基于多云存储的 Android 密钥管理技术

余宇劲 凌 捷
(广东工业大学计算机学院 广东 广州 510006)

摘 要 针对 Android 客户端的密钥管理难题,提出一种结合多云存储和秘密共享的 Android 密钥管理技术。与传统的 Android 端加密技术不同,其不在移动设备上存储密钥、用户指纹、图案密码等敏感信息,而是转化为在云端存储秘密份额。选择对称加密算法,结合一次性口令生成密钥,加密文件并存储密文;通过拉格朗日插值多项式为 n 个云服务端秘密共享者产生各自的秘密份额,删除口令和密钥信息,上传秘密份额至 n 个云服务端;用户解密时,客户端随机向 $m(m \leqslant n)$ 个云服务端发出请求,进行验证,返回相应的份额,在客户端进行重组,得到密钥进行解密。在多云场景下,对 KB 级至 MB 级的数据进行实验,结果表明:该技术增加了攻击者逆向获得密钥的难度,能够抵抗单个云客户端和云服务端的合谋攻击,且时间开销成本不高。

关键词 多云存储 秘密共享 Android 数据安全 密钥

中图分类号 TP309 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2020.09.047

ANDROID KEY MANAGEMENT BASED ON MULTI-CLOUD STORAGE

Yu Yujing Ling Jie
(*School of Computer*, *Guangdong University of Technology*, *Guangzhou 510006*, *Guangdong*, *China*)

Abstract Aiming at the key management problem of Android client, this paper proposes an Android key management technology that combines multi-cloud storage and secret sharing. It is different from the traditional Android encryption technology. Instead of storing sensitive information such as keys, user fingerprints, and pattern passwords on mobile devices, it is converted to store secret shares in the cloud. It chose a symmetric encryption algorithm, combined a one-time password to generate a key, encrypted the file, and stored the ciphertext. Through Lagrange interpolation polynomials, the respective secret shares of n cloud servers were generated, and passwords and key information were deleted. The secret shares were uploaded to n cloud servers. When the user decrypts, the client randomly sent a request to $m(m \leqslant n)$ cloud servers for verification, returned the corresponding share, reorganized in the client, and obtained the key for decryption. In the multi-cloud scenario, the data from KB level to MB level were tested, and the comparison was made with the scheme in literature. The technology in this paper increases the difficulty for the attacker to obtain the key in reverse. It can resist the collusion attack of single cloud client and cloud server, and the time cost is not high.

Keywords Multi-cloud storage Secret share Android Data security Secret key

0 引 言

Android 是一款基于 Linux 的,拥有开源性和可移植性的智能移动操作系统。2008 年 9 月,谷歌公司发布了第一版 Android 1.0 系统,至今 Android 系统已经占据了全球智能手机操作系统份额的 85.9%。随着 Android 设备的广泛使用,Android 平台上的恶意软件数量急剧增加,腾讯安全实验室在 2019 年公布的手机安全报告中指出,去年国内 Android 新增病毒包达 800.62 万个,Android 手机病毒感染用户数近 1.13 亿。2018 年手机病毒类型占比中,隐私获取与远程控制的综合比例为 17.57%。Android 设备中隐私数据的存储是一个难题,隐私数据包含用户敏感信息、设备信息、

收稿日期:2019 - 05 - 24。广东省重点领域研发计划项目(2019B010139002);广州市重大攻关计划项目(201902020006,201902020007)。**余宇劲**,硕士生,主研领域:移动与无线网络安全。**凌捷**,教授。

