



南京邮电大学
Nanjing University of Posts and Telecommunications

基于应用程序的取证分析

汇报人：李昕
学号：1024041115
汇报日期：2025/5/7



1.基本概念

基于应用程序的取证分析是指针对计算机和移动设备上安装的各类应用程序及其生成的数据进行系统性收集、检验和分析，以获取与调查相关的数字证据的过程。这种分析方法专注于应用程序在系统中的行为模式、数据存储方式、日志记录以及与操作系统和其他软件的交互特征。

从技术层面看，应用软件取证分析包含三个核心维度：

静态分析

对应用程序的安装文件、配置文件、数据库等静态数据进行检验，不涉及程序运行状态的分析

通过监控应用程序在运行时的行为，包括文件访问、注册表修改、网络通信等活动

动态分析

跨平台分析

针对同一应用在不同操作系统上的数据存储和行为差异进行比较研究



1.基本概念--取证对象范围

结构化数据取证

结构化数据指具有预定义模型的数据，通常以表格、键值对或层级关系存储，可直接通过查询语言或解析工具提取。

SQLite数据库

典型应用

微信聊天记录

手机短信

JSON/XML配置文件

典型应用

应用设置（如settings.json）

移动APP的本地缓存

01

非结构化数据取证

非结构化数据无固定格式，需依赖文件签名、内容扫描或元数据分析提取信息。

日志文件

典型格式

文本日志（app.log）

二进制日志

缓存图片/视频

02

内存残留数据取证

应用运行时临时存储在内存中的数据，可能包含未持久化的敏感信息。

剪贴板内容

未保存的编辑内容

举例：

WPS文档的自动保存副本

浏览器表单输入缓存

03

网络流量取证

捕获应用与服务器之间的通信数据，用于分析API调用、文件传输等。

HTTP/HTTPS流量抓包

解密HT流量

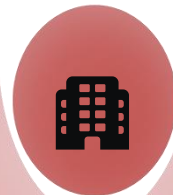
04

2.应用软件取证意义



刑事犯罪

举例：
通过微信已删除聊天记录
的SQLite碎片还原关键对话
即使嫌疑人删除聊天记录，
仍可从手机存储的
EnMicroMsg.db数据库碎片
中恢复数据
关键字段：message表的
content、createTime（、
talker



商业泄密

从ERP系统操作日志锁定
批量导出CSV的违规行为
分析ERP后台日志文件，
筛选导出CSV操作记录
关键字段：操作人ID、导
出时间、导出数据表名

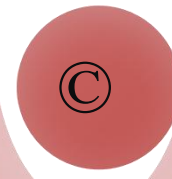
例：发现员工A在离职前
夜批量导出客户清单，直
接对应泄密事件



金融欺诈

解析支付宝trade.db中的交易
流水确认资金流向
定位数据库中的交易记录表，
提取转账金额、对方账户
订单状态

例：发现嫌疑人伪造"转账成
功"截图，实际数据库无相应
记录



知识产权侵权

通过注册表键值和激活日
志证明盗版使用

例：企业使用盗版设计软
件，注册表显示破解组织
"Team XYZ"的签名



2.应用软件取证意义

数据爆炸时代的精准定位

- 效率需求：传统全盘分析耗时呈指数增长，应用定向分析可提升效率

维度	系统取证	应用软件取证
数据来源	操作系统、内核、驱动	应用程序代码、业务逻辑、用户输入
取证目标	系统级行为（文件操作、进程）	用户级行为（聊天记录、交易、操作）
技术难点	需处理底层二进制结构	需理解业务逻辑与自定义数据格式

全盘取证与应用软件取证不是替代，而是互补

- ✓ 全盘取证提供全局数据支撑，发现潜在关联证据
- ✓ 应用取证聚焦核心数据，加速关键信息提取
- ✓ 交叉验证增强证据可信度，确保调查全面且高效



案例分析

2023年7月，红花岗公安分局成功破获一起利用虚拟货币（U币）洗钱案件，涉案金额达3800余万元。该案通过支付流水分析追踪涉案资金流向，锁定广东梅州、福建泉州等地的犯罪团伙，并查获大量作案工具。案件侦办中，警方结合电子支付数据、银行卡交易记录及虚拟货币交易链分析，精准打击洗钱犯罪，凸显了资金流追踪在新型网络犯罪侦查中的关键作用。



