$(C 为正常数). 通过频繁改变 m 和 n 的值，使 x,y 发生改变，获得距离 l，那么随着 (x,y) 沿着圆 $x^2+y^2=C$ 的变化，l 的值呈现如图 5

所示的马鞍线变化趋势.取 $C=1$,则当 $x=0$ 时,l 的值为 b,当 $y=0$ 时,l 的值为 a.于是我们通过大量的实验便可以确定 a、b 的值,并找到 a、b 的取值规律.

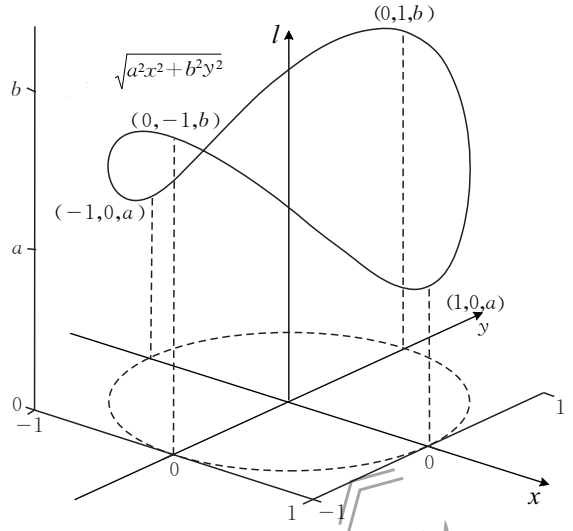

图 5 平面距离计算方法的距离变化趋势

对于 3.2 节中介绍的观众方坐标旋转偏移的平面距离计算方法而言,当 (x,y) 沿着圆 $x^2+y^2=C$ 变化时,l 的值呈现图 6 所示的发生偏移的马鞍线变化趋势,但我们依然可以通过 l 的最大值和最小值确定 a 和 b 的值(当 $C=1$ 时,a 和 b 的取值只会是 l 的最大或最小值之一),并且我们通过分析 l 的最大值和最小值所对应的坐标便可以确定观众方坐标旋转偏移的角度,例如图 6 对应的 θ.

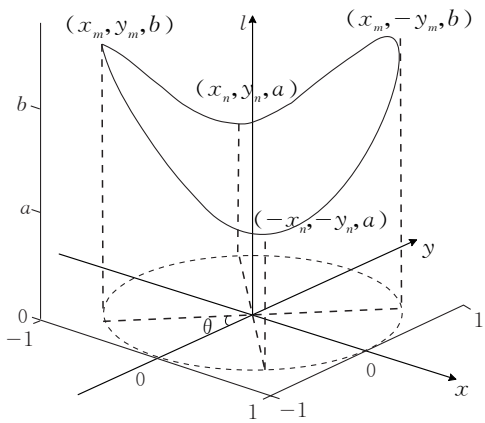

图 6 观众方坐标旋转偏移后的距离变化趋势

4.4 低精度距离的解决

使用低精度距离是一种常用的距离模糊手段[12].有些 App 并不返回主播与观众的准确距离,而是返回大致距离(例如返回精度为 0.1 km 的距离,或显示距离在 0.1 km 以内等),以达到距离模糊的效果.这个大致距离可能是通过四舍五入后保留的,也可能是向上或向下取整后获得的.我们称服务器返回的这种距离为低精度距离,并且称这个距离精度为绝对误差.

对于低精度距离,即使通过大规模穷举探索的方法也无法得到主播的准确位置,因为当观众的位置非常靠近主播时,服务器也只能告诉观众其与主播的距离在什么范围之内.因此,要想通过位置定位算法准确地计算出主播的位置,就必须提高距离精度.历史上已经有学者提出了低精度距离的解决办法[7,9,13],然而现有的方法都是建立在模糊距离相对于真实距离单调递增的基础上的,即在已知 A、B 两点之间的真实距离大于 A、C 两点之间的真实距离的情况下,A、B 两点之间的被模糊后的模糊距离大于 A、C 两点之间的模糊距离.然而,这个条件并不是对所有 App 都成立,例如我们通过实验发现,映客直播和一直播不满足这种单调递增关系,因此,现有的解决低精度距离的方法对其不适用.

为了提高距离精度,我们采用求转折点的办法.转折点是指在无限小的距离范围内,使服务器返回的距离发生变化的观众方坐标点.以采用四舍五入得到的低精度距离为例,当两个观众方坐标 $A_i(m_i,n_i)$ 和 $A_j(m_j,n_j)$ 无限靠近,但服务器返回的它们与主播方的距离 l_i 和 l_j 不相同时,我们认为 A_i 和 A_j 之间发生了四舍五入的转折.A_i 和 A_j 称作转折点.例如,若距离精度为 0.1 km,$l_i=1.1$ km,$l_j=1.2$ km 时,我们认为 $l_i=1.149$ km,$l_j=1.150$ km.此外,由于 A_i 和 A_j 无限靠近,并不适合同时做三点定位中的两个点,我们只能取其中之一作为转折点.每一个转折点都是一个可以获取较高距离精度的点.对于向上或向下取整得到的低精度距离而言,也可以通过类似的求转折点的办法提高距离精度.

我们用具体的算法介绍我们求转折点的方法.若已知一个观众方坐标 $A_1(m_1,n_1)$ 和其与主播方的低精度距离 l_1.我们首先在 A_1 附近选取一点 $A_2(m_2,n_2)$,保证 $l_1\neq l_2$,根据 A_1 和 A_2,我们通过两个步骤便可以求得一个转折点.

步骤 1.求邻近点.邻近点指与目标点的距离相差绝对误差的两个点.步骤 1 可以通过算法 1 来说明,算法 1 中的 *maxError* 表示绝对误差,Coo⟨*lat*, *lon*, *distance*⟩表示观众方的纬度、经度和服务器返回的距离组成的三元组.

算法 1. 寻找邻近点.

输入:Coo A, Coo B, *maxError*

输出:⟨Coo E, Coo F⟩

PROCEDURE *nearbyPoint*(Coo A, Coo B)

1. IF($|A.distance-B.distance|=maxError$) THEN

```
2.    RETURN 〈A,B〉
3.  END IF
4.  C.lat←(A.lat+B.lat)/2
5.  C.lon←(A.lon+B.lon)/2
6.  C.distance←Get distance by network query
7.  IF(C.distance=A.distance) THEN
8.    RETURN nearbyPoint(B,C)
9.  ELSE IF(C.distance=B.distance) THEN
10.   RETURN nearbyPoint(A,C)
11. ELSE IF(|C.distance−A.distance|<|C.distance−
    B.distance|) THEN
12.   RETURN nearbyPoint(A,C)
13. ELSE
14.   RETURN nearbyPoint(B,C)
15. END IF
```

步骤 2. 求转折点. 步骤 2 可以用算法 2 表示. 算法 2 中的 *minDistance* 表示一个极小的距离值，可以人工设定. 若两个观众方坐标 $A_i(m_i,n_i)$ 和 $A_j(m_j,n_j)$ 的距离(球面距离)小于 *minDistance*，但服务器返回的他们与主播方的距离 l_i 和 l_j 不相同时，A_i 和 A_j 二者之一便可以被选作转折点.