密钥等,其中密钥的存储是关键问题。

Android 系统中的密钥一般可分为两种:一是安卓应用中,用来与服务器通信,或是加密 App 自身数据的密钥;二是安卓应用中因为加密文件产生的密钥。目前针对 Android 设备的安全问题,国内外学者已经做了大量的研究。文献[1-3]指出了近年来 Android 设备安全问题研究优点和不足,详细归纳了 Android 系统的安全缺陷以及所面临的威胁。在企业移动管理系统上,文献[4]提出了密钥容器的概念,保护文件的加密密钥,实现了移动端的安全存储,但该方案需要对已有 App 进行函数重写以及重打包,实现成本高。文献[5]提出了基于 RFID 智能卡的 Android 移动终端数据保护方案,可以增加攻击者破解难度,把密钥信息存放在 RFID 卡上,采用机卡配合的形式,增加安全性,但是需要额外的硬件如智能卡、设备的 NFC 支持。文献[6]提出了基于指纹与移动端协助的口令认证方法,利用指纹和口令结合生成密钥,避免了密钥的存储,但其需要存储指纹信息,存在安全隐患。文献[7-8]提出了一些攻击手段,可以远程使存储在设备上的密钥泄露。文献[9]提出了在 Android 平台上进行多云存储的思想,将密文分片后进行压缩,然后分存到不同云服务商上,可以避免厂商锁定的危害,有一定的可用性,但其密文分片的计算量很大,对于计算力有限的安卓设备会是一个巨大的开销。文献[10]提出了一种基于位置的移动设备加解密模型,使用位置信息结合密钥进行加密,用户只能在有效位置内进行解密。文献[11]提出在安卓应用开发中将密钥分存在 SharedPreferences、文件存储、Setting. System 中,提高了数据的安全性,创新性地将密钥分存的思想引入 Android客户端的开发中。文献[12]在亚马逊 S3(传统云平台)的基础上提出一种密钥管理(Key Manager, KM)的概念,KM 向用户请求密钥并与云服务商交互。但此方案没有考虑用户端是否存在恶意操作,如果用户端受到入侵,那么数据的安全不能得到保障。

为了解决 Android 设备中密钥的安全存储问题,本文提出一种结合密钥分块和多云存储的密钥安全存储技术,实现了密钥的安全存储,无须额外硬件支持,在终端设备上不涉及密钥信息存储,对文件加解密时间的影响较小,实验证明了其安全性、可靠性和实用性。

1　预备知识

1.1　秘密共享

秘密共享方案中,有一个加密者和一组共享者 $\{P_1,P_2,\cdots,P_n\}$,加密者持有一个秘密 $s \in S$,为每个共享者产生一个保密的秘密份额,使得共享者的任何一个授权集合能够通过他们各自掌握的份额恢复出 s。

(1) 秘密分割。秘密分割是由加密者实现的一个映射:

$$F:S \cdot R \rightarrow S_1 \cdot S_2 \cdot \cdots \cdot S_n$$

其中:S 是秘密所在的集合;R 是随机输入集;$S_i(i=1,2,\cdots,n)$ 是 P_i 的秘密份额集合。对 $\forall s \in S, \forall r \in R$,映射 $F(s,r)$ 得到一个 n 元组 $(s_1,s_2,\cdots,s_n)$,使得 $s_i \in S_i(i=1,2,\cdots,n)$。$s_i$ 称为 P_i 的份额,记为 $F_i(s,r)=s_i$,加密者以秘密方式将 s_i 给 P_i。

(2) 重构。s 能被任一授权集合重构,即对 $\forall G \in A$,A 为授权集合,设 $G=\{i_1,i_2,\cdots,i_{|G|}\}$,有一个重构函数:

$$h_G:S_{i_1} \cdot S_{i_2} \cdot \cdots \cdot S_{i_{|G|}} \rightarrow S$$

使得对 $\forall s \in S, \forall r \in R$,如果 $F(s,r)=(s_1,s_2,\cdots,s_n)$,则有:

$$h_G(s_{i_1},s_{i_2},\cdots,s_{i_{|G|}})=s$$

方案的安全性要求:任一非授权集合都不能得到 s 的任何信息[13]。即一个秘密被 n 个人分享,只有 $m(m \leqslant n)$ 个或更多的参与者联合可以重构该秘密,而任意小于 m 个参与者不能得到该秘密的任何信息,以上定义为 (m,n)-门限秘密分割方案,m 称为方案的门限值。

