IMPLEMENTATION OF APPLICATIONS TO IMPROVE IPHONE FORENSIC ANALYSIS AND INTEGRITY OF EVIDENCE

GRADUATE THESIS REPORT

Submitted to the Faculty of
The Department of Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, Texas

In Partial Fulfillment of Requirements for the Degree of
Masters of Science in Computer Science

By

Rajavardhan Mallepally
Summer 2011

Committee Members

Dr. Mario Garcia
Committee Chairperson

Dr. Ahmed Mahdy
Committee Member

Dr. Long-zhuang Li
Committee Member
Abstract

iPhone release witnessed a significant impact on Forensic research community. iPhone Forensic research in many ways paved contributed to Smartphone research. Forensic investigators facing many challenges while analyzing the evidence on iPhone because of factors like frequent version upgrades, file system on iPhone and lack of sophisticated tools or methods for iPhone forensic analysis. On the other hand increasing number of applications for iPhone is becoming the source for bad guys to escape from the forensic investigation. The purpose of this research is to study and analyze evidence on iPhone. The research started with study on iPhone architecture and file system. This research presents analysis on iPhone forensic tools and techniques. This research proposes a technique to recover the evidence on iPhone irrespective of increasing number of applications and an application to provide data integrity check for the iPhone forensic tools and techniques. Then research turned into bad guys perspective to foreshadow possible ways of escaping from forensic investigation there by proposes two anti-forensic applications; to delete the evidence on iPhone and to hide the secret text in password key chain.
TABLE OF CONTENT

ABSTRACT .......................................................................................................................... II

TABLE OF CONTENT ............................................................................................................. III

LIST OF TABLES .................................................................................................................... V

LIST OF TABLES ..................................................................................................................... VI

1) INTRODUCTION ................................................................................................................. 1

2) IPHONE ARCHITECTURE ............................................................................................... 3

   2.1) IPHONE KERNEL ......................................................................................................... 5

   2.1) SYSTEM RAM AND FLASH RAM ............................................................................... 6

3) FILE SYSTEM ON IPHONE ............................................................................................ 7

   3.1) HFS PLUS FILE SYSTEM .......................................................................................... 7

   3.2) HFS PLUS VOLUME .................................................................................................. 8

   3.3) METADATA .................................................................................................................. 10

4) EVIDENCE ON IPHONE .................................................................................................. 11

5) CHALLENGES INVOLVED IN IPHONE FORENSICS .................................................. 13

6) IPHONE FORENSIC TOOLS AND METHODS ............................................................ 15

   6.1) EXISTING TOOLS AND METHODS .......................................................................... 15

       6.1.1) Wolf ...................................................................................................................... 15

       6.1.2) UFED .................................................................................................................... 16

       6.1.3) XRY ....................................................................................................................... 16

       6.1.4) Secure View 3 ..................................................................................................... 16

       6.1.5) CellIDEK ................................................................................................................ 16

       6.1.6) Oxygen-Forensic Suite ....................................................................................... 16

       6.1.7) Device Seizure .................................................................................................... 16

       6.1.8) Zdziarski Technique ........................................................................................... 16

   6.2) NEW TECHNIQUE ...................................................................................................... 21

7) ANTI-FORENSIC TECHNIQUES ...................................................................................... 25
7.1) Hiding Data on Jailbroken iPhone ................................................................. 26
   7.1.1) Method 1 ................................................................................................. 26
   7.1.2) Method 2 ................................................................................................. 27
   7.1.3) Method 3 ................................................................................................. 27
7.2) Hiding Data on Non-Jailbroken iPhone ......................................................... 27
7.3) Anti-Forensic Applications Proposed ............................................................ 28
   7.3.1) HideSecret ............................................................................................. 28
   7.3.2) KillEve ........................................................................................................ 38
8) Data Integrity Check Solution .............................................................................. 41
9) Conclusion and Future Work ............................................................................ 43
REFERENCES: ........................................................................................................ 45
List of Tables

3.1) METADATA ZONE ...................................................................................................................... 11
6.1) COMPARISON OF IPHONE FORENSIC TOOLS ...................................................................... 17
6.2) DIRECTORY LIST ON IPHONE USER DATA PARTITION .............................................................. 21
7.1) ADDING SECRET TEXT TO KEY CHAIN .................................................................................... 29
7.2) SEARCHING FOR THE SECRET TEXT IN THE DIRECTORY ...................................................... 30
7.3) CODE BLOCK TO SEND PERIODIC MEMORY ALLOCATION REQUESTS ................................... 40
8.1) CODE BLOCK TO HASH THE CONTENT IN PLIST ...................................................................... 42
# List of Figures

