新型智能终端取证技术研究

金 波, 吴松洋, 熊 雄, 张 勇

信息网络安全公安部重点实验室(公安部第三研究所) 上海 中国 201024

摘要 随着移动互联网的广泛应用, 智能手机、平板等新型智能终端设备在各种各样的违法犯罪活动中开始扮演越来越重要的角色, 从涉案手机中提取的数据常常包含与违法犯罪行为相关的重要线索和证据。然而, 移动智能终端设备不断提升的安全设计可能使得取证人员无法从设备中提取数据, 给电子数据取证鉴定工作提出了新的挑战。本文详细分析当前主流的 iOS、Android 和 Windows Phone 等平台下的移动设备的安全机制, 研究了主要的安全机制破解和取证技术及其在目前电子数据取证工作中的应用。最后, 对未来面向新型移动智能终端电子数据取证技术研究发展方向进行了探讨。

关键词 智能终端取证; 电子数据取证鉴定; 数据恢复

中图法分类号 TP309 DOI 号 10.19363/j.cnki.cn10-1380/tn.2016.03.004

Research on Digital Forensics of Smart Devices

JIN Bo, WU Songyang, XIONG Xiong, ZHANG Yong

Key Lab of Information Network Security of Ministry of Public Security (The Third Research Institute of Ministry of Public Security), Shanghai 201024, China

Abstract As the widespread application of mobile Internet, smart devices such as smartphones and pads also increasingly play an important roles in a wide variety of criminal activities. It has been proven that the extracted data from smartphone of suspects often critical clues and evidence of digital investigation. However, the improved security design of smartphones in recent years seriously hinders the evidence acquisition for mobile phone forensics and brings us new challenges. This paper analyzed current security mechanisms of popular mobile devices of iOS, Android and Windows Phone, studied the major crack technologies of smartphones and their application in digital investigation. This paper also discussed research directions of smartphone forensics technologies in the future.

Key words Smartphone forensics; digital investigation; data recovery

1 引言

随着移动网络、智能终端软硬件设计制造的不断创新和进步, 在过去的几年中, 全球移动智能终端产业、市场规模和用户数量都获得了持续强劲增长。根据《中国移动互联网发展状况及其安全报告(2016)》[1], 截至 2015 年末, 在我国境内活跃的智能手机联网终端达 11.3 亿部, 活跃的手机网民数量达 7.8 亿, 超过了世界上大多数国家人口数。九成以上市场份额被 Android 和 iOS 系统占据, 而这两者占比分别是 78.9%和 13.08%。移动互联网、智能手机等智能设备已完全融入到了人们的日常生活中。流行的社交、即时通讯、搜索和电子商务等领域的移动应用甚至开始塑造出新的社会生活形态并影响全国大部分民众的生活。

然而, 移动智能终端与人们工作生活日益密切的关系, 也使得它们在各种各样的违法犯罪活动中开始扮演越来越重要的角色[2]。一方面, 移动智能终端的通信工具属性使其必然成为犯罪活动中的联络工具; 另一方面, 智能终端丰富的功能也会被利用, 从而成为实施违法犯罪行为的工具, 例如用于电信诈骗、木马病毒等恶意程序的传播等。2016 年 5 月, 360 公司就发现全球首款专用于网络电信诈骗的 Android 木马。该木马伪装成“公安部案件查询系统”, 具备隐私数据窃取、钓鱼欺骗和远程控制等多种恶意功能, 能够窃取银行账户中的资金[3]。在大量案件处理过程中, 从涉案智能终端中提取的数据等常包含与违法犯罪行为相关的重要线索和证据。例如可以通过短信、通话记录、电子邮件了解嫌疑人之间的关系,可以通过 GPS 定位功能确定用户的活

通讯作者: 吴松洋, 博士, 副研究员, Email: wusongyang@stars.org.cn。

本课题得到国家发改委 2013 年信息安全专项(发改办高技[2015]298 号)项目资助。

收稿日期: 2016-06-05; 修改日期: 2016-06-29; 定稿日期: 2016-07-08

动信息。

随着用户对智能终端安全性要求也不断上升, 目前移动智能终端厂商普遍加强了设备系统安全的设计, 以保护用户的重要数据。设备安全性不断提升的同时, 也对涉及智能终端的电子数据取证鉴定工作提出了新的要求。例如自 Apple 采用了全盘加密和独立硬件级加密以来, 从被锁定的 iOS 智能终端上获取数据就成为取证工作面临的一大难题。在 2016 年初, 美国联邦调查局(FBI)为了破解圣贝纳迪诺枪击案凶手使用的 iPhone 5c, 甚至不惜寻求通过法律手段要求苹果去解锁这部 iPhone[4]。后来虽然在第三方取证公司的帮助下该手机成功解锁。但此案也预示着在不久的将来, 不断增强的智能终端安全保护措施将是电子数据取证技术面临的最大挑战之一, 从设备上获取关键数据终究是取证工作中最重要的工作。

本文对面向 iOS、Android、BlackBerry 等平台智能终端的主要取证技术难点进行了梳理, 详细分析了各类型智能终端设备的数据保护安全机制, 结合目前针对安全机制的破解技术提出了面向新型智能终端的取证技术和方法, 并给出了实际取证案例研究。最后, 本文也对未来面向移动智能终端的电子取证技术研究发展方向进行了探讨。

2 国内外相关研究进展

随着智能手机、平板电脑等终端设备的普及, 自 2007 年起, 面向智能终端的电子数据取证技术引起了学术界和产业界的广泛重视[5]。2007 年, NIST 就发布了指南《Guidelines on cell phone forensics (800-101)》, 用以指导对手机等设备的取证操作及技术研究方向。Morrissey 等[6]和 Hoog 等[7]编著了指南性技术书籍, 分别介绍面向 iOS 设备(包括 iPhone、iPad 和 iPod touch)和 Android 设备(包括智能手机和平板设备)的数据提取和分析技术。在电子证据提取方面, 书中[6, 7]整理汇总了当时较新的技术方法, 着重讨论了“人工提取”、“逻辑提取”和“物理提取”三个方向。人工提取包括通过拍照、直接连接设备或安装相应应用程序等手段提取设备内容, 人工提取出错概率大, 可能会遗漏重要证据, 已很少被采用; 逻辑提取指提取拷贝逻辑存储区域的文件或整个分区镜像, 以及通过其他手段获取短信、通信录等记录信息。涵盖了通过备份、提权后镜像整个分区以及 Android 设备的 Recovery 模式来获取数据的技术手段。物理提取指从物理存储介质上获取存储镜像。可用的技术包括摘取存储芯片通过专业设备提取存储镜像, 或通过设备的 JTAG(Joint Test Action Group)接口驱动设备 CPU 直接读取存储芯片中的数据。上述文献为面向智能终端的电子数据取证实践提供了重要的操作指南, 其中所论述的技术路线也为后续研究提供了重要参考。但随着智能终端设备的升级换代, 其中具体的技术方案也不再应用目前的取证工作需求。

对智能终端设备数据保护机制的突破一直是近年来取证技术的关注重点。Abalenkovs 等 2012 年的论文[8]面向 iOS 和 Android 的数据提取和分析技术进行了综述。文献首先分析了当时 Android 和 iOS 设备的数据保护机制原理。Android 方面, 介绍了通过屏幕划动痕迹破解手势密码、通过 Recovery 模式获取数据、通过 JTAG 和 chip-off 技术获取闪存物理数据等技术方案。iOS 方面, 重点是 iPhone4 及以下版本的锁屏密码破解、通过 iTunes 逻辑备份提取数据等方案。利用针对 iPhone4 及以下版本的锁屏密码破解技术, 邱等学者[9]提出了针对 iPhone4 设备 NAND flash 的镜像提取和解密方案, 并基于 FTL(file translation layer)原理的分析给出了逻辑数据恢复方案。文献[10]提出了基于 iPad 一些未公开的“特性”, 对 iPad 进行越狱、安装 SSH 服务并通过苹果 iPad 相机连接套件提取存储镜像的方法, 但文献测试的 iPad 和 iOS 版本较早, 已不支持目前主流 iOS 设备。针对 iOS 设备备份解密问题, 文献[11]提出了基于 CPU+GPU 平台和并行随机搜索策略的解密算法, 有效提高了破解加密 iOS 备份文件的效率。Oestreicher[12]提出了 iCloud 数据取证方案, 这包括了确定同步文件, 提取、验证同步文件完整性等过程。在无法访问 iOS 智能终端设备的情况下, 通过 iCloud 取证也是一个可行的方案。Vidas 等人[13]讨论了通过创建定制的引导镜像来进行电子证据提取的技术, 并详细阐述了数据获取过程。文献[14]评估了 Android Recovery 模式下提取操作对数据完整性的影响, 并设计了支持数据完整性保护的取证工具。Yang 等人[15]基于 Android 设备的固件更新协议实现了提取 Android 存储数据的技术方案。每个厂商均提供了固件更新程序, 但没有公布更新协议。经过在数个机型上对更新协议进行逆向分析, 解析出了闪存读取命令, 同通过该命令获取 Android 设备的存储镜像。但不同的厂商固件更新程序会存在较大差异, 该方向的研究还需要一直跟进研究各厂商不同版本的固件更新程序。