1.2　Android 系统中的存储

Android 系统中存储方式包括文件存储、SharedPreferences 存储、数据库存储等。其中:文件存储是 Android 中最基本的一种数据存储方式,它不对存储的内容进行任何格式处理,所有数据都原封不动地保存在文件中,适合存储一些简单的数据;SharedPreferences 使用键值对的方式存储数据,即保存数据时,给此条数据提供一个键,在读取数据时通过对应的键取数据,如果把密钥直接存储在 SharedPreferences 很容易被窃取;数据库存储,Android 内置了数据库 SQLite,适合存储量大、结构性复杂的数据。

要在设备上安全存储密钥,目前常用的办法是不断加固客户端,增大逆向成本。为了提高安卓设备上密钥的安全性,本文不在本地上做密钥的任何存储,将密钥按照秘密共享的思想,拆分成 n 块秘密份额,上传到 n 个云平台上,即使是逆向成功,或是某云服务器与云客户端以某种方式进行合谋攻击,也不能获得密钥去解密密文。

1.3　多云平台

云存储是在云计算概念的基础上发展起来的一种

新的存储方式,它是指通过网格计算、集群文件系统、分级存储等现有技术,将网络中大量的存储设备通过硬件、软件的方式集合在一起,并对外提供标准的存储接口,以供用户调用并存储数据的存储方式[14]。国内已经兴起了多种云平台如阿里云、腾讯云等。云存储的最大特点在于存储即服务,用户可以通过开放的API将自己的数据上传到云端保存,如百度 Pesonal Cloud Storage(PCS)、阿里开放云平台。但由于用户丧失了数据的绝对控制权,一些数据安全的隐患也由此产生[15]。恶意云服务器与云客户端进行合谋窃取用户隐私,以及目前已知的多种攻击手段如拒绝服务攻击、僵尸网络攻击、音频隐写攻击等都会造成用户数据泄露。单云环境下还可能出现厂商锁定或者服务器损坏的场面,用户的数据安全得不到足够的保障。

单云下存在诸多潜在的安全问题,如服务可用性故障、内部人员的恶意破坏等,为了提高云存储可靠性,充分利用各个云的优势,多云架构应运而生,通过在多个云存储服务中分散存储用户数据的方式来避免和缓解云存储提供商垄断问题。这样既可解决单云故障问题,又能提高安全性、可靠性,降低更换云服务提供商的成本。

