第5章

macOS取证技术

虽然 Windows 仍是当下绝大多数个人计算机用户操作系统的首选，但 macOS 也拥有一个相对稳定的用户群。随着 iPhone 浪潮的驱动和在中国市场的需求投入，苹果设备独有的工业设计风格、便捷的系统用户体验、日趋完善的应用程序体系和相对安全的数据保障，博得了越来越多的用户青睐。由于 mac 计算机硬件和 macOS 操作系统的特殊性，相应的取证技术和方法与在 Windows 操作系统环境下截然不同，这也给电子数据调查取证人员带来了很多困难，本章将针对苹果设备的桌面操作系统 macOS 取证技术进行介绍。

5.1 macOS 取证概述

苹果公司的软硬件一体化个人计算机平台统称为 Macintosh（简称 Mac），macOS 是一套由苹果公司开发的运行在该计算机平台上的操作系统，类似于个人计算机上安装的 Windows 或 Linux 操作系统。

5.1.1 Mac 的发展

1976 年，史蒂夫·乔布斯等创立了苹果公司，同年推出首款产品 Apple I（外形如图 5-1 所示）。

相比打字机一样的 Apple I，1977 年推出的 Apple Ⅱ成为全球首款真正意义上的个人计算机。得益于 Apple Ⅱ的成功，苹果公司也迅速实现扩张，于 1983 年推出第三代计算机 Lisa，作为全球第一款图形化计算机问世，为 Mac 的发展确定了方向。

1984 年 Macintosh 的上市标志着一个苹果时代的开始。其操作系统延续 Lisa 的图形化设计，虽然仍是个黑白系统，但光标、窗口等元素一应俱全，为以后桌面操作体系奠定了基础。1991 年，苹果公司首款便携式 Mac 设备 PowerBook 发布，同时苹果公司还向微软公司开放了图形界面授权。

1990—1997 年，苹果推出了多款 Mac 产品，但都没有得到社会的普遍认可。与此同时，Windows 操作系统在个人计算机领域的普及，一度让苹果公司失去了该领域的主导地位。1997 年，乔布斯回归苹果，大胆启用了设计师乔纳森·伊夫，推出了首款 iMac 产品（图 5-2）。该产品将 CRT 显示器、机箱融为一体，配以半透明外壳，现代苹果计算机的独特产品风格开始形成，得到了用户的认可。

图 5-1　Apple Ⅰ（1976）

图 5-2　iMac（1998）

2000 年，运行有 macOS X 操作系统的 iBook 推出，成为首款支持 Wi-Fi 的计算机；2001 年，PowerBook G4 上市，确定了苹果笔记本电脑的计风格；2002 年推出的 iMac G4 成为 Mac 发展史上的一个里程碑；2003 年，Power Mac G5 问世，采用了充满未来感的塔式设计，它对日后的 Mac Pro 系列产品产生了深远的影响，标志着苹果系列产品开始走向成熟；2005 年，iMac G5 问世，苹果公司在这款计算机上明确了 iMac 系列的发展方向，同年，苹果公司推出了一款低端计算机 Mac Mini。

2006 年，苹果公司放弃了 IBM 的 Power PC 处理器，转而使用英特尔产品，诞生了沿用至今的 MacBook Pro 系列产品。2011 年，苹果公司对 MacBook Pro 进行了一次重新设计，增加了 Thunderbolt（雷电）接口，并更改了键盘设计，2012 年又配备了 Retina 超高清屏幕。

2008 年问世的 MacBook Air 是苹果公司第一款一体化设计的笔记本电脑，实现了超轻薄机身和出众的质感，彻底改变了移动计算机的体验，其经典外形一直沿用至今。

2016 年，MacBook Pro 推出了全新的版本，该版本舍弃了之前的键盘快捷键区，取而代之的是支持触控操作的全新 Retina 屏幕条，苹果公司将其称为 Touch Bar。苹果公司认为 Touch Bar 可用来替代所有的右键操作，并在默认状态下禁用了右击操作。此外，键盘上有 Siri 的专属按键，还可以直接用 Touch Bar 上的 Touch ID 解锁 Mac。

2017 年，苹果公司发布了 iMac 系列硬件产品 iMac Pro，它搭载了最多 18 核至强处理器，首次配备了 Apple T2 安全芯片，同时还采用了 AMD 最新的 Vega 显卡。

2020 年，基于 Arm 架构的 Apple M1 芯片问世，同时首批搭载该芯片的 MacBook Pro、MacBook Air 和 Mac mini 等硬件也相继推出。

5.1.2　Apple T2 安全芯片

Apple T2 安全芯片是 Apple 公司设计的第二代定制化 Mac 芯片。通过对其他 Mac 计算机中的几款控制器进行重新设计与整合，例如系统管理控制器、图像信号处理器、音频控

制器和 SSD 控制器，T2 芯片为 Mac 带来了多项新功能，同时安全隔区协处理器为加密存储和安全启动功能的实现提供了支撑。T2 芯片采用专用的 AES 硬件为存储在固态硬盘上的数据加密，这既不会影响固态硬盘的性能，还能让 Intel Xeon 处理器专心处理运算任务。此外，安全启动功能确保底层软件不会被篡改，而且只有 Apple 公司信任的软件才能在开机时启动。

对于取证而言，T2 芯片的出现使关机状态下 Mac 计算机硬盘被加密，即便能够获取其镜像，也无法读取其中数据，为取证工作带来了困难。下列 Mac 计算机配备了 Apple T2 安全芯片：

（1）2020 年起推出的 iMac。

（2）iMac Pro。

（3）2019 年推出的 Mac Pro。

（4）2018 年推出的 Mac mini。

（5）2018 年起推出的 MacBook Air。

（6）2018 年起推出的 MacBook Pro。

在开启的 Mac 计算机中也可以通过“系统信息”来确认有没有配备 T2 芯片。具体操作方法为：首先，在按住 Option 键的同时，选取控制栏中的“系统信息”，然后选择“控制器”或 iBridge（具体视 macOS 版本来确定），在出现的如图 5-3 所示的界面中，如果看到“Apple T2 芯片”信息，即表示该 Mac 配备了 Apple T2 安全芯片。

图 5-3　在“系统信息”中查看 Apple T2 芯片

5.1.3　macOS 的发展

macOS 是首个在商用领域成功应用的图形用户界面操作系统。当年苹果公司推出图

形界面时，微软公司还停留在 DOS 时代，Windows 操作系统还没有正式推出。

Mac 的操作系统可以划分为两个系列：一个是较早推出且已停止服务的 Classic macOS，系统搭载首款 Mac 计算机于 1984 年问世，采用 Mach 内核，最初以 System 来命名，历经的版本有 System 1(1984)、System 2(1985)、System 3(1986)、System 4(1987)、System 5(1987)、System 6(1988)、System 7(1991)、macOS 8(1997)、macOS 9(1999)，终极版本为 macOS 9.2.2；另一个则是全新的 macOS。

2001 年，具有划时代意义的 macOS X 10.0 发布。macOS X 主要由两部分组成：一是 Darwin，是以 BSD 源码和 Mach 微核心为基础，类似 UNIX 的开放源码环境；二是由苹果开发、命名为 Aqua 的有版权的 GUI 界面。在 macOS X 发布的同年，macOS X Server 也同时发售，两个版本架构上完全相同，只是在包含的工作群组管理和管理软件工具上存在一些差异。也就是从这个版本开始，以后的每个 macOS X 系列的后缀都是以一种大型猫科动物来命名，包括后来的 10.1 Puma(美洲狮)、10.2 Jaguar(美洲虎)、10.3 Panther(黑豹)、10.4 Tiger(虎)、10.5 Leopard(美洲豹)、10.6 Snow Leopard(雪豹)。其中，2009 年发布的 Snow Leopard 是苹果第一个 Intel Mac 专用系统，从此使用 PowerPC 处理器的 Mac 设备就无法继续使用新系统。

2011 年，苹果推出对应 10.7 版本的 OS X Lion(狮子)，自此在操作系统正式名称中去掉了 Mac 字样和版本号；2012 年又推出了对应 10.8 版本的 OS X Mountain Lion(山狮)。2013 年起，由于大型猫科动物名称即将用尽，苹果发布 10.9 版本时，将系统定名为“Mavericks”(冲浪湾)，即加州北部的一处冲浪胜地。也正是从这个版本开始，苹果决定今后用加州景点名称作为系统代号，包括后来的 10.10 版本 Yosemite(优胜美地国家公园)、10.11 版本 El Capitan(酋长岩)等。

2016 年，苹果推出 10.12 版本的 Sierra(内华达山脉)，操作系统名称由 OS X 变更为 macOS，与 iOS、tvOS、watchOS 形成呼应。次年，macOS 10.13 High Sierra(内华达高脊山脉)发布，APFS 文件系统正式取代 macOS 上使用多年的 HFS+文件系统。

2018 年，macOS X Mojave 10.14(莫哈韦沙漠)发布；2019 年，macOS 10.15 Catalina(卡特琳娜岛)发布；2020 年，macOS 11.0 Big Sur(大瑟尔)发布，被称为是 macOS 发布以来最大的一次更新，同时此次更新将更好地实现与 Intel 平台、ARM 平台以及 iOS 平台 App 的兼容。

5.2 macOS 的系统特性

因为 Classic macOS 是一个不提供“命令行”模式的全图形操作系统，所以它虽然易于操作，但因为没有提供有效的内存管理、协同式多任务和扩展冲突处理等机制而被诟病。

新的 macOS 引入了一种新型的文件系统，一个文件包括了两个不同的“分支”，分别把参数保存在“资源分支”，而把原始数据保存在“数据分支”，这在当时是非常创新的。但是，该系统由于文件系统识别问题，与其他操作系统的兼容成为挑战。

为了解决这些问题，macOS X 使用了基于 BSD UNIX 的内核，并带来 UNIX 风格的内存保护和抢占式多任务处理。同时，改进了内存管理方式，允许同时运行更多软件，这从本质上消除了因一个程序崩溃而导致其他程序同时崩溃的可能性。这也是首个包括“命令行”

模式的 macOS。macOS X 有一个兼容层负责执行早期的 Mac 应用程序，名为 Classic 环境，也就是开发人员所熟知的“蓝盒子”(the blue box)，它把早期的 macOS 9. x 系统的完整复制作为 macOS X 里一个程序执行，但执行应用程序的兼容性在当前的硬件下难以达到理想的效果。

与之前的版本相比较，macOS 操作系统界面具有特色，突出了形象的图标和人机对话功能；同时，提供了独特的桌面、菜单栏、状态栏、程序坞、调度中心、通知中心、启动台等功能界面。本节主要基于 macOS 介绍其系统架构和主要功能。

5.2.1　系统架构

macOS 系统架构如图 5-4 所示，除硬件(hardware)外，主要包括操作系统内核(core OS)、应用程序接口(API)和图形用户界面(GUI)3 个组成部分。学习和理解其组成结构和主要组成部分的功能，对电子数据取证工作是非常重要的。

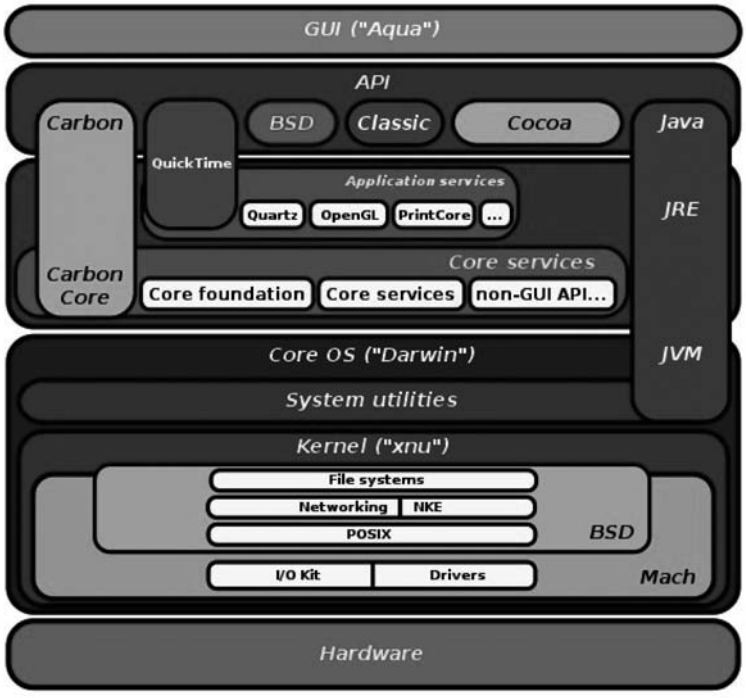


图 5-4　macOS 系统架构

1. Core OS

Core OS 是 macOS 的基础部分(也称为 Darwin)，它也是一款开放源代码的类 UNIX 操作系统，操作界面如图 5-5 所示。Darwin 主要由两部分组成：XNU 内核和 UNIX 工具。

通常讲的 macOS 内核是 Darwin，其实这是一个不严谨的说法，因为 Darwin 不只包含内核，还包括其他内容。严格地讲，macOS 的内核是 XNU(XNU 据说是“XNU's not UNIX”的缩写)。XNU 是一款结合了微内核与宏内核特性的混合内核，包括 Mach、BSD 和 I/O Kit。

图 5-5 uname 命令显示的 UNIX 或类 UNIX 系统信息

1) Mach

Mach 原来是一款微内核，XNU 中的 Mach 来自 OSFMK 7.3(open software foundation mach kernel)，它负责实现 CPU 调度、内存保护等功能。它是 macOS 内核中最重要的部分，XNU 中大部分代码来自 Mach，而且 macOS 中的可执行文件也是 mach-o 格式。

2) BSD

XNU 中包含一个经过修改的 BSD，它负责进程管理、UNIX 文件权限、网络堆栈、虚拟文件系统、POSIX 兼容。macOS 之所以符合单一 UNIX 规范，也正是因为 BSD。

3) I/O Kit

I/O Kit 是 XNU 内核中的开源框架，可帮助开发人员为 macOS 和 iOS 操作系统编写设备驱动程序代码。I/O Kit 框架由 NeXTSTEP 的 DriverKit 演变而来，与 macOS 9 或 BSD 的设备驱动程序框架没有任何相似之处。

除了内核以外，Darwin 还包括一些 UNIX 工具(图 5-6)，这些工具一些由苹果开发，一些来自第三方(例如 FreeBSD Project、GNU Project、Apache 等)。

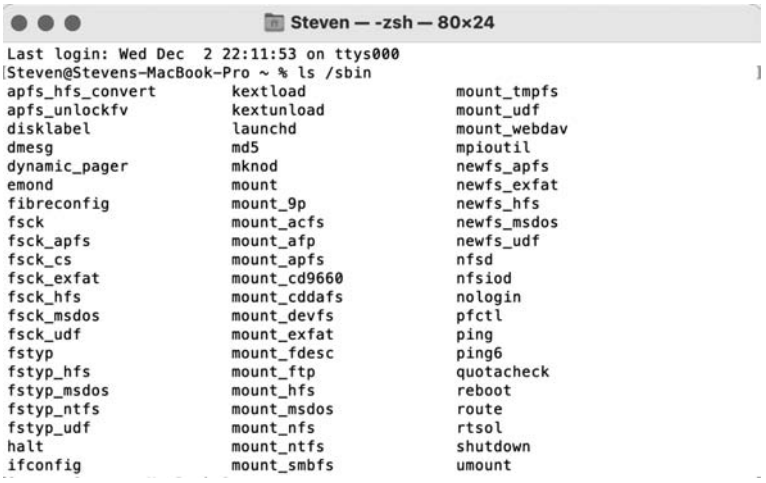


图 5-6 系统管理的必备工具

2. API

与 Windows、Linux 等操作系统一样，macOS 的 API(application programming interface，应用程序接口)为操作系统内核与上层应用程序之间提供了预设的接口和规范。

1) launchd

launchd 由苹果开发，它是一款统一服务管理框架，用于启动、停止和管理 macOS 中的守护进程、应用程序、进程和脚本。由于 launchd 支持多线程，所以它比传统的 UNIX 初始

化程序 SysVinit 要高级，launchd 同时正在被移植到 FreeBSD 平台，它的设计思想也被 systemd 所借鉴，后者成为目前 Linux 发行版中的主流系统初始化程序。