根据电子数据取证司法实践需要以及取证技术研究的进展, 国内外相关组织和机构均发布了面向

智能终端电子数据取证的行业标准和规范。NIST 于 2007 年发布了指南《Guidelines on cell phone forensics(800-101)》。根据相关技术的发展演进, NIST 于 2013 年发布了《Guidelines on Mobile Device Forensics (800-101 Revision 1)》, 代替了“800-101”。SWGDE(Scientific Working Group on Digital Evidence)组织也发布了系列最佳实践, 包括《SWGDE Best Practices for Chip-Off》、《SWGDE Best Practices for Examining Mobile Phones Using JTAG》、《SWGDE Best Practices for Mobile Phone Forensics》等。我国, 《法庭科学电子物证手机检验技术规范(GA/T 1069-2013)》、《移动终端取证检验方法(GA/T 1170-2014)》和《手机电子数据提取操作规范(SF/Z JD0401002-2015)》等标准规范相继发布。标准规范的发布对智能终端取证过程进行了详细解释指导, 从而确保获取电子物证的有效性。规范性文件中的技术元素较少, 但对于技术研究和工具开发而言, 规范性文件能够提供重要方向性指引, 确保研发成果更好地满足司法实践需求。

通过回顾和分析上述研究文献, 可以发现具体的取证技术方案主要还是依赖于当时设备、环境的特性, 以及安全研究社区的成果(如越狱、ROOT、锁屏密码破解等技术和工具)。而智能终端软硬件技术的不断创新和频繁更新换代缩短了相关电子数据取证技术研究文献的时效性。例如文献[6, 7, 8, 9, 10]等所讨论的具体技术方案已明显不适用于目前较主流的智能终端设备。这也正是本文撰写的主要动机。即本文对目前 iOS、Android、Windows Phone、BlackBerry 等平台智能终端仍存在的取证技术难点进行了梳理, 详细分析了各类型智能终端设备的数据保护安全机制, 最后结合目前安全机制的破解技术提出了面向新型智能终端的取证技术和方法, 并给出实例研究。

3 新型智能终端取证技术

3.1 iOS 智能终端的取证技术

3.1.1 取证中的难点问题

iOS 智能终端是目前最为流行的智能终端产品之一。iOS 设备具备较高的数据安全性, 从简单锁屏密码、复杂锁屏密码到全盘数据加密, 从多个层次保护用户的隐私数据。

iOS 操作系统采用了严格的权限机制保护数据安全性。智能终端的数据存储主要分为三个类别: 照片、视频等可共享的数据、已安装 App 的应用数据以及系统数据。不同类别的数据需要不同的数据访问权限。照片、视频等可共享的文件能通过 WPD 模式直接读取; 已安装 App 应用的数据存储在 iOS 系统的 AppDomain 区域, 需要通过 iOS 的数据传输服务访问数据; 系统分区则存储了重要的各类系统数据及服务应用, 对于未获取到 Root 权限的 iOS 智能终端而言, 系统分区的数据无法读取。

AppDomain 区域存储了第三方应用的可共享数据。iOS 8.2 版本之前, 所有第三方应用运行过程中产生的用户数据, 包括信息记录(存于 SQLite 数据库)和音视频缓存文件等, 均可以从 AppDomain 区域中第三方应用的 Documents、Library 等目录下提取。iOS8.2 及之后版本, 仅有少量第三方应用的数据仍可以通过 AppDomain 区域提取数据。iOS 系统的系统分区存储了内置应用的使用痕迹数据, 如短信、通信录、通话记录、电子邮件、键盘使用记录、账号信息等数据。

当前针对 iOS 智能终端的取证分析技术主要从直接访问设备的“在线取证”、通过 iOS 备份的“离线取证”和通过 iOS 智能终端存储镜像的“离线取证”等方面展开研究。近年来 iOS 智能终端的取证技术取得了较大的提高, 但是在数据取证、锁屏破解等安全防护机制的突破方面仍然存在较多困难。

3.1.2 iOS 智能终端的取证技术方案

(1) 针对 iOS 智能终端的“在线取证”

“在线获取”是 iOS 智能终端的主要取证技术手段。通过 USB 连接 iOS 智能终端, 解除锁屏并信任取证设备与 iOS 设备建立连接。取证设备直接从 iOS 智能终端中提取各类数据及 App 的使用数据进行分析, 将取证分析结果进行呈现。过程如图 1 所示。

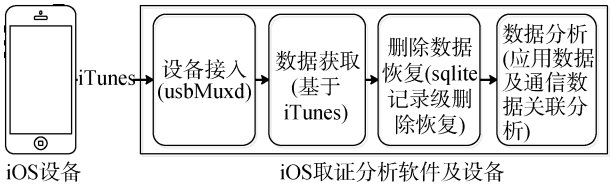

图 1 在线取证操作

iOS 智能终端中的 AppDomain 区域存储了 iOS 操作系统已安装 App 的应用数据, 是重要的电子证据数据来源。iOS 智能终端的在线取证, 主要是获取 AppDomain 区域的数据并进行检验分析。

iOS 操作系统各版本均提供了 com.apple.afc 数据传输服务[16], 是获取 AppDomain 区域存储数据的重要途径。该服务基于 iOS 系统的 USBMuxd 协议进行数据传输, 使用服务的时候必须通过 USB 数据线连接 iOS 设备与取证设备。com.apple.afc 服务由 iOS

系统的 lockdown 守护进程负责创建, 监听端口通常为 62078, com.apple.afc 服务的通信过程[17]如图 2 所示:

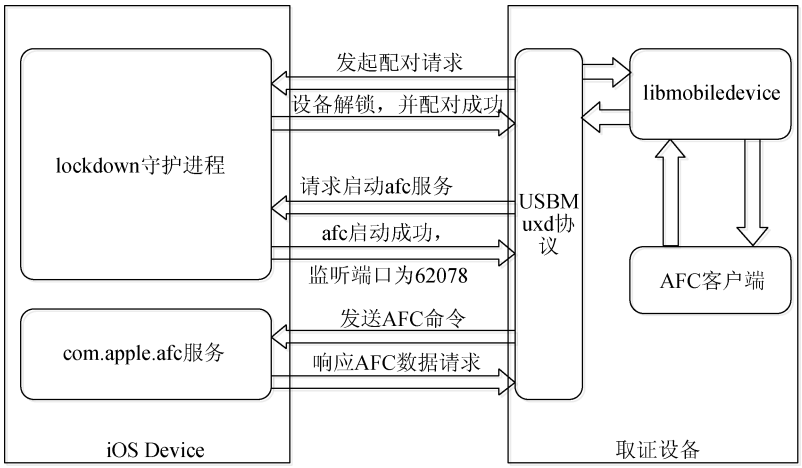

图 2 com.apple.afc 服务的通信过程

afc 服务通信主要分为三个步骤:

a) 基于 USBMuxd 协议, 取证设备发起设备配对请求, 将 iOS 设备解锁并响应配对请求, 成功建立信任关系;

b) 取证分析设备请求启动 com.apple.afc 数据传输服务, iOS 系统通过 lockdown 守护进程启动 afc 服务, 启动后的 afc 服务开始监听网络端口 62078;

c) 取证设备向端口 62078 发送数据请求命令, 并获得 iOS 设备返回的数据。

iTunes 软件的 iTunesMobileDevice.dll 中完整封装了 afc 服务的通信调用接口, 取证分析设备可通过这些接口完成与 iOS 终端设备的数据通信, 之后调用封装的 AMDeviceLookupApplications 接口即可取证 AppDomain 目录的数据。该接口如下所述。

```
public unsafe static extern int
    AMDeviceLookupApplications(void* device,
      void* options, ref MobileDevice.
      __CFDictionary* appBundles);
```

AppDomain 区域目录结构如图 3 所示, 以 App 的包名为目录分别存储不同 App 的数据, 用户应用数据通常存储在不同对应 App 目录中的 Documents 目录下。如微信应用数据就存储在“/AppDomain/com.tencent.xin/Documemnts”目录中, 微信的数据库文件路径为“/AppDomain/com.tencent.xin/Document/***/DB/MM.sqlite (其中***为不同微信账户的 MD5 散列值) ”。

对于未越狱的 iOS 智能终端, iOS8.2 版本系统之前, 可通过 com.apple.afc 服务获取 AppDomain 区域中存储的第三方应用数据。iOS8.2 版本之后, AppDomain 区域存储的数据大大减少了, 通过 com.apple.afc 服务仅能获取 Coco 等少数应用的数据。

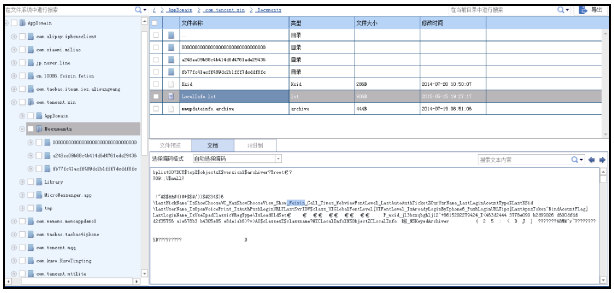

图 3 AppDomain 目录数据结构

“越狱”是突破 iOS 设备权限访问控制的唯一手段[18], 越狱后能够全面提取 iOS 智能终端的数据, 对取证工作极为重要。具体方法是利用越狱软件解除原有固件对手机系统的限制束缚, 使用户获取最高操作权限, 从而读写 iOS 系统中任何区域的数据。通常“越狱”行为会少量更改 iOS 设备中的系统数据, 某些极端情况下会造成 iOS 系统崩溃并导致证据信息丢失, 因此“越狱”操作必须谨慎。“越狱”过程是利用 iOS 操作系统的漏洞, 绕过 iOS 系统的安全机制, 并向 iOS 操作系统中“植入”了重要的 com.apple.afc2 服务。相比于前面介绍的 com.apple.afc 服务, com.apple.afc2 服务提供了 Root 权限, 可以访问到 iOS 设备的所有区域的数据文件, 是 iOS 设备取证的强有力手段[19]。