3. 相关证据的主要存储位置

Windows 平台

- 注册表配置
 - 存储内容：软件设置、用户偏好、安装信息
- 本地文件存储
 - 存储内容：缓存、日志、临时文件
 - 典型路径：
 - %AppData%\AppName\ (用户数据)
 - Program Files\AppName\ (程序文件)

macOS 平台

- Plist 文件 (Property List)
 - 存储内容：应用配置、用户设置
 - 典型路径：
 - ~/Library/Preferences/com.app.plist (用户级)
 - /Library/Preferences/com.app.plist (系统级)
- 应用沙盒数据
 - 存储内容：缓存、数据库、用户生成内容
 - 典型路径：
 - ~/Library/Containers/com.app/Data / (沙盒隔离)

iOS 平台

- Keychain (加密存储)
 - 存储内容：密码、Token、敏感信息
- SQLite 数据库 & Plist 文件
 - 存储内容：应用数据、缓存
 - 典型路径：
 - /var/mobile/Containers/Data/Application/AppGUID/

云端存储

- AWS S3 / 阿里云 OSS
 - 存储内容：用户行为日志、备份数据
 - 典型路径：
 - /logs/user_activity/ (需合法调取权限)



3.相关证据的主要存储位置

各个系统的共同点

用户级配置
系统级配置

结构化存储



均采用键值对或表结构。

用户数据与系统数据分离

加密与权限控制



本地加密 + 云端权限管理

多数据关联

外键关联



南京邮电大学

3.相关证据的主要存储位置

Windows 平台数据加密

核心机制:

注册表: 通常明文, 敏感数据可能用DPAPI加密。

文件存储: 日志/缓存多明文, 密码等敏感数据依赖BitLocker或CryptoAPI。

典型场景:

Chrome密码用DPAPI加密, 企业软件使用CNG保护配置。

macOS 平台数据加密

核心机制:

Keychain: 硬件级加密, 绑定生物识别。

文件/沙盒: Plist默认明文, 但Notes/iCloud数据端到端加密。

典型场景:

Safari 密码存Keychain, FileVault全盘加密保护所有文件。

Android 平台数据加密

核心机制:

SharedPreferences: 默认明文, 安全方案用

EncryptedSharedPreferences。

数据库: 可选 SQLCipher 或 Android Keystore)。

典型场景:

Signal 加密聊天记录, 微信支付依赖 TrustZone。

云端存储加密机制

核心机制:

传输层: HTTPS强制加密。

存储层:

服务端加密 (SSE): 用户管理密钥或托管密钥。

客户端加密: Cryptomator 等工具先加密再上传。

典型场景:

iCloud 部分数据端到端加密, 企业云盘用 SSE-MS。



4.相关证据的获取方法

注册表取证

01

(1) 注册表取证

工具:

RegEdit (手动查看)、
Registry Explorer (自动化解析
时间戳和隐藏键值)。

PowerShell 脚本批量导出

本地文件提取

02

工具:

磁盘镜像: FTK Imager (生成
AD1或E01镜像, 保留元数据)。

文件级提取:

SQLite数据库: DB Browser for
SQLite。

日志文件: Notepad++ (支持大
文件)、LogParser (结构化分
析)。

内存取证

03

工具:

Magnet RAM Capture (低干
扰内存转储)、Volatility
(分析内存镜像)。

网络流量捕获

04

工具:

Wireshark (全局抓包)、
Fiddler (针对应用层HTTPS
解密, 需安装根证书)。



4.相关证据的分析方法

(1) 熵值检测原理：加密数据通常具有高熵值，而普通文本或未加密文件的熵值较低。

适用场景：

快速筛选可疑文件。

结合binwalk进一步分析文件结构。

加密识别与分类

(2) 文件签名分析

原理：加密文件通常覆盖原始文件头。

工具：binwalk、HxD

(1) DPAPI解密

工具：提取MasterKey文件

(2) SQLite数据库解密

场景：微信Msg.db、Skypemain.db等。

解密工具：SQLite3命令行

解密技术

(3) 自定义加密破解

工具链：

静态分析：Ghidra/IDA Pro反编译应用，定位加密函数（如CryptEncrypt调用）。

动态分析：x64dbg下断点捕获密钥或明文。

关联分析与时间线重建

工具：

时间线工具：Plaso（log2timeline）、MFTECmd（解析\$MFT时间戳）。

可视化：Timeline Explorer（按时间排序事件）。



实验案例

不同操作系统下，应用程序（以微信为例）数据的存储位置和方式存在显著差异：

一、微信在Windows系统中的核心数据存储位置

1. 主安装目录
 2. 用户数据存储目录
 3. AppData相关目录
- 二、微信数据库文件



Msg目录下：

ChatMsg.db：存储文字聊天记录(加密)