2）Core Foundation

Core Foundation 是 macOS 和 iOS 中的 C 应用程序编程接口，是低级例程和包装函数的混合。

3）Cocoa

Cocoa 是苹果公司为 macOS X 所创建的原生面向对象的 API，是 macOS X 上 5 大 API 之一（其他 4 个是 Carbon、POSIX、X11 和 Java）。Cocoa 是苹果的面向对象开发框架，用来生成 macOS X 的应用程序。主要的开发语言为 Objective-C。Cocoa 开始于 1989 年 9 月上市的 NeXTSTEP 1.0，当时没有 Foundation 框架，只有动态运行库，称为 kit，其中最重要的是 AppKit。1993 年 NeXTSTEP 3.1 被移植到了 Intel、Sparc、HP 的平台上，Foundation 首次被加入，同时 Sun 和 NeXT 合作开发的 OpenStep 也可以运行在 Windows 系统上。

3. GUI

Aqua GUI 是 macOS 的桌面环境，类似 Linux 中的 GNOME。不过，不是所有 macOS X 都是 Aqua GUI，在 macOS X 早期测试版 Rhapsody 中，用的还是经典的 Classic macOS 界面。

5.2.2 时间戳

时间信息是操作系统中非常重要的环节，它贯穿于各类系统数据、用户数据、应用程序及文件系统数据等内容。macOS 中使用的时间戳主要有 Mac 绝对时间、UNIX 时间戳和“HFS+时间戳”3 种类型，如表 5-1 所示。

表 5-1 macOS 中使用的时间戳

名 称	长 度	起始时间	单 位
Mac 绝对时间	32 位	2001 年 1 月 1 日 00:00:00UTC	秒
UNIX 时间戳	32 位	1970 年 1 月 1 日 00:00:00UTC	秒
HFS+时间戳	32 位	1904 年 1 月 1 日 00:00:00UTC	秒

在取证过程中，macOS 时间戳提供了大量有用信息，图 5-7 所示的是在一个.docx 文件上运行 mdls 命令的显示信息，其中重要的时间戳如下。

（1）内容创建时间：kMDItemContentCreationDate。

（2）内容修改时间：kMDItemContentModificationDate。

（3）添加时间：kMDItemDateAdded（这是文件被添加到当前目录的时间）。

（4）内容改变时间：kMDItemFSContentChangeDate。

（5）文件创建时间：kMDItemFSCreationDate。

（6）上次使用时间：kMDItemLastUsedDate。

```
Steven — -zsh — 91×49
Steven@Stevens-MacBook-Pro ~ % mdls /Users/Steven/Desktop/chap5.docx
_kMDItemDisplayNameWithExtensions      = "chap5.docx"
kMDItemAuthors                         = (
    "Microsoft Office User"
)
kMDItemContentCreationDate             = 2020-12-13 16:09:48 +0000
kMDItemContentCreationDate_Ranking     = 2020-12-13 00:00:00 +0000
kMDItemContentModificationDate         = 2020-12-28 10:47:26 +0000
kMDItemContentModificationDate_Ranking = 2020-12-28 00:00:00 +0000
kMDItemContentType                     = "org.openxmlformats.wordprocessingml.document"
kMDItemContentTypeTree                 = (
    "org.openxmlformats.wordprocessingml.document",
    "org.openxmlformats.openxml",
    "public.data",
    "public.item",
    "public.composite-content",
    "public.content"
)
kMDItemDateAdded                       = 2020-12-28 14:37:31 +0000
kMDItemDateAdded_Ranking               = 2020-12-28 00:00:00 +0000
kMDItemDisplayName                     = "chap5.docx"
kMDItemDocumentIdentifier              = 0
kMDItemFSContentChangeDate             = 2020-12-28 10:47:26 +0000
kMDItemFSCreationDate                  = 2020-12-13 16:09:48 +0000
kMDItemFSCreatorCode                   = "MSWD"
kMDItemFSFinderFlags                   = 0
kMDItemFSHasCustomIcon                 = (null)
kMDItemFSInvisible                     = 0
kMDItemFSIsExtensionHidden             = 0
kMDItemFSIsStationery                  = (null)
kMDItemFSLabel                         = 0
kMDItemFSName                          = "chap5.docx"
kMDItemFSNodeCount                     = (null)
kMDItemFSOwnerGroupID                  = 20
kMDItemFSOwnerUserID                   = 501
kMDItemFSSize                          = 11098508
kMDItemFSTypeCode                      = "WXBN"
kMDItemInterestingDate_Ranking         = 2020-12-28 00:00:00 +0000
kMDItemKind                            = "Microsoft Word document (.docx)"
kMDItemLastUsedDate                    = 2020-12-28 14:18:37 +0000
kMDItemLastUsedDate_Ranking            = 2020-12-28 00:00:00 +0000
kMDItemLogicalSize                     = 11098508
kMDItemPhysicalSize                    = 11100160
kMDItemSubject                         = "Label 1, Label 2, Label 3"
kMDItemTitle                           = "Central topic"
kMDItemUseCount                        = 2
kMDItemUsedDates                       = (
    "2020-12-27 16:00:00 +0000"
)
```

图 5-7　docx 文件的时间戳

5.3　文件系统

与 Windows 操作系统一样，在 macOS 中文件系统是对文件实现管理的一种机制和方法。不同类型的操作系统中，对文件系统的描述在概念上基本相似，但概念名称和实现方法有所差异。

5.3.1　卷与分区

卷和分区是操作系统中对磁盘进行管理的基本单位，对卷和分区的理解和正确操作是取证工作的基础。

1. 分区方案

Mac 磁盘分区方案包括三种：GUID 分区图、主引导记录和 Apple 分区图。如图 5-8 所示，在系统的“实用工具→磁盘工具”中抹掉（格式化）磁盘时，即可选择相应方案。

（1）Apple 分区图。用于基于 PowerPC、Open Firmware 的 Mac 设备。

（2）GUID 分区图。用于 Intel、Apple 芯片的 Mac 设备及基于 EFI 的 Windows 设备。

图 5-8　“磁盘工具”中的磁盘分区方案

(3) 主引导记录。即 MBR,用于所有 Windows 设备。

2. 容器与宗卷

macOS 10.14 或更高版本使用 APFS(Apple file system,Apple 文件系统)时,允许在磁盘上的宗卷间共享空间。通常情况下,大部分 Mac 设备自身包含一块物理磁盘,可以将其划分成多个单独的部分,每个部分称为一个“容器”(container)。如图 5-9 所示,对照磁盘工具,可以看到物理磁盘之下,首先是一个“APFS 容器”,其次才是 APFS 逻辑卷(即“宗卷”),这些宗卷是放在 APFS 容器里面的。

图 5-9　磁盘工具中查看磁盘层级

但是在大多数情况下,使用 APFS 时不应将磁盘分区,而应在单个容器内创建多个宗卷(Volume),宗卷没有固定大小,容器内所有宗卷共享可用空间(空间共享会影响克隆和稀疏文件)。举个例子,如果在一个 1000GB 的 APFS 格式磁盘(单个容器)上,有一个 A 分区和一个 B 分区。目前 A 和 B 分区分别占用 300G 和 400G 空间,但两者都可以任意使用整个容器剩余的 300GB 空间。A 和 B 分区可以不设固定分区大小,这样在 A 或 B 分区中,即使再存储一个小于 300GB 的文件都没问题。APFS 中的磁盘(disk)、容器(container)、宗卷

(volume)和命名空间(namespace)之间的关系如图 5-10 所示。

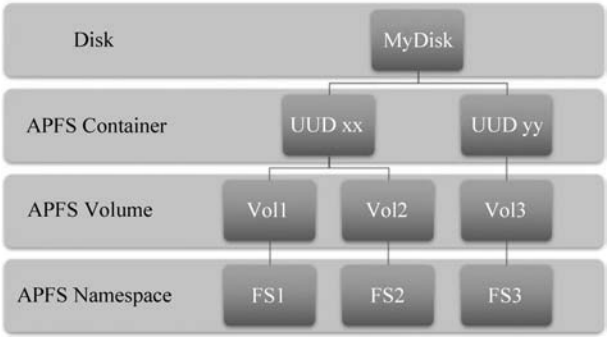


图 5-10 APFS 中的磁盘、容器、宗卷、命名空间关系图

APFS 共享空间时,甚至可以在 APFS 宗卷上安装其他版本的 macOS。当然,如果要安装 Windows 系统,还是可以考虑将内置物理磁盘进行分区。

5.3.2 根目录

macOS 系统卷的根目录通常只有可见的 4 个目录:系统、应用程序、用户和资源库,其他目录均处于隐藏状态,需要通过"Command+Shift+."组合键来显示。随着 APFS 文件系统的普及,根目录较 HFS+时代也发生了变化,主要目录及功能描述如表 5-2 所示。

表 5-2 macOS 主要目录及功能描述

名 称	功 能 描 述
系统(system)	通常包含仅属于操作系统的数据
应用程序(applications)	所有程序都默认安装在该目录下,可以直接启动
用户(users)	包含所有用户目录,其下的共享目录可用于不同用户间的数据共享
资源库(library)	包含应用程序相关数据,这里安装的软件会应用于整个系统、所有用户
隐私(private)	隐藏状态,包含系统级的日志、打印记录、密码信息、睡眠交换文件等信息
卷(volumes)	隐藏状态,连接的存储介质的加载点,加载点是动态的,断开后自动消失,与 Linux 下的"/mnt"或"/media"目录相似
网络(network)	隐藏状态,域控制、活动目录等
DocumentRevisions-V100	隐藏状态,保存之前文件的位置信息,APFS 中无此目录
.fseventsd	隐藏状态,包含 HFS+文件系统日志,APFS 中无此目录
.Spotlight-V100	隐藏状态,保存 Spotlight 索引信息,APFS 中无此目录
.DS_Store	隐藏状态,保存目录的设置和视图信息,APFS 中无此目录
.Trashes	隐藏状态,启动卷以外的卷中删除的数据,APFS 卷中无此目录

5.3.3 文件系统的发展

Mac 常用的文件系统有 HFS、HFS+和 APFS,它们之间的主要区别如表 5-3 所示。

表 5-3 HFS、HFS+和 APFS 之间的主要区别

名 称	HFS	HFS+	APFS
推出时间	1984	1998	2017
位数	16	32	64
最大文件大小	2GB	8EB	8EB
最大硬盘容量	2TB	8EB	8EB
文件保险箱	无	有	有
时间机器	无	有	有
写时复制	无	无	有
磁盘快照	无	无	有
共享空间	无	无	有
多密钥加密	无	无	有

1. HFS 与 HFS+

1) HFS

macOS 标准格式(HFS)是 Mac 设备在较长时间里使用的专有文件系统,也是苹果 Open Firmware 所支持的最基本的文件系统,常见于早期的 Mac 计算机的硬盘、软盘和光盘等。HFS 不但保留了之前文件所设定的基本概念,更带来了极大的功能增强,如应用了全新的目录文件、支持子目录结构、支持最初的权限概念、支持更多的 Mac 文件属性等。HFS 一直使用至今,但在 macOS X 10.5 之前,HFS 可以挂载为读写,而之后只可以挂载为只读。

2) HFS+

macOS 扩展格式(HFS+或 HFS Plus)是在 HFS 的基础上进行改进的文件系统格式。在 1998 年 macOS 8.1 中,苹果首次带来了对 HFS+的支持。HFS+放弃 HFS 中使用的 MacRoman,从而执行 Unicode 作为文件命名的编码标准; HFS+通过增加可分配块数目,降低了单一文件的最大大小值。HFS+推出的主要目标是优化大容量硬盘的存储能力,同时 HFS+扩大了文件数量大小,将单个文件的大小和硬盘容量扩展到 8EB。

在 macOS X 10.2 中,HFS+文件系统增加了日志(journaling)功能。日志功能就是在写文件时,先在日志中做个记录,记录完后才真正写入文件中。日志也是磁盘中的一个二进制位置,它是专门开辟出用于读写日志的一个空间区域,该区域不用来存储文件数据。HFS+增加了文件系统的安全性,因为不带日志功能的 HFS 文件系统缺乏安全性,在文件系统不完整时(比如丢失索引),丢失的数据很难恢复。

HFS+支持的分区格式主要包括以下几类:

(1) macOS 扩展(日志式)。使用 Mac 格式(日志式 HFS+)来保护分层文件系统的完整性。如果不需要加密或区分大小写格式,则选取此项。

(2) macOS 扩展(日志式,加密)。使用 Mac 格式要求加密,并加密分区。

(3) macOS 扩展(区分大小写,日志式)。使用 Mac 格式,并区分文件夹名称的大小写。例如,名称为 ABC 和 abc 的两个文件夹不同。

(4) macOS 扩展(区分大小写,日志式,加密)。使用 Mac 格式,区分文件名称的大小写,要求加密,并加密分区。

在 macOS X 10.13 中，HFS+被 AFPS 取代。

2. APFS

Apple 文件系统(APFS)是运行 macOS 10.13 或后续版本的 Mac 计算机所使用的默认文件系统，它具有改进的文件系统基础、强加密、空间共享、磁盘快照、快速目录大小统计等特性。macOS 10.13 及以上版本支持将 APFS 用于启动盘和数据盘。APFS 具有以下特点：

1) 按照需求分配容器中的磁盘空间

单个 APFS 容器包含多个宗卷时，容器的可用空间会共享，并且会自动按需分配到任意单独的宗卷。用户可以指定每个宗卷的保留大小和配额大小。每个宗卷仅使用整体容器的一部分，这样一来，可用空间即为容器的总大小减去该容器中所有宗卷的大小。

2) 支持写时复制

在 APFS 中，同一个文件，无论复制多少份，假如不修改，这个文件实际只会在磁盘中存储一份，只是其索引有多份而已。换言之，当一个对象更新时，将创建并引用一个新对象，旧的对象仍然可以被扫描。对于电子数据取证而言，写时复制技术提供了数据恢复的可能性，可以直接找到某个文件的历史版本。

3) 支持磁盘快照

快照意味着来自时间节点的对象将被“锁定”，并创造了在区块级重建整个历史卷状态的可能。比起苹果自己的 Time Machine(时间机器)，快照更接近于微软的 NTFS 卷影复制技术。对于电子数据取证而言，快照技术与写时复制技术起到的作用基本一致，提供了数据恢复的可能。

macOS 可以通过“磁盘工具”添加或删除 APFS 容器中的宗卷分区，APFS 容器中的每个宗卷都可以拥有其 APFS 格式，主要包括 APFS、APFS(加密)、APFS(区分大小写)或 APFS(区分大小写，加密)。

(1) APFS。使用 APFS 格式。如果不需要加密或区分大小写格式，则选取此项。

(2) APFS(加密)。使用 APFS 格式且加密宗卷。

(3) APFS(区分大小写)。使用 APFS 格式并区分文件和文件夹名称的大小写。例如，名称为“ABC”和“abc”的两个文件夹不同。

(4) APFS(区分大小写，加密)。使用 APFS 格式，区分文件和文件夹名称的大小写且加密宗卷。

3. 兼容性

macOS 对不同文件系统格式的支持情况如表 5-4 所示。

表 5-4 macOS 对不同文件系统格式的支持情况

文件系统	支持情况	说明
MFS	部分支持	macOS 8 起取消对 MFS 的支持
UFS	部分支持	macOS X 10.7 起取消对 UFS 的支持
HFS	支持	macOS X 10.5 起仅支持只读
HFS+	读写	
APFS	读写	

续表

文件系统	支持情况	说明
FAT16/32	读写	macOS中称为MS-DOS
ExFAT	读写	
NTFS	读	可开启系统自带支持或安装第三方应用支持
EXT/2/3/4	不支持	安装第三方应用支持
BtrFS	不支持	安装第三方应用支持
XFS	不支持	安装第三方应用支持
ZFS	读	

NTFS是Windows和大容量移动存储设备经常使用的文件系统格式，那么如何打开macOS原生自带的NTFS读写功能，让用户能够在Mac计算机上方便地读写NTFS格式的设备呢？可以通过修改/etc目录中的.fstab文件来实现：复制hosts文件到桌面，将其重命名为.fstab，然后写入如下内容：

```
LABEL = My\040Disk none ntfs rw,auto,nobrowse
LABEL = HD001 none ntfs rw,auto,nobrowse
LABEL = 移动硬盘 002 none ntfs rw,auto,nobrowse
```