2.1) IPHONE ARCHITECTURE ........................................................................................................... 3

2.2) SYSTEM RAM AND FLASH RAM USAGE ON SMART PHONES ........................................ 7

3.1) HFS PLUS VOLUME .................................................................................................................. 9

6.1) TEXTFREE.PLIST IN XML FORMAT ....................................................................................... 23

6.2) WHERE.PLIST IN XML FORMAT ......................................................................................... 24

6.3) SIMCAM.PLIST IN XML FORMAT ........................................................................................ 25

7.1) ADDING SECRET TEXT TO KEY CHAIN .................................................................................. 31

7.2) RETRIEVING SECRET TEXT FROM KEY CHAIN ................................................................. 33

7.3) DELETING SECRET MESSAGE FROM KEY CHAIN ............................................................... 35

7.4) ERROR MESSAGE FOR WRONG ACCESS ID OR PASSWORD ............................................... 36

7.5) DELETE FAIL ON WRONG ACCESS ID OR PASSWORD ..................................................... 37

7.6) XML FORMAT OF HIDESECRET.PLIST ................................................................................. 38

7.7) KILLEVE APPLICATION ON IPAD SIMULATOR .................................................................. 40

8.1) HASHING APPLICATION FOR DATA INTEGRITY CHECK ..................................................... 42
1) Introduction

iPhone is the revolutionary product of the year 2007 by Apple Inc. Apple iPhone is the solution to carrying the three devices iPod, mobile phone and Internet in pocket. This combination of three different devices made iPhone such a popular; that from note one of the user who bought his first iPhone in the year 2007 when the first iPhone released; “I’ve actually been up for about 72 hours, Not because of the iPhone, but because I had to study for a test. I just wasn’t going to let sleeplessness deter me from being the first in line.” [8] Such an impact iPhone created in the modern digital era of the world. What makes the iPhone such an attracting device? On reviewing the features of the iPhone, it is the clear way to get such an impression. The alluring features of iPhone replace the use of many electronic devices, like PDAs, palm tops; besides acting as mobile device iPhone includes many features, works as music player, Internet browser and many more applications. These extensive features were built on Apple’s Leopard platform, which is based UNIX. [11] [5] The exiting features available iPhone undoubtedly attracts many users to its user pool. As there is increase growth in number of users, more the information shared on iPhone, through different forms of communication. For example the communication may be a phone call, text message, email message etc. Along with innocent user nefarious guys also were attracted to the iPhone to gain access to the private information of other iPhone users [4]. iPhone in the hands of bad guys is going to be a threat for common users. Bad guys can make use of iPhone in different perspectives. Some of the examples are pornography, identity theft, harassing people, and many more. It is going to be challenging part for the forensic investigator recover the evidence from any iPhone, which involved in illegal activities.
Here iPhone Forensics plays key role in presenting the evidence in the court of law. This process of presenting evidence involves different phases like, collecting the hardware device (iPhone involved in crime scene), securing the device, analyzing the content on the device, and finally reporting. All these phases need to be documented for future reference and is called chain of custody. One of the major risks associated with the investigation process is data contamination. This research on iPhone forensic analysis focuses on analyzing the data on iPhone and extracting the evidence from the iPhone without any alterations to the evidence. Because of the file system used by the Apple Inc., the architecture used develop the iPhone and operating system (iOS) that used on iPhone makes it difficult to extract the information from the iPhone with regular computer forensic tools. The goal of this research is to study the iPhone architecture and the file system of iPhone, information stored on iPhone and its significance in forensic analysis, methods used for forensic analysis and an approach to extract the evidence from iPhone. Along with the analysis of forensic recovery of evidence from iPhone, this research is also focused on Anti-Forensics techniques, to foreshadow the possible ways a bad guy can escape from the forensic investigation. The following sections present the detailed version of the research. Section 2 describes the architecture of the iPhone, Section 3 presents file system on iPhone, Section 4 discusses the evidence that can be present on iPhone followed by the challenges involved in extraction of evidence from iPhone in Section 5, Section 6 is the summery of the capabilities and functionalities of the existing forensic tools and methods, and also introduces a new techniques for iPhone forensic investigation. Section 7 presents different anti-forensic techniques exists for hiding the evidence from forensic investigation and also introduces two new methods for killing the
evidence on iPhone and hiding the evidence on iPhone, Section 8 provides a solution to the data integrity, which the presents tools lacks in.