对于已越狱的 iOS 智能终端, 可通过 com.apple.afc2 服务访问包括短信、通信录、通话记录数据的系统分区, 第三方应用的数据则从目录“/User/ Containers/Data/Application/”下获取, 该目录下的各个第三方应用的文件夹均以该应用的 GUID 命名, 通过 Application 目录下的 *com.apple.mobile_container_manager.metadata.plist* 文件可分析出对应的 APP 名

称[20]。

此外, iOS 操作系统自身还存在一些“安全漏洞”, 也可以被用于在取证分析工作中获取电子证据。经媒体披露[21]美国国家安全局在 iOS 智能终端中强制安装了“DROPOUTJEEP”服务用以收集用户信息, 且此服务无需 iOS 智能终端处于“越狱”状态即可运行调用。“DROPOUTJEEP”服务包含了三个重要的后台服务: com.apple.mobile.file_relay、com.apple.mobile.house_arrest 和 com.apple.pcapd。

com.apple.mobile.file_relay 服务可以在非越狱状态下绕过 iOS 智能终端的安全机制, 向取证分析人员提供一个功能强大的数据读取接口, 直接提取用户的应用数据。由于此后台服务能够获取的数据过于广泛。 Apple 公司自 iOS8 就关闭了该服务。图 4 是 com.apple.mobile.file_relay 服务的应用示例。

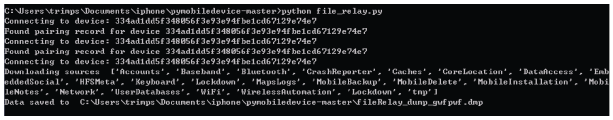

图 4　通过 com.apple.mobile.file_relay 获取数据

com.apple.mobile.house_arrest 服务功能相对简单, 主要用于获取应用程序的 Library、Caches、Cookies 以及 Preferences 目录下的数据。

com.apple.pcapd 服务用于强制开启 iOS 智能终端上内置的 libpcap 服务, libpcap 服务可用于监听 iOS 智能终端的网络通信, 并能够 dump 所有的 HTTP 请求和响应数据。图 5 为 libpcap 服务 dump 文件数据结构。Dump 数据主要包括两部分内容, 文件头和 IP 数据报文。文件头主要包括格式标记、版本信息、时间信息以及数据包大小和数据包类型。紧跟着文件头的数据则是按顺序依次记录的数据报文。数据报文包括 IP 数据报长度、原始 IP 数据报长度、时间信息以及数据报正文内容。

(2) 针对 iTunes 备份进行“离线取证”

iTunes 备份的“离线取证”即为通过 iOS 智能终端的 iTunes 备份数据进行取证分析。iTunes 可对用户数据备份保全, 备份数据中包含了大量用户通话记录、短信、App 使用数据等信息。在无法直接对 iOS 智能终端“在线取证”的情况下, iOS 设备的备份是极为重要的检材。

可通过 iTunes 软件附带的 AppleMobileBackup.exe 程序备份设备。调用命令为: *AppleMobileBackup.exe –backup –target [设备 ID]*。获得的备份数据存储在以设备 ID 命名的文件夹中, 主要包括 Manifest.mbdb、Status.plist、Manifest.plist、Info.plist 以及数据备份文件等。

dump数据格式标记 (4Byte)	版本信息(4Byte)	时间信息(8Byte)	数据包大小(4Byte)	数据包类型(4Byte)
保存的IP数据报长度 (4Byte)	原始IP数据报长度 (4Byte)	时间信息(8Byte)	IP数据报内容(n Byte)	
.....				
保存的IP数据报长度 (4Byte)	原始IP数据报长度 (4Byte)	时间信息(8Byte)	IP数据报内容(n Byte)	

图 5　libpcap 服务 dump 文件的数据结构

Manifest.mbdb 文件是 iTunes 备份文件的核心数据库, 存储了备份文件的详细存储路径信息, 解析 Manifest.mbdb 文件即可得到名称与存储位置的映射关系, 进而可以分析还原出整个文件系统。

Manifest.mbdb 文件以固定的 6 字节格式标记“*mbdb\5\0*”开头。紧跟着是每个备份文件的详细信息。备份文件详细数据结构如图 6 所示。

Mbdb 文件的数据结构包括三个部分的内容:

域长度 (2Byte)	域数据(n Byte)	路径长度 (2Byte)	路径(n Byte)	路径长度 (2Byte)	绝对路径(n Byte)
摘要长度 (2Byte)	文件内容摘要(n Byte)		摘要长度 (2Byte)	解密密钥(n Byte)	
文件权限信息 (2 Byte)	文件iNode信息 (4 Byte)		uid信息 (4 Byte)	gid信息 (4 Byte)	
修改时间 (4 Byte)	最后访问时间 (4 Byte)		创建时间 (4 Byte)	文件长度 (8 Byte)	
保护级别 (1 Byte)	属性数量 (1 Byte)	属性内容(n Byte)			

图 6　Manifest.mbdb 的数据结构

a) 文件路径信息:包括文件所属的域, 文件路径及绝对路径;

b) 文件内容及权限信息:主要包括文件内容的长度、文件内容的 SHA-1 摘要值, 对于加密的 iTunes 备份还包含了解密所需的密钥信息, 此外还记录了文件的权限信息、文件拥有者 id、用户组 id 和文件的修改、访问和创建时间;

c) 文件的额外属性信息: 主要包括属性数量及属性内容。

iTunes 的备份文件均以编码的名称来命名。编码方法如下:

加密文件名 = SHA-1(<域>-<文件路径>)

其中, 域为 iOS 系统中的存储空间概念, 主要包括“AppDomain”、“HomeDomain”、“MediaDomain”等十余种固定字符串。因此, 依据 Manifest.mbdb 文件的解析结果, 可以使用上述算法得到编码文件名称与原始路径的映射关系, 如下例所示。

003f52579f0ae40aaee7f066aae75505a9d7e002
→ Library/SMS
3bb707eaa713ae7fbe515fda15ac6a084936a2f7
→ Media/DCIM/100APPLE/IMG_0021.JPG
75bff5e106fd1d632d4f9d2257dc0503bb45c62e
→ Documents/iLyric
2857a36ddc4ae8e7e3cf54f84f2d63d232227d4f
→ Media/PhotoData
33e6cc0ba0441a0c73b75e5916fa8c6de17dcd4a
→ Documents

Info.plist 存储了 iTunes 备份的 iOS 智能终端基本信息, 包括设备名称、IMEI、GUID 等信息。plist 文件是 iOS 操作系统普遍采用的一种数据存储格式, 可以直接转换为 XML 文件格式。

Info.plist 文件转换为 XML 格式, 内容摘要如下:

```
<Dict>
  <key>Build Version</key>
  <string>12A405</string>
  <key>Device Name</key>
  <string>“Administrator”的 iPhone</string>
  <key>Display Name</key>
  <string>“Administrator”的 iPhone</string>
  <key>GUID</key>
  <string>36A492******</string>
  <key>IMEI</key>
  <string>35875******</string>
</Dict>
```

Manifest.plist 存储了加密的 iTunes 备份中极为重要的解密密钥数据 BackupKeyBag 以及是否加密的标识 IsEncrypted。IsEncrypted 设置为 True 的备份文件是经过加密的, 无法直接进行文件内容读取和分析, 必须先进行解密操作。

iTunes 备份的“离线取证”因受到第三方应用对于 iTunes 备份协议的支持度影响, 相较于“在线取证”, 获取的数据存在局限性[20]。

(3) 针对 iOS 智能终端的镜像进行“离线取证”

iOS 智能终端采用了高强度数据全盘加密技术, 对于 iPhone4s 及之后版本的 iOS 智能终端, 目前业界还没有有效的数据解密方案, 因此针对 iOS 智能终端的镜像的“离线取证”技术有着较为苛刻的条件。目前, 仅在已越狱的 iOS 智能终端上能获取设备存储镜像。利用 Cydia 在已越狱的 iOS 智能终端中安装 dd、nc、ssh 等工具, 在取证设备中通过 ssh 远程执行 dd 与 nc 命令, 对 iOS 智能终端进行镜像, 并通过 nc 将镜像数据传送到取证设备中。镜像命令为:

dd if=/dev/disk0s1s2 | nc [设备 ip] [监听端口]

针对 iOS 智能终端镜像的“离线取证”, 目前仅能够对 iPhone4 及之前的 iOS 智能终端镜像进行解密分析, iPhone4s 及之后的 iOS 智能终端因修复了 bootloader 中已知的漏洞并更换了加密芯片, 目前尚无任何方法对 iOS 智能终端镜像进行解密分析。相比较前述的取证方法, iOS 设备存储镜像是逐 bit 获取存储数据, 所以可以基于“文件特征签名”来恢复已删除数据文件, 获得更多的电子证据[22,5]。