注意：My\040Disk、HD001、移动硬盘002分别代表几个外接硬盘的名称，其中\040代替空格。

5.3.4 文件保险箱

文件保险箱(FileVault)加密程序最初在macOS X 10.3中引入，它只允许加密用户主目录。

1. FileVault与FileVault 2

从macOS X 10.7开始，苹果公司重新设计并加强了加密方案，并作为FileVault 2发布。FileVault 2使用全磁盘、128位AES加密(XTS-AESW模式)来保证数据安全，该加密软件自动实时加密和解密，对具备权限的用户来说是透明的。但对于取证人员而言，即使磁盘被移除并连接到其他计算机，或者不拆机直接获取到磁盘镜像，内部的APFS宗卷也会保持加密状态，在未解密的情况下，也无法查看其中的信息。

此外，可根据开机启动界面判断其是否开启了文件保险箱功能。开启了文件保险箱功能的目标计算机在其待机界面中有睡眠标签，且能显示其他账户。

2. 加密与解密

1) 加密

文件保险箱功能通过“系统偏好设置”中的“安全性与隐私”进行管理。通过选取“安全性与隐私”中的“文件保险箱”标签(macOS High Sierra 10.13.2之前系统默认使用英文名称FileVault)来启用或停用文件保险箱，如图5-11所示。

macOS X 10.9(Mavericks)及以前版本中，可以选择通过存储文件保险箱恢复密钥的方法(需提供三个安全提示问题及答案)来便于以后解锁，如图5-12所示。

图 5-11 “安全性与隐私”中的“文件保险箱(FileVault)”功能

图 5-12 macOS X 10.9 及以前版本的文件保险箱恢复密钥设置

在 macOS X 10.10(Yosemite)及以后版本中，可以通过 iCloud 账户来解锁磁盘并重设密码。在加密时，如果不使用 iCloud 账户来解锁文件保险箱，也可以创建一个本地恢复密钥来解锁，类似 Windows 的 BitLocker，密钥保存在加密启动磁盘以外的其他安全位置，如图 5-13 所示。

图 5-13 macOS X 10.10 及以后版本的文件保险箱解锁设置

在搭载 Apple T2 芯片的 Mac 计算机上，所有文件保险箱密钥的处理都发生在安全隔区中；加密密钥绝不会直接透露给 CPU。所有 APFS 宗卷默认使用宗卷密钥创建。宗卷和元数据内容使用此宗卷密钥加密，此宗卷密钥使用类密钥封装。文件保险箱启用时，类密钥受用户登录密码和硬件 UID 共同保护，此保护在搭载 T2 芯片的 Mac 计算机上为默认设置。移动存储介质的加密不使用 T2 芯片的安全性功能，而是采用与未搭载 T2 芯片的 Mac 计算机相同的方式执行加密。

在搭载 T2 芯片的 Mac 上，如果在“设置助理”初次运行过程中未启用文件保险箱，宗卷仍然会加密，但这时宗卷密钥仅由安全隔区中的硬件 UID 保护。如果随后启用了文件保险箱（由于数据已加密，该过程可立即完成），反重放机制会阻止旧密钥（仅基于硬件 UID）被用于解密宗卷。然后类密钥将受用户登录密码和硬件 UID 共同保护。

macOS 10.13 及以上版本中的 APFS 改变了“文件保险箱”加密密钥的生成方式。在 CoreStorage 宗卷上的 macOS 旧版本中，文件保险箱加密过程中使用的密钥是在 Mac 上启用文件保险箱时创建的。在 APFS 宗卷上的 macOS 中，该类密钥在用户创建过程中或 Mac 用户首次登录过程中创建。加密密钥的实施方式、生成时间和存储方式都是 SecureToken 的一部分。具体来说，SecureToken 是密钥加密密钥（KEK）的封装版本，由用户密码保护。

macOS 10.15 还引入了一项新功能 Bootstrap Token，以帮助将 SecureToken 授予移动账户和设备注册时创建的管理员账户。

2）解密

通常情况下，可以根据启用文件保险箱时的设置进行解锁。目前，macOS 支持 3 种解锁方式：用户登录口令、iCloud 账户密码和恢复密钥。

当目标计算机处于开机状态时，可以根据实际情况在上述 3 种方式中选择最合适的进行解密（即“停用文件保险箱”），然后进行数据固定，如图 5-14 所示；当目标计算机处于关

图 5-14　macOS 10.10 及以后版本的文件保险箱解密提示

机状态时，可以先获取磁盘镜像，再通过专业的取证分析工具加载后进行解密，如图 5-15 所示。例如，通过盘古石计算机取证分析系统(SafeAnalyzer)可以实现相关操作。

图 5-15　使用 SafeAnalyzer 加载 Mac 镜像后解密文件保险箱

5.3.5　关键文件格式

macOS 中最常见的系统及应用数据文件格式是 SQLite、Plist 和 iOS 类似。

1. SQLite 文件

SQLite 是一种轻型的基于 SQL 的关系数据库，macOS 中的 SQLite 格式特征为 SQLite format 3，通常情况下大家关注的第三方应用程序的核心记录一般存储在 SQLite 数据库中，如 QQ、微信通讯录、聊天记录等。

需要说明的是，虽然 QQ、微信通讯录等数据使用 SQLite 数据库，但这些 SQLite 存储文件的扩展名并不统一，常见的有. db、. sqlite3、. sqlite 等。通常，每个 SQLite 文件中会保护多个数据表(table)，每个表中又会包含多个字段。

与众多个人计算机和智能手机应用一样，Mac 应用出于安全性保护，对核心数据库也进行了加密，不同开发商的加密方式也各不相同，未加密或解密后的 SQLite 文件可以使用 SQLite 查看器直接打开，macOS 下常用的 SQLite 查看器有 DB Browser for SQLite(图 5-16)、SQLite manager(火狐浏览器扩展)、sqlite3(命令行工具)等，Windows 下还有 SQLite Browser、SQLite Expert 等工具。

2. Plist 文件

属性表文件(property list files，Plist 文件)是 macOS 常用的一种文件形式，通常用于存储用户设置，也可以用于存储部分应用数据。最早推出 Plist 的 NeXTSTEP 只使用一种文件格式，macOS 10.0 中采用了一种新的 XML 格式，XML 格式支持非 ASCII 格式，也可存储 NSValue 对象。由于 XML 文件在存储时的空间效率并不算高，macOS 10.2 引入了 Plist 文件格式用于存储二进制文件。从 macOS 10.4 开始，Plist 文件成为系统的默认格式。

macOS 10.2 中引入了 plutil 工具，可以用来检查 Plist 的语法，或者对其进行格式转换。对于 XML 格式的 Plist 文件，可以使用任意文本编辑器进行编辑。同时，苹果公司提

	INT	Z_OPT	ZORD	ZTYPE	ZCREATETIME	ZMODIFYTIME	ISPLAYDESCRIPTI	ZIDENTIFIER	ZNAME
		过滤	过滤	过滤	过滤	过滤	过滤	过滤	过滤
1		1	10	1	616244323.486...	614686058	您好，产品是否满...	cntaobao尊贵汇...	尊贵汇家居专营店
2		1	22	1	616244323.489...	609753372		cntaobao小鞋子1...	小鞋子1006
3		1	19	1	616244323.488...	610418835	求求您了，宝宝公...	cntaobao古岗古...	古岗古旗舰店
4		1	7	1	616244323.486...	615282733	不客气哈[表情]	cntaobao雕塑翡翠	雕塑翡翠
5		1	5	1	616244323.485...	615714887	亲亲 小妹求求您...	cntaobaoaspor...	aspor旗舰店
6		1	23	1	616244323.489...	609497863	亲，您看下安装视...	cntaobao萨格慕...	萨格慕旗舰店
7		1	6	1	616244323.485...	615612747	[redacted][表情]亲...	cntaobao琴道旗...	琴道旗舰店
8		1	0	1	616244323.483...	616244289	嗯好的	cntaobao韩科旗...	韩科旗舰店
9		1	8	1	616244323.486...	615103710	[图片]	cntaobao和风雨...	和风雨旗舰店
10		1	11	1	616244323.486...	614659967		cntaobao无比圆...	无比圆睿必特专卖...
11		1	9	1	616244323.486...	614926839	不客气	cntaobao慕然回...	慕然回首的你701...
12		1	25	1	616244323.489...	609069612	亲亲，您可以发送...	cntaobaohazzys...	hazzys官方旗舰店
13		1	26	1	616244323.489...	608536949	您在本店购买的商...	cntaobao芳夕旗...	芳夕旗舰店
14		1	27	1	616244323.490...	608526481	这款是老茶树的品...	cntaobao狮峰茶...	狮峰茶叶官方旗舰...
15		1	24	1	616244323.489...	609312353	穿不了	cntaobao迪启蜜爱	迪启蜜爱

图 5-16 使用 DB Browser for SQLite 查看 SQLite 文件

供了 Property List Editor 应用程序(作为 Apple developer tools 的一部分),它是一个树状的查看器与编辑器,并可以处理二进制格式的 Plist 文件。

目前,最常用的第三方 Plist 查看及编辑工具是 PlistEdit Pro(操作界面如图 5-17 所示),它有 Mac 和 Windows 两个版本。PlistEdit Pro 支持 macOS 中 XML 和二进制两种格式的 Plist 文件,同时还支持 macOS 外的 JSON 和 ASCII 格式。

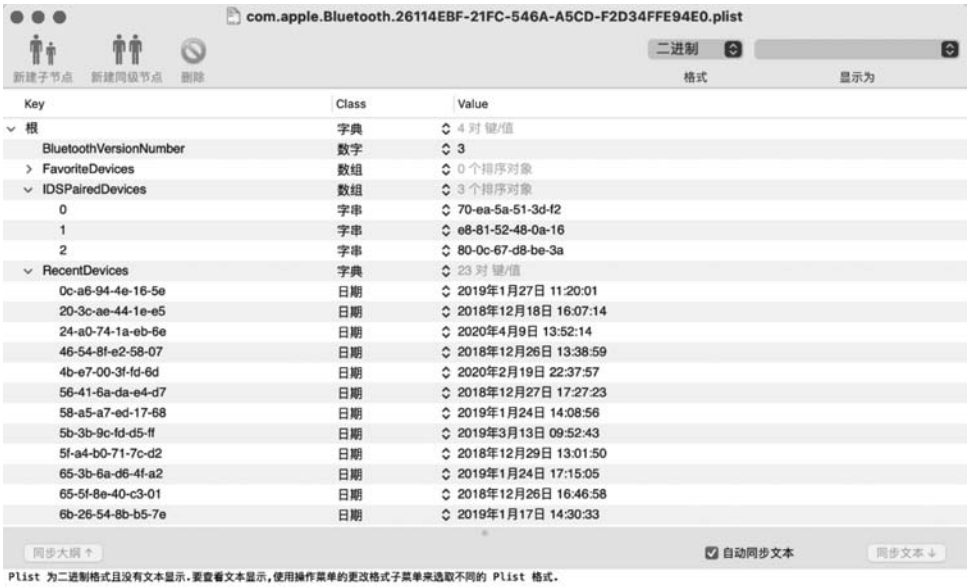


图 5-17 使用 PlistEdit Pro 查看 Plist 文件

5.4 数据的提取与固定

数据的提取与固定是电子数据取证工作的关键环节。此项工作的重点是针对具体的取证对象，将注意力集中在数据这个主要目标上，采取必要的技术手段对可能蕴藏着案件信息的数据进行提取和固定。

5.4.1 准备工作

不同于 Windows 计算机的取证，Mac 硬件和 macOS 显得复杂而多变，因此针对 Mac 计算机的取证准备工作显得尤为重要。取证人员需要考虑到可能涉及的各种设备及其存储介质，以便选择合适的取证方法和工具。还要能够快速确定不同型号的 Mac 计算机（MacBook 系列、iMac、Mac mini 或者 Mac Pro 等）的具体版本和年份，同时判断其中的操作系统版本及其功能特性可能对取证工作的影响。

1. 关键因素

为了更好地开始 Mac 取证工作，取证人员需要厘清以下关键因素：

1）开关机状态

现场计算机开关机状态的不同影响着取证方法和工具的选择，不同情况下可获取的数据范围和有价值数据的数据量也都不同。针对不同状态，可以采取“在线取证”或“离线取证”中的一种方式。

2）硬盘情况

针对关机状态的计算机，通常建议采用拆机读硬盘的方式。但不同年份的 Mac 计算机配备了不同类型的存储介质（硬盘），早期的设备还允许用户自由更换硬盘，实际情况也许是机械盘，也许是固态盘；硬盘也许能拆卸，也许固化在主板上不能拆卸。拆卸后是否具备硬盘接口的只读连接条件，不能拆卸或没有条件读取时又该如何进行取证，这些问题都需要进行充分考虑，尤其在“离线取证”环节中应加以注意。

3）文件保险箱

随着 Mac 设备的广泛应用以及设备自身的更新，越来越多的涉案 Mac 设备启用了文件保险箱（File Vault）功能，取证人员需对其有清晰的认知，关于文件保险箱特性及其加密、解密操作已在 6.3.4 节中进行了介绍。

4）登录口令

用户登录口令是开机状态取证的“拦路虎”，同时也与文件保险箱及取证应用程序或命令操作的权限等因素都息息相关，应尽可能在询问、讯问或其他线索证据中获得明文。关于系统口令的特性及解密方法，在“在线取证”过程中都会关注到。

5）磁盘挂载

当针对开机状态的目标 Mac 计算机取证时，取证人员需要确认好连接到目标计算机上的磁盘的文件系统，选择读写均能够兼容的文件系统类型。

当使用取证专用的 Mac 计算机读取目标计算机磁盘时，需要在自动挂载前启动专用的只读软件以确保数据的原始性不被破坏。

6）启动转换助理

启动转换助理(boot camp)是苹果公司于2006年推出的基于Intel处理器的Mac计算机运行Windows操作系统的软件。Boot Camp功能已经整合到macOS 10.5及以后版本中。

当关机状态的Mac计算机进入引导磁盘选择界面时，取证人员需留意通过Boot Camp安装的Windows分区，以便选择合适的方式单独提取和分析。

当针对开机状态的Mac计算机取证时，取证人员需留意磁盘的分区情况及系统偏好设置中“启动磁盘”功能中的选项，及时发现可能存在的其他操作系统，以便选择合适的方式单独提取和分析。

2. 环境处置

在Mac计算机处于开机状态时，要特别注意计算机上是否正在运行数据擦除软件，防止数据丢失。当发现有运行数据擦除软件时要及时停止软件的运行，或根据实际情况选择直接硬关机。Mac计算机与iPhone类似，可以在iCloud设置中开启“查找我的Mac”功能(图5-18)，当计算机处于联网状态时，需要判断计算机是否有被远程擦除或锁定的风险，如果有风险，应当考虑断开网络链接，断开网络连接时要注意有线网络和无线网络两种情形。

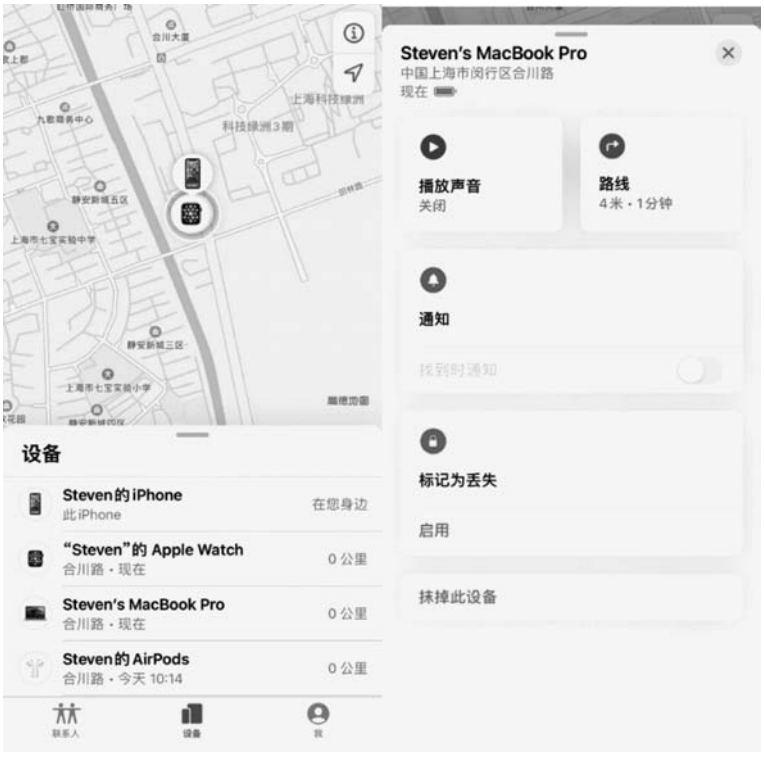


图5-18　开启“查找我的Mac”功能后即可实现对其远程控制

3. 流程规范

由于Mac设备及macOS的特殊性，针对Mac计算机取证时，需遵循以下规范：

(1) 在开始取证前，需明确具体实施操作的技术人员，无关人员需要远离物证。

(2) 采取稳定的方式屏蔽网络信号,以防远程锁定和数据擦除。

(3) 开机不关,及时获取易失性数据和已解密的数据,结束后硬关机。

(4) 关机不开,选择合适方法拆机取证或免拆机取证。

5.4.2 在线取证

在线取证即通常所说的针对处于开机状态的目标计算机的取证操作。当目标计算机处于未锁屏(或锁屏可解开)状态时,在进入桌面后,根据案件需求先进行易失性数据提取,主要包括网络数据和运行状态下物理内存中的数据等。针对苹果计算机的内存数据提取,可以使用专用的内存镜像工具进行固定,在提取过程中需要注意按照取证规范进行操作并使用合理的方式进行记录。