MediaMsg.db：记录所有媒体文件的元数据

MicroMsg.db：核心数据库，包含联系人、聊天列表等

Emotion.db：表情数据

Sns.db：朋友圈数据

二、微信在ios系统中的核心数据存储位置

/var/mobile/Containers/



实验案例

\WeChat Files\[微信号]\Msg: 存储所有聊天记录的加密数据库文件

ChatMsg.db: 文字聊天记录数据库

MediaMsg.db: 媒体文件信息数据库

MicroMsg.db: 核心消息数据库

ChatMsg	2025/5/6 16:43	Data Base File	96 KB
ChatRoomUser	2025/5/6 19:00	Data Base File	928 KB
ClientGeneral	2025/5/6 19:00	Data Base File	36 KB
CustomerService	2025/5/6 16:43	Data Base File	52 KB
Emotion	2025/5/6 19:00	Data Base File	524,208 KB

\WeChat Files\[微信号]\FileStorage: 各类文件存储

\Image: 聊天图片(按年月分类)

\Video: 聊天视频

\File: 传输的文件

\Voice: 语音消息

Image	2024/10/27 14:58	文件夹
MPPageFastLoad	2024/10/27 14:58	文件夹
MsgAttach	2025/4/15 23:01	文件夹
PAG	2024/10/27 14:57	文件夹
Sns	2024/10/27 14:58	文件夹
Temp	2025/5/6 20:27	文件夹
TempFromPhone	2024/10/29 11:35	文件夹
Video	2025/5/6 17:50	文件夹



实验案例

WeChatWin.dll 核心动态链接库

WeChatWin.dll 存储了微信的核心逻辑，主要包括：

加密相关：AES密钥、RSA密钥、自定义算法

数据存储：SQLite密钥、文件缓存路径

反调试：代码混淆、动态加载、签名校验



实验案例

微信主程序（WeChat.exe） 需要关注版本信息和签名时间等信息



```
C:\Sigcheck>sigcheck.exe "D:\WeChat\WeChat.exe"
```

```
Sigcheck v2.90 - File version and signature viewer  
Copyright (C) 2004-2022 Mark Russinovich  
Sysinternals - www.sysinternals.com
```

```
d:\wechat\WeChat.exe:
```

```
Verified: Signed  
Signing date: 22:49 2025/3/5  
Publisher: Tencent Technology (Shenzhen) Company Limited  
Company: Tencent  
Description: WeChat  
Product: WeChat  
Prod version: 3.9.12.1000  
File version: 3.9.12.51  
MachineType: 64-bit
```

```
1 import pefile  
2 import datetime  
3 pe = pefile.PE(r'D:\WeChat\WeChat.exe')  
4 timestamp = pe.FILE_HEADER.TimeDateStamp  
5 print("编译时间:", datetime.datetime.utcfromtimestamp(timestamp).strftime('%Y-%m-%d %H:%M:%S'))
```

```
C:\Users\Lenovo\PycharmProjects\pythonProject\venv\Scripts\python.exe D:\pythonProject8\a.py  
D:\pythonProject8\a.py:5: DeprecationWarning: datetime.datetime.utcfromtimestamp() is deprecated and scheduled  
for removal in a future version. Use datetime.datetime.fromtimestamp() instead.  
print("编译时间:", datetime.datetime.utcfromtimestamp(timestamp).strftime('%Y-%m-%d %H:%M:%S'))  
编译时间: 2025-03-05 14:33:16
```



实验案例

1. 获取加密的聊天数据库

进入 Msg 文件夹，找到以下文件：

- ChatMsg.db → 最新聊天记录数据库。

```
D:\WeChat Files\wxid_bdpmrwcwp2e722\Msg>sqlite3 ChatMsg.db
SQLite version 3.50.1 2025-06-06 14:52:32
Enter ".help" for usage hints.
sqlite> .tables
Error: file is not a database
sqlite>
```