2) iPhone Architecture

iPhone architecture may not sounds worthy in terms of forensic investigation but helps in understanding the mechanism of iPhone and helps in analyzing the information accessed through iPhone. Apple Inc. developed iPhone in seven layers as shown in figure 2.1.

![iPhone Architecture Diagram](image)

Even though understanding of the hardware functionality much involves in analyzing the evidence it is important to know the hardware changes that effects the access to the device. Here in case of iPhone, architecture can be analyzed in two perspectives; one from application layer to hardware layer and the second is from hardware layer to application layer.

*Application Layer to Hardware Layer:*
This part describes a user call to the application and processing the user request. The application layer is the interface between the user and the iPhone hardware, user request from application layer processed through all the layers down the hardware layer. Through user request application layer makes an API or framework call. The Framework/API layer makes the necessary translation and makes a system call to Objective C runtime, which makes a call to dynamically running C library. Then the C library converts the call into assembly level system call, which can be processed by iPhone OS kernel. The iPhone OS kernel call upon the appropriate service depending the call. For example the call can be display drivers, storage drivers etc. These calls are made to the appropriate registers to words the user request. Now it’s the hardware, which makes user request to get finished.

*Hardware Layer to Application Layer:*

The hardware units called gyros detect the movement and indicate the change by raising or modifying the registers. Firmware plays a key role in detecting the change and notices the state of the iPhone and generates an interrupt call to the processor through system bus. The ARM processor built in upon receiving the interrupt call, sends an in-memory interrupt service routine (ISR), which the iPhone OS setup during the driver installation. The iPhone OS sends the signal to the active application in UNIX-style based the signal received from the ISR by running the subroutine function located in iPhone OS code. The rest of the functionality is carried out according to specification of Objective C or C runtimes. The application responds to the framework call bade by the Objective C. [5] On continuing with architecture of iPhone, iPhone kernel and flash RAM plays very important role.
2.1) iPhone Kernel:

Apple used the same I/O kit architecture for iPhone’ Mach kernel that used for Mac OS X. This gives the flexibility in reusing the code that proved to work properly. By using these components Apple Inc. able to design strong base for iPhone. To support low-level hardware components the I/) kit Mach uses kexts or kernel extensions. When some thing crashes out side the kernel, it is to restore the OS by simply restarting. If the crash is inside the kernel the system cannot be restored back, and makes the system unusable. That’s the reason Apple Inc. does not allow third party applications to be installed on iPhone.

Apple supports only two sets of kexts, first one refers to USB input and the second one refers to Virtual Private Networks (VPN). The USB input refers to mainly user interface; multi-touch screen is used for user interface in iPhone.

- IOUSBDeviceFamily
- IOHIDFamily
- AppleMultitouchSPI

VPN used for securely tunnel the encrypted traffic over public network. The protocols used:

- L2Tp.ppp
- PPTP.ppp

iPhone kernel is slightly different from Mac’s kernel even though Apple used the same components used Mac’s kernel to build iPhone’s kernel. The differences lie in supporting the low-level hardware components. For example Mac’s kernel have the extensions to iSight camera, audio and video hardware etc., where iPhone does not
support any of this kind. The extensive feature of the Mac and I/O kit, can accommodate extensions to wide range of hardware units without rebuilding the “mobile OS kernel”. Apple iPhone got the efficient power saving mechanism; the device goes into sleep mode or standby mode when not in use. [9] System/Library/Extensions is the location where the kernel cache files stored. [6]

2.2) System RAM and Flash RAM:
The substantial difference between the iPhone and other smart phones is that iPhone does not use any physical drives; instead it uses Flash RAM. Flash RAM acts as a DMG file on Mac OS. On mounting this DMG file acts as physical drive. The speed of the device is increased because of Flash RAM. The restoration process in iPhone is similar to iPod, which involves mounting the clean DMG file on to iPhone. IPhone is installed with 4096 MB (4GB) or 8192 MB (8GB) of Flash RAM, in which 500 TO 700 MB is, reserved for iPhone OS and its applications. IPhone also accommodated 128 MB of system RAM, which can be used by running applications.