算法 2.　寻找转折点.

```
输入：Coo A, Coo B, minDistance
输出：〈Coo E,Coo F〉
PROCEDURE turningPoint(Coo A,Coo B)
1.  IF(Spherical distance of A and B<=minDistance)
    THEN
2.    RETURN 〈A,B〉
3.  END IF
4.  C.lat←(A.lat+B.lat)/2
5.  C.lon←(A.lon+B.lon)/2
6.  C.distance←Get distance by network query
7.  IF(C.distance==A.distance) THEN
8.    RETURN turningPoint(B,C)
9.  ELSE IF(C.distance==B.distance) THEN
10.   RETURN turningPoint(A,C)
11. END IF
```

5　漏洞利用技术方案

这一节我们介绍针对各大直播类 App 存在的主播地理位置泄露漏洞所采用的漏洞利用技术方案. 由于各大直播 App 泄露主播位置的方式不一样，因此，我们根据各大直播 App 的不同特点，从简便、高效地利用漏洞的角度，设计了三套漏洞利用方案.

5.1　位置信息请求响应模型

通过对各个直播类 App“附近的主播”功能进行分析，我们总结出它们共有的主播位置信息请求响应模型，如图 7 所示. 这些 App 都使用第三方定位 SDK 查询移动设备所在的地理位置信息，App 使用当前设备的 GPS、基站信号、WiFi 信号等生成定位依据，并将定位依据发送到相应的定位服务器. 定位服务器对定位依据进行计算得到定位结果，最后将结果返回给 App. App 获得了移动设备的地理位置信息后，取出地理位置坐标(经纬度)，并构造网络请求，发送给主播信息服务器，查询设备所在的地理位置附近的主播信息. 主播信息服务器接收并验证请求后向 App 返回附近的主播信息.

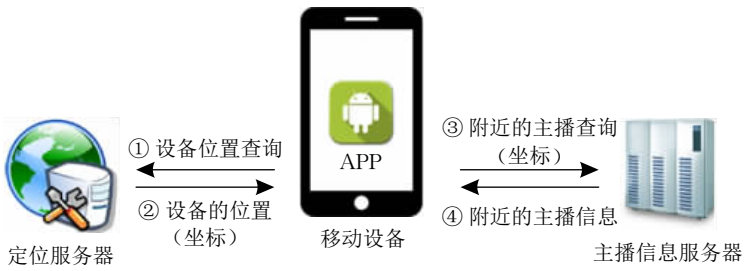

图 7　位置信息请求响应模型

为了提高漏洞利用的效率，我们开发的漏洞利用程序主要运行在电脑端，由电脑构造和发送请求，并接收和处理服务器返回的主播位置信息. 结合第 3 节介绍的位置定位方法，我们可以得出，漏洞利用的关键是漏洞利用程序是否能够向服务器传送伪造的观众方坐标并通过服务器端的验证，以及服务器传回的主播位置信息是否可以被漏洞利用程序读懂并处理. 然而，在具体实现时，我们发现，许多 App 为了保障网络通信的安全和防止观众方伪造网络请求，采用了加密通信的方式，使我们的漏洞利用程序在解决上述两个关键问题上遇到了困难. 为了便于描述，我们将网络通信数据中所有或部分内容的加密，或在通信中增加加密字段、校验码等保密通信的措施统称为加密.

针对加密通信的问题，考虑到不同开发者所采用的加密方案的多样性和破解加密的难度，我们采取的漏洞利用思路是绕过加密. 通过图 7 的整个业务流程可以看出，漏洞利用需要绕过的主要加密是图 7 的②③④三步，根据图 7 的②③④三步是否加密，我们提出了三种漏洞利用方案，并在 Android 设备中进行了实现，在接下来的三节中我们将依次进行介绍.

5.2 单一 C-S 漏洞利用方案

若图 7 的第③和第④步通信数据都没有加密，那么我们采用单一 C-S 方案，即电脑直接向服务器发送携带观众方坐标的请求，并接收服务器端返回的主播位置信息. 通过破解各大直播 App 具体的请求格式，伪造观众方坐标，发送请求，并接收和处理主播信息服务器返回的主播位置信息. 我们在定位一个主播时，通过多次改变观众方坐标，并从服务器返回的主播位置信息中找到该主播的位置信息，结合第 4 节介绍的位置定位方法，便可以计算出该主播的地理位置.

5.3 方法 Hook 方案

若图 7 的第④步通信数据被加密，我们采用方法 Hook 方案. 如图 8 所示，我们通过逆向工程，找到图 7 第③步用到的观众方坐标(Coordinate)在 App 代码中被读取的位置(类名、方法名等)，采用方法 Hook 的方法，注入代码，使 App 在运行中读取 SdCard 中的文本文件所提供的 Coordinate，并以该 Coordiante 作为图 7 第③步网络请求的参数. 同时，我们也使用方法 Hook 的方法，对 App 代码中展示主播位置信息的方法进行拦截，使主播的位置信息输出到 Android 系统日志中.

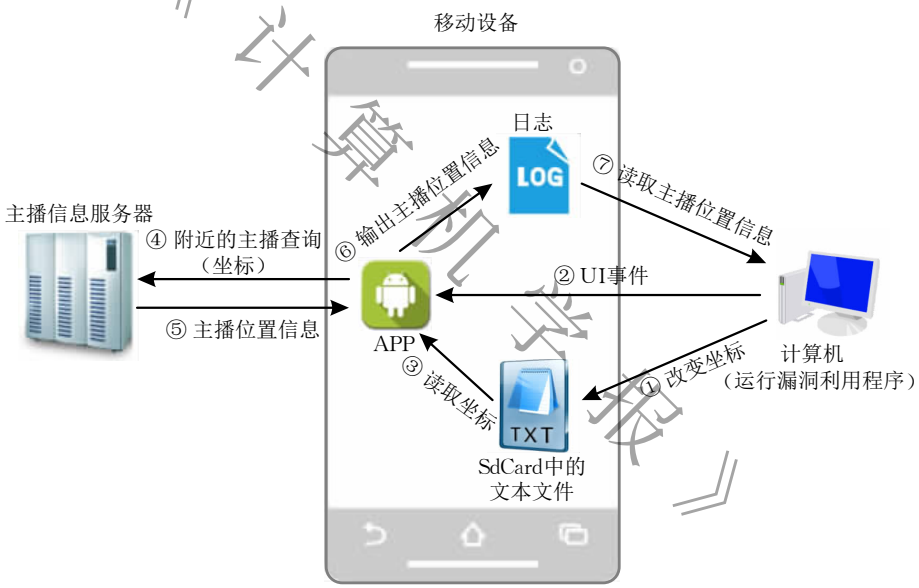

图 8 方法 Hook 模型

我们采用基于 Xposed 框架的 Hook 技术，使用 Root 后的手机，针对 App 的特点编写基于 Xposed 的 Hook 代码. 电脑通过 adb push/pull 命令改变 Coordinate 的值，并通过 adb logcat 命令读取日志中的主播位置信息. 为了实现自动化，减少人工操作的工作量，根据各 App 操作界面的具体特点，电脑端通过 adb input 命令构造特定的 UI 事件，对 App 进行模拟 UI 事件操作，使 App 不断地阅读 sdcard 的 Coordinate，发送“附近的主播”查询请求，并显示主播信息. 在这个过程中，电脑端便可以频繁获取不同的观众方坐标对应的主播位置信息，从而根据第 4 节介绍的定位算法计算出主播的位置.