1. 在线取证的注意事项

macOS 10.12 及以上版本的系统需要关闭系统完整性保护(SIP)才能获取内存和磁盘镜像,这是因为当开启 SIP 时,系统会对内部的数据进行保护,镜像工具将无法获取全盘镜像数据。查看 SIP 是否开启可以在终端中输入“csrutil status”命令,就可以看到是“enabled”(已开启)还是“disabled”(未开启)。关闭系统保护的操作需要先重启系统,按住 Command+R 组合键进入恢复模式,打开终端运行“csrutil disable”命令即可。

针对较新 Mac 计算机的内存提取还需要管理员权限,所以需要先行获取该计算机的管理员账户密码。

需要注意的是,针对提取到的内存数据,其后期解析工作其实是复杂的。目前已有的工具并不能支持所有版本的 Mac 内存数据分析,而且能支持的版本也不能将内存中所有的数据都完整地解析出来。那么在提取易失性数据时就不能仅仅提取内存镜像,而是要充分利用系统环境中可信的取证程序进行易失性数据的提取固定,如使用命令行进行网络连接情况的提取等。

目标计算机处于开机状态时还要根据实际情况及时进行网络信号的屏蔽和网络连接的断开,以防止苹果计算机被远程擦除或锁定。在保证数据安全的前提下执行完易失性数据提取后,一般再进行非易失性数据的固定。在执行非易失性数据的固定任务时,通常采取硬关机操作,在关机后直接对存储盘进行离线镜像。但是,需要注意的是,在硬关机前应先检查磁盘的加密情况,如果启用了文件保险箱加密,那么就需要在开机状态下将解密的数据进行及时固定提取,针对磁盘无法拆卸(或不具备接口只读连接条件)和搭载 T2 芯片加密的 Mac 计算机,应当在线提取或使用专用的 USB 启动系统进行不拆机镜像。

需要注意的是,macOS 默认不支持写入到 NTFS 文件系统的存储介质,因此用于存储数据的外接设备的文件系统通常选择 exFAT。

当目标机处于锁屏状态时,需要进行屏幕解锁,然后按照未锁屏状态的情况进行提取。现场确实无法解锁的,可以先行扣押,后续根据实际案件情况及当时技术情况进行处理。

2. 常用的在线取证工具

常用的 Mac 在线取证工具有奇安信集团的盘古石现场取证系统(SafeImager)、美国 BlackBag 公司(已被 Cellebrite 公司收购)的 MacQuisition 和美国 SUMURI 公司的

RECON ITR 等，操作界面分别如图 5-19、图 5-20 和图 5-21 所示。

图 5-19 盘古石现场取证系统(SafeImager)

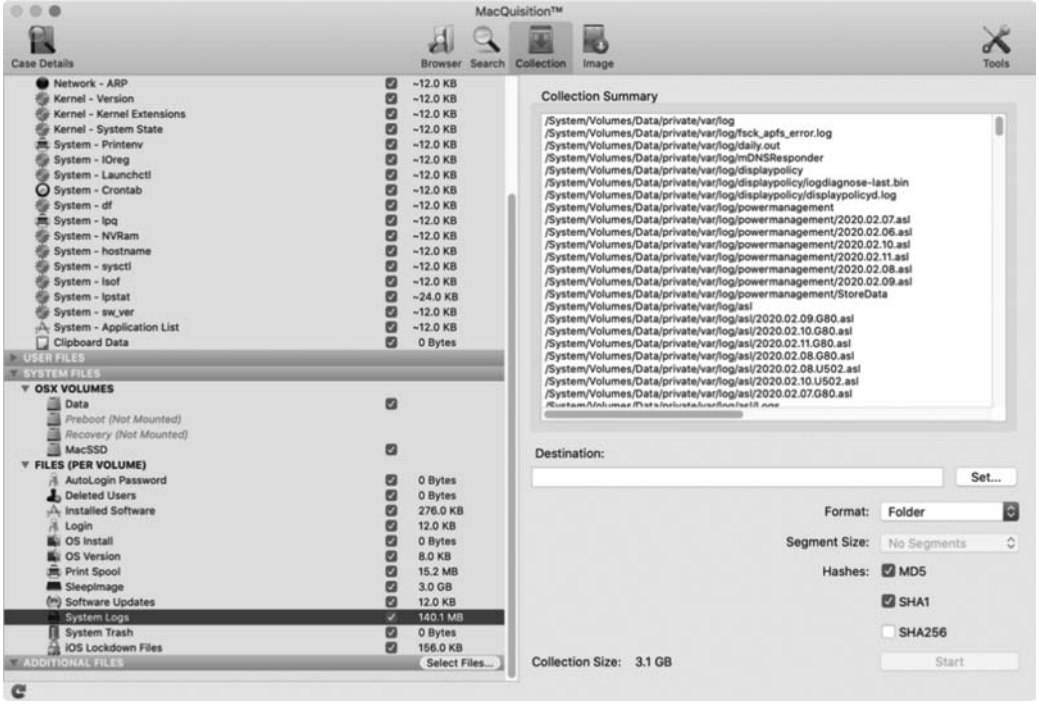


图 5-20 BlackBag MacQuisition

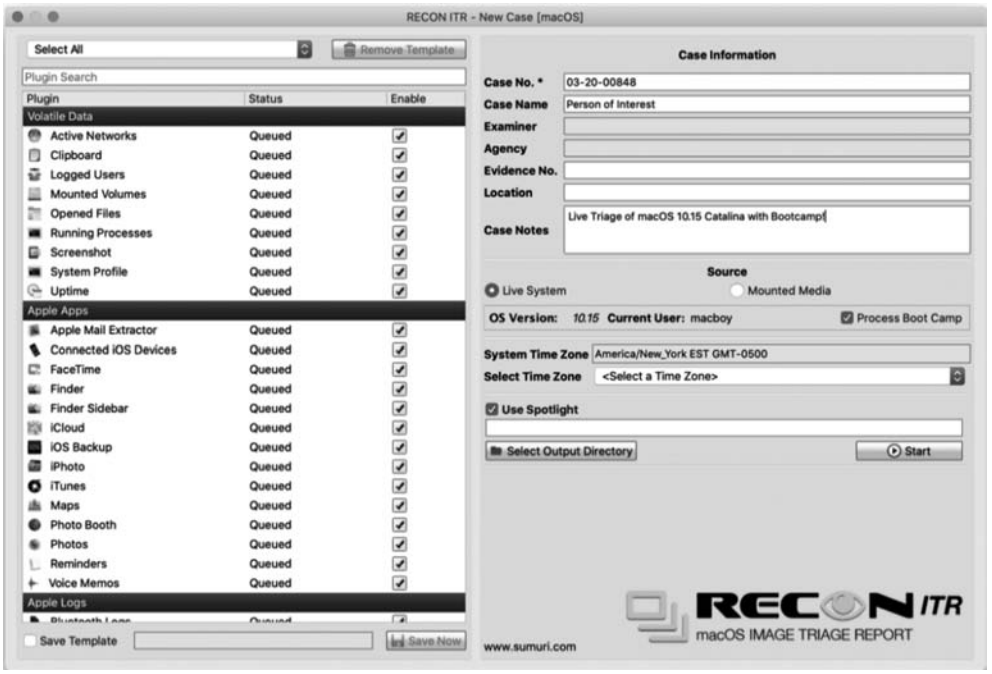


图 5-21 SUMMURI RECON ITR

3. 登录口令

登录口令也称账户密码或用户密码，是用户登录对应账户时的必需品。在传统 UNIX 系统中，密码和账户信息是分开存储的，例如“/etc/passwd”文件存储了账户名、密码是否加密的标记、UID、默认群组、主目录、默认登录 shell 以及注释信息，而“/etc/shadow”文件存储了 UID、密码密文、最近一次修改的天数、密码有效时限等信息。

1）macOS 用户登录口令的存放位置

在 macOS 上，与 UNIX 略有不同，且随系统版本的不同而有所变化，在 macOS 10.4～10.6 版本中账户密码存储在/private/var/db/shadow/hash 目录下以每个账户的 UUID 为名的文件中；在 macOS 10.7 及其以后的版本中，用户口令的散列值保存在/var/db/dslocal/nodes/Default/users/文件夹下的 plist 文件中。

在加密算法上，macOS 主要使用 SHA-1 或 SHA-512 进行加密，并加上长度为 4 字节的 salt(“盐值”)，最终生成密文。macOS 甚至用二进制代码来记录该账户使用密码的历史情况，例如，该账户曾经被尝试登录时密码输入不匹配的次数、上次登录时间、上次更改密码日期等信息。如图 5-22 所示，通常密文中的第 28～32 字节为“盐”，第 32～96 字节为 SHA-512 哈希值。显然，如果得到了登录口令的密文，配合已知的算法，理论上都可以进行破解。

2）解锁 Mac 计算机锁屏密码的方法

由于文件保险箱和 Apple T2 芯片等诸多限制，macOS 锁屏密码的密文很难获取，因此解锁 Mac 的锁屏密码一直是取证领域的难题，目前可行的解决方案主要有 4 类。

(1) 社会工程学方法，即通过其他周边线索判断或推测可能的密码。

(2) 绕过密码认证机制。

图 5-22　账户 Plist 文件中的密文

(3) 指纹解锁(2016 年以后的部分机型)。

(4) Apple Watch 解锁。

其中,绕过密码认证有两种方式:修改内存绕过密码认证、读取内存获取密码明文。

3) 修改内存绕过密码认证

修改内存绕过密码认证根据系统是否开启文件保险箱(FileVault)和虚拟化技术(VT-d)功能的情况采取不同的方式。无论采取什么方式,其原理都是以 DMA 方式修改 macOS 的物理内存,绕过密码认证机制。

2012 年下半年后生产的 Mac 计算机,默认都会开启 VT-d 功能。在 macOS 10.10 以后的版本中,用户进行系统初始配置时,默认都会开启 FileVault 功能。

(1) 未开启 FileVault 和 VT-d 的 Mac 计算机密码绕过方法。在未开启 FileVault 和 VT-d 的 Mac 计算机中,可通过基于火线(1394)或雷电(ThunderBolt)接口的直接内存访问(DMA)方式读取其物理内存,绕过屏保待机状态密码认证机制的方法如下。

步骤 1:使用内存分析技术,从内存里将 loginwindow 的可执行文件转储出来,记录转储文件中每个物理页的虚拟地址与物理地址。

步骤 2:对 loginwindow 的可执行文件样本进行反汇编,寻找字符串修改位置。

步骤 3:找到实现密码认证功能的函数 methImpl_LWBuiltInScreenLockAuthLion verifyPassword,将最终结果返回位置处的返回值设置为 1,即可实现绕过密码认证直接进入系统。

步骤 4:修改物理内存,绕过密码认证。

以上实现过程中的具体操作为:首先,使用特征字符串搜索的方法,从 loginwindow 的可执行文件样本中搜索出函数 methImpl_LWBuiltInScreenLockAuthLion_verifyPassword 对应返回值的特征字符串;接着,将记录的物理页的物理地址与特征字符串在物理页的偏移相加,获得特征字符串在内存中的位置,将返回值由原来的 0 修改为 1。

(2) 开启 VT-d 未开启 FileVault 的 Mac 计算机密码绕过方法。对于开启了 VT-d 功能,但没有开启 FileVault 功能的 Mac 计算机,处于关机状态时,可首先关闭 VT-d 功能,然后再进行以下的操作。

步骤 1:开机按住 Command+R 组合键,进入恢复模式。

步骤 2:打开终端,输入命令 nvram boot-args="dart=0"。

步骤 3:重启计算机,使用"(1)未开启 FileVault 和 VT-d 的 Mac 计算机密码绕过方法"中描述的步骤破解密码。

(3) 开启 VT-d 和 FileVault 的 Mac 计算机密码绕过方法。在开启 VT-d 和 FileVault

功能的情况下，为了绕过密码认证功能，需要两次接触目标计算机，前两步操作与“(2)开启VT-d未开启FileVault的Mac计算机密码绕过方法”操作相同，都是为了关闭VT-d功能。具体步骤如下。

步骤1：开机按住Command＋R组合键，进入恢复模式。

步骤2：打开终端，输入命令nvram boot-args＝“dart＝0”，并离开。

步骤3：等待目标计算机处于屏保状态(用户已经输入过登录密码)时，再执行“(1)未开启FileVault和VT-d的Mac计算机密码绕过方法”中的密码绕过步骤。

综上所述，对于开启VT-d功能的Mac计算机，首先关闭VT-d，然后根据其是否开启FileVault判断下一步工作：如果没有开启FileVault功能，开机后就能绕过密码(这种方式需要先关机再重启，破坏了检材原始性，严格意义上讲并不符合取证的合法性要求)；如果开启了FileVault功能，等待目标计算机用户登录后，才能绕过密码(显然，该情形对于取证意义不大)。

4) 读取内存获取密码明文

在macOS 10.12.2之前版本的系统中，用户输入的密码会在内存中以明文的Unicode形式存在，而且在每次重新热启动且没有载入操作系统时，这些内存不会被刷新，密码仍存在于内存中的某个地方。当系统热启动到FileVault开机登录界面时(此时没有启动操作系统)，通过DMA方式读取内存，可从内存中搜索到密码明文。

如前所述，支持VT-d的计算机对DMA方式访问内存做了一定的限制。此时，又可以在Mac计算机开机但操作系统没有被加载的情况下以DMA方式获取内存，这是因为，只有在操作系统启动后，那些系统保护区域的内存和DMA才会被系统接管。下面通过“PCILeech”软件(GitHub上的开源项目)实现从内存中读取密码明文。破解设备及连接方式如图5-23所示，操作步骤如下：

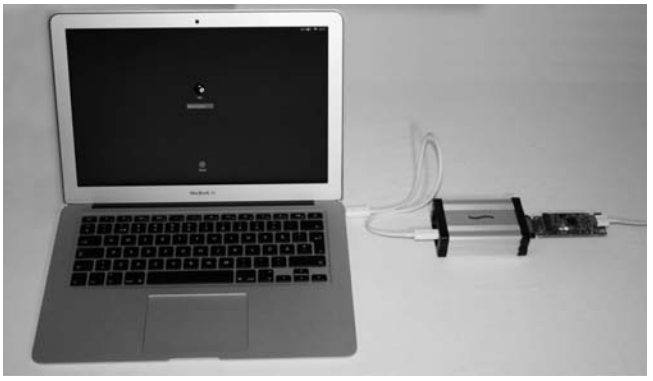


图5-23　破解设备及连接方式

(1) 将设备的一端连接待破解计算机，另一端连接Windows计算机(取证工作站)。

(2) 在Windows计算机上运行相关程序，程序提示重启Mac计算机系统时，同时按Ctrl＋Command＋Power组合键重启Mac计算机。

(3) 稍等片刻，程序会提示待破解计算机上的登录密码。

(4) 在FileVault2的开机登录界面中输入此密码，即可进入系统。

该方式利用了macOS 10.12.2以前的漏洞，但是该漏洞在macOS 10.12.2及其以后

的版本中被修复了(见图 5-24 中 WARNING 后的提示信息)。

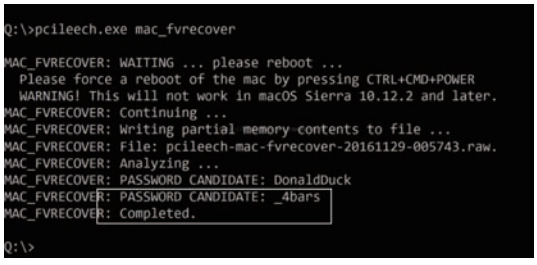


图 5-24 执行 mac_fvrecover 命令后提示重启并获得可能密码

当然,业内还有一种通过修改内存的方式向 Mac 计算机系统植入可执行文件的方法。即使在未知目标计算机系统登录密码的情况下,该技术也可以方便地操作目标计算机系统,突破目标计算机的安全限制,被植入的可执行文件具备管理员权限,可以执行任意功能。考虑到该方法对原始检材的破坏性,本书中不做详述。