Figure 2.2 shows the difference between the iPhone and other smart phone in using the System RAM and Flash RAM.
3) **File System on iPhone**

iPhone is partitioned into two sectors; first partition of the iPhone is MBR (Master Boot Record), and the second sector or second partition is used to store the user data. The second partition is the good source for evidentiary files such as user notes, movies, pictures, SMS, contacts and etc. [24] Even though the partitions are different they use the same file system, called HFSX. HFSX is the extension for HFS Plus file system. HFSX and HFS Plus are the versions of Hierarchical File System (HFS). There no much difference between HFSX and HFS Plus file systems, except some minor differences in the wrapper. [10]

3.1) **HFS Plus File System:**

To overcome some of the limits in HFS file system Apple introduced HFS Plus file system. One of the major advantage with HFS Plus file system is it supports larger file sizes; supports block address of length 32-bit instead 16-bit supported by
HFS. [20] HFS Plus file system uses journaling as the default setting; Logging of every transaction to the disk is called journaling, such as unclean shut down. The concept journaling helps to prevent file system corruption.

In the HFS Plus file system the storage medium is represented as volumes. A single physical disk may include a single volume or multiple volumes through disk partitioning. One single logical volume is capable of including multiple physical disks. HFS Plus file system treats a single logical volume as a single logical entry that can be accessed at random. [24] The key characteristic features on which HFS Plus file system designed are:

- Efficient use of disk space; uses 32-bit block address space, which gives the scope to efficient use disk space with small size of blocks. With the small size of block size the average wasted space can be reduced.
- File names; uses 255 Unicode characters to name files on storage medium, which makes it easier to have descriptive names. File name can occupy up to 512 bytes in HFS Plus file system to support B-tree structure; B-tree index node must store at least two keys (plus pointers and done descriptor).
- HFS Plus file system holds another B-tree to store additional information for a file or directory.
- Booting on Other Operating Systems; a special startup file defined and useful for the systems that does not have HFS or HFS Plus support in ROM. [20]

3.2) HFS Plus volume:

The volumes of data are allocated on the storage medium. These data allocation units are also called as allocation blocks. Size of allocation blocks will be multiples of
storage medium sectors. HFS Plus file system supports Unicode format file names (file name accepts up to 255 characters). Maximum file size can be 263 bytes.

HFS Plus uses a number of interrelated structures to manage the organization of data on the volume. These structures include:

- Volume header
- Allocation file
- Extents overflow file
- Catalog file
- Attributes file
- Startup file

Figure 3.1 gives the overview of HFS plus volume. The first 1024 bytes and the last 512 bytes of the volume are reserved. The first 1024 bytes of volume are used to store
boot load information. Volume header contains information about the volume as a whole, including the location of other key structures in the volume. A copy of the volume header and the alternate volume header are stored at the starting 1024 bytes and before the end of the volume. Allocation file provides a bitmap of used and unused blocks in the volume. Extents overflow file contains pointers when file size exceeds eight blocks, in this case that helps in locating the actual data when referred. Catalog file maintains information about the hierarchy of files and folders on a volume. A catalog file is organized as a B-tree file. The B-tree file structure allows fast searches on arbitrary keys. The attribute file contains custom file attributes. The startup file is a special file intended to hold information needed when booting a system that does not have built-in (ROM) support for HFS Plus.

Collection of allocation blocks forms forks, used to store both special and user files. Space for the forks is reserved in clumps. The size of the clump is multiple of allocation block size. User files can have multiple forks. At least two forks are assigned to each file namely resource fork and data fork. Resource fork contains information such as icon, and other metadata. Actual data stored in data forks.

### 3.3) Metadata:

In order to reduce the seek time for typical accesses on files, in latest Mac OS X version introduced metadata zone. The metadata zone contains both the frequently used small files near each other on disk and the volume metadata. The metadata includes allocation bitmap file, extents overflow file, the catalog file, and the journal file. This volume metadata provides the flexibility in file system managing the
contents of the volume. The metadata zone is established when the volume is mounted. The size of the zone is based on the list shown in table 3.1.