(4) 锁屏密码和 iTunes 加密备份的破解

iOS 智能终端自 iOS7 版本开始, 必须首先建立信任关系才可以进行数据通信。而信任关系建立的前提是设备必须处于解锁状态。锁定的 iOS 设备给取证分析工作带来了极大的困扰。iOS 智能终端自 iPhone4s 开始使用 AES-GCM 加密算法将用户的锁屏密码加密并存储在 KeyChain 文件中, 解密及解锁流程如图 7 所示:

iOS 智能终端中存在两个关键的 AES 密钥:一个是 GID, 一个系列的所有 iOS 智能终端共享这个密钥; 还有一个是 UID, 是每部 iOS 智能终端独有的密钥, 并且 UID 是嵌入在 iOS 智能终端硬件中无法被直接获取。因此, 对 iOS 智能终端的锁屏密码进行破解从解密技术上十分困难。目前针对 iOS 智能终端的锁屏密码破解主要有三种思路:

a) DFU 模式下暴力破解 iPhone4 及以下的 iOS 智能终端。

DFU(Development Firmware Upgrade)是 iPhone 固件的强制升降级模式, 也称之为工厂模式。因 iPhone4 及以下版本的 iOS 智能终端的 Bootloader 中存在引导漏洞, 允许使用经过特殊修改的第三方

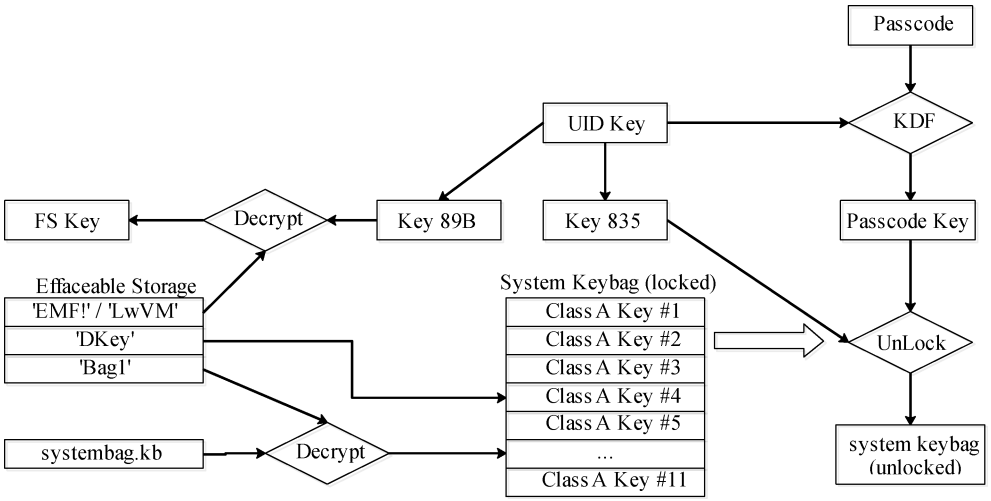

图 7　iOS 智能终端解密和解锁流程

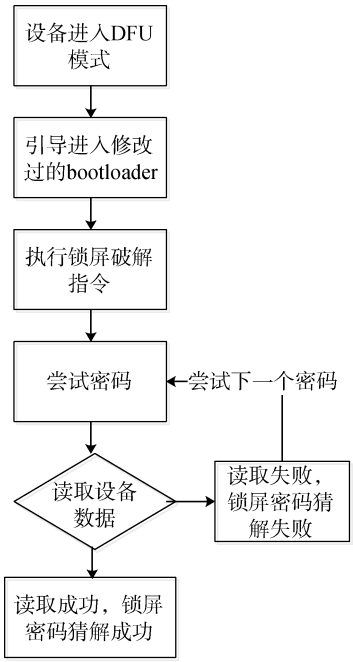

图 8　DFU 模式破解锁屏密码

Bootloader 程序。通过将密码破解程序打包到 Bootloader 程序, 进入 DFU 模式, 绕过 iOS 操作系统自身的密码错误次数限制, 可以暴力猜解 iOS 智能终端的锁屏密码。如图 8 所示。4 位数字锁屏密码猜解需要近 25 分钟, 6 位数字需要近 22 小时, 10 位数字近 25 年。

b) 使用屏幕锁破解辅助工具

专用破解工具[23]利用了 iOS 智能终端判断密码错误次数的漏洞, 每猜测数次锁屏密码就强制重启设备并重置设备错误密码计数。专用破解工具已经支持到 iOS8.2 操作系统及 iPhone5s 设备。屏幕锁破解辅助工具的工作流程如图 9 所示。

锁屏破解工具的破解核心在于依据屏幕亮度变化来检测密码是否输入正确, 并自动化的输入要猜测的密码, 从而将暴力破解锁屏密码的过程全自动化。根据文献[23]给出的资料, 不同的 iOS 设备破解耗时存在差异, 破解 4 位锁屏密码的最大耗时从 17 小时到 110 小时不等。。

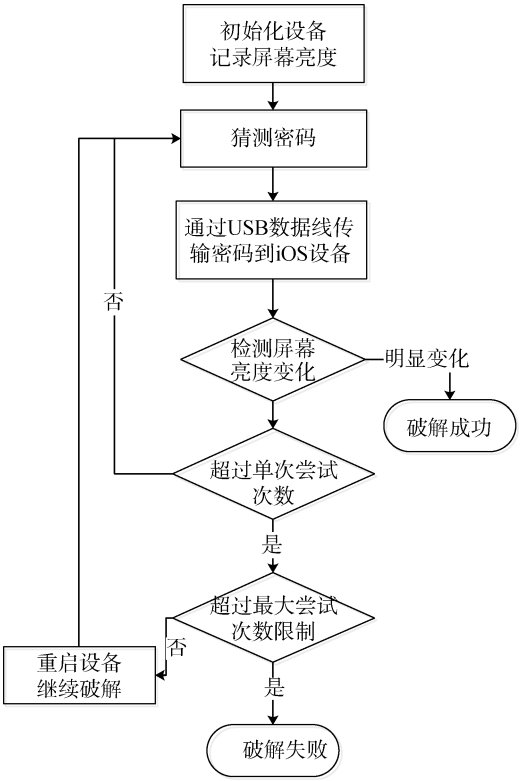

图 9　锁屏专用破解工具的破解流程

c) 破解 iTunes 备份的密码, 尝试使用 iTunes 备份的密码解锁 iOS 智能终端

iTunes 允许用户在进行 iOS 智能终端备份时, 指定一个密码保护用户的备份数据。其加密算法是利用用户输入的备份密码对所有的备份数据使用

PBKDF2 算法进行加密之后再进行 AES 加密。利用彩虹表不断尝试不同的密码组合, 重复 iTunes 的数据加密过程可猜解出 iTunes 加密备份的解密密码[6]。

具体过程为: 解析 iTunes 加密备份的 Manifest.plist 文件, 并从该文件的 BackupKeyBag 中获取解密数据 keybag。解析 keybag 数据并从中提取 SALT、ITER 以及 WPKY 等几个重要字段的值。通过 PBKDF2 算法, 使用盐值 SALT、迭代次数 ITER 对假设的解密密码进行加密。采用加密结果和 AES-ECB 算法对数据 WPKY 进行解密, 然后验证 WPKY 的解密结果前 8 个字节是否是"0xA6A6A6A6A6A6A6A6"。如果是, 则猜测的解密密码是正确的, 否则需要尝试下一个解密密码。

破解 iTunes 加密备份不受 iOS 智能终端的运算能力的限制, 在 PC 机(CPU 为 i5-4590, 8GB 内存)上测试, 单线程破解 iTunes 加密备份, 4 位的加密密码可以在 5 到 10 分钟之内破解出来。

3.2 Android 智能终端的取证技术

3.2.1 取证中的难点问题

Android 操作系统主要从四个方面保护系统和应用数据的安全[24]:

a) 应用程序层: Android SDK 提供应用程序访问系统资源的接口, 程序运行时需要向操作系统申请访问权限。

b) 应用框架层: APK 数字证书, 同一包名且采用同一数字证书的应用才被认为是同一应用, 是应用升级和设置应用间通信的权限的身份证明。

c) Android 运行环境层: 每个应用均拥有一个虚拟机, 应用间无法相互访问私有数据, 保证程序独立安全性。数据共享通过 ContentProvider 进行。

d) 操作系统层: Android 在权限管理采用 Linux 的 ACL(Access Control List)权限机制, 文件访问区分群组和用户, 权限包括可读、可写、可执行。

由于 Android 操作系统是开源的, 各智能终端设备厂商可进行不同程度的功能和安全机制调整。如 Android 4.0 操作系统开始就已经支持存储系统全盘加密, 但厂商出于性能、成本等各方面的考虑, 迄今为止仅有少数 Android 设备默认开启了此功能。同 iOS 相比, Android 智能终端设备还有一个显著的特征就是存在着明显的"碎片化"。即 Android 设备类型繁多, 其上运行的 Android 操作系统版本各异。

不同的 Android 智能终端设备, 取证技术手段存在一定差异。目前 Android 智能终端的取证技术主要包括 Android 智能终端在线取证、通过 Recovery 模式取证、Android 设备的芯片级取证、Android 智能终端 Backup 数据离线取证以及 Android 智能终端镜像文件离线取证等。

3.2.2 Android 智能终端的取证技术方案

(1) Android 智能终端在线取证

类似于 iOS 智能终端设备, Android 智能终端设备在 Android4.0 操作系统以来, 也采用了 Android 设备主动授权连接的模式。通过 USB 数据线将手机与电脑连接, Android 设备首先需要开启"USB 调试模式", USB 调试又称为 USB Debug, 是 Android 为开发者提供的用于开发工作的功能, 通过该功能可实现在计算机和移动设备之间复制数据、安装应用程序与读取日志数据等功能。Android4.0 及以上版本的设备还需要在 Android 设备上允许同取证设备建立连接, 连接后通过 *adb.exe devices* 命令即可枚举所有已识别的 Android 设备[7]。