各大云平台开放的 API 使开发者能够把自己的 App 和云服务端建立连接,根据 API 类型进行不同的操作,如图 1 所示。

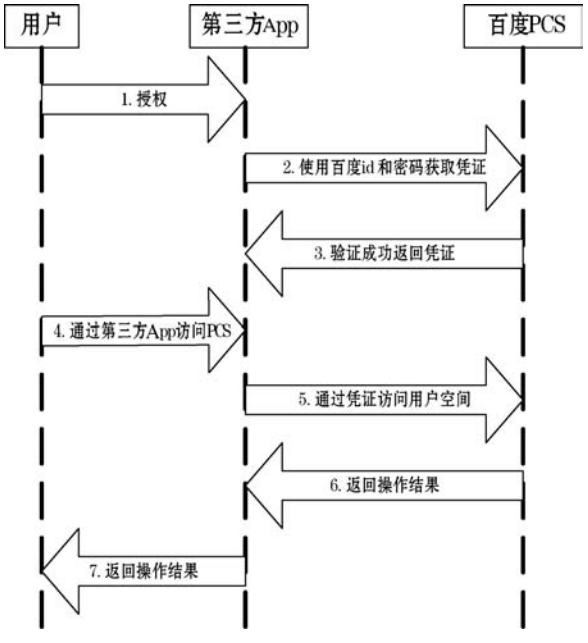

图 1 与云平台建立连接

2 方法描述

本文提出的结合多云存储和秘密共享的密钥管理技术由三个部分组成,分别是加密部分、解密部分、处理密钥部分,其中处理密钥部分是核心。

加密部分:负责结合用户输入的一次性口令产生随机密钥,对文件进行加密。

解密部分:与处理密钥部分进行交互,利用传回的重组的密钥进行解密。

处理密钥部分:进行拉格朗日插值多项式运算,选择服务器,验证服务器,传输完毕后删除密钥。

实现原理如图 2 所示。

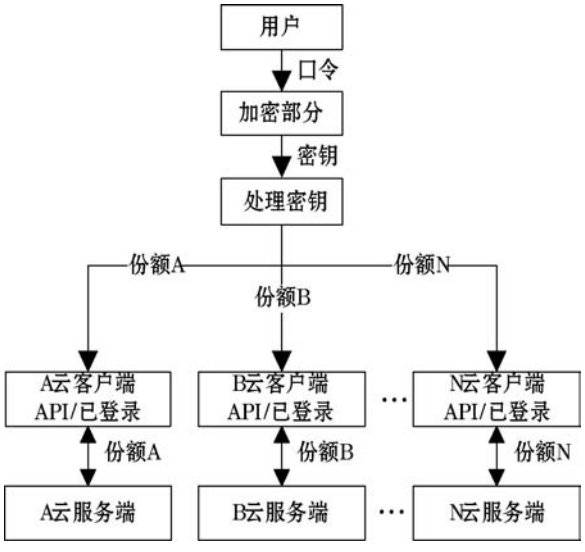

图 2 技术原理图

2.1 加密阶段

安卓端启动加密程序,结合用户输入的一次性口令 p 生成随机密钥 s,对明文 M 进行 AES 加密,得到密文 C,存储密文完毕,删除 p,初步加密完成。

2.2 密钥保护阶段

设门限法方案选择 $(m,n)=(3,5)$,将密钥通过拉格朗日插值多项式进行切片,即生成 5 个秘密共享项,任意大于等于 3 个共享项即可解密。具体步骤如下:

(1) 生成随机数 R_1、R_2、x_1、x_2、x_3、x_4、x_5。

(2) 计算对应的 $F(x_i)$:

$$F(x_i) = R_1 \times x_i^2 + R_2 \times x_i + s$$

(3) 给不同的 $(x_i, F(x_i))$ 分配唯一的 ID,发送给 5 个云服务端,删除 R 和 s。ID 的作用是在多密钥共享的情况下,区分不同的密钥对应的 $(x_i, F(x_i))$。

在密钥保护阶段完毕后,无论是本地还是云盘中,都不涉及到口令和密钥的存储,而是转换成 $(x_i, F(x_i))$ 的存储,无须上传密文,实现了用户数据的私有化。

2.3 解密阶段

当用户需要解密时,安卓客户端向随机的三个服务端发出请求,进行各自的验证,验证通过后,根据 ID

依次返回(x_i, $F(x_i)$)给安卓客户端,客户端进行方程组求解,得到 s,进行解密。本文解密阶段涉及到与不同的云进行交互,随机选取 m 个云服务商进行验证,查找 ID,返回(x_i, $F(x_i)$),因为传输体积小,所以网络消耗和时间成本较低。

2.4　安全性分析

文献[11]将秘密份额存储在 SharedPreference、文件存储和系统数据库 Setting. Systeam 中,提高了攻击者的破解难度,但是因为在 Java 层实现,攻击者可以从 apk 文件包中解压提取出 classes. dex 文件,利用反编译工具(如 apktool、Jadx 等)获得 Java 源码,进而分析出秘密份额存储位置进行重组密钥。本文方法需要获得凭证与多个云服务端建立连接,才能够进行密钥重组操作,假设攻击者逆向成功,只能得到密钥的拆分方式,用户的一次性口令和秘密份额都已被删除。用户与 n 个云客户端之间的通信,通过 Android 沙箱机制相互隔离,所以攻击者与 m 个云服务端建立连接的可能性相当低,极大地降低了攻击者获得密钥的可能性。

即使某个云服务商想窥探存储在该云的数据,或者是黑客攻击云存储平台,也只能得到部分的秘密份额(x_i, $F(x_i)$),这对整个解密过程是无意义的。假设运行在客户端上的云客户端得到了 root 权限,在未经用户许可的情况下,与对应的云服务商进行合谋,窃取用户密文,因为小于 m 个云服务端不能恢复密钥,所以不能进行解密,攻击者没有办法得到密钥和明文信息。