5.4 Fiddler 代理方案

若图 7 第④步通信没有被加密，而第③步被加密，我们采用 Fiddler 代理方案. 不过 Fiddler 代理方案的前提是图 7 第③、④步所采用的通信协议是 Http 或 Https，因为我们目前还没有找到有效的对手机 TCP 请求进行通信代理的方案.

Fiddler 代理方案是为了降低漏洞利用的难度而产生的，体现了我们所追求的漏洞利用的简便性原则. 使用 Fiddler 代理方案可以避免对 App 进行方法 Hook 或减少方法 Hook 的代码量. 因为方法 Hook 的技术实现较为复杂，需要对 App 进行反编译，看懂反编译后的源码，并根据 App 的代码流程

编写 Hook 代码. 若 App 采用一些特殊化处理(例如:加壳、代码混淆、抗反编译等),逆向工作量大增,漏洞利用的难度也随之增大.

我们所采用的 Fiddler 代理方案的策略是:

(1)若图 7 第②步通信没有加密,我们采用图 9 展示的 Fiddler 代理模型. 这样,就可以避免方法 Hook 操作. 设置手机的 Http 代理为 Fiddler,所有的 Http 请求都经过 Fiddler,我们根据图 7 第②步主播信息服务器返回的数据的格式,保存一个本地文件(以 $f1$ 表示). 每当 Fiddler 代理接收到手机的图 7 第①步请求时,就返回 $f1$ 中记录的位置信息,而不是把这个请求传递给第三方定位服务器. 电脑端就可以方便地更改 $f1$ 中记录的 Coordinate. 当手机使用该 Coordinate 作为图 7 第③步请求的参数时,Fiddler 同样拦截该请求,并将该请求传给电脑端,电脑端再将这个请求的所有参数取出,并构造"附近的主播"查询请求传递给主播信息服务器. 由于这个请求的参数已经被 App 正确的加密,查询请求可以顺利地被服务器端验证通过,并将主播位置信息传给电脑端. 电脑端便可以读取和处理主播位置信息,计算下一步的 Coordinate,修改 $f1$. 此外,和 5.3 节中介绍的一样,电脑端通过 adb input 命令构造特定的 UI 事件对 App 进行模拟事件操作,使其不断地读取 $f1$ 中记录的 Coordiante,并发送"附近的主播"查询请求.

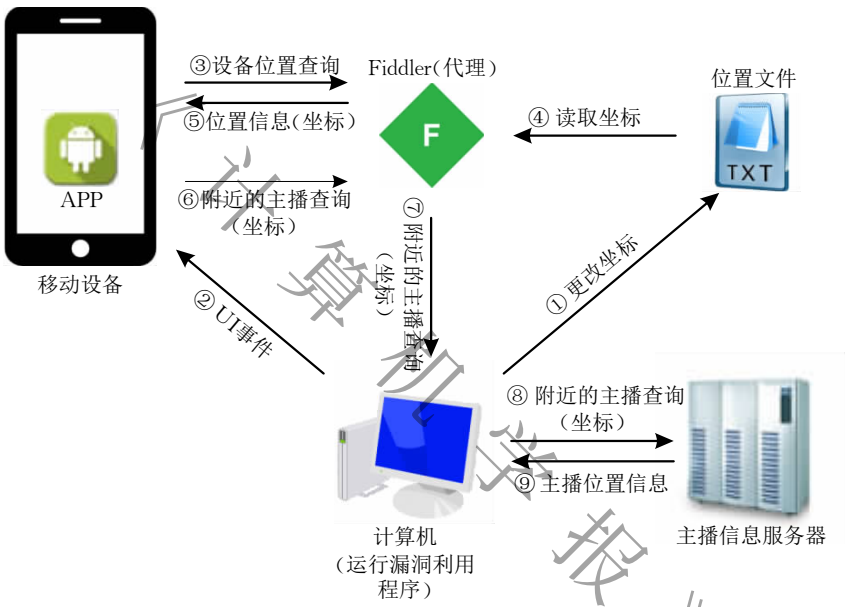

图 9　Fiddler 代理方案模型

(2)若图 7 第②步被加密,就仍然需要采用 Hook 技术绕过加密,但此时只需要对 App 读取 Coordinate 的方法进行 Hook,而不对显示主播位置信息的方法进行 Hook,以减少方法 Hook 的工作量,然后采用 Fiddler 代理的方法,通过 Fiddler 拦截并转发图 7 第③步的"附近的主播"的查询请求. 具体实现方法是将图 9 中的第④步换成图 8 中的第③步即可.

6　实验与分析

我们根据中国报告网所做的"2016 年中国直播类 App 年度排行榜"①,测试了用户活跃程度排名前 15 的直播 App,其中有 7 个 App 存在"附近的主播"功能,我们对这 7 个 App 的安装下载量进行了统计,结果如表 1 所示. 表 1 的第 2 列展示了这 7 个直播 App 的用户活跃程度在排行榜上的排名. 后两列是截至 2017 年 4 月,它们在豌豆荚、360 手机卫士等著名应用市场上的下载量. 从表中我们可以看出,这 7 个 App 是广为流行的直播 App,具有很大的用户量,因此,对它们的研究具有很强的代表性.

表 1　各大直播 App 的排名和下载量

App 名称	排名	豌豆荚/万	360 手机助手/万
YY 直播	2	3446	10619
映客直播	3	625	822
斗鱼直播	4	1706	2299
一直播	8	124	218
花椒直播	9	290	9758
龙珠直播	12	413	249
Now 直播	15	124	47

① 中国报告网. 2016 年中国直播类 App 年度排行榜. http://data.chinabaogao.com/it/2017/01162A9222017.html

经过我们的测试,这 7 个 App 全部存在泄露主播地理位置的漏洞.我们已经将漏洞情况告知相关厂商,并获得了这些厂商的感谢,漏洞也得到了修复.这一节我们将对各个 App 的漏洞情况和我们所采用的漏洞利用技术方案进行介绍.

6.1 各直播 App 的距离隐藏

通过对各个直播 App 进行网络抓包分析,我们发现,一些 App 在 UI 界面上显示的距离精度和实际能够获得的距离精度是不一样的,例如,Now 直播在界面上只显示精度为 100 m 的距离数据,而我们通过网络抓包发现,主播信息服务器在图 7 第④步传回的距离数据已经精确到 1 m,但是在界面上却没有显示这种精度的距离数据.我们推测这也是直播 App 对距离进行模糊处理的手段.我们将 App 客户端对服务器返回的距离数据进行隐藏处理的手段称为客户端距离隐藏.表 2 展示了我们分析出的各个直播 App 的距离隐藏的情况,第 3 列为各个直播 App 在界面上显示的距离精度,第 4 列为主播信息服务器实际返回的距离精度.其中 YY 和斗鱼直播实际返回的距离精度为 1 m,而界面上显示的是 10 m.映客和一直播没有进行客户端距离隐藏,服务器返回的距离和界面显示的距离均为 100 m.