2. 解密数据库（需获取密钥）





微信的数据库使用 SQLCipher 加密，密钥存储在 Config 文件夹的 config.data 文件中。

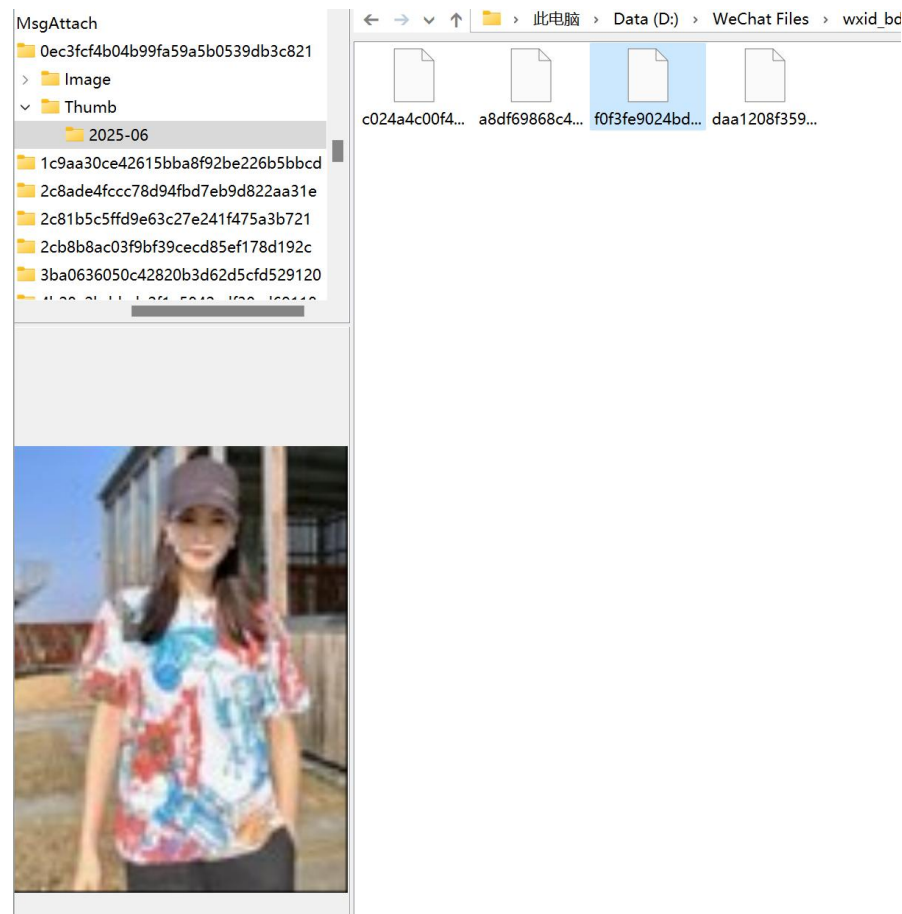
- 方法一：使用 WeChatViewer 工具自动提取密钥
 - 工具会自动提取密钥并解密 MSG.db，生成明文 HTML 或 TXT 文件。
- 方法二：手动提取密钥
- 若工具失效，可手动从 config.data 中提取密钥，但过程复杂。



实验案例--提取聊天记录

使用WeChatViewer工具可以解密.dat文件

	a8df69868c46fc26d02a6cd66872d723...	2025/6/7 17:03	DAT 文件	4 KB
	c024a4c00f4363873af5547ae06348ac_...	2025/6/7 17:10	DAT 文件	3 KB
	daa1208f3598e2b3071ad345356f8ca4...	2025/6/7 16:55	DAT 文件	4 KB
	f0f3fe9024bdaf5ad9035c39395f70ec_t...	2025/6/7 17:01	DAT 文件	6 KB



🔑 WxDatViewer 的用途

解密微信 .dat 文件（图片、视频、表情包等）

✗ 不支持：

直接查看聊天记录（文字、语音、转账等）



南京邮电大学

实验案例

1. 检查数字签名 & 版本一致性（确保DLL未被篡改）

```
C:\Sigcheck>sigcheck.exe "D:\WeChat\[3.9.12.51]\WeChatWin.dll
```

```
Sigcheck v2.90 - File version and signature viewer  
Copyright (C) 2004-2022 Mark Russinovich  
Sysinternals - www.sysinternals.com
```

```
d:\wechat\[3.9.12.51]\WeChatWin.dll:
```

Verified:	Signed
Signing date:	22:49 2025/3/5
Publisher:	Tencent Technology (Shenzhen) Company Limited
Company:	Tencent
Description:	WeChat
Product:	WECHAT
Prod version:	3.9.12.1000
File version:	3.9.12.51
MachineType:	64-bit

```
1 import pefile  
2 import datetime  
3 pe = pefile.PE(r'D:\WeChat\[3.9.12.51]\WeChatWin.dll')  
4 timestamp = pe.FILE_HEADER.TimeDateStamp  
5 print("编译时间:", datetime.datetime.utcfromtimestamp(timestamp).strftime('%Y-%m-%d %H:%M:%S'))
```

```
C:\Users\Lenovo\PycharmProjects\pythonProject\venv\Scripts\python.exe D:\pythonProject8\a.py  
D:\pythonProject8\a.py:5: DeprecationWarning: datetime.datetime.utcfromtimestamp() is deprecated and scheduled  
for removal in a future version. Use timezone-aware objects to represent datetimes in UTC: datetime.datetime.from  
timestamp(timestamp, datetime.UTC).  
print("编译时间:", datetime.datetime.utcfromtimestamp(timestamp).strftime('%Y-%m-%d %H:%M:%S'))
```

```
编译时间: 2025-03-05 14:47:16
```