3　实验与结果分析

本实验开发环境使用了 Android Studio 2.2 进行界面和逻辑编写,Eclipse 4.10 用于 Java 代码调试;测试手机型号为 Nexus 6,手机参数为 Android 7.0 系统,3 GB 内存,CPU 类型为骁龙 805。利用本文方法对 KB 级到 MB 级的 14 个文件进行加解密测试,结果如表 1 所示。

表 1　文件加解密时间

大小/KB	加密/ms	处理密钥/ms	解密/ms	总时间/ms
4	298	179	84	561
6	298	175	179	652
73	313	174	83	570
75	283	178	91	552
142	308	179	134	621
162	327	179	84	590

续表 1

大小/KB	加密/ms	处理密钥/ms	解密/ms	总时间/ms
395	352	184	98	634
557	335	171	104	610
1 579	349	180	117	646
1 667	354	175	125	654
8 320	515	179	198	892
8 462	507	176	491	1 174
10 991	536	173	533	1 242
16 785	761	175	537	1 472

处理密钥时间为加密之后的分片时间加上解密之前的组装时间,和密钥长度有关。基于设备和实验环境的原因,加解密时间不是线性增长的,但总体是随着文件大小而上升的。

本文中,处理密钥时间与密钥长度有关,当文件大小增大时,处理密钥时间占总时间的比重会缩小,对加解密时间的影响越小,如图 3 所示。

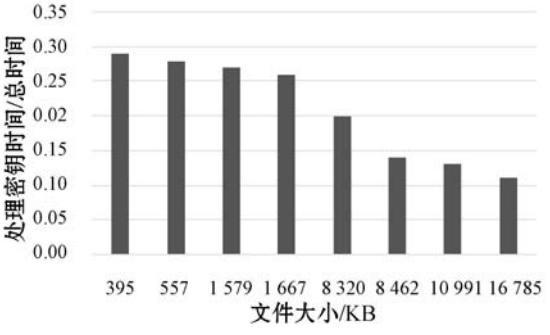

图 3　对加解密时间影响

文献[8]提出在将密钥分存在 Android 设备中的不同位置,如存储在 SharedPreferences、文件存储、Setting. System 等,选取该方案作为对比,处理相同位数的密钥,所用时间对比如图 4 所示,纵坐标为本文方案所用时间减去文献[8]方案所用时间。

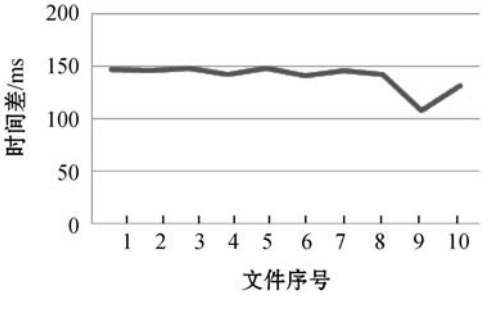

图 4　与文献[8]方案的密钥处理时间对比

文献[8]方案在 Java 层实施,分存在同一设备的不同地方,相对于密钥直接存储和密钥硬编码,提高了安全性,与本文方案对比,不需要与云服务商交互,节省了传输时间,但是将秘密份额分存在设备上,只要逆

向了源文件,得到 Java 代码,便可以分析秘密份额存储位置,从而获取密钥。本文方案则是在密钥保护阶段,通过与多云平台交互,将秘密份额存在不同的云服务端上,逆向不能得到密钥存储的相关信息,具有更高的安全性。

由于实验环境、设备参数、代码复杂度等因素存在差异,所以实验结果会有一定的误差,但是总体来看时间差稳定在 100 ~ 150 ms 内,不会影响用户体验。

4 结 语

本文提出了一种新的 Android 密钥管理技术,通过结合多云存储和秘密共享的思想,在密钥处理阶段,将秘密份额存放在 n 个云服务端上,与现有方案相比,减少了逆向得到密钥信息的可能性,能够避免云客户端和云服务端的合谋攻击,实现了 Android 客户端密钥的安全存放。在大文件加解密时,对加解密总时间的影响会显著减少,具有实用性。

参 考 文 献

[1] 张玉清,王凯,杨欢,等. Android 安全综述[J]. 计算机研究与发展,2014,51(7):1385 - 1396.