表 2 直播 App 的距离隐藏

App 名称	距离计算方法	界面显示距离精度/m	真实距离精度/m
YY 直播	球面距离计算方法	10	1
映客直播	平面距离计算方法	100	100
斗鱼直播	球面距离计算方法	10	1
花椒直播	直接返回坐标	100	—
一直播	平面距离计算方法(旋转)	100	100
Now 直播	球面距离计算方法	100	1
龙珠直播	直接返回坐标	100	—

此外,我们还发现花椒直播和龙珠直播采用另一种距离隐藏措施,在图 7 第④步网络通信中,由主播信息服务器返回的主播位置信息中直接包含了主播的地理位置坐标,而没有包含距离数据,客户端进行距离计算后将距离显示在界面上.经过我们的测试发现,这些坐标是没有经过模糊处理的准确的主播地理位置坐标.花椒直播返回的坐标的经度被保留 5 位小数,纬度被保留 6 位小数;龙珠直播的经度被保留 4 位小数,纬度被保留 6 位小数.这样的坐标精度足以将主播的位置定位精确到 2 m 以内(可以通过计算每经度和纬度对应的最大距离偏移计算得到).这两个直播 App 在界面上显示的是用户与主播的距离,精度均为 100 m.这实际上是一种非常不安全的距离显示策略,因为这使得我们的漏洞利用程序省略了第 4 节中介绍的定位计算.

6.2 各直播 App 的通信加密情况与漏洞利用技术方案的选择

在第 5 节中我们提到,我们所采用的漏洞利用技术方案追求简单和高效.因此,对于各个直播 App,我们都是选择可以达到的最为简单和高效的漏洞利用技术方案进行漏洞利用.此外,在第 5 节中我们也介绍过,漏洞利用技术方案是根据图 7 第②③④三步网络通信是否加密来选择的.表 3 展示了各个直播 App 与图 7 对应的第②③④三步网络通信的加密情况和我们所采用的漏洞利用技术方案.

表 3 直播 App 的漏洞利用技术方案

App 名称	定位 SDK	②加密	③加密	④加密	漏洞利用方案
YY 直播	Baidu	否	否	否	C-S 模型
映客直播	Amap	否	否	否	C-S 模型
斗鱼直播	Baidu	否	是	否	Fiddler 代理
花椒直播	Haosou	否	是	否	Fiddler 代理
一直播	Amap	是	是	是	Hook
Now 直播	Tencent	是	否	否	C-S 模型
龙珠直播	Tencent	是	否	否	C-S 模型

经过我们的分析,YY、映客、Now、龙珠等 4 个直播 App 在图 7 第③和第④步网络通信中都没有加密,我们对它们的漏洞利用采用单一 C-S 方案.YY、映客和龙珠直播的漏洞利用程序只需要简单地改变网络请求中的观众方坐标(Coordinate),并发送携带有伪造 Coordinate 的请求就可以顺利接收到服务器端的回复,这个过程不需要手机的参与.但 Now 直播需要手机登录后提供一个用户 Cookie,否则图 7 第③步网络通信无法被服务器验证通过.一个有效的 Cookie 可以被多次使用,而不用每次发送请求都改变 Cookie.

斗鱼直播在图 7 第③步网络通信中对请求数据加密,但第②④步未加密,因此我们采用图 9 所示的 Fiddler 代理方案.花椒直播在图 7 第③步网络请求中携带有加密字符串,这个加密字符串由 App 在运行中按照一定的策略生成,为了绕过加密,我们同样采用 5.4 节中介绍的 Fiddler 代理方案对其进行漏洞利用,截获图 7 第②步的网络请求,使其访问本地的伪造位置文件,并转发图 7 第③步的网络请求给电脑端.一直播在图 7 第③和第④步都采取加密,因此我们采用图 9 所示的 Hook 漏洞利用方案对其进行漏洞利用.

6.3 各直播 App 的距离计算方法与定位算法的选择

不同的 App 采用的距离计算方法不同,我们根据各 App 所采用的距离计算方法选择不同的位置定位算法对主播进行定位,除了花椒和龙珠这两个

直接返回主播坐标的直播 App，我们判断其他各直播 App 的距离计算方法所采取的措施是，对于某个直播 App，选取任意三个不同的观众方坐标和对应的距离，若能够使用 4.1 节中的方法准确定位，则我们认为其采用的是球面距离方案. 若不能，我们采用 4.3 节中介绍的马鞍线测试方法测试其是否采用了平面距离方案.

经过我们的分析，总结出了各个直播 App 的距离计算方法，如表 2 的第 2 列所示. 其中，YY、斗鱼、Now 直播采用的是球面距离计算方法，我们对它们采用 4.1 节中介绍的基于球面距离计算方法的位置定位算法进行主播位置定位. 映客直播和一直播采用了平面距离计算方法，我们对其采用 4.2 节中介绍的基于平面距离计算方法的位置定位算法进行主播位置定位.

为了对映客和一直播的距离计算方法的相关参数进行确定，我们采用了 4.3 节中介绍的方法，在北京、西安和武汉三地进行了实地实验，实验结果如表 4 所示. 表 4 列出了我们在三个城市进行实地实验后测出的相关参数. 在每个城市，我们通过自己做主播，然后向服务器端发送伪造观众方坐标的请求以获取主播位置信息. 对于主播方和观众方坐标 $D(m_d,n_d)$ 与 $A(m,n)$，令 $x=n-n_d$，$y=m-m_d$，通过频繁改变 m 和 n 的值，使 x，y 发生改变，并获得 A 与 D 的距离 l. 图 10 和图 11 分别展示了北京地区映客直播和一直播的测试效果图. 从图中可以看出，随着 (x,y) 沿圆 $x^2+y^2=C$（C 分别取 3 种值）的变化，l 的值呈现马鞍线变化趋势.

表 4　映客与一直播距离计算相关参数的确定

城市	主播方坐标	映客		一直播		
		a	b	a	b	θ
北京	(40.4107,116.6809)	49.9	111.3	69.8	130.6	54.9
西安	(34.2332,108.9167)	36.1	111.3	70.2	129.9	55.0
武汉	(40.4107,116.6809)	46.0	111.3	70.3	130.1	55.3

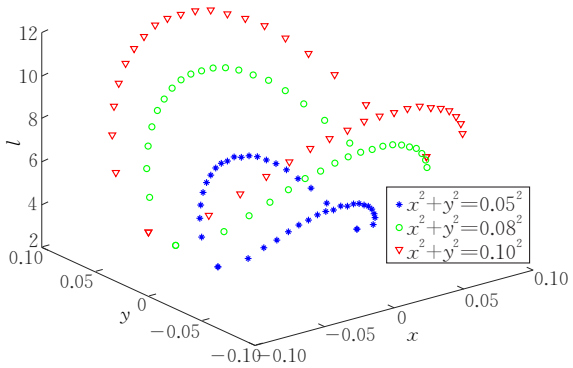

图 10　映客参数确定实验结果(北京)

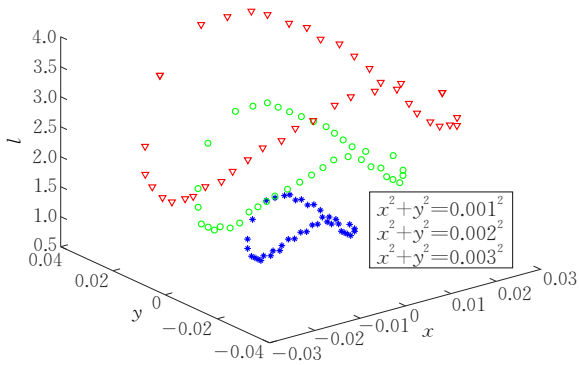

图 11　一直播参数确定实验结果(北京)

测试结果表明，映客直播在进行平面距离计算时，采用的 a 和 b 的值分别为

$$a=\frac{2\pi R}{360},\quad b=\frac{2\pi R\times\cos n}{360}.$$

一直播采用了 3.2 节中介绍的观众方坐标旋转偏移的方法，进行了距离模糊化处理，因此我们采用 4.2 节中介绍的针对观众方坐标旋转偏移的位置定位算法对其进行主播位置定位. 一直播采用的 a、b 和 θ 的值对全国各地来说基本是一样的（$a=70$，$b=130$，$\theta=55$），映客和一直播的这种 a 和 b 的取值使距离计算的结果基本是错误的，我们推测这也是一种距离模糊手段.