另外,直接读写物理内存的设备需要目标计算机具有 1394、PCI Express 以及雷电等接口,但是,近年来生产的计算机几乎都不再配有 1394 和 PCI Express 接口;在新款 MacBook 系列计算机中,雷电接口也逐渐被 USB Type C 接口取代。因此,借助硬件接口和植入可执行程序的方式都受到了限制。

当待取证的 Mac 计算机具备指纹解锁功能且嫌疑人(或被害人)也可以提供指纹进行登录时,也是一种可选的办法。但指纹解锁并不是一直可行,下列情形下无法直接使用指纹解锁登录:

(1) Mac 计算机重启或冷启动。

(2) Mac 计算机超过 48 小时未解锁。

(3) 在过去 156 小时内未使用密码解锁设备,且 4 小时内未使用指纹进行解锁。

(4) Mac 计算机收到远程锁定指令。

(5) 指纹尝试超过 5 次。

当待取证的 Mac 计算机的操作系统为 macOS 10.12 或更新版本时,系统提供了一种全新的解锁登录方式,即通过蓝牙有效范围内绑定了同一 Apple ID(且开启了 2FA)的 Apple Watch 实现自动解锁。因此,在现场搜寻机主可能持有的 Apple Watch 也是一种可选的办法,如图 5-25 所示。当然,该方法对 Mac 计算机重启后的首次解锁无效。

图 5-25 “安全性与隐私”设置中使用 Apple Watch 解锁 Mac 的相关选项

4. 系统信息

开机状态下获取系统信息的方式有多种，不借助专门的取证工具时，取证人员可以通过macOS系统自带的命令行工具实现系统信息的获取。如图5-26所示，打开终端程序，通过表5-5所示的命令查看系统信息。

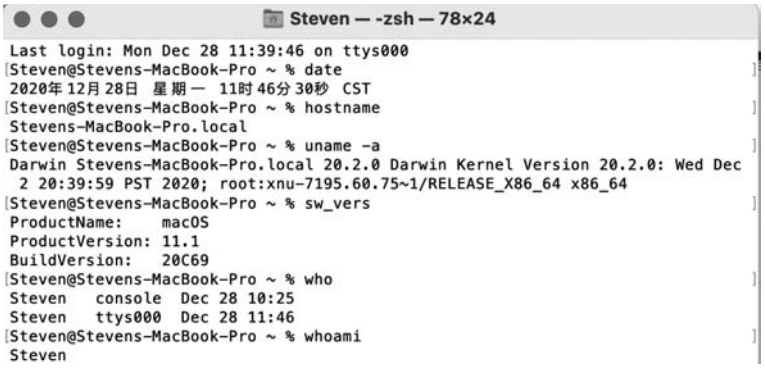


图5-26 通过命令行工具查看系统信息

表5-5 查看系统信息的命令

命 令	功 能 描 述
date	显示系统当前的日期和时间
hostname	显示主机名
uname -a	显示操作系统的有关信息
sw_vers	显示操作系统版本
who	列出当前登录的所有用户
whoami	显示当前正进行操作的用户名

当然，除了上述命令外，macOS下的命令还有很多，常用于在开机状态下快速查看系统信息的命令如表5-6所示。

表5-6 开机状态下快速查看系统信息的命令

操 作 要 求	命 令
列出用户登录信息	last
列出最近执行过的命令及编号	history
显示当前系统活动的总信息	w
显示打开的文件	lsof|more
显示运行的进程	ps aux |more
显示进程的活动网络连接	lsof -i
显示网络配置信息	ifconfig
显示路由表	netstat -rn|more
显示ARP表	arp -an

5. 易失信息

易失信息，也称为易失性数据，即容易丢失的数据。当目标计算机处于开机状态时，易

失信息的提取对有些案件来说特别重要。提取易失性数据需要一定的时间，在提取易失性数据时一定要先确保数据的安全，然后根据实际情况进行提取。

提取物理内存运行时的镜像可以使用盘古石现场取证系统(Safe Imager)，操作界面如图 5-27 所示。在较新的 macOS 中提取内存数据时需要输入管理员账户的密码以取得权限。

图 5-27　使用 SafeImager 获取内存镜像

6. 其他信息

当目标计算机处于开机状态时，还需仔细检查计算机中是否有特殊应用程序在运行，例如，远程控制软件、虚拟机控制软件等。如果存在远程控制，而且被控制计算机中的数据特别重要，应当采用合适的方法及时对被远程控制的计算机进行数据提取，防止断开连接后无法获取关键数据。

5.4.3　离线取证

离线取证，即目标计算机处于关机状态的取证操作。此时不要轻易启动目标计算机的系统，要对其进行合法合规的封存并做好记录，最后选用合适的方式进行数据提取与分析。

1. 拆机镜像

在最初版本的 Mac 计算机中，数据没有全盘加密，且硬盘方便拆卸，取证人员可以直接拆机后使用对应的只读接口对磁盘进行数据提取和固定。

随着 MacBook 系列的推出，特别是 MacBook Air 的上市，苹果在 2013 年起出厂的 MacBook 系列笔记本电脑都采用了固态硬盘(SSD)配置，且支持自由拆卸，如图 5-28 所示。因此，这个时期的 Mac 计算机都可以拆机取证，前提是配备好适配 Mac 选用的 SSD 的 PCIe 转接口即可。2016 年新款 MacBook Pro 的推出后，SSD 固化在主板上无法拆卸，自此，拆机镜像的方式不再可行。

图 5-28　拆卸 MacBook Pro 中的硬盘(左侧为 SATA 机械硬盘,右侧为 PCIe 固态硬盘)

2. 不拆机镜像

针对无法拆卸或不具备磁盘转接条件的情况,可以使用不拆机工具直接启动取证专用系统后对 Mac 计算机内部磁盘进行镜像。常见的不拆机镜像工具有 SafeImager、RECON ITR、Paladin 等。具体方法是:连接已经制作好的专用 USB 设备,按住 Option 键选择外部设备启动(如果是普通 Windows 键盘,则使用 Alt 键代替)。

值得注意的是,2016 年及以后推出的 MacBook 系列具有开盖即启动的功能,当目标机笔记本为合盖状态时,在打开前要提前外接键盘并按住 Option 键,防止直接启动进入嫌疑人系统破坏原始证据。

如图 5-29 所示,在从 USB 外部启动过程中可能遇到设备有固件加密、被锁定(开机有 PIN 码)或 T2 芯片验证,此时外部启动不能成功,目前从技术层面暂时无法解决。

(a)　　(b)

图 5-29　有固件加密(a)和被擦除/锁定的 Mac 计算机(b)

搭载 T2 芯片的 Mac 计算机在安全启动设置上默认开启了完整性验证,在启动期间 Mac 计算机会验证启动磁盘上操作系统的完整性,以确保操作系统是合法的。启动安全性设置也可以决定是否允许从外部介质进行启动。

如图 5-30 所示,启动安全性设置需要在恢复模式下进行,同时按住 Command+R 组合键开机进入,在设置时输入管理员密码。

3. 目标磁盘模式

目标磁盘模式(即 TDM 模式)是在两台 Mac 计算机之间进行数据传输的一种特殊模

图 5-30　启动安全性的设置工具

式。该模式需要在开机时按住 T 键进入或直接在系统中重启进入(图 5-31)。该模式下 UEFI 固件会启动内建的 UEFI 应用程序,该程序会将内置存储设备显示为基于块的无格式存储设备,这些设备可以通过雷电或 FireWire 接口进行连接(具体取决于 Mac 机型)。

图 5-31　开机状态下在系统偏好设置中选择目标磁盘模式

开启该模式的 Mac 计算机,可以在开机状态下在线提取,也可以使用目标磁盘模式进行数据提取。当 Mac 计算机进入目标磁盘模式时,屏幕上会显示大的雷电或 FireWire 符

号。如果两台计算机之间的连接正确，则以目标磁盘模式启动的 Mac 计算机会在另一台取证专用 Mac 计算机上显示为磁盘，此时就可以对磁盘数据进行提取和固定。如果目标磁盘启用了文件保险箱加密，系统则会要求取证人员输入密码来解锁并挂载这个磁盘。

要退出目标磁盘模式，需要首先从取证专用 Mac 计算机中退出磁盘，或者直接关闭取证专用 Mac 计算机。然后，按住目标 Mac 计算机上的电源按钮 10s，关机后松开按钮。

5.5 数据的分析

域(Domain)是 macOS 管理所有文件系统资源的方式，它不是体现在某个看得见摸得着的文件或者界面上的。如果说一个企业内部的管理有等级制，那么 macOS 就有“域”制。“域”是 Apple 管理文件系统的机制，传统 UNIX 以及 Linux 系统上很少提到。

在 macOS 中共有 4 个域：用户(User)、本地(Local)、网络(Network)和系统(System)，这 4 个域几乎涵盖了 macOS 操作系统的全部内容。

5.5.1 用户域

用户域中包含了一个用户登录到系统后可以支配的文件资源。简单地说，用户域是由用户的主目录来定义的。主目录通常是/Users/下的某个目录，比如用户 Steven，他的主目录很可能是/Users/Steven。当然，这个主目录也可以重新指定到另一个位置，比如另一个分区，或者一个分区上的某个目录，甚至是网络上的某个目录或者移动硬盘上的一个目录。在一些公司或者学校等组织中，通常做法是把全部用户的主目录集中存放在一台服务器上，这样用户就可以在任何一台客户端机器上登录访问自己的主目录下的资源。

用户的主目录下存放了用户的一切信息，比如系统偏好信息、终端的命令历史记录、应用程序配置信息等，还包括由该用户修改和产生的信息。

5.5.2 用户与群组

macOS 中的用户和群组的功能类似于 Windows 操作系统中的用户和工作组，可以联系 Windows 中的知识，再结合 macOS 中的操作来理解本节的内容。

1. 用户账户

用户指人，而账户指 macOS 赋予某用户的系统身份。为了便于理解，假设不存在几个用户共享一个账户登录的情况，这样用户跟账户(指常规登录账户)便一一对应起来，下文提到的账户与用户，如果没有特别标明，那么两者不加区分。

1) root 账户

类似于 UNIX/Linux 下的 root 账户和 Windows 中的 Administrator 账户，macOS 中的 root 账户是系统超级管理员，拥有系统最高权限，如果以 root 身份登录了 macOS，就相当于进入了“超级用户”，这是个万能也因此十分危险的角色。默认情况下，macOS 会将 root 账户禁用。事实上，这里的禁用是将 root 账户的密码设为一个随机值。在这种情况下，可以通过以下步骤启用 root 用户：

(1) 进入“系统偏好设置”→“用户与群组”。

(2) 单击窗口左下角锁形图标,输入管理员名称和密码。

(3) 单击“登录选项”→“加入”→“打开目录实用工具”。

(4) 单击窗口左下角锁形图标,输入管理员名称和密码。

(5) 选取菜单栏上的“编辑”→“启用 Root 用户”,如图 5-32 所示。

(6) 在弹出的窗口中设置 root 账户的密码。

图 5-32　启用 root 账户

2) 管理员账户

系统管理员账户是除 root 之外的第一个用户,该账户可以修改系统配置需要中的任意项目、安装应用程序、在系统中添加可以被所有用户共享的任何资源,但不包括关键的系统文件以及其他用户除了 Public 和 Sites 目录中的文件。这里因为管理员账户也不过是被加入了 Admin 用户组的普通用户,用户域要确保用户之间有一定的独立性。在 macOS 中,至少要有一个 Admin 用户组的普通用户,因为 macOS 不鼓励 root 账户来管理系统,普通用户也不能管理系统配置。

3) 普通账户

普通账户只能使用全部应用程序的一个子集,并修改与本账户有关的系统设置。但像安全性与隐私、电池、打印机与扫描仪、网络、共享、用户与群组、日期与时间以及启动磁盘这些系统范围的设置,普通用户是无法进行设置的。根据“域”的管理要求,普通用户无法看到或者更改其他普通用户的私有资源。

4) 访客账户

访客账户是从 macOS 10.5 开始新增的一类账户类型。类似于 Windows 系统中的 guest 账户,macOS 中的访客账户没有自己的密码,所以任何人都能以访客身份使用 macOS,一旦访客离开了系统,访客的主目录就会被删除。

5）共享用户

共享用户是 macOS 中权限最低的一类账户，因为这类用户没有自己的私有空间（主目录）。共享用户一般不能登录系统，因为默认情况下该类账户没有自己的密码，能访问的地方也只有其他账户的公共目录和共享目录（除非管理员来改变这一切）。

6）系统账户

系统账户是系统针对特定进程进行操作的一类账户，如果用户在另一台计算机上登录 Mac 设备，那么前提是该 Mac 设备上一定要有个系统账户在后台默默地维护着对应进程。这些系统账户通常以“_”开头，也有自己的主目录，但主目录不在/Users 下面，而是在其他应用程序的安装目录中。

跟传统 UNIX 不同，macOS 把本地所有账户信息集中存储在一个目录/private/var/db/dslocal/nodes/Default/Users 中。这个目录就是该机器的“户口中心”，每个用户都有一个对应的 property list 文本文件（后缀名为.plist）记录着该账户的内部信息，仅在 root 权限下可查看，显示信息如图 5-33 所示。

Key	Class	Value
> authentication_authority	数组	3 个排序对象
⌄ generateduid	数组	1 个排序对象
0	字串	50E55F80-0DB7-4DE8-B716-EE632CA66910
⌄ gid	数组	1 个排序对象
0	字串	20
⌄ home	数组	1 个排序对象
0	字串	/Users/Steven
⌄ inputSources	数组	1 个排序对象
0	字串	<?xml version="1.0" encoding="UTF-8"?>
⌄ jpegphoto	数组	1 个排序对象
0	数据	474,172 字节: FFD8FFE0 00104A46 494600...CC4904E3
⌄ name	数组	2 个排序对象
0	字串	Steven
1	字串	com.apple.idms.appleid.prd.000343-05-b3527510-c2fd-45
⌄ passwd	数组	1 个排序对象
0	字串	••••••••
⌄ realname	数组	1 个排序对象
0	字串	Steven Pi
⌄ record_daemon_version	数组	1 个排序对象
0	字串	4850000
⌄ shell	数组	1 个排序对象
0	字串	/bin/zsh
⌄ uid	数组	1 个排序对象
0	字串	501

图 5-33　账户 Plist 文件中的用户信息

2. 群组

组仅仅是用户的一个分类管理列表。把一个或者多个用户逻辑上划分为一个组，可以针对文件和一些系统级的操作形成更为细化且方便的控制机制。例如，一个公司有一个“人力资源部”，处于这一部门（逻辑上的组）的员工（个人账户）就可以被赋予查看或处理公司的一些敏感信息（文件或者某些程序）的权限，用于给员工加薪、发放工资或者培训之类的操作。一个员工可以属于多个部门，所以一个账户可以属于多个组。

系统默认下，普通用户被指定为 staff 组，管理员被指定为 admin 组和 staff 组，而 root 账户则会被指定为 wheel 组。

需要说明的是，组是可以嵌套的，即可以把一个小组整体作为另一个小组的成员。

5.5.3　主目录

主目录指管理员和普通账户所在的目录，但不包括 root、访客、共享和系统账户。用户可以在自己账户下的主目录中建立任意文件和目录。默认情况下，有一些已经存在的子目录（图 5-34）。

名称	种类	修改日期
公共	文件夹	2018年11月8日 16:36
图片	文件夹	2020年12月10日 11:17
文稿	文件夹	2020年11月16日 12:01
下载	文件夹	今天 13:16
音乐	文件夹	2020年11月18日 11:20
应用程序	文件夹	2019年10月8日 11:36
影片	文件夹	2020年11月9日 10:35
桌面	文件夹	2020年12月10日 09:36