WeChatWin.dll 属性

常规 数字签名 安全 详细信息 以前的版本

 WeChatWin.dll

文件类型: 应用程序扩展 (.dll)

打开方式: 未知应用程序 [更改\(C\)...](#)

位置: D:\WeChat\[3.9.12.51]

大小: 96.6 MB (101,355,616 字节)

占用空间: 96.6 MB (101,359,616 字节)

创建时间: 2025年3月7日, 18:02:50

修改时间: 2025年3月24日, 22:50:57

访问时间: 2025年5月5日, 20:20:39

















签名时间应 \geq 编译时间, \leq 修改时间








南京邮电大学

实验案例

2. 静态分析密钥相关函数（IDA Pro）

 mars::cronet::CancelCronetTask(std::basic_string<char,...
 mars::cronet::StartCronetRequest(CronetTaskProfile::Cr...
 mars::cronet::StartCronetUploadRequest(CronetTaskPr...
 mars::cronet::UnInit(void)
 GetHandleVerifier
 ShouldDoUpdate
 **SignWith3Des**
 StartWechat
 TlsGetData
 TlsStoreData
 VerifyDllReport
 WechatLog
 TlsCallback_0
 TlsCallback_1
 TlsCallback_2
 TlsCallback_3

	.rdata:000000...	00000016	C	pbeWithSHA1AndDES-CBC
	.rdata:000000...	00000021	C	pbeWithSHA1And3-KeyTripleDES-CBC
	.rdata:000000...	00000021	C	pbeWithSHA1And2-KeyTripleDES-CBC
	.rdata:000000...	00000015	C	pbeWithMD5AndDES-CBC
	.rdata:000000...	00000015	C	pbeWithMD2AndDES-CBC

表明微信可能使用 PBE + 3DES 组合加密（如用户数据或密钥派生）。
密钥派生方式：通过用户密码（或设备指纹）生成加密密钥。



实验案例

- 3、逆向 pbeWithSHA1And3-KeyTripleDES-CBC 的实现，获取密钥派生方式
- 4、动态调试捕获密钥（x64dbg）/从内存提取密钥（ Volatility ）/使用专业工具如GoWxDump自动获取
- 5、使用 SQLite 工具执行查询语句，完整导出聊天记录

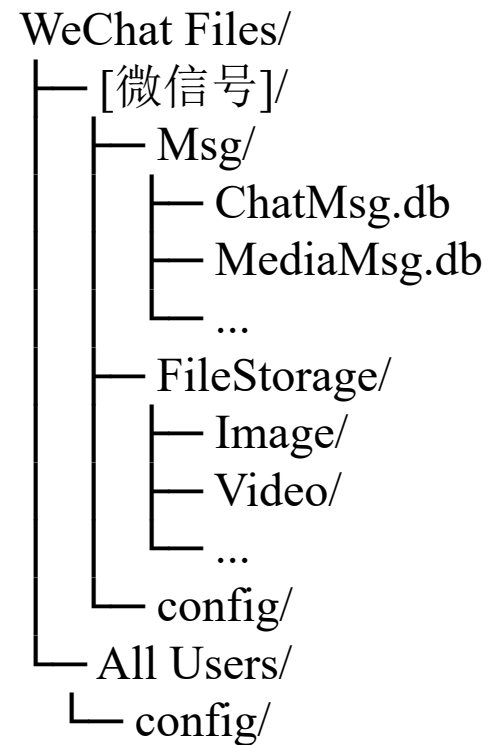


微信数据取证流程

数据库文件提取具体流程

- ① 创建证据镜像 (FTK Imager)
- ② 提取数据库文件
 - 完整目录结构复制
 - 保持原始路径关系
- ③ 证据固定
 - 计算哈希
 - 数字签名
 - 时间戳认证

获取密钥 → 数据库还原





谢谢!



计算机学院 软件学院
网络空间安全学院
School of Computer Science