此外，映客直播采用的距离模糊策略还包括最小显示距离策略，即当观众方与主播方的距离小于 2 km 时，界面上只显示 2 km，超过 2 km 时，才返回按照其平面距离计算方法算出的距离数据，并保留距离精度为 100 m. 为了保证计算的准确性，我们的漏洞利用程序只采用服务器端返回的 2 km 以上的距离数据，而排除 2 km 内的距离数据.

通过表 2 我们也可以看出，不同的 App 获得的服务器端返回的距离精度不同，因此，我们对低精度距离按照 4.4 节中介绍的求转折点的办法进行了距离精度提升，这 7 个 App 需要进行距离精度提升的只有映客直播和一直播.

6.4　定位准确性分析

为了验证我们对于各个直播 App 的漏洞利用的准确性，我们采用了两种验证方式：

第一种方式是自己做主播，并使用漏洞利用程序对自己进行定位. 具体实现方法是分别在这 7 个 App 中注册了主播账号，在北京地区进行直播测试. 对于每个直播 App，我们在 100 个不同的地址开播，并对这 100 个地址进行了位置定位的准确性测

试.在开播时,我们在 Google 地图上查询到开播的地理位置坐标,该坐标便是主播的实际位置坐标.然后使用漏洞利用程序进行主播位置定位,并将定位计算的结果与实际的主播位置坐标进行比较,并通过 3.1 节中介绍的球面距离计算方法计算距离偏差.

由于花椒和龙珠直播不需要采用位置定位算法进行主播位置定位,而是服务器端直接返回主播方坐标.因此,我们只需要统计服务器端返回的地理位置坐标的准确性,对每个地址只做一次位置定位,统计结果如图 12 所示.图 12 展示了花椒和龙珠直播对应于 100 个不同的开播地址的定位偏差.从图中我们可以看出,花椒和龙珠直播对应的大多数位置定位精度在 50 米以内,这个精度已经足够精确.经过分析,我们认为产生定位距离偏差的主要原因是直播 App 所使用的第三方定位 SDK 本身存在的定位精度偏差.

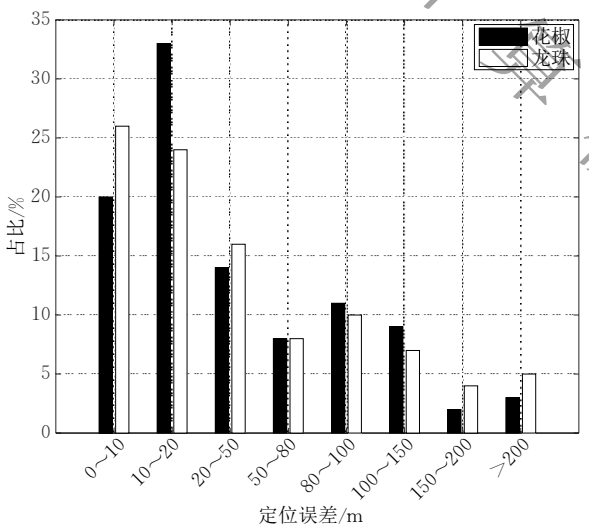

图 12 花椒和龙珠直播定位准确性统计

其余 5 个直播 App 我们采用第 4 节中介绍的三点定位算法进行主播位置定位.在实验中我们发现,用于三点定位的探测点(观众方坐标点)与主播方的距离影响着定位计算的准确性.为了分析探测点的位置与定位误差之间的关系,对于所测试的 100 个不同的开播位置点,我们在每个开播位置点上做 9 次三点定位计算,每次三点定位的探测点与主播方的平均距离分布在 9 个距离区段内(图 13 的横轴所示的 9 个区段).实验结果如图 13 所示,图 13 展示了定位误差随着探测点和主播方之间的距离的变化趋势,横轴表示用来进行三点定位的探测点与主播方的平均距离.

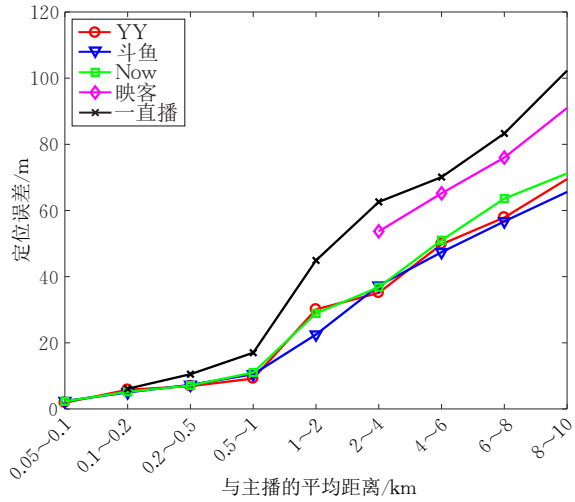

图 13 5 个直播 App 定位准确性统计

实验结果表明,探测点与主播方的距离越大,定位误差越大.我们经过分析认为,这种定位误差是由定位运算本身存在的误差引起的.不管是基于球面距离计算方法的位置定位还是基于平面距离计算方法的位置定位,计算方法本身会导致一定程度的误差.YY、斗鱼和 Now 直播的定位误差随着探测点的距离变化趋势很接近,在同一距离区段内,三者相对于映客和一直播的定位误差要小.映客和一直播由于采用的是平面距离算法,且低精度距离求精的过程也对定位准确性造成影响.因此,二者的定位误差较大.但 5 个直播的定位误差依然在可以接受的范围内.

第二种验证方式是对大量真实的主播进行实际测试,对于每个直播 App,我们都测试了 20 个以上的主播,定位准确性的验证方法是直接与主播进行网上交谈,由主播确定我们的定位计算结果的准确性.因此,定位准确性依赖于主播的主观因素.我们曾尝试着将我们计算出来的地理位置坐标传给主播,并让主播通过查询 Google 地图等方式大致给出计算结果的偏差.然而,由于主播的自我保护意识等原因,愿意配合的主播并不多.此外,由于不同的上网方式本身就存在不同的定位偏差(例如:有研究者经过研究发现,GPS、WiFi、基站三种定位方式的平均精度分别为 10 m,80 m,600 m[14]),因此,这种验证方式难以对定位准确性进行量化分析.但是通过与大量主播进行的实际的交流,我们依然能够确定定位计算结果的准确性.

6.5 漏洞利用的时间复杂度分析

实验表明,漏洞利用的主要时间集中在网络通信、UI 事件、本地数据转发等三个方面.

我们对网络通信的时间复杂度的分析是以观众方发送“附近的主播”查询请求(图 7 第③步通信)的次数为基准的.各直播 App 的时间复杂度如表 5 第 4 列所示.花椒和龙珠直播只需要 1 次“附近的主播”查询即可获得主播方的地理位置坐标,因此其网络通信的时间复杂度为 1.