图 5-34　用户的主目录

1）应用程序

应用程序（applications）默认安装在 macOS 系统容器的“应用程序”目录下，用户目录下的应用程序目录默认为空。如果用户选择将第三方应用程序手动安装到“应用程序”目录中，macOS 将会自动在该目录找到安装的程序，且该用户独自拥有该程序的使用权。

2）桌面

传统类 UNIX 系统基于命令界面，没有“桌面”（Desktop）这个概念，而图形界面的产生使主目录下多了一个文件夹，用来存放用户在图形界面登录后看到的内容。

3）文稿

文稿（documents）包含用户的个人文档。某些应用程序默认会在该目录创建文件夹存放相关文档。

4）下载

系统默认会设置一个下载（downloads）目录，用于临时存放下载的文件。例如，通过浏览器下载了一个文件，浏览器默认会把该文件存放在 downloads 目录中。

5）资源库

资源库（library）用于存放用户的应用程序以及与该用户相关的其他系统资源，这些资源包括字体、联系人、邮件、屏保等，用户的所有个人信息及对应的资源都存放在这里。

6）影片

影片（movies）是视频文件的默认存储目录，iTunes、iMovie 等程序默认将视频文件存放在该目录下。

7）音乐

音乐（music）目录用于存放音乐文件，iTunes、网易云音乐等程序默认将音乐文件存放在该目录下。

8）图片

图片（pictures）目录用于存放图片文件，iTunes、iPhoto 等程序默认将图片文件存放在

该目录下。

9）公共

公共(public)目录用于存储与其他用户共享的文件，系统默认状态下，这个目录可以被其他用户访问。

既然主目录归用户所有并管理，那么用户确实是可以随意存放文件的，例如把音乐放在图片目录下、把私人视频文件放在公共目录下、把文稿统统放在资源库目录下等，这种做法虽然不是很合理，但这些在 macOS 看来都是“合法”的。所以，在取证过程中关键是要看文件的内容，而不能简单地用目录来归类。

5.5.4 资源库

资源库(library)包含了每一个独立用户专有的系统资源、针对个人爱好的设置和数据，被誉为“取证数据中的金矿”。默认情况下，macOS 中的用户“资源库”文件夹在 macOS 10.7 及以后的版本中处于隐藏状态。

值得注意的是，这里的资源库是用户资源库，与根目录下的资源库不同，后者为系统全局资源库，不能混淆。

1. Containers

Containers 中包含从 macOS 的 App Store 下载的应用程序的支持文件、缓存的数据和临时文件。自 macOS 10.7 开始，App Store 上的应用程序通过沙箱来保护数据。

应用数据存储的传统位置是“～/Library/Application Support/”(其中，“～”代替用户目录)，而使用沙箱保护的应用数据会存储在“～/Library/Containers/< Bundle ID >/Data/Library/Application Support/”中，因此，应用数据可能保存在一个或两个目录中，如图 5-35 所示。

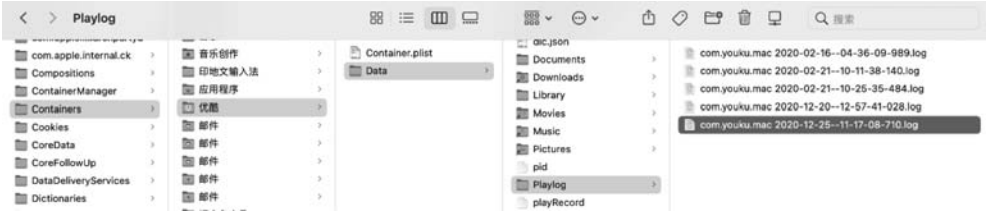


图 5-35 用户资源库中 Containers 目录下存储的应用数据

2. Preferences

顾名思义，Preferences 目录存储应用和系统设置的 plist 文件，如图 5-36 所示。类似地，相关. plist 文件可能存储在“～/Library/Preference/ByHost/”和“～/Library/Containers/< Bundle ID >/Data/Library/Preference/”两个目录中。其中，“～/Library/Preference/ByHost/”目录下存储的 Plist 文件的命名采用了“反向 DNA”格式。

3. Application Support

Application Support 目录是应用存储数据的位置，其中包括系统应用程序和安装的第三方应用程序。

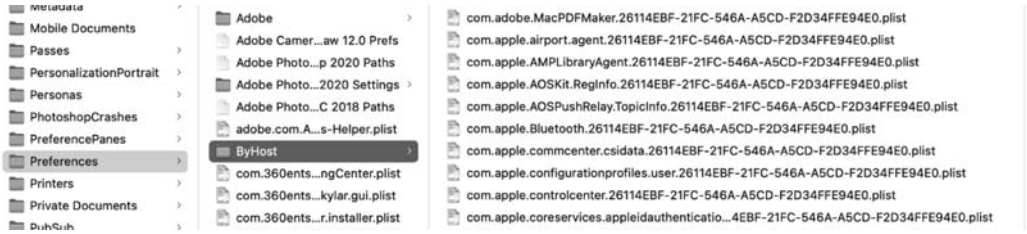


图 5-36　用户资源库中 Preferences 目录下存储的.Plist 文件

4. Caches

Caches 目录存储着每个应用的缓存数据，大部分缓存文件存储在“～/Library/Caches/<Bundle ID>/fsCachedData/”下，并采用“反向 DNA”格式命名，如图 5-37 所示。

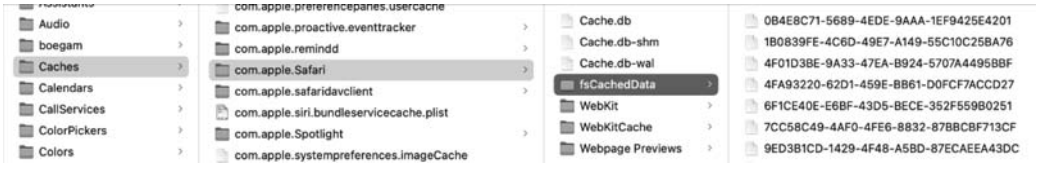


图 5-37　用户资源库中 Caches 目录下存储的缓存数据

5.5.5　钥匙串

钥匙串(keychain)是 Mac 计算机的密码信息管理系统，自 macOS 8.6 开始作为系统自带功能推出并沿用至今。钥匙串相当于一个加密的容器，用以安全地存储 Mac、App、服务器和网站的账户名、密码以及机密信息，如信用卡号或银行账户 PIN 码等。用户访问网站、电子邮件账户、网络服务器或其他受密码保护的任意 App 时，可以选择将账户密码存储在钥匙串中，这样在每次登录时都无须输入密码。同样，在运行 iOS、iPadOS 的移动设备上以及 iCloud 中也支持对应的钥匙串功能。

1. 存储与访问

钥匙串文件存储在～/Library/Keychains/(图 5-38)和/Library/Keychains/(图 5-39)中，访问钥匙串的系统程序是应用程序目录下“实用工具”中的“钥匙串访问”，也有一个称为 security 的访问钥匙串的命令行工具。

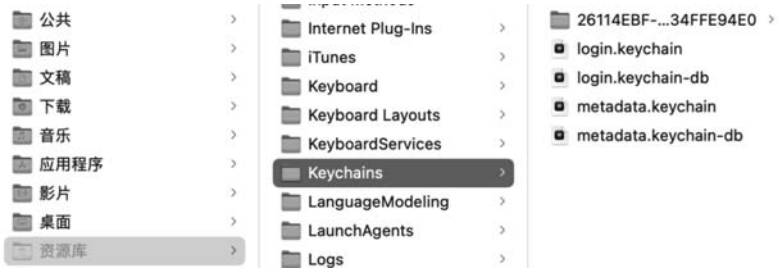


图 5-38　“～/Library/Keychains/”中的钥匙串文件

图 5-39 /Library/Keychains/中的钥匙串文件

“钥匙串访问”是一个 macOS 应用程序，它允许用户访问和配置钥匙串的内容，加锁或解锁钥匙串，显示系统存储的密码、证书、密钥和安全备注等操作。“钥匙串访问”的图形用户界面显示了多种类型的钥匙串，通常包括登录、iCloud、系统和系统根证书，如图 5-40 所示。

图 5-40 “实用工具”里的“钥匙串访问”App

在钥匙串访问 App 中双击某一项目行或单击窗口上方的“显示所选项目的简介”快捷按钮，即可查看存储在钥匙串中的信息(选中“显示密码”)，如图 5-41 所示。

图 5-41 存储在钥匙串中的密码信息

2. 加锁与解锁

默认钥匙串文件是登录(Login)钥匙串,通常在登录时使用用户口令解锁,但该钥匙串的密码也可以不同于用户口令,通过牺牲部分便利性来提高安全性。钥匙串访问程序不允许为钥匙串设置空密码。

钥匙串可以被设为在计算机闲置一定时间后自动"加锁",也可以通过钥匙串访问程序手动加锁。如果钥匙串已被加锁,则访问钥匙串需要再次输入密码以解锁。以新文件覆盖~/Library/Keychains/中的文件(如系统恢复)也会导致钥匙串被锁住。

如果登录钥匙串被用户口令所保护,则当用户口令被更改时该钥匙串密码也会随之更改。然而,在共享的 Mac 或非 Mac 局域网中,从非 macOS 更改用户口令,则可能存在两个密码不同步的情形。

5.5.6 系统痕迹

系统痕迹是操作系统运行过程中系统自主产生的使用痕迹。macOS 中通常需要关注的系统痕迹有操作系统信息(如系统版本、时区信息、用户信息、开关机记录、蓝牙连接记录、Wi-Fi 连接记录、USB 连接记录)、系统日志、系统活动记录等。

1. 操作系统信息

macOS 中主要的系统信息存储位置如下。

系统版本:/System/Library/CoreServices/SystemVersion. plist

语言和时区:/Library/Preferences/. GlobalPreferences. plist

系统设置:/Library/Preferences/SystemConfiguration/preferences. plist

最后更新:/Library/Preferences/com. apple. SoftwareUpdate. plist

最后休眠时间:/Library/Preferences/SystemConfigurations/com. apple. PowerManagement. plist

蓝牙记录:/Library/Preferences/com. apple. Bluetooth. plist

键盘设置:/Library/Preferences/com. apple. HIToolbox. plist

打印机设置:/Library/Preferences/org. cups. printers. plist

时间机器:/Library/Preferences/com. apple. TimeMachine. plist

防火墙:/Library/Preferences/com. apple. alf. plist

AirPort:/Library/Preferences/SystemConfigurations/com. apple. airport. preferences. plist

用户信息:/private/var/db/dslocal/nodes/Default/Users/

最后登录的用户:/Library/Preferences/com. apple. loginwindow. plist

MAC 地址:/private/var/log/daily. out

启动项:/Library/LaunchAgents/、/Library/LaunchDaemons/、/System/Library/LaunchAgents/、/System/Library/LaunchDaemons/

隔空投送(AirDrop):/var/db/uuidtext/

USB 连接记录:/private/var/db/diagnostics/persist/XXXXX. tracev3

2. 系统日志

不同的 macOS 文件系统所对应的系统日志存储路径也各不相同。例如，在 HFS＋文件系统中日志存储在系统根目录“/. fseventsd”文件夹下，而在 APFS 文件系统中，日志则存储在系统根目录“/private/var/log”文件夹下。

针对已经普及的 APFS 环境，macOS 的日志目录下较清晰地存储着操作系统日志(system. log)、应用程序日志(install. log)、文件系统日志(fsck_apfs. log)、Wi-Fi 日志(wifi. log)等。其中，应用程序日志记录着程序的安装、更新、卸载等信息，文件系统日志记录着卷的挂载、卸载以及文件和文件夹的创建、修改等信息，如图 5-42 所示。

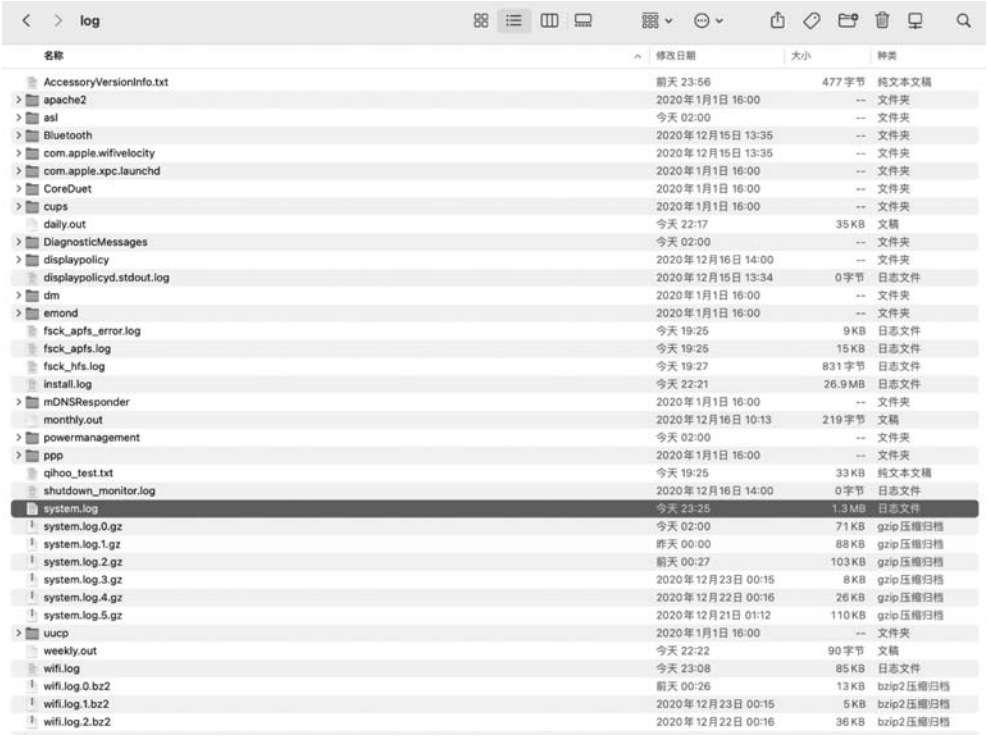


图 5-42　APFS 文件系统环境下的系统日志目录

文件系统日志对于每个苹果取证人员而言都是非常宝贵的证据，它记录着文件系统发生的变化，它被时间机器(Time Machine)和聚焦搜索(Splotlight)等操作系统的多个组件使用。类似 Windows 系统的 NTFS 日志(该日志记录了 NTFS 文件系统的变化，并将数据存储在 UsnJrnl：$J 中)，通过解析文件系统日志，可以分析出文件系统曾发生过的事件，如文件、文件夹、符号链接和硬链接的创建、删除、重命名、修改、权限更改等。其中有价值的日志包括：安装和卸载外部驱动器和磁盘镜像、用户配置目录中的活动、文档编辑、互联网活动、移动到回收站的文件、下载的文件等。

macOS 的文件系统结构并没有开源，所以大部分取证工具虽然支持对 macOS 文件系统的读取，但是对于 macOS 文件系统的元数据等底层结构的解析能力却有所欠缺，在必要时，需要借助 macOS 自身的解析能力来解析更多的数据，例如 macOS 自带的 mdls 命令。

mdls 命令可以显示 macOS 里某个文件的元数据，包括详细的时间信息，这对于有些案

件的证据分析有很大的帮助。mdls 命令使用方式较为简单，输入“mdls <文件所在路径>”即可，如图 5-43 所示。

```
Steven — -zsh — 83×53
Steven@Stevens-MacBook-Pro ~ % mdls /Users/Steven/Pictures/personal/IMG_5675.jpg
_kMDItemDisplayNameWithExtensions      = "IMG_5675.JPG"
kMDItemAcquisitionMake                 = "Canon"
kMDItemAcquisitionModel                = "Canon EOS 700D"
kMDItemAperture                        = 4.625
kMDItemAuthors                         = (
    ""
)
kMDItemBitsPerSample                   = 32
kMDItemColorSpace                      = "RGB"
kMDItemContentCreationDate             = 2020-11-02 07:35:42 +0000
kMDItemContentCreationDate_Ranking     = 2020-11-02 00:00:00 +0000
kMDItemContentModificationDate         = 2020-11-02 07:35:42 +0000
kMDItemContentModificationDate_Ranking = 2020-11-02 00:00:00 +0000
kMDItemContentType                     = "public.jpeg"
kMDItemContentTypeTree                 = (
    "public.jpeg",
    "public.image",
    "public.data",
    "public.item",
    "public.content"
)
kMDItemCopyright                       = ""
kMDItemDateAdded                       = 2020-11-10 08:15:33 +0000
kMDItemDateAdded_Ranking               = 2020-11-10 00:00:00 +0000
kMDItemDisplayName                     = "IMG_5675.JPG"
kMDItemDocumentIdentifier              = 0
kMDItemEXIFVersion                     = "2.3"
kMDItemExposureMode                    = 0
kMDItemExposureProgram                 = 2
kMDItemExposureTimeSeconds             = 0.01666666666666667
kMDItemFlashOnOff                      = 1
kMDItemFNumber                         = 5
kMDItemFocalLength                     = 50
kMDItemFSContentChangeDate             = 2020-11-10 08:15:26 +0000
kMDItemFSCreationDate                  = 2020-11-10 08:15:24 +0000
kMDItemFSCreatorCode                   = ""
kMDItemFSFinderFlags                   = 0
kMDItemFSHasCustomIcon                 = (null)
kMDItemFSInvisible                     = 0
kMDItemFSIsExtensionHidden             = 0
kMDItemFSIsStationery                  = (null)
kMDItemFSLabel                         = 0
kMDItemFSName                          = "IMG_5675.JPG"
kMDItemFSNodeCount                     = (null)
kMDItemFSOwnerGroupID                  = 20
kMDItemFSOwnerUserID                   = 501
kMDItemFSSize                          = 2663068
kMDItemFSTypeCode                      = ""
kMDItemHasAlphaChannel                 = 0
kMDItemInterestingDate_Ranking         = 2020-11-02 00:00:00 +0000
kMDItemISOSpeed                        = 400
kMDItemKind                            = "JPEG图像"
```