USB 调试模式启动后, 取证设备即可与检材进行数据通信, 但是由于 Android 设备的权限控制, 对于未 Root 的 Android 设备, 仅能够获取到 SdCard 目录下的文件。如想访问系统目录或者应用程序数据目录则需要 Root 权限。ROOT 是一种 Android 系统中的"提权"方式, 获取 Root 权限后, 就可以通过 ADB 读写整个文件系统[25]。从 Android 设备中获取数据主要通过 ADB pull 命令:

Adb –s [设备 ID] pull [设备路径] [本地存储路径]

(2) Android 智能终端 Recovery 模式取证

对于部分不能 ROOT 的手机可以通过 Recovery 模式获取数据。Recovery 是 Android 设备的特殊启动模式, 通常用于为 Android 设备重新安装操作系统、清空数据并恢复出厂设置。与正常启动 Android 操作系统不同, Recovery 模式不需要输入锁屏密码解锁, 不需要打开 USB 调试模式。Recovery 模式是以 Root 权限运行的, 因此即使设备没有被 ROOT 也可以获取全部分区的数据访问权限[26,27]。

通常而言, 在 Android 设备加电启动时同时按住"电源键"和"音量+键", 稍等片刻即可进入 Recovery 模式, 之后再通过 USB 连接 Android 设备与取证设备即可。Recovery 模式下, 可直接使用 adb 连接 Android 设备, 并获得一个 shell[26,27]。

Recovery 模式取证优势明显, 但是也存在一些现实困难。通常 Android 智能终端并不自带 Recovery 模式, 且人工向 Android 智能终端安装 Recovery 也很复杂。部分机型刷入 Recovery 需要先解锁设备, 而解锁将会清空设备上的数据[26,27]。

"小米"系列手机在国内 Android 市场的占有率处于领先地位[1]。对于存在锁屏密码并且没有开启 USB

调试的“小米”手机无法进行 Root。这是就可以借助 Recovery 模式来取证在开机加电时。同时按住“电源键”及“音量下键”进入设备的 FastBoot 模式。在 FastBoot 模式下, 输入引导 recovery 启动的命令:

Fastboot.exe boot [Recovery.img 路径]

稍等片刻, 即可进入 Recovery 模式[28]。Recovery 模式下 Android 设备的取证分析过程与已获取 Root 权限的 Android 设备在线取证分析过程一致。

(3) Android 智能终端的芯片级取证

“芯片级”取证是终结 Android 智能终端数据保护的最后手段, 当所有的破解技术手段都失去作用时, “芯片级”取证将成为最后的“救命稻草”。“芯片级”取证利用了 Android 设备硬件上设计的 JTAG “调试接口”。JTAG[7,29]是一种国际标准测试协议, 主要用于芯片的测试。在电子证据取证工作, 可用于提取难以解锁或损坏的手机数据, 特别适用于提取高通芯片组的 Android 系统手机存储芯片中的数据镜像。提取数据时, 需要将手机外壳拆开, 并找到主板上的测试点。JTAG 接口“隐藏”在 Android 设备的主板上, 需要手工焊线连接接口与取证设备。JTAG 接口主要利用到核心的 5 个触点 TDI、TDO、TCK、TMS 以及 TRST, 如图 10 所示。JTAG 取证没有固定的模式和线缆插槽, 需要仔细寻找确认关键的触点是否焊接正确, 取证过程中需要保持恒定的输入电流电压, 对取证人员和环境都有较高要求。JTAG 取证直接利用调试协议驱动 CPU 读取存储区域的数据, 读出的数据即为 Android 设备的芯片级数据镜像[29]。通常 Android 设备的芯片级镜像包含若干个分区, 每个分区可能采用了不同的文件系统。

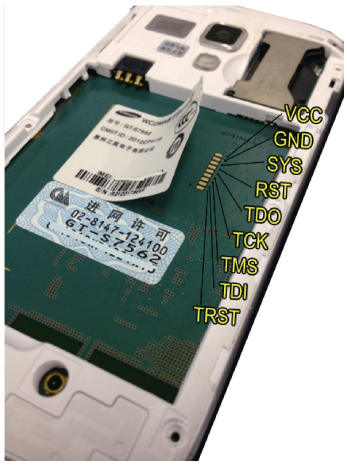

图 10　手机 JTAG 接口示意

(4) Android 智能终端 Backup 数据离线取证

对 Android 设备的检材进行 Root 并不总是成功或者被允许的, 而进入 Recovery 模式也不是所有品牌手机都支持的, 这种场合就需要通过 Android 设备的“ADB Backup 模式”取证[30]。

Android4.0 版本以后增加了备份功能, 连接 Android 设备与取证设备之后输入如下命令即可进行备份: *adb.exe backup [备份路径]*。ADB Backup 命令备份结果为*.ab 文件, 实际是.tar.gz 格式的数据压缩包, *.ab 文件包含了 Android 设备中近 80%的用户使用数据。

随着 App 的版本升级, 部分 App 如 QQ、微信的新版本已经不支持通过 Backup 方式获取到用户使用痕迹数据。但是 Android 系统的 App 安装过程存在漏洞可以使得取证分析人员绕过新版本 App 禁止备份的限制。首先通过 Adb 提供的 uninstall 命令卸载新版本的 App, 在卸载时加上 –k 参数, 即可使得卸载 App 时保留用户的使用痕迹数据。卸载完成之后再安装老版本的 App, 最后再通过 adb 的 Backup 命令备份 App 的数据即可获取到相关证据信息。

(5) Android 智能终端镜像文件离线取证

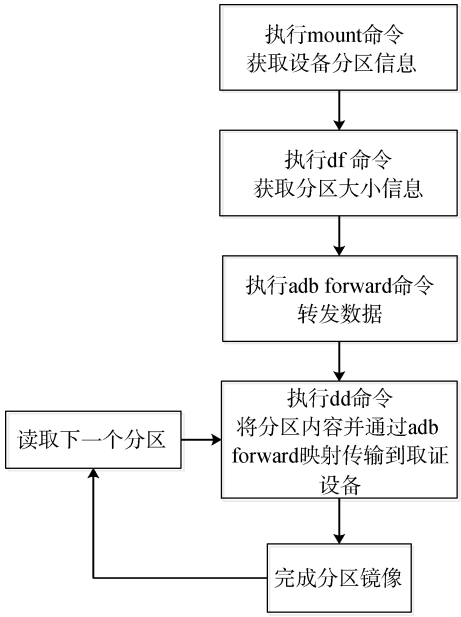

图 11　获取 Android 智能终端存储镜像

通过 ADB Backup 备份的镜像只包含逻辑文件数据, 如果需要对用户删除文件进行恢复则需要对整个 Android 设备的存储镜像。采用 DD 工具, 能够忽略文件系统, 以磁盘扇区为单位复制整个存储或分区的数据。镜像 Android 设备存储的同时, 需要把数据发送给电脑端。数据通信基于 socket。具体过程为: 首先在手机端运行镜像程序, 以后台服务方式运行并监听指定端口。然后电脑端通过调用 mount 命令获取手机所有分区, 以 socket 方式连接手机端服务程序, 连接之前需要通过命令将端口数据进行

转发:

ADB -s [设备 ID] forward tcp: port tcp: port

然后发送数据告知需要镜像的分区。手机在收到命令后, 开始进行设备镜像, 并将数据实时发送给电脑端。Android 设备镜像过程如图 11 所示。通过获取的镜像, 可利用文件签名特征来扫描镜像文件, 恢复出已删除的照片、视频等文件。

(6) Android 设备锁屏密码破解

Android 设备的密码保护主要包括“手势锁屏密码”、“简单数字密码”、“复杂密码”。如图 13 所示。Android 设备的锁屏密码存储在/data/system 目录下, 名为 password.key 或 gesture.key, 其中 Password.key 存储的是数字密码和复杂密码, Gesture.key 存储的是手势密码。

对于手势密码, Android 最终将其转换为数字序列, 并使用 SHA-1 对数字序列进行加密存储。九宫格样式的手势密码, 其排列组合仅有 389112 可能。可将九宫格手势序列与 SHA-1 散列值预先计算好。在需要进行解密时, 通过 Recovery 模式读取/data/system/gesture.key 文件的内容, 并在预算计算的数据表中直接查表对比即可。

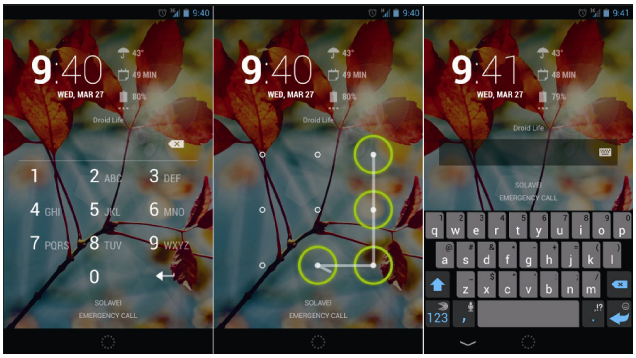

图 12 Android 锁屏密码保护

对于开启了 ADB 调试, 并且已经 Root 过的 Android 设备, 直接通过 ADB shell 进入/data/system 目录, 删除密码存储文件(password.key 或 gesture.key)即可破除 Android 的锁屏密码保护。

对于未开启 ADB 调试且没有 Root 的 Android 设备, 根据不同设备有不同的解决方案。“小米”系列手机可以进入 Recovery 模式下删除/data/system/gesture.key 和/data/system/password.key 文件; “三星”系列手机可以通过 USB 数据线利用三星特有的 mobex 协议[31]进行通信, 删除 password.key、gesture.key 文件, 或者通过 Odin 模式刷入指定的 Recovery 文件, 之后再通过 adb 命令删除上述密码文件。移除密码配置文件之后, 重启设备输入任意密码即可进入 Android 系统。