表 5　直播 App 漏洞利用的时间复杂度分析

App 名称	需要 UI 事件	本地数据转发	网络通信时间
YY 直播	否	无	$3N+1$
映客直播	否	无	$3N\times\mathrm{Log}_2100$
斗鱼直播	是	手机、fiddler、电脑	$3N+1$
花椒直播	是	手机、fiddler、电脑	1
一直播	是	手机、电脑	$3N\times\mathrm{Log}_2100$
Now 直播	否	否	$3N+1$
龙珠直播	否	否	1

注:本地数据转发指电脑、手机、Fiddler 三者之间的数据转发,N 表示定位计算的次数.

对于 YY、斗鱼和 Now 直播,我们采用 4.1 节中介绍的基于球面距离计算方法的位置定位算法进行主播位置定位.三次改变观众方坐标即可定位一个主播,但 6.4 节中的实验结果表明,定位精度随探测点(主播方坐标)与主播方的距离的增加而增加.因此,一次三点定位往往不能精确确定一个主播的位置,往往需要进行多次三点定位才能以较高的精度对主播进行定位,在定位计算的过程中不断缩小主播方和观众方的距离.因此,网络请求次数为 $3N+1$(N 表示三点定位计算的次数),最后一次网络请求用于定位准确性验证,即将定位计算后获得的坐标伪装为观众方坐标传给服务器端,服务器返回的距离数据即为定位偏差.

映客直播和一直播都需要采用 4.4 节中介绍的求转折点的办法进行距离精度提升.由于所采用的算法(算法 1 和算法 2)属于二分算法[15],而二分算法的时间复杂度为对数阶.漏洞利用程序需要将这两个直播 App 的距离精度从 100 m 提升到 1 m,而每次精度提升需要的网络请求次数平均为 Log_2100.因此,这两个直播的网络通信时间复杂度均为 $3N\times\mathrm{Log}_2100$.

对于 Fiddler 代理方案和 Hook 方案而言,每次发送主播位置信息查询请求,都需要漏洞利用程序处理电脑、手机、Fiddler 代理等三者之间的数据转发,而这些数据转发均需要耗费较多的时间.表 5 第 3 列展示了各个 App 的本地数据转发所涉及的对象,花椒直播和斗鱼直播由于采用 Fiddler 代理方案,需要处理电脑、手机、Fiddler 代理三者之间的数据转发,一直播使用 Hook 方案,只需要处理电脑和手机之间的数据转发.

UI 事件也是耗费漏洞利用时间的一个重要方面,电脑端通过 adb input 命令向 App 发送 UI 事件流,通过 UI 事件的驱动,App 便可以频繁的发送主播位置信息查询请求.如表 5 第 2 列所示,在这 7 个直播 App 中,只有花椒、斗鱼和一直播需要进行 UI 事件操作.

从表 5 可以看出,映客和一直播定位主播的时间复杂度是最高的,而花椒直播和龙珠直播的时间复杂度最低.其他 3 个直播 App 大致相同,时间复杂度适中.

7　漏洞风险分析与防御方法

根据第 6 节中的实验结果,这一节我们对各个直播 App 进行漏洞风险分析,并提出可行的漏洞防御方法.

7.1　漏洞风险分析

对于第 6 节中被测试的 7 个 App,虽然它们各自或多或少采取了一些距离模糊措施,例如客户端距离隐藏、低精度距离、通信加密、观众方坐标旋转偏移等,防止主播的地理位置被计算出来,但我们通过具体的实验证明它们都是不安全的,主播的地理位置均可以被破解.攻击者能够做到,对于某一个区域(城市、地区)的一个正在直播的主播,只要他(她)能够出现在“附近的主播”中(有些直播公司出于各种原因,使某些主播不出现在“附近的主播”中),那么我们就能计算出主播所在的地理位置坐标.而一旦获得了地理位置坐标,便可以通过 Google 地图等在线地图对主播进行定位.给定一个区域,就可以计算出这个区域内所有正在直播的主播的位置;频繁改变观众方坐标,就可以对全国任何位置的主播进行定位.

主播的地理位置被破解的危害不言而喻,主播一般都不希望自己的地理位置被泄露,受到疯狂粉丝的骚扰.但最令我们惊讶的是,各个直播 App 却都没有给主播预留一个选项,使主播可以选择是否允许自己被显示在“附近的主播”功能项中.而这样的选项设置我们在一些社交类 App 如微信、陌陌中都有见到,是一种比较成熟和有效的用户地理位置保护措施.此外,我们还在实验中发现一些直播 App 采取了对主播进行有选择的保护措施,例如,YY 直播的“附近的主播”的功能中基本上见不到拥有较高人气的大主播,这或许是 YY 直播对于大主

播的一种保护措施,类似的现象在其他直播 App 中也有所体现,但我们不认为这种只对部分人的隐私进行保护的隐私保护策略是正确和完整的.

7.2 漏洞的防御方法

主播位置的泄露是一种比较严重的安全漏洞,主播的个人隐私安全也应当得到保护.这一节,我们分别从 App 开发者和主播两个角度来讲述漏洞的防御方法.

从 App 开发者的角度,我们认为开发者可以考虑以下 3 点:

(1)增强距离模糊手段.我们认为这些直播 App 存在主播位置泄露漏洞的根本原因在于,缺乏有效的距离模糊手段.历史上已有学者提出较为有效的距离模糊手段[7,13,16],因此本文不打算提出新的距离模糊手段,而是推荐网络直播厂商采用一些安全性高的距离模糊手段.例如,文献[7,13]等均提出了采用地理网格技术对距离进行模糊操作,是当前比较有效的距离模糊手段.按照一定规则对地面进行网格划分,位于不同网格的两个点的距离等于它们之间的网格长度.如图 14 所示,主播和观众之间相距 3 个网格,那么模糊后的距离数据为 3 个网格的距离.

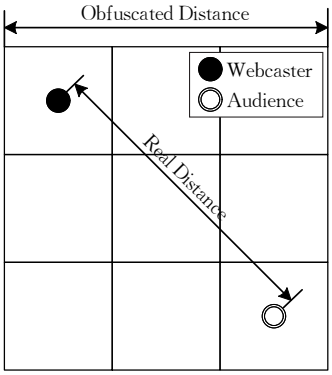

图 14 基于地理网格的距离模糊

此外,我们认为,虽然使用这些安全性较高的距离模糊方法能够取得一定的防御效果,但不管采用什么样的距离模糊手段,只要是参与距离计算的是主播的真实坐标,便存在巨大的隐患.因此,我们认为最简单有效的距离模糊方法是不使用真实的主播坐标参与距离计算.服务器端可以对每个主播的位置坐标进行随机的模糊处理后,产生一个错误坐标,并在服务器端保存一个主播与错误坐标的映射关系.在计算主播与观众之间的距离的时候,使用这个错误的坐标参与距离计算.若采用这种方式,客户端是无法破解主播的地理位置坐标的,并且只要这个错误坐标与真实的主播坐标相差不是太大,不至于会对用户体验造成很大影响.

(2)识别和限制恶意网络请求.没有对恶意网络请求进行有效识别和限制也是 7 个直播 App 存在泄露主播地理位置漏洞的原因.例如,在 6.5 节中我们介绍过,映客直播和一直播破解一个主播的坐标,都需要向服务器端发送多次请求,每次请求均需要伪装客户端坐标.这些请求具有明显的恶意性质,而映客直播和一直播均没有对网络请求进行辨别,导致主播地理位置的泄露.因此,服务器端应当加强对类似的恶意请求的识别和限制,当监测到恶意的主播信息请求后,可以采取一定的策略提供非真实信息,以阻碍攻击者对主播的定位计算.