图 5-43　mdls 命令的结果

3. 系统活动记录

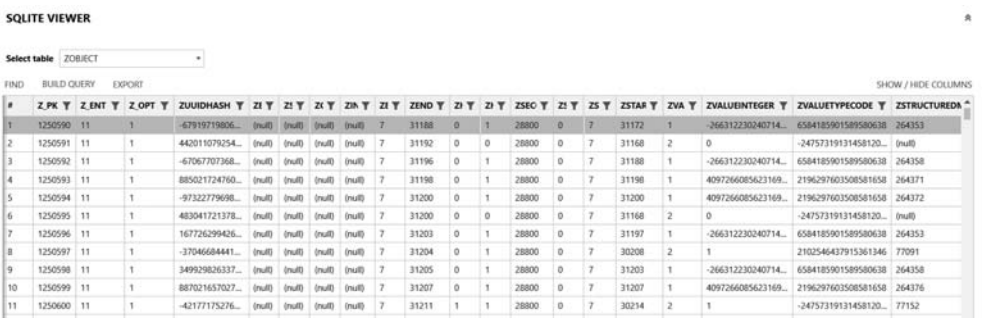
macOS 近年来一直不断更新功能，这些新的功能特性对于取证来说，有的带来了困难，有的则带来了更多的机会。

1）KnowledgeC. db

KnowledgeC. db 是一个记录了苹果计算机大量活动的 SQLite 数据库，包括应用程序活动、设备插入焦点、Safari 历史记录、屏幕背光状态等。这些记录可以间接地反映用户使用操作系统时留下的行为痕迹。该文件存储于“/private/var/db/CoreDuet/Knowledge/”下，如图 5-44 所示。

2）已隔离文件

已隔离文件(quarantined files)记录了使用苹果计算机从互联网下载文件的相关信息，包括日期、时间和文件下载位置。该文件位于“～/library/Preferences/com. apple.



#	Z_PK	Z_ENT	Z_OPT	ZUUIDHASH	ZI	ZS	ZC	ZIN	ZI	ZEND	ZI	ZI	ZSEC	ZS	ZS	ZSTAR	ZVA	ZVALUEINTEGER	ZVALUETYPECODE	ZSTRUCTUREDA
1	1250590	11	1	-679197198O6...	(null)	(null)	(null)	(null)	7	31188	0	1	28800	0	7	31172	1	-266312230240714...	6584185901589580638	264353
2	1250591	11	1	442011079254...	(null)	(null)	(null)	(null)	7	31192	0	0	28800	0	7	31168	2	0	-24757319131458120...	(null)
3	1250592	11	1	-6706770736B...	(null)	(null)	(null)	(null)	7	31196	0	1	28800	0	7	31188	1	-266312230240714...	6584185901589580638	264358
4	1250593	11	1	885021724760...	(null)	(null)	(null)	(null)	7	31198	0	1	28800	0	7	31198	1	40972660856231691...	2196297603508581658	264371
5	1250594	11	1	-97322779698...	(null)	(null)	(null)	(null)	7	31200	0	1	28800	0	7	31200	1	40972660856231691...	2196297603508581658	264372
6	1250595	11	1	483041721378...	(null)	(null)	(null)	(null)	7	31200	0	0	28800	0	7	31168	2	0	-24757319131458120...	(null)
7	1250596	11	1	167726299426...	(null)	(null)	(null)	(null)	7	31203	0	1	28800	0	7	31197	1	-266312230240714...	6584185901589580638	264353
8	1250597	11	1	-37046684441...	(null)	(null)	(null)	(null)	7	31204	0	1	28800	0	7	30208	2	1	2102546437915361346	77091
9	1250598	11	1	349929826337...	(null)	(null)	(null)	(null)	7	31205	0	1	28800	0	7	31203	1	-266312230240714...	6584185901589580638	264358
10	1250599	11	1	887021657027...	(null)	(null)	(null)	(null)	7	31207	0	1	28800	0	7	31207	1	40972660856231691...	2196297603508581658	264376
11	1250600	11	1	-42177175276...	(null)	(null)	(null)	(null)	7	31211	1	1	28800	0	7	30214	2	1	-24757319131458120...	77152

图 5-44 查看 knowledgeC.db 的内容

LaunchServices.QuarantineEvensV2/”下，当需要打开这些文件时，系统会提示这些文件是从互联网下载的，需要进行安全性确认。

3）Diagnostics

Diagnostics 位于/var/db/diagnostics/，是 macOS 的诊断报告，其中记录了系统、应用程序和用户的操作行为，在系统诊断和使用信息中可以看到大量的行为记录。

5.5.7 用户痕迹

用户痕迹是操作系统运行过程中由用户操作产生的使用痕迹。macOS 中通常关注的用户痕迹有系统功能的使用痕迹（如通讯录、回收站记录、命令历史、程序状态、最近打开的项目、打印记录等）、浏览器记录、邮件记录、即时通信记录等。

1. 系统功能使用痕迹

1）通讯录

通讯录是 macOS 自带的应用程序，其主要数据存储在～/Library/Application Support/AddressBook/AddressBook-v22.abcddb 文件中，该文件为 SQLite 格式。通讯录中的联系人信息分散地存储在～/Library/Application Support/AddressBook/Sources/<string>/目录中，其中 Metadata 文件夹中包含联系人明细信息，文件扩展名为.abcdp，这些文件是 Plist 格式，可以直接用 PlistEdit Pro 打开查看。Images 文件夹包含的图片是与联系人相关的头像图片，实际为.jpg 格式，可以直接预览查看，如图 5-45 所示。

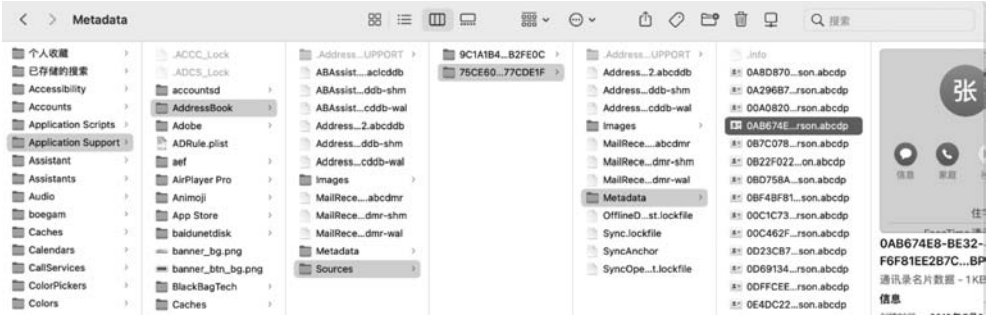


图 5-45 联系人信息的存储路径

2）聚焦搜索记录

聚焦搜索（Spotlight）是 macOS 10.4 开始推出的快速搜寻引擎。Spotlight 使用 Metadata 搜索引擎，可以广泛查找任何位于计算机中的项目，包含文件、图片、音乐、应用程序等，还可以搜索到文档中指定的文字内容。Spotlight 历史记录存储在～/Library/Preferences/com.apple.Spotlight.plist 中。

3）用户行为痕迹

在用户对应账户目录中隐藏着如下一些重要的行为信息。

（1）终端输入的历史命令：～/.bash_history、～/bash_sessions、～/.zsh_history、～/.zsh_sessions。

（2）默认打印机：～/.cups/lpoptions。

（3）默认文本编码和区域语言设置：.CFUserTextEncoding。

（4）回收站记录：～/.Trash。

回收站（.Trash）是一个隐藏文件夹，包含所有删除的文件，通常位于用户目录，.Trashes 目录也存在其他的外部存储介质中，如移动硬盘。回收站位于程序坞的最右方（或最下方），删除文件会被放入其中，除非用户主动清空，否则这些文件会一直存在里面。

位于回收站中的文件无法预览，也无法使用，如需要预览和使用必须移出到另一个位置。从 macOS 10.6 开始的回收站都具备了“放回原处”功能。这个过程其实并不是“恢复”，只需把文件从回收站里拖到桌面或者其他目录即可。

4）程序痕迹

用户对应用程序进行设置或启动运行时，会产生很多程序痕迹数据，主要信息包括：

（1）程序坞记录。程序坞记录用于记录当前用户放置在程序坞上的应用程序信息，具体存储在～/Library/Preferences/com.apple.dock.plist 文件中。其中，persistent-apps 字段对应程序坞上的常驻应用程序，如图 5-46 所示；recent-apps 字段对应最近打开的应用程序，如图 5-47 所示。

图 5-46　persistent-apps 字段对应程序坞上的常驻应用程序

（2）保存的程序状态。应用程序窗口在启动运行时产生的状态记录存储在～/Library/Saved Application State/目录下，分布于程序各自目录中，实际上大部分应用程序目录在此处仅仅是替身（类似于 Windows 的快捷方式），真正的文件位于～/Library/Containers/

图 5-47　recent-apps 字段对应最近打开的应用程序

<Bundle ID>/Data/Library/Saved Application State/目录中，主要包含 windows. plist、data. data 和 window_#. data(该文件在部分目录中可能没有，“#”代表数字(下同)，从 1 开始计数)，如图 5-48 所示。其中，windows. plist 文件存储着应用程序窗口信息，如 Microsoft Word 中打开的文档名称、最近打开的文件名称、Safari 浏览的网页等，如图 5-49 所示。

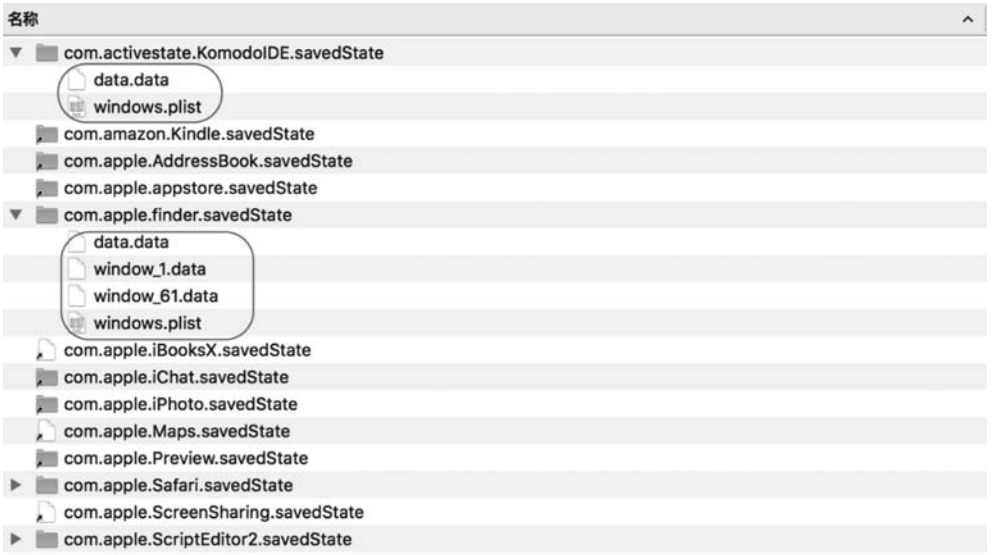


图 5-48　应用程序状态信息

(3) 开机时重新运行的程序。macOS 关机时，会记录下当前正处于打开状态的应用程序窗口，并弹出关机确认窗口，其中包含选项“再次登录时重新打开窗口”，如勾选确认，则下次开机时自动打开上述未关闭的应用程序，如图 5-50 所示。记录这些开机时需要重新运行的程序的信息存储在～/Library/Preferences/ByHost/com. apple. loginwindow. *. plist 文件中，其中 * 代替反向 DNA 字符串。

5) 最近使用痕迹

如图 5-51 所示，macOS 中包含大量最近使用痕迹，主要信息及存储位置如下。

(1) 最近使用的项目：～/Library/ApplicationSupport/com. apple. sharedfilelist。

(2) 最近使用的程序：com. apple. LSSharedFileList. RecentApplications. sfl。

(3) 最近使用的文稿：com. apple. LSSharedFileList. RecentDocuments. sfl。

Key	Class	Value
NSDockMenu	数组	3 个排序对象
0	字典	5 对 键/值
command	数字	1
mark	数字	2
name	字串	chap5-macOS取证技术.docx
system-icon	数字	1735879022
tag	数字	191
1	字典	1 对 键/值
2	字典	3 对 键/值
indent	数字	0
name	字串	打开最近的文件
sub	数组	12 个排序对象
0	字典	4 对 键/值
1	字典	4 对 键/值
command	数字	2
indent	数字	0
name	字串	盘古石手机取证分析系统使用说明_V6.2.docx
tag	数字	1

图 5-49 Windows.plist 中的应用程序窗口信息

Key	Class	Value
TALAppsToRelaunchAtLogin	数组	8 个排序对象
0	字典	4 对 键/值
1	字典	4 对 键/值
BackgroundState	数字	2
BundleID	字串	com.microsoft.word
Hide	布尔值	否
Path	字串	/Applications/Microsoft Word.app
2	字典	4 对 键/值
3	字典	4 对 键/值
BackgroundState	数字	2
BundleID	字串	com.microsoft.powerpoint
Hide	布尔值	否
Path	字串	/Applications/Microsoft PowerPoint.app
4	字典	4 对 键/值
BackgroundState	数字	3
BundleID	字串	com.tencent.xinwechat
Hide	布尔值	否
Path	字串	/Applications/WeChat.app

图 5-50 开机时重新运行的应用程序信息

图 5-51 系统默认设置的 10 个最近使用的项目

（4）最近访问过的服务器：com. apple. LSSharedFileList. RecentHosts. sfl 和 com. apple. LSSharedFileList. RecentServers. sfl。

（5）每个程序的最近使用记录：com. apple. LSSharedFileList. ApplicationRecentDocuments。

（6）访问最近使用的文件夹：～/Library/Preferences/com. apple. finder. plist。

2. 打印记录

与 Windows 一样，用户在连接打印机执行打印任务时，会留下大量打印痕迹，如图 5-52 所示。主要信息及存储位置如下。

（1）最近使用的打印机：～/Library/Preferences/org. cups. PrintingPrefs. plist。

（2）打印日志：/var/log/cups/。

（3）打印 job control 文件：/private/var/spool/cups/（只在打印过程中出现 PDF 文件）。

图 5-52 最近使用的打印机信息

3. 浏览器记录

Safari 是 macOS 自带的浏览器程序，绝大多数核心数据以 Plist 文件存储，自 V3 版本开始引入了 SQLite 格式，开创了一种全新的缓存方式。大部分 Safari 的数据存储在以下 4 个地方：～/Library/Safari/、～/Library/Caches/com. apple. Safari/、～/Library/Caches/Metadata/和～/Library/Preferences/。Safari 中的缓存页面如图 5-53 所示。