Table 3.1: Meta Data Zone [1]

<table>
<thead>
<tr>
<th>Item</th>
<th>Contribution to the Meta Data Zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocation Bitmap File</td>
<td>Physical size (total Blocks times the volume's allocation block size) of the allocation bitmap file.</td>
</tr>
<tr>
<td>Extents Overflow File</td>
<td>4MB, plus 4MB per 100GB (up to 128MB maximum)</td>
</tr>
<tr>
<td>Journal File</td>
<td>8MB, plus 8MB per 100GB (up to 512MB maximum)</td>
</tr>
<tr>
<td>Catalog File</td>
<td>10 bytes per KB (1GB minimum)</td>
</tr>
<tr>
<td>Hot Files</td>
<td>5 bytes per KB (10MB minimum; 512MB maximum)</td>
</tr>
</tbody>
</table>

This policy is only applied to volumes whose size is at least 10GB, and which have journaling enabled. [24] HFSX is an extension to the HFS Plus, which allows some additional features that are incompatible with HFS Plus file system. Case sensitive files names such feature defined as of now. HFSX version number and signature are saved in volume header. If any new feature is added the version number will be changed. [1]

4) Evidence on iPhone

iPhone memory is divided into two partitions. First partition is about 300-500MB and reserved for storing the Operating System files. This space strictly restricted for users, does not store any user data. Second partition is rest of the iPhone memory, which can be accessed by user to store files and is called as user data partition. User data partition is the rich source of evidence, which stores all the activities of user of the iPhone. User data partition contains the following information:

- **Address Book**: Address book on iPhone contains the contact details like phone numbers, email, and mailing address.
• **Calendar:** Calendar is the place that stores the events specific to date, month and year. User schedules and appointments can be stored in calendar.

• **Call History:** Stores the call made, received and dialed through the iPhone used. Call history can store up to 100 entries in it. But only 60 entries are visible to iPhone users. Call History deletion process is non atomic, that is no single entry cannot be deleted individually. Clear call list option clears all the 60 entries in Call History.

• **SMS:** Short message service on iPhone stores the messages in grouped according to contact numbers. Individual messages cannot be deleted per every contact rather clear operation deletes all the messages associated with that contact number.

• **Voicemail:** iPhone stores voicemails up to 20 entries.

• **Mail:** Mail application can be configured to any number of emails that user desired. Undo operation is not supported with email deletion on iPhone.

• **Safari:** Safari is the default Internet browser on iPhone. Safari is the access point to web world.

• **Notes:** Notes is the application that can stores user notes and lists with date and time.

• **Utilities:** Default utilities on iPhone contain clock, calculator, voice memos and compass applications.

• **Stocks:** Is the application that lists the stocks and rates, it is a live application which gives the current list rates on the stock market.
• *YouTube*: YouTube application is the access point to the YouTube videos on the web.

• *Photos*: Photos is the place where all pictures are stored. These pictures are imported from different applications like camera, email, and third party applications. It even stores the video files.

• *Weather*: Weather channel application gives the current weather reports to iPhone user.

• *Cookies*: Cookies store the web related information especially web information on Safari and YouTube.

• *App Store*: Application store on iPhone is the access point to all the apps on App Store. There were around 200,000 applications for iPhone [iPad 2 Keynote]

• *Maps*: “Google maps” is a default application on iPhone. This application is capable of search and track the route between two places.

• *Settings*: iPhone allows custom settings. The user can apply his/her own settings to different applications. For example user can block own identity while using the phone.

• Along with default applications iPhone supports third party applications. These applications can be downloaded from App Store.

5) **Challenges involved in iPhone Forensic Analysis**

*Challenges that can be faced by forensic investigator while investigating the iPhone:*

• Regular forensic tools cannot be used to investigate the iPhone, because of the file system and operating system present on iPhone.
• Automatic mounting process when the iPhone connected with the forensic workstation can alter the evidence on iPhone.

• Flash RAM used in constructing the iPhone memory unit makes it difficult in creating the iPhone forensic image.

• Because of Flash memory iPhone cannot be detected as physical device on forensic workstation, which restricts the access to the information on iPhone.

• Upgrades in the iPhone firmware versions by Apple, Inc. makes it difficult for the existing forensic tools and methods to cope up with new firmware versions. The tools and methods need to be updated as per the version upgrade of iPhone firmware.

• Increasing number of third party applications increasing the complexity in forensic investigation. Most of the existing tools and methods are having limited capacity of extracting information from third party applications. More clearly discussed in Section 6.