3.3 BlackBerry、Windows Phone 和 Symbian 系统的取证

BlackBerry、Windows Phone 和 Symbian 智能终端的市场占有率已处于较低的水平。BlackBerry 操作系统的安全机制极为出色, 不仅采用了高强度的存储芯片进行全盘数据加密, 而且从 BlackBerry OS 10 版本开始对备份数据也采取了默认加密策略; Windows Phone 智能操作系统的安全性也相对较高, 仅有极少数 Windows Phone 智能终端能够“越狱”获取 Root 权限; Symbian 系统的安全机制相对薄弱, 取证方案主要采取在 Symbian 手机上安装代理 App, 同分析备份协同进行取证。

BlackBerry 使用的是 Research In Motion 专用的操作系统, 与 Symbian、Windows Phone、Android 相比, 存储芯片强制采用数据加密, 使得芯片级取证无法适用于黑莓设备。BlackBerry 系统的取证可以通过直连 BlackBerry 设备, 或者对备份 BlackBerry 后通过备份进行取证分析。

对于搭载了 BlackBerry OS 10 之前版本操作系统的 BlackBerry 设备, 通过 USB 连接到取证设备之后, 可选择通过 USB Drive 类型挂载。这样 BlackBerry 设备会被识别为可移动的磁盘, 然后通过取证分析软件即可分析设备持有人的上网行为、多媒体数据信息等。

此外, 可以通过黑莓公司提供的 BlackBerry Desktop Software(适用于 BlackBerry OS 10 之前版本系统)或 BlackBerry Link(适用于 BlackBerry OS 10 及之后版本系统)管理软件备份设备数据。备份数据为 ipd 或者 bbb 格式, 本质上是一种压缩文件, 包含了用户使用 BlackBerry 设备时产生的短信、通信录、通话记录等证据信息。

自 BlackBerry OS 10 版本开始, BlackBerry 设备在备份时会被强制使用设备持有人的 BlackBerryID 通过黑莓提供的 BES 在线服务生成加密密钥, 对备份出来的数据使用 BGP 算法加密存储。因此, 针对 BlackBerry OS 10 版本的黑莓设备取证时, 还需要获取设备持有人的 BlackBerryID 及密码[32]。

Windows Phone 智能操作系统由微软在 2010 年发布, 版本从 7.0 历经 7.8、8.0、8.1 直到最新的 Windows Phone10。Windows Phone 智能终端采用了十分安全的操作系统设计、对外暴漏极少的外部接口, 从发布至今仅少数型号的设备能够获取 Root 权限。Windows Phone 设备的取证通常采取设备直连、数据备份和芯片级取证三种手段。

通过 USB 数据线连接 Windows Phone 设备与取证设备, Windows Phone 设备以 WPD 模式连接, 可以取证分析部分用户多媒体数据数据。

搭载 Windows Phone8 以下版本的设备, 可以通过微软提供的 Zune 同步软件进行数据备份, 但是仅能通过备份取证分析到设备持有人的通信录及部分多媒体数据。通过设备直连方式获取 Windows Phone 设备的数据也仅仅能够获取到照片、音乐等少量多媒体数据。

Windows Phone 设备可以利用主板上遗留的 JTAG 接口进行芯片级的数据取证。通过 JTAG 接口驱动 CPU 读取存储芯片的全部数据, 即可获取到 Windows Phone 设备的存储镜像。Windows Phone 设备的存储层采用了 NTFS 文件系统, 存储镜像可支持删除文件恢复, 以及短信、通信录、通话记录、上网记录以及 QQ 等数据的取证分析[33,34]。

Symbian 是诺基亚公司自 1997 开始开发的针对智能终端使用的操作系统。Symbian 操作系统的取证, 可以通过 Nokia PC Suite 制作的设备备份来进行。设备备份为 nbu 格式, nbu 格式的结构较为简单。从 nbu 文件 0x14 字节开始记录了 Symbian 设备的类型、名称、IMEI、固件版本、操作系统版本、语言以及地区信息, 从设备区块之后的 0x14 字节读取到 nbu 文件的分区数量, 紧接着为不同的分区块。如第 2 区块存储了联系人信息、第 4 区块存储了日历信息、第 6 区块存储了短消息记录等[35]。

3.4 智能终端设备删除数据恢复

智能终端删除恢复技术是取证过程中的重要步骤。智能终端通常采用软硬件一体的设计, 难以采用 PC 对硬盘制作的镜像的方式获取智能终端存储镜像。目前智能终端在删除数据恢复方面的研究集中在记录级删除数据恢复和文件级删除数据恢复。

记录级删除数据恢复主要针对智能终端中广泛应用的 SQLite 数据库进行分析。Sqlite 是轻量级嵌入式数据库, 所有数据均存储在同一个文件中, 为了加快数据的存取速度, Sqlite 在删除记录时并不会彻底销毁掉数据记录, 而是采用标记位的方式将删除的记录设置为空闲空间。通过采用层次递归遍历算法, 扫描 Sqlite 文件的空闲空间, 即可查找到所有已删除的数据记录。2009 年, Pereira 等[36]提出了一个基于 SQLite 内部记录格式的删除数据恢复方案, 利用运行时日志文件信息恢复 Firefox 浏览器的被删除记录。2011 年 Jeon 等人提出了从 SQLite 数据库未分配区域中, 如空闲块等, 恢复被删除记录的方案[37]。

文件级删除数据恢复针对用户使用过程中所删除的图片、视频、语音等数据进行删除恢复。文件级删除数据恢复基于文件格式签名, 对存储镜像中的数据进行匹配, 从而从镜像中恢复出所有未被覆盖的文件。文件级删除数据恢复首先需要获取到智能终端设备完整的存储镜像文件。获得了完整的智能终端设备存储系统镜像之后, 即可采用基于文件系统的删除恢复取证出被删除的文件名、文件路径等文件属性信息, 或者采用基于签名恢复的技术手段从存储镜像中恢复出所有未被完全覆盖内容的已删除文件。

文件级删除数据恢复依赖于智能终端的存储镜像。不同类型的智能终端采取的安全机制不同, 对于文件级删除数据恢复的支持程度也各有不同。

对于 iOS 智能终端, iPhone4 及之前版本的 iOS 智能终端存在 BootLoader 引导漏洞, 因此可以对存储镜像进行解密, 实现文件级删除数据的恢复。iPhone4 之后的 iOS 智能终端仅仅能够对已越狱的设备制作镜像, 而无法进行文件级删除数据恢复。

Android 智能终端, 可以对成功 Root 的设备制作完整存储镜像, 或者通过芯片级取证制作存储镜像。对于采取了全盘加密技术的 Android 设备尚无法进行文件级删除数据恢复, 但是采取全盘加密技术保护的 Android 智能终端目前还较少。

BlackBerry 智能终端, 由于采取了全盘加密技术, 尚无法进行文件级删除数据恢复。Windows Phone 设备, 数据存储层采用的是 NTFS 文件系统且并未加密, 能够进行文件级删除数据恢复。Symbian 设备, 也能获取存储镜像并应用文件级删除数据恢复技术。

4 智能终端取证技术的应用

4.1 面向新型智能终端取证软件的设计

基于对新型智能终端取证技术的研究, 笔者所在的实验室设计开发了面向新型智能终端的取证分析软件。软件首要考虑的是需要支持对市面主流智能终端设备的勘查取证, 需要支持对主流智能终端应用程序的取证分析。因此软件多采用分层架构设计, 并采取插件接口支持对新设备、新应用的取证功能扩展。

新型智能终端取证软件的框架设计如图 13 所示, 自底向上分为设备接入层、数据提取层、数据分析层以及 UI 展示层。设备接入层适配各型智能终端设备的接入; 数据提取层向上层数据分析层提供统一的数据读取接口; 数据分析层基于数据提取层获取的证据信息对用户的通信行为、虚拟身份、轨迹信

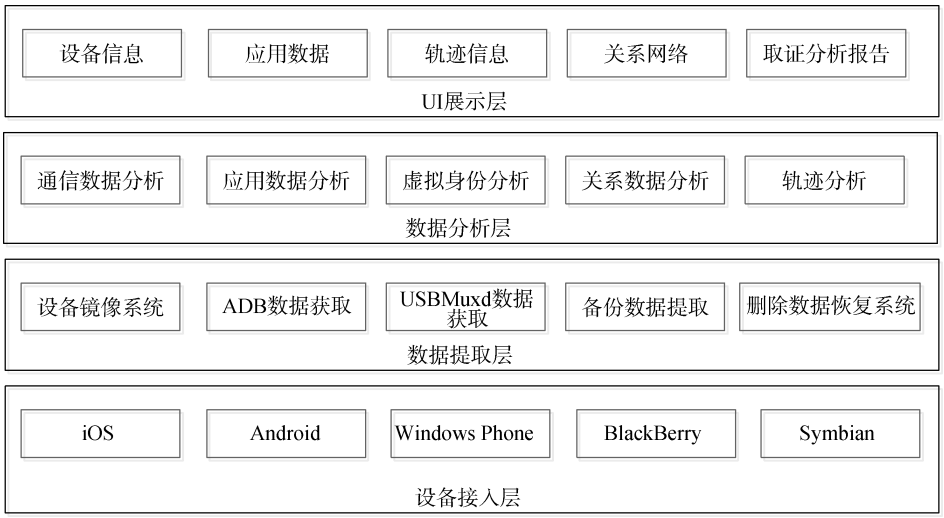

图 13　新型智能终端取证软件框架设计

息以及关系网络进行取证分析; UI 展示层则提供了取证分析人员的人机交互界面, 向取证分析人员提供各类取证分析结果及取证分析报告。

4.2　实际案例研究

本节采用取证鉴定司法实践中的实际案例来剖析智能终端取证技术的应用。案例中待取证的智能手机有三台, 具体型号分别是 iPhone 6(iOS 版本 9.1)、小米 4c(android 5.1.1 版本)以及 Lumia 620 (Windows Phone 8.1 版本), 其中 iPhone6 手机未设置锁屏密码、未越狱, 小米 4c 和 Lumia620 手机均设置了锁屏密码。三部手机中均需取证短信、微信和浏览器中的电子证据, 由于智能终端操作系统、设备类型不同, 对电子证据的取证分析方法也存在较大差异。