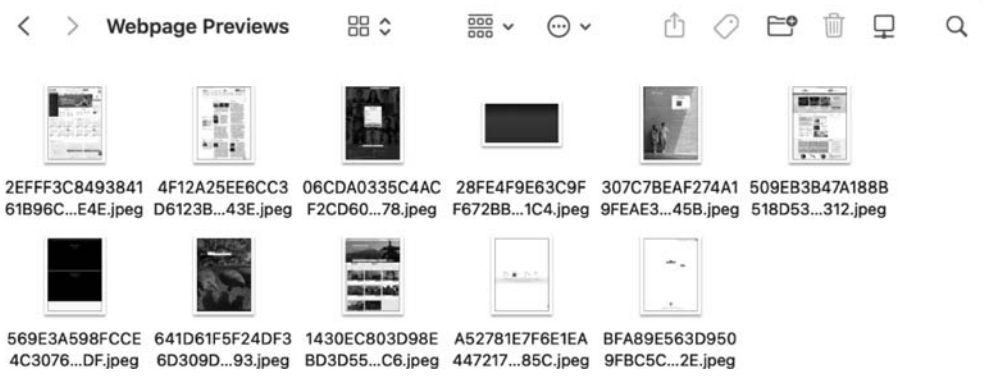


图 5-53 Safari 中缓存的页面

重点痕迹具体分布如下。

(1) 浏览器配置：～/Library/Preferences/com. apple. Safari. plist。

(2) 下载历史：～/Library/Safari/Downloads. plist。

(3) 访问历史：～/Library/Safari/History. db(History_items 存放 URL 和访问次数，History_visits 保存时间和网页的标题)。

(4) 隐私浏览记录：～/Library/Preferences/com. apple. Safari. SafeBrowsing. plist。

(5) 上次打开的窗口：～/Library/Safari/LastSession. plist。

(6) 书签：～/Library/Safari/Bookmarks. plist。

(7) 缓存：～/Library/Caches/com. apple. Safari/Cache. db。

(8) 缓存的页面：～/Library/Caches/com. apple. Safari/Webpage Previews/。

(9) Cookies：～/Library/Cookies。

(10) 最常浏览的网站：TopSites. plist(Safari V4～V6 使用 coverflow 窗口显示用户最常访问(最近访问))的网址，其中包括最后修改的日期、网站 URL 和网站标题。

4. 邮件记录

用户在使用系统邮件客户端时产生的数据存储在“～/Library/Mail/V”中，目前最新版本为 V8，因此实际存储位置为～/Library/Mail/V8，如图 5-54 所示。V8 目录下，Maildata/Signatures/记录了邮件客户端中添加的邮件账户及其签名信息，主要分布在 AccountsMap. plist 和 AllSignatures. plist 文件中。V8 下存储的两个名称较长的目录即 AccountsMap. plist 中对应的账户，账户目录下包含发件箱、收件箱、垃圾邮件、草稿、已发送邮件、已删除邮件等重要数据。

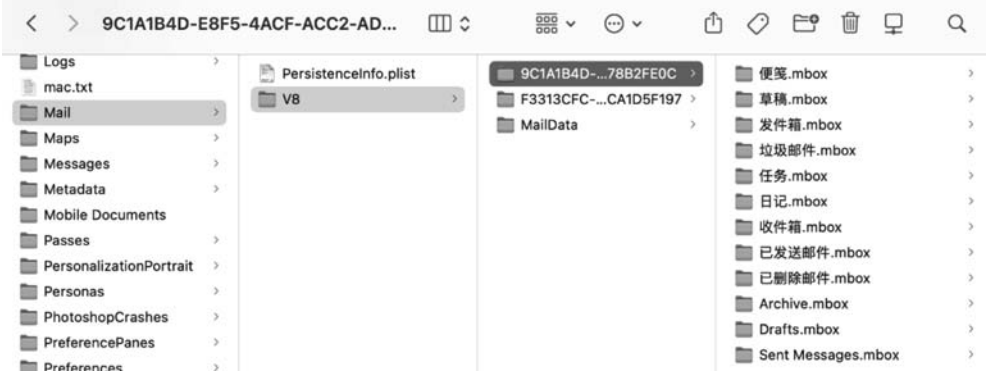


图 5-54　邮件数据目录

如图 5-54 右侧内容所示，邮件的核心数据均存储在这些以. mbox 结尾的目录中。以收件箱. mbox 为例，Data 目录中每个数字子目录代表一个组(macOS 邮件客户端的智能邮箱分组)，每个组中均有 Attachments 和 Messages 两个目录，分别存储邮件附件(支持直接打开查看)和邮件文件(即. emlx 文件，默认通过邮件客户端程序打开，支持文本方式查看邮件头信息)，如图 5-55 所示。

5. 即时通信记录

目前，即时通信是互联网的主要应用功能，大家几乎离不开微信、QQ 等即时通信工具，

图 5-55　收件箱数据

这些工具产生的痕迹信息对于取证工作来说是非常宝贵的。

1）iMessage

iMessage 是苹果公司推出的即时通信软件，可以发送短信、视频等，拥有非常高的安全性。iMessage 虽然与短信/彩信共同使用系统自带的“消息”应用进行数据展现，但实际上用户仅需要通过 Wi-Fi 或蜂窝数据网络进行通信，依托同一个 Apple ID，短信可以在 iPhone、iPad 和 Mac 之间实现同步推送。

iMessage 的消息记录存储在～/Library/Messages/chat. db 文件中，彩信附带的多媒体附件（如图片、视频等）则存储在～/Library/Messages/Attachments/目录中。

2）微信

微信是腾讯公司推出的一款综合社交工具，广受用户欢迎。相比其他平台，Mac 版微信虽然用户基数不大，但作为与手机微信交互的重要数据源，往往在案件侦破中发挥着意想不到的作用。

微信的用户数据存储在“～/Library/Containers/com. tencent. xinWeChat/Data/Library/Application Support/com. tencent. xinWeChat/<版本号>/< MD5 字符串>/”目录下，其中的关键信息包括：

（1）账户信息：/Account/userinfo. data。

（2）好友请求：/Account/friendRequest. data。

（3）联系人消息记录：/Contact/wccontact_new2. db。

（4）群组消息记录：/Group/group_new. db。

（5）所有消息记录：/Message/msg_＃. db。

（6）消息中的多媒体文件缓存：/Message/MessageTemp/。

（7）撤回的消息：RevokeMsg/revokemsg. db。

（8）头像缓存：/Avatar（包括群组图标和群组内非好友用户的头像缓存）。

（9）同步的聊天：/ChatSync/ChatSync. db。

（10）收藏列表：/Favorite/favorites. db。

（11）收藏缓存：/Favorite/data/。

(12) 语音通话记录：/voip/。

当然，微信出于安全性保护，上述关键信息中绝大部分.db格式的SQLite数据库都处于加密状态，解密密钥并不在计算机本地离线存储，且加密方式与手机微信截然不同，暴力破解的可行性微乎其微。因此需要在联网状态下借助常规登录方式(账号密码、手机验证等)登录后进行取证。

3) QQ

QQ的数据分布相比微信更加简单，主要的用户数据存储在"～/Library/Containers/com.tencent.qq/Data/Library/Application Support/QQ/< QQ号>/"目录下，与Windows版类似，消息记录也存储在msg3.0.db文件中，附件存储于并行目录中，如image存储图片、video存储视频等。

4) 其他应用数据

(1) 百度网盘：～/Library/Preferences/com.baidu.BaiduNetdisk-mac.plist

～/Library/Application Support/com.baidu.BaiduNetdisk-mac/<字符串>/file.db

(2) 迅雷：～/Library/Preferences/com.xunlei.Thunder.plist

(3) 印象笔记：～/Library/Containers/com.evernote.Evernote/Data/Library/Application Support/com.evernote.Evernote/accounts/app.yinxiang.com/< ID >/localNoteStore/LocalNoteStore.sqlite

5) iTunes备份

通常情况下Mac计算机用户同时使用iPhone的概率很高，因此其中很有可能存储着机主的iPhone手机的备份数据。iPhone在macOS中默认存储在～/Library/Application Support/MobileSync/Backup/目录中。

5.5.8　时间机器

时间机器(Time Machine)是Mac的内置备份功能。如图5-56所示，Time Machine可

图5-56　系统偏好设置中的"时间机器"

以对用户的所有文件进行自动备份，包括应用、音乐、照片、电子邮件、文稿和系统文件。如果能获取到备份，即便原始文件被永久性删除或者磁盘被抹掉或更换，取证人员还有机会从备份中恢复相关数据。

通常获取时间机器的数据需要找到备份数据时的存储介质，存储介质的查找可以通过查看 Mac 计算机的备份记录寻找线索，找到备份的存储介质直接加载分析即可。当然，备份是支持加密的，如果找到的备份数据被加密，则先解密再分析。

5.5.9 常用工具

通过本节前面内容的介绍，重点对用户主目录、钥匙串、系统痕迹和用户痕迹等内容进行了梳理，读者对 macOS 中关键信息的存储位置已经逐渐掌握，结合前文介绍的 SQLite 文件和 Plist 文件的查看方法和工具，可以尝试手工分析。当然，由于系统和用户数据的存储分散且信息琐碎，手工分析工作量较大且容易出错，为了提高分析效率和结果精确度，国内外的专业取证分析软件已经针对 macOS 数据分析实现了自动化，下面主要介绍 3 款常用的数据分析工具。

1. SafeAnalyzer

盘古石计算机取证分析系统(SafeAnalyzer)是奇安信集团自主研发的计算机综合取证分析工具，也是国内首款计算机取证分析工具(2007 年发布)。支持 Windows、Linux、macOS 三大系统，集快速提取、恢复、挖掘、分析、搜索、过滤、校验和报告等功能于一体，其操作界面如图 5-57 所示。

图 5-57 盘古石计算机取证分析系统(SafeAnalyzer)

2. Magnet AXIOM

Magnet AXIOM 是美国 Magnet 公司研发的一款综合的电子数据取证分析工具，也是全球知名的一款互联网电子证据发现、数据深度挖掘及分析的专业工具。AXIOM 基于 Magnet IEF 的分析功能打造而成，通过处理磁盘镜像及文件转储中的原始数据，提取使用痕迹的相关数据，将数字证据的搜索过程自动化。目前 Magnet AXIOM 可以对计算机、手

机、云端数据三种数据源进行提取分析，其操作界面如图 5-58 所示。

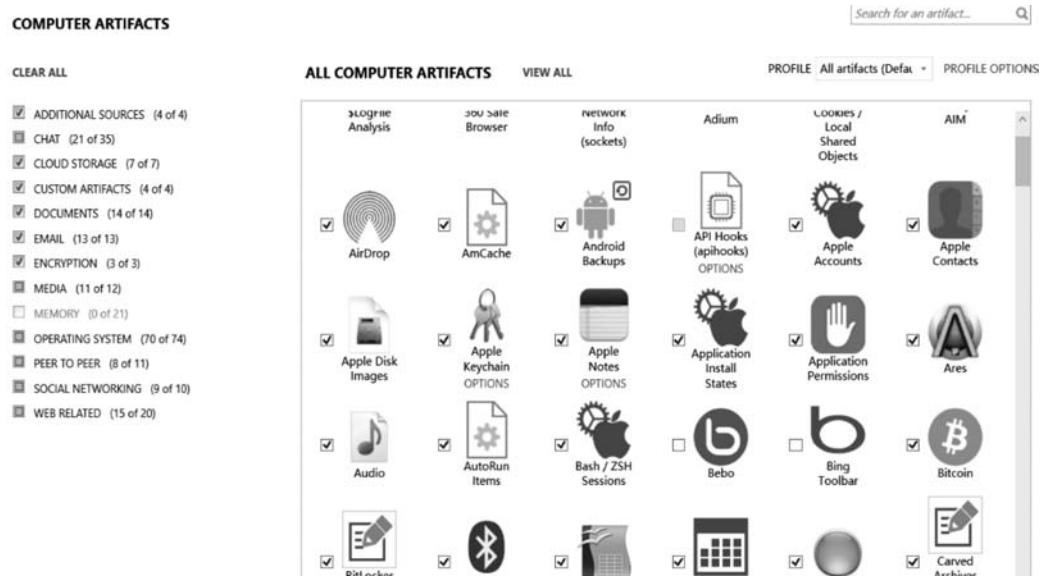


图 5-58 Magnet AXIOM

3. BlackBag Blacklight

Blacklight 是美国 BlackBag 公司（已被 Cellebrite 公司收购）研发的系列工具中针对 macOS 数据分析的专用工具，针对系统痕迹及用户痕迹的分析效果广受国内外使用者的好评，其操作界面如图 5-59 所示。

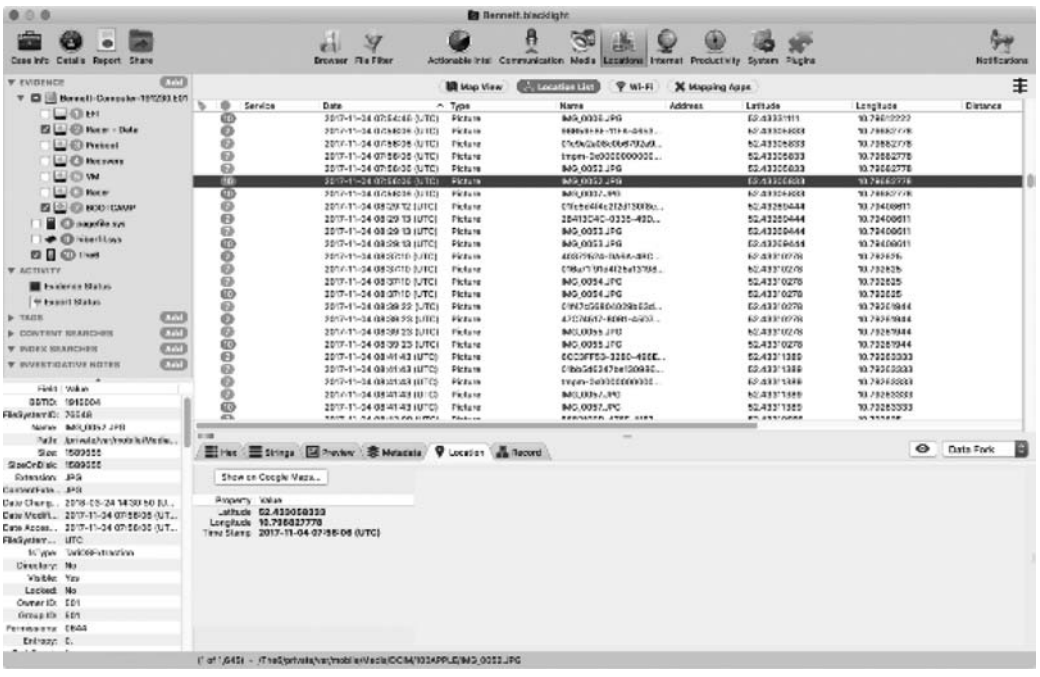


图 5-59 BlackBag Blacklight

习题

5-1 Mac 计算机操作系统更换过哪些名称？列举出具有里程碑意义的版本号。

5-2 Apple T2 安全芯片给取证带来了哪些困难？

5-3 Mac 键盘与通用 PC 键盘按键有哪些不同？有无替代键？

5-4 如何启动 macOS 的单用户模式？该模式对于取证有何意义？

5-5 macOS 中常用的时间格式有哪些？

5-6 解释 APFS 中的容器、宗卷、分区的概念。

5-7 HFS、HFS+和 APFS 文件系统有哪些区别？

5-8 文件保险箱的解密方式有哪些？

5-9 macOS 中的应用数据文件有哪些常见格式？各有什么特点？如何查看？

5-10 针对 macOS 取证前的准备工作有哪些？

5-11 目前可用于破解 macOS 用户登录口令的方法有哪些？

5-12 macOS 的易失性数据包括哪些？如何获取？

5-13 什么情况下采用拆机方式获取 Mac 数据？什么情况下采用不拆机方式？

5-14 目标磁盘模式对于电子数据取证有何意义？

5-15 macOS 中有哪几种账户？分别有什么特点？

5-16 用户主目录中存储了哪些主要数据？

5-17 钥匙串对于取证的意义有哪些？

5-18 时间机器是什么？如何获取其中的电子数据？