[2] 谢佳筠,伏晓,骆斌. Android 防护技术研究进展[J]. 计算机工程,2018,44(2):163 - 170,176.

[3] Sufatrio,Tan D J J,Chua T W,et al. Securing android:a survey,taxonomy,and challenges[J]. ACM Computing Surveys,2015,47(4):1 - 45.

[4] 李成吉,雷灵光,林璟锵,等. 安全的 Android 移动终端内容保护方案[J]. 计算机工程与设计,2016,37(3):591 - 596.

[5] 秦文仙,王琼霄,高能,等. 基于 RFID 智能卡的 Android 移动终端数据保护方案[J]. 计算机工程与应用,2016,52(2):112 - 116,126.

[6] 安迪,杨超,姜奇,等. 一种新的基于指纹与移动端协助的口令认证方法[J]. 计算机研究与发展,2016,53(10):2399 - 2410.

[7] Wang D,Cheng H B,He D B,et al. On the challenges in designing identity-based privacy-preserving authentication schemes for mobile devices[J]. IEEE Systems Journal,2018,12(1):916 - 925.

[8] He D B,Zeadally S,Wu L B,et al. Certificateless public auditing scheme for cloud-assisted wireless body area networks[J]. IEEE Systems Journal,2018,12(1):64 - 73.

[9] 孔琰,赵帅兵,刘若琳,等. 基于安卓平台的多云存储系统[J]. 计算机应用,2017,37(S1):39 - 44,48.

[10] You L,Chen Y L,Yan B,et al. A novel location-based encryption model using fuzzy vault scheme[J]. Soft Computing,2018,22:3383 - 3393.

[11] 刘培鹤,闫翔宇,何文才,等. 基于 Android 的密钥分存方案[J]. 计算机应用与软件,2018,35(2):320 - 324,333.

[12] 王志中,周城,牟宇飞. 基于分离密钥的云存储加密解决方案[J]. 电信科学,2013,29(1):51 - 56.

[13] 杨波. 密码学中的可证明安全性[M]. 清华大学出版社,2017.

[14] 张玉清,王晓菲,刘雪峰,等. 云计算环境安全综述[J]. 软件学报,2016,27(6):1328 - 1348.

[15] 傅颖勋,罗圣美,舒继武. 安全云存储系统与关键技术综述[J]. 计算机研究与发展,2013,50(1):136 - 145.

(上接第 285 页)

激活图像的时间阈值也需要进一步研究,以改善交互体验。并且用户测试也存在一定的局限性,无法对该认证系统进行全面评估。未来将进一步探索基于注视的 PII 输入策略,同时在系统设计时考虑用户本身的固有习惯,以提高认证系统的效率和可用性。

参 考 文 献

[1] Valles P A S, Villalobos-Serrano J G, Martinez-Pelaez R, et al. My personal images as my graphical password [J]. IEEE Latin America Transactions, 2018, 16(5): 1516 - 1523.

[2] Mihajlov M, Jerman-Blazic B. Eye tracking graphical passwords [C]//International Conference on Applied Human Factors and Ergonomics, 2017.

[3] Maeder A, Fookes C, Sridharan S. Gaze based user authentication for personal computer applications [C]//Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004.

[4] Hoanca B, Mock K. Secure graphical password system for high traffic public areas [C]//Proceedings of the Eye Tracking Research and Application Symposium, 2006.

[5] Dunphy P, Fitch A, Olivier P. Gaze-contingent passwords at the ATM [C]//4th Conference on Communication by Gaze Interaction (COGAIN), 2008.

[6] Dan W H, Ji Q. In the eye of the beholder: a survey of models for eyes and gaze [J]. IEEE Trans Pattern Anal Mach Intell, 2010, 32(3): 478 - 500.

[7] Zhang Y, Mou X. Survey on eye movement based authentication systems[C]//CCF Chinese Conference on Computer Vision, 2015.

[8] Hansen J P, Rajanna V, MacKenzie I S, et al. A fitts' law study of click and dwell interaction by gaze, head and mouse with a head-mounted display [C]//Workshop on Communication by Gaze Interaction, 2018: 1 - 5.