(1) iPhone6 手机取证

通过基于 USBMuxd 协议从设备的 AppDomain 区域获取到微信应用数据, 通过 iTunes 备份提取了短信和浏览器使用数据。

iPhone6 的短信数据存储于 SQLite 数据库 sms.db 中, 通过 SQLite 删除数据恢复技术对其进行检验是取证中的必要过程, 从中能恢复不少重要的被删除短信记删除恢复录。从 sms.db 数据库的 message 数据表中读取所有短信记录, 表中 date 字段包含短信的收发时间、text 字段包含正文内容、account 字段指明了短信通信对象。

iPhone6 的微信数据存储在 com.tencent.xin 目录下的 MM.sqlite 数据库中。微信数据库也采用了 SQLite 数据库存储, 数据库未加密。微信的好友、群组关系存储于 Friend 数据表中, 表字段包括了好友的用户名、昵称、邮件、手机号码以及关键的 UsrName 字段。微信的聊天数据均存储在采用 *Chat_[MD5 (UsrName)]*命名的数据表中。通过 Friend 表解析出好友关系之后, 即可进一步找出同该好友的聊天记录表。聊天记录表主要字段包括好友 ID “friendId”、消息内容“Message”、消息类型“Type”等字段。消息类型 Type 指明该条聊天记录是图片、语音还是文本消息。基于以上数据的取证, 能够完整分析出微信的账号、关系以及好友、群组的聊天内容。

iPhone6 的 Safari 浏览器将书签存储在 Bookmarks.db 数据库中, 浏览历史记录则存储在 History.plist 文件中。Bookmarks.db 同样是 SQLite 数据库, 其中 bookmarks 表存储了浏览器的书签内容, 主要字段为 title、type、id、url 分别存储了书签的标题、类型以及书签 URL。浏览器的历史记录文件 History.plist 是 plist 格式的文件, 可以取证浏览器历史记录, 重要的字段主要包括 title、lastVisitedDate、visitCount 以及 redirectURLs 字段, 分别存储了历史记录的标题、最后访问时间、访问次数和 URL 信息。

(2) 小米 4C 手机取证

通过 Recovery 模式也顺利绕过它的锁屏密码保护, 并成功提取到短信、微信及浏览器中的电子证据。小米 4C 的短信、微信和浏览器使用数据均以 SQLite 数据库进行存储。短信数据存储在 mmssms.db 数据库的 sms 表中, 主要字段包括 address、date、body、type 等, 分别记录了短信的收件地址、收发时间、短信正文内容以及短信内容。

微信应用数据存储在/data/data/com.tencent.mm/MicroMsg 文件夹中。在 Android 系统中, 微信应用的数据(数据库 EnMicroMsg.db)是采用 Sqlcipher 加密存储的, 无法直接进行检验分析。首先需要从 system_config_prefs.xml 配置文件中获取微信账户的

uin 值, 并提取设备 IMEI 数据。计算字符串 IMEI-uin 的 MD5 值, 并取前 7 位作为解密密码来解密 EnMicroMsg.db 数据库。解密后微信数据库的 rcontact 表存储了微信好友和群组信息, 主要包括 username、nickname 和 type 字段, 分别记录了用户名、昵称和好友类型。与 iOS 系统的版本不同, Android 微信的所有聊天记录均存储在 message 表中, 通过 message 表的 talker 字段来区分不同好友、群组的聊天记录。

小米 4C 手机的浏览器使用数据存储在 browser2.db 数据库中, 其中 history 表、bookmarks 表分别记录了浏览器的历史访问记录和书签记录。History 表的关键字段 title、date 和 url 分别存储了历史记录的标题、访问时间和 URL 信息。Bookmarks 表的 title、created、url 则分别记录了书签的标题、书签创建时间和书签的 URL。

(3) Lumia620 手机取证

通过芯片级取证技术提取了手机的完整存储镜像。Lumia620 使用了 windows phone 8.1 操作系统, 数据常用存储格式与 iPhone6、小米 4c 明显不同。Lumia620 手机的短信、浏览器历史记录均存储在微软开发的 ESE 数据库中(Windows 可扩展存储引擎 Windows Extensible Storage Engine)。

Lumia620 手机的短信数据存储在路径为 WPCOMMSERVICES\APPDATA\Local\Unistore\store.vol 的 ESE 数据库文件中。ESE 数据库的解析相对复杂, 短信的正文内容主要存储在数据库的 Message 表中。表的关键字段“0e060040”、“0c1f001f”和“0037001f”分别存储了短信的收发时间、短信联系人和短信的消息内容。

Lumia620 手机的微信应用数据存储在路径为 /Users/DefApps/APPDATA/{23E1505B-9383-4ED4-9195-DA23A3442820}/Local/storage/database.sdf 的文件中。微信应用数据采用了 SQLServer 数据库存储, 数据库中的 Contact 表和 ChatMsg 表分别存储了微信的好友、群组关系和聊天正文内容。Contact 表的关键字段主要包括 strUsrName、strNickName、strMobile 等, 分别存储了好友的用户名、昵称和手机号码等信息。ChatMsg 表主要字段为 strTalker、strContent、nMsgType、nCreateTime 等, 分别存储了微信消息的发送者、消息内容、消息类型和消息收发时间。对这两个数据表进行关联分析, 取证 Lumia620 手机中微信的聊天记录。

Lumia620 手机的浏览器使用记录存储在 APPDATA\Local\Microsoft\Windows\WebCacheV01.dat 文件以及 SharedData\InternetExplorer\Favorites 目录下, 这两个位置分别存储了浏览器的访问历史记录和书签记录。WebCacheV01.dat 文件为 ESE 数据库, 历史访问记录存储于 History 数据表, 表关键字段包括 AccessCount、ModifiedTime、AccessedTime 和 Url, 分别存储访问次数、创建时间、最后访问时间和访问的 URL。Lumia620 手机的浏览器书签存储在 Favorites 目录下, 浏览器书签均以 url 链接文件的形式存储, 直接读取链接文件的内容可获取书签的 URL 信息、链接文件的创建时间对应书签的创建时间。

5 总结与展望

本文详细分析了当前主流的 iOS、Android、Windows Phone 等平台下的移动设备的安全机制及取证技术难题, 结合目前主要的安全机制破解方案研究并提出了面向新型智能终端的取证技术方案。本文的研究将有利于解决新型智能终端的安全防护机制给电子数据取证工作带来的难题。

目前, 移动智能终端终端所采用的安全技术仍然是电子数据取证工作所面临的重要挑战, 主要体现在:

首先, 对于智能终端取证而言, 访问权限不足会导致关键数据无法获取, 给整个取证鉴定工作带来很多障碍。相关技术的研究总是滞后于智能终端设备的发展, 如 iOS9.0-9.1 越狱工具推出时, 苹果向用户推送 iOS9.2.1 已经有一段时间, 且至今仍未有针对 iOS9.2 以上版本的越狱工具, 同时 App 访问权限的突破和解密数据本身也具有较高的技术挑战性。

其次, 移动智能终端数据存储形式、安全设置、操作方法复杂多样, 各厂商尚无统一的标准。手机取证往往只能对检材手机进行直接操作。特别是需要破解获取系统 ROOT 权限时需要往设备内部写入程序, 存在破坏证据数据的可能性。由此可见对移动智能终端无损取证难度很大, 电子数据取证鉴定的工作人员的意外操作直接会影响所获取的电子证据的证据力。

再次, 随着用户隐私保护意识的加强, 智能终端厂商和第三方安全类公司都推出了众多安全工具, 如远程数据抹除(防被盗后泄密)、文件保险箱、文件粉碎机等。这类应用同样也给取证分析带来了不少影响, 如犯罪嫌疑人可利用文件保险箱、粉碎机作为反取证的手段。

最后, 随着云计算应用、产品的普及。智能终端应用逐渐出现了将数据存储迁移至云端的趋势。已有的智能终端取证分析技术和产品可能会“遗漏”大量云端的重要电子证据。

针对上述问题。在技术方面要求电子数据取证技术研究者加大以下几个方向的研究: 首先是对移动智能终端安全机制、安全应用软件及相应破解关键技术的研究, 以及对反取证手段的检测与反制技术研究; 其次, 面向智能终端设备电子物证保护和终端、服务端协同取证的需求, 研究基于仿真执行等技术的智能终端取证分析方案; 最后, 积极跟上云服务等新型应用模式的发展, 研究云环境中的电子证据固定保全和检验分析技术, 并支持在海量取证数据中对案、事件线索关系的分析与查证。确保取证技术方法能适应安全技术发展, 能为电子数据的取证鉴定司法实践提供适合的技术方案和设备。同时, 还需要紧跟信息技术的发展和演进, 加快建设和完善的面向新型智能终端设备、新型应用环境(如云服务)的取证标准体系, 建立适合我国国情的统一的技术规范。为操作复杂纷繁的智能终端电子数据取证鉴定工作提供最佳实践指南, 保证智能终端取证的工具、技术方法和操作方法更具有科学性, 保证电子证据的法律效力。

参考文献

[1] "Report on mobile internet development and security in China," Internet Society of China and CNCERT/CC, http://www.isc.org.cn/download/2016report.pdf, May, 2016.
(中国移动互联网发展状况及其安全报告(2016),中国互联网协会, 国家互联网应急中心, http://www.isc.org.cn/download/2016report.pdf, 2016.05)

[2] Z. Yang, B. Liu and R. Xu, "Current Situation and Trend of Digital Forensics Research," *E-science Technology & Application,* vol. 6, no. 1, pp. 3-11 (in Chinese), 2015.
(杨泽明, 刘宝旭, 许榕生. 数字取证研究现状与发展态势[J]. 科研信息化技术与应用, 2015(1).)