(3)App 开发者应当给用户一个选项,使主播自己选择是否出现在观众方"附近的主播"功能界面中.在 7.1 节中我们也提到过,这种措施被目前许多有类似功能的社交类 App 所采用,效果良好,但在直播类 App 中,我们却没有见到过这种选项设置,这是直播类 App 的一个安全缺陷.

从主播的角度,我们认为,在主播无法确定直播 App 是否可以保证自己的地理位置隐私安全的情况下.可以采取措施向服务器提供伪造的坐标,例如可以使用 Root 后的手机,安装位置伪装 App(如天下游、位置伪装大师等),对直播 App 进行地址伪装操作.或者若 App 在不被授予地理位置权限时依然可以运行的话,可以在手机的应用设置中撤销直播 App 的地理位置查询权限.

8 讨 论

对于具有"附近的主播"功能的 7 个直播类 App,我们已经成功使用自动化的漏洞利用程序对各 App 进行了主播位置定位,并取得了良好的效果.然而,我们的研究工作依然存在以下缺陷有待改进:

(1)对定位计算结果进行量化分析较为困难.我们在 6.4 节中介绍过,为了验证定位计算结果的准确性,我们实施了两种测试,第一种是自己做主播进行测试,第二种是测试实际的主播.然而这两种方式在对计算结果的量化分析上都做的不到位,但我们目前还没有找到更好的量化分析办法.对于第一种测试方法而言,自己做主播会受到地域的限制,测试的地域有限(目前只限于实验团队所在的地区),测试结果量化分析的说服力不强.对于第二种测试

方法而言，由于受到主播的主观意识和自我保护意识的影响，也难以对定位计算结果进行准确量化分析.

(2)位置定位方法有待扩展. 由于本文主要是对直播类 App 进行研究和归纳总结，因此第 4 节介绍的位置定位方法对我们所测试的 7 个 App 都是有效的，但实际上，我们通过分析其他一些社交类 App 时发现，某些具有类似功能(例如“附近的人”功能)的社交类 App 采用了一些其他的距离模糊手段，但我们现有的定位计算方法还不能够解决.

在今后的研究工作中，我们将找到更有效的办法对定位计算结果进行更充分、更准确的量化分析，并对位置定位算法进行扩展，使其能够应对更多的距离模糊手段，深入剖析现有的位置定位方法的安全问题，提出更有效的位置隐私保护措施. 此外，为了深入研究主播位置泄露的危害，我们下一阶段还打算从宏观角度研究对全国范围内的主播进行全局攻击的安全问题.

9　相关工作

随着移动互联网的快速发展，移动应用面临的用户位置隐私泄露的安全问题也越来越严重，用户的地理位置隐私受到移动广告、移动系统平台、App 开发商等多方面的安全威胁[17]. 甚至有研究者披露，许多移动操作系统中的位置信息访问控制策略本身是无效或低效的[18]，导致移动应用用户面临的位置隐私泄露的安全问题更为突出和严重. 为了保护用户的地理位置隐私，一些研究者采用伪装地理位置的方式，向服务器发送虚假的地理位置和路线来防止使用位置服务的用户被追踪[19-21]. 为了增强移动操作系统对用户地理位置隐私的保护，一些研究者提出了修改系统源码的方式. 例如，Amini 等人[22]提出了为移动操作系统增加位置缓存，使 App 访问位置信息时优先访问该缓存，通过控制缓存的内容达到保护用户位置信息的效果. Fawaz 等人[23]提出了 LP-Guardian 框架，对系统中的各个 App 进行独立的、有区别的保护，在保护用户的地理位置信息不被泄露的同时，保障各个 App 的正常运行. 为了防止用户遭遇地理位置试探攻击，一些研究者提出了定位验证的办法来限制攻击者的地址伪装[24-27]，然而它们需要可信的网络基础设施的支持，并且只能在较小的距离范围内才有效，因此，其实用性不强.

一些研究者对社交类 App“附近的人”功能进行了研究，并研究了其中存在的用户位置隐私泄露的问题. 我们认为这些工作是与本文的研究内容最为相关的. Li 等人[7]发现了一些社交类 App 如 Wechat、Skout、Momo 等的“附近的人”功能存在泄露用户位置信息的问题，并提出了用于位置定位的算法. 但他们的方法只考虑到了球面距离算法，没有考虑到我们在 3.2 节中所介绍的一些厂商所使用的平面距离算法，甚至是旋转后的平面距离算法. 因此，对平面距离算法来说，他们的位置定位方法是无效的. Polakis 等人[13]用形式化的方法描述了当前普遍存在的三种用户位置隐私泄露模型，并提出了相应的攻击和防御方法. 然而其针对距离模糊的位置定位方法具有局限性，只对模糊距离相对于真实距离单调递增的距离计算方法有效，而对映客和一直播等不满足单调递增关系的平面距离计算方法无效. Zhao 等人[8]也指出了一些社交类 App 的“附近的人”功能存在的位置泄露问题，并介绍了对用户进行定位的技术手段. 然而他们并没有提出有效的定位算法，并对定位算法的效率进行分析. 没有考虑一些厂商所采用的平面距离算法，没有解决 4.4 节中提出的低精度距离的问题. 对待加密通信的问题所采用的方法是反编译 App，然后破解加密，该方法在通用性上要低于我们所提出的加密绕过策略. Xue 等人[9]提出了一种通过数论的方法解决我们在 4.4 节中提出的低精度距离的问题，虽然其时间复杂度也为对数阶，但他们所采用的方法在观众方坐标的探针选择上并不能保证严格遵循二分原则，降低了计算速度. 此外，他们的算法只对球面距离计算方法或标准的平面距离计算方法有效，而对 a、b 扭曲取值的平面距离算法或观众方坐标旋转偏移的距离计算方法无效. Hoang 等人[10]对 3 个 App——Grindr、Jack'd 和 Hornet 进行了分析，发现尽管这 3 个 App 采用了一些距离模糊措施，甚至是设置了取消距离显示的功能，用户的地理位置还是能够被定位，并对定位方法进行了具体实现. 然而，他们针对 3 个 App 的特点进行的研究不具有通用性，没有总结厂商常用的距离计算方案. 没有研究漏洞利用的通用解决方案. 其漏洞利用方法不能运用在部分直播类 App 的漏洞利用上.

10 结束语

本文对当前流行的直播类 App 的“附近的主播”功能进行了研究,实验证明具有这项功能的各直播 App 均存在泄露主播地理位置的漏洞.我们对各直播 App 的距离计算方法进行了分析和概括,并针对这些距离计算方法提出了相应的位置定位算法,从漏洞利用的简便性和高效性出发,提出了三套漏洞利用方案,同时我们也提出了相应的防御措施.随着网络直播在中国的快速发展,主播个人的隐私保护显得越来越重要,我们希望通过这项研究呼吁直播厂商以至于社会各界人士关注主播的个人信息安全.