[3] "Depth analysis of fraud crossing platforms of networks and telecommunications," 360, http://zt.360.cn/1101061855.php?dtid=1101061451&did=1101783053, May. 2016.
(深入分析跨平台网络电信诈骗, 360, http://zt.360.cn/1101061855.php?dtid=1101061451&did=1101783053, 2016.05)

[4] "FBI–Apple encryption dispute," Wikipedia, https://en.wikipedia.org/wiki/FBI%E2%80%93Apple_encryption_dispute, 2016.

[5] K. Barmpatsalo, D. Damopoulo, G. Kambourakis and V. Katos, "A critical review of 7 years of Mobile Device Forensics," *Digital Investigation*, vol. 10, no. 4, pp. 323-349, 2013.

[6] S. Morrissey and T. Campbell, "iOS forensic analysis for iPhone, iPad, and iPod touch," Berkeley, Calif.: New York: Apress, 2010.

[7] A.Hoog, "Android forensics: investigation, analysis and mobile security for Google Android," Elsevier, 2011.

[8] D. Abalenkovs, P. Bondarenko, V K. Pathapati, et al. "Mobile forensics: Comparison of extraction and analyzing methods of ios and android, " [Master Dissertation], GjÃÿvik University College, 2012.

[9] W. Qiu, Q. Su, B. Liu, et al. "iOS Data Recovery Using Low-Level NAND Images," *IEEE Security & Privacy*, vol. 11, no. 5, pp. 49-55, 2013.

[10] L. Gómez-Miralles, J. Arnedo-Moreno. "Versatile iPad forensic acquisition using the Apple Camera Connection Kit," *Computers & Mathematics with Applications*, vol. 63, no. 2, pp. 544-553, 2012.

[11] L. Ge, L. Wang. "Decryption and Forensic System for Encrypted iPhone Backup Files Based on Parallel Random Search," *International Conference on Applications and Techniques in Information Security*, Springer Berlin Heidelberg, pp. 347-358, 2015.

[12] K. Oestreicher, "A forensically robust method for acquisition of iCloud data," *Digital Investigation*, vol. 11, pp. S106-S113, 2014.

[13] T. Vidas, C. Zhang, N. Christin, "Toward a general collection methodology for Android devices," *Digital Investigation*, vol. 8, pp. S14-S24, 2011.

[14] N. Son, Y. Lee, D. Kim, et al. "A study of user data integrity during acquisition of Android devices," *Digital Investigation*, vol. 10, pp. S3-S11, 2013.

[15] S J. Yang, J H. Choi, K B. Kim, et al. "New acquisition method based on firmware update protocols for Android smartphones," *Digital Investigation,* vol. 14, pp. S68-S76, 2015.

[16] "AFC," The iPhone wiki, https://www.theiphonewiki.com/wiki/AFC, October. 2014.

[17] "Usbmux", The iPhone Wiki, https://www.theiphonewiki.com/wiki/Usbmux, Oct, 2011.

[18] "IOS jailbreaking," Wikipedia, https://en.wikipedia.org/wiki/IOS_jailbreaking. May. 2016.

[19] Y. Wang and L. Xu, "Research on Digital Forensics of iOS platform," *Digital Technology and Application*, no. 2, pp. 49-49 (in Chinese), 2015.
(王晨语, 薛亮, iOS平台的数字取证技术研究, 数字技术与应用, 2015(2), pp.49-49)

[20] Q. Su, "Research on Digital Forensic with iOS Devices [Master Dissertation] ," Shanghai Jiao Tong University, 2013.
(苏芊, iOS 终端数字取证研究[硕士学位论文], 上海交通大学, 2013)

[21] "Shopping for Spy Gear: Catalog Advertises NSA Toolbox," DER SPIEGEL, http://www.spiegel.de/international/world/catalog-reveals-nsa-has-back-doors-for-numerous-devices-a-940994.html, December 2013.

[22] M. Epifani and P. Stirparo, "Learning iOS Forensics," Packt Publishing Ltd, 2015.

[23] "IP-BOX with iOS Adapter iPhone Password Unlock Tool", TEEL technologies,http://www.teeltech.com/mobile-device-forensic-tools/ip-box-iphone-password-unlock-tool/

[24] P. Faruki, A. Bharmal, V. Laxmi, et al, "Android security: a survey of issues, malware penetration, and defenses," *Communications Surveys & Tutorials, IEEE*, vol. 17, no. 2, pp. 998-1022, 2015.

[25] X. Wan, J. He, G. Liu, et al, "Survey of Digital Forensics Technologies and Tools for Android based Intelligent Devices," *International Journal of Digital Crime and Forensics (IJDCF)*, vol. 7,

no. 1, pp. 1-25, 2015.

[26] H. Cui and Q. Zhao, “Lockscreen Password Cracking on Android Smart Mobile Phone Using MTP Protocol,” *Forensic Science and Technology,* vol. 40, no. 1, pp. 82-83 (in Chinese), 2015.
(崔鹤群, 赵强, 巧用 MTP 协议破解 Android 智能手机屏幕锁密码, 刑事技术, 2015(1): 82-83.)

[27] S. Shi, M. Li and M. Lei, “Lockscreen Passcode Decryption on Android Smartphone Using a Custom Recovery,” *Forensic Science and Technology*, vol. 40, no. 4, pp. 327-327(in Chinese), 2014.
(石穗东, 李蒙, 雷鸣, 运用第三方 recovery 破解安卓手机屏幕锁, 刑事技术, 2015(4):327-329.)

[28] J. Wang, C. Ji, and H. Pei, “Bypass for Screen Lock of Android Smart Phone,” *Forensic Science and Technology* ,vol. 40, no. 2, pp. 142-145(in Chinese), 2015.
(王即墨, 计超豪, 裴洪卿, Android 智能手机锁屏密码及破解方法研究, 刑事技术, 2015(2): 142-145)

[29] MF. Breeuwsma, “Forensic imaging of embedded systems using JTAG (boundary-scan),” *Digital Investigation*, vol. 3, no. 1, pp. 32-42, 2006.

[30] H. Liu, “Research on Android Device Forensic,” *Netinfo Security*, no. 9, pp. 29-32(in Chinese), 2015.
(刘浩阳, Android 设备取证研究, 信息网络安全, 2015(9):29-32)

[31] “OBEX and other Transfer Protoco”, xda-developers, http://forum.xda-developers.com/showthread.php?t=929206, January, 2011.

[32] SK. Sasidharanand KL. Thomas, “BlackBerry Forensics: An Agent Based Approach for Database Acquisition,” *Advances in Computing and Communications*, Springer Berlin Heidelberg, pp. 552-561, 2011.

[33] T. Schaefer, H. Höfken and M. Schuba, “Windows Phone 7 from a Digital Forensics’ Perspective,” *Digital Forensics and Cyber Crime,* Springer Berlin Heidelberg, pp. 62-76, 2011.

[34] S. Kumar, J. Kumar, S. Jithin, “A Novel Method for Windows Phone Forensics,” *International Journal of Scientific & Engineering Research*, vol. 6, no. 1, 2015

[35] Savoldi, P. Gubian, A. Savoldi and P. Gubian, “Symbian forensics: an overview,” Intelligent Information Hiding and Multimedia Signal Processing, 2008. International Conference on. IEEE (IIHMSP'08), pp. 529-533, 2008.

[36] M.T. Pereira. “Forensic analysis of the Firefox 3 Internet history and recovery of deleted SQLite records,” Digital Investigation, vol. 5, no. 3, pp. 93-103, 2009.

[37] S. Jeon, J. Bang, K. Byun, etc, “A recovery method of deleted record for SQLite database,” *Pers Ubiquit Comput*, vol. 16, no. 6, pp. 707-715, 2012.

金波　于 2000 年 2 月份在华东理工大学控制理论与控制工程专业获得博士学位, 现任公安部第三研究所 研究员。研究领域为信息网络安全。Email: jinbo@stars. org.cn

吴松洋　于 2011 年在同济大学计算机应用专业获得博士学位。现任公安部第三研究所 副研究员。研究领域为信息网络安全、电子数据取证、大数据。Email: wusongyang@stars.org.cn

熊雄　于 2011 年在华北计算技术研究所计算机软件与理论专业获得硕士学位。现任公安部第三研究所 研究实习员。研究领域为电子数据勘查取证技术。Email: xiongxiong@stars.org.cn

张勇　于 2013 年在华东师范大学计算机软件与理论专业获得博士学位。现任公安部第三研究所 助理研究员。研究领域为信息安全、电子数据勘查取证。Email: zhangyong@stars.org.cn