参 考 文 献

[1] Kushwaha A, Kushwaha V. Location based services using android mobile operating system. International Journal of Advances in Engineering & Technology, 2011, 1(1): 14-20

[2] Dhar S, Varshney U. Challenges and business models for mobile location-based services and advertising. Communications of the ACM, 2011, 54(5): 121-128

[3] Grace M C, Zhou W, Jiang X, et al. Unsafe exposure analysis of mobile in-app advertisements//Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks. Tucson, USA, 2012: 101-112

[4] Book T, Wallach D S. A case of collusion: A study of the interface between ad libraries and their apps//Proceedings of the 3rd ACM Workshop on Security and Privacy in Smartphones & Mobile Devices. Berlin, Germany, 2013: 79-86

[5] Zhou X, Demetriou S, He D, et al. Identity, location, disease and more: Inferring your secrets from android public resources//Proceedings of the 20th ACM Conference on Computer & Communications Security. Berlin, Germany, 2013: 1017-1028

[6] Lu L, Li Z, Wu Z, et al. CHEX: Statically vetting android apps for component hijacking vulnerabilities//Proceedings of the 19th ACM Conference on Computer and Communications Security. Raleigh, USA, 2012: 229-240

[7] Li M, Zhu H, Gao Z, et al. All your location are belong to us: Breaking mobile social networks for automated user location tracking//Proceedings of the 15th ACM International Symposium on Mobile ad Hoc Networking and Computing. Philadelphia, USA, 2014: 43-52

[8] Zhao S, Luo X, Bai B, et al. I know where you all are! exploiting mobile social apps for large-scale location privacy probing//Proceedings of the 21st Australasian Conference on Information Security and Privacy. Melbourne, Australia, 2016: 3-19

[9] Xue M, Liu Y, Ross K W, et al. I know where you are: thwarting privacy protection in location-based social discovery services//Proceedings of the 2015 IEEE Conference on Computer Communications Workshops. Hong Kong, China, 2015: 179-184

[10] Hoang N P, Asano Y, Yoshikawa M. Your neighbors are my spies: Location and other privacy concerns in dating apps//Proceedings of the 18th International Conference on Advanced Communication Technology. Pyeongchang, Korea, 2016: 715-721

[11] Murphy W, Hereman W. Determination of a position in three dimensions using trilateration and approximate distances. Department of Mathematical and Computer Sciences, Colorado School of Mines, Golden, Colorado: Technical Report: MCS-95-07, 1995

[12] Ardagna C A, Cremonini M, Damiani E, et al. Location privacy protection through obfuscation-based techniques//Proceedings of the 21st IFIP Annual Conference on Data and Applications Security and Privacy. Los Angeles, USA, 2007: 47-60

[13] Polakis I, Argyros G, Petsios T, et al. Where's wally?: Precise user discovery attacks in location proximity services//Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. Denver, USA, 2015: 817-828

[14] Zandbergen P A. Accuracy of iPhone locations: A comparison of assisted gps, wifi and cellular positioning. Transactions in GIS, 2009, 13(s1): 5-25

[15] Bentley J L. Multidimensional binary search trees used for associative searching. Communications of the ACM, 1975, 18(9): 509-517

[16] Ardagna C A, Cremonini M, di Vimercati S D C, et al. An obfuscation-based approach for protecting location privacy. IEEE Transactions on Dependable and Secure Computing, 2011, 8(1): 13-24

[17] Andrienko G, Gkoulalas-Divanis A, Gruteser M, et al. Report from dagstuhl: The liberation of mobile location data and its implications for privacy research. ACM SIGMOBILE Mobile Computing and Communications Review, 2013, 17(2): 7-18

[18] Fawaz K, Feng H, Shin K G. Anatomization and protection of mobile apps' location privacy threats//Proceedings of the 24th USENIX Security Symposium. Washington, USA, 2015: 753-768

[19] Krumm J. Realistic driving trips for location privacy//Proceedings of the 7th International Conference on Pervasive Computing. Nara, Japan, 2009: 25-41

[20] Meyerowitz J, Roy Choudhury R. Hiding stars with fireworks: Location privacy through camouflage//Proceedings of the 15th Annual International Conference on Mobile Computing and Networking. Beijing, China, 2009: 345-356

[21] Gao S, Ma J, Shi W, et al. LTPPM: A location and trajectory privacy protection mechanism in participatory sensing. Wireless Communications and Mobile Computing, 2015, 15(1): 155-169

[22] Amini S, Lindqvist J, Hong J, et al. Caché: Caching location-enhanced content to improve user privacy//Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services. Bethesda, USA, 2011: 197-210

[23] Fawaz K, Shin K G. Location privacy protection for smartphone users//Proceedings of the 21st ACM SIGSAC Conference on Computer and Communications Security. Scottsdale, USA, 2014: 239-250

[24] Brassil J, Manadhata P K, Netravali R. Traffic signature-based mobile device location authentication. IEEE Transactions on Mobile Computing, 2014, 13(9): 2156-2169

[25] Narayanan A, Thiagarajan N, Lakhani M, et al. Location privacy via private proximity testing//Proceedings of the 18th Annual Network & Distributed System Security Symposium. San Diego, USA, 2011

[26] Saroiu S, Wolman A. Enabling new mobile applications with location proofs//Proceedings of the 10th Workshop on Mobile Computing Systems and Applications. Santa Cruz, USA, 2009: No. 3

[27] Zheng Y, Li M, Lou W, et al. Sharp: Private proximity test and secure handshake with cheat-proof location tags//Proceedings of the 17th European Symposium on Research in Computer Security. Pisa, Italy, 2012: 361-378

YUE Hong-Zhou, Ph. D. candidate. His research interests include mobile Internet security and web security.

ZHANG Yu-Qing, Ph. D., professor. His research interests include network and information system security.

Background

Location privacy is always a heated topic in academic circles. With the rapid development of mobile Internet, the location privacy of mobile app (application) user is facing more and more security threats. These security threats come from various aspects, such as the app vendors, third party advertisers and malicious attackers. It is studied by several researchers that, for some LBSN (location-based social networks) apps, even without hacking the server, an attacker can calculate out a user's location by legal network request. The function of "People Nearby" in some LBSN apps is such a case. Researchers found that some LBSN apps which have this function can leak user's location privacy by showing distance information on the UI (User interface). With this distance information, the locations of nearby users can be calculated by the method of trilateration localization. However, none of these works propose a general calculation model and practical vulnerability exploitation framework. Therefore, their methods cannot solve more problems of location privacy leakage caused by similar functions in other mobile apps.

This paper shows a new scenario of location privacy leakage which is also caused by showing distance information of nearby users. But this time, the users whose location privacy be leaked are webcasters, a newly developing and more privacy sensitive social group. Nowadays in China, webcasting is becoming the hottest topic in Internet. Many webcasting mobile apps (applications) have the function of searching webcasters nearby. When searching nearby webcasters somewhere, server will return webcasters' information nearby, along with the distance between webcaster and audience. 7 of the top 15 most popular webcasting apps are found to have this function. It is found through this study that all of them have vulnerabilities of leaking webcasters' location privacy. This paper analyzes and sums up developers' commonly used distance calculation methods. Corresponding localization methods are proposed based on these distance calculation methods. From the viewpoint of simplicity and efficiency of vulnerability exploitation, three vulnerability exploitation frameworks are proposed. Experimental results show that the localization methods and vulnerability exploitation frameworks are effective and efficient.

This work is supported by the National Natural Science Foundation of China (61272481, 61572460) and the National Key R&D Program of China (No. 2016YFB0800703).