• Security implications by third party application developers decreasing the readability of information accessed through their application. For example encryption standards applied on the information makes the information unreadable by encrypting it.

• Due to lack of data integrity check in the present tools and methods, evidence does not meet the requirements of court of law.
6) iPhone Forensic Tools and Methods

6.1) Existing Tools and Methods

6.1.1) Wolf: Wolf is iPhone forensic tool that works only on Mac designed by Sixth Legion. Wolf can recover the evidence from 2G and 3G only with any firmware version. This tool cannot recover any deleted data but able to bypass the passcode on iPhone without Jailbreaking [3]. The information that can be extracted from iPhone (2G and 3G) with Wolf is call history, device information, SMS, music/video, Internet information, contacts and photos.

6.1.2) UFED: UFED developed by Cellebrite, comes in three versions, UFED Physical Pro, UFED Ruggedized and UFED Standard. This tool not only works with iPhone but also with many other mobile devices including handheld PDAs. This tool only capable of retrieving the standard information like contacts, SMS, music/videos, call logs, browser history, pictures, ringtones and basic device information. This tool generated the reports with user-friendly interfaces in HTML, XSL, CSV and XML formats [7].

6.1.3) XRY: Micro Systemation developed XRY to investigate the mobile devices. This tool only works on windows operating system. This tool supports in recovering the standard evidence like call logs, text messages, contacts, device information etc. This tool comes with SIM cloner, which can be used to make exact copy of SIM card from the mobile device [17].

6.1.4) Secure View 3: Secure View 3 is the upgrade version to the Secure View 2 developed by Susteen for mobile forensics. This tool is capable of retrieving Facebook, Twitter and MySpace information along with standard mobile information.
This tool supports backup service to prevent loss of data. This tools also supports time line analysis and web history analysis [26].

**6.1.5) CellDEK:** A portable forensic investigating tool can be connected to any forensic workstation, developed by Logicube. This tool provides the secure erase option to prepare the forensic workstation. CellDEK can analyze different mobile device including iPhones and PDAs. Automatic information extraction will be performed as soon as the evidence device connected with CellDEK. CellDEK able to recover the information like, call logs, SMS, handset information, SIM card details, Calendar events, To do lists, and etc [13].

**6.1.6) Oxygen-forensic suite:** Oxygen forensic suite is specifically designed for mobile forensic investigation with capability of retrieving the information not only limited to standard mobile information but beyond it can retrieve information resides in Web history, GPRS, Wi-Fi information etc. This tool also provides data integrity check [21].

**6.1.7) Device Seizure:** Device Seizure developed by Paraben Corporation to investigate mobile device, GPS devices and PDAs. Device Seizure comes as both software and hardware, which works only with windows operating systems. This tool is capable of recovering the listed information; call history, SMS, Email, web history, Calendar, to do list, scheduler etc. [22].

**6.1.8) Zdziarski technique:** One of the most powerful techniques among the iPhone forensic tools and techniques exists. This technique developed by Janothan Zdziarski specially to investigate iPhone. This method involves Jailbreaking of iPhone; Jailbreaking is the concept of removing the Apple’s restrictions on third party
applications; Apple allows applications to install only that were there in App Store. Jailbreaking allows application not only from App Store but also from other sources. With Jailbreaking and Cydia packages (incudes forensic tool kit) installed on suspect’s iPhone gives the complete access to information on suspect’s device through SSH connection between suspect’s iPhone and the forensic workstation [12].

Table 6.1 gives the details and comparison reports of some of the tools for iPhone forensic investigation.

Table 6.1: Comparison of iPhone Forensic tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Vendor</th>
<th>Data Acquisition</th>
<th>Additional features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wolf</td>
<td>Sixth Legion</td>
<td>Contacts (modified, accessed, changed), basic phone information.</td>
<td>Fast, direct access to SQLite files, which increases the percent of retrieving, deleted files.</td>
</tr>
<tr>
<td>UFED</td>
<td>Cellebrite</td>
<td>Contacts, SMS text messages, deleted text messages (SIM/USIM), call history (received, dialed, missed), audio files, video files, pictures, ringtones and phone details.</td>
<td>Supports USB and Bluetooth. Works with majority of carriers. Reports in different formats (HTML, XML, XLS, CSV)</td>
</tr>
<tr>
<td>CellDEK</td>
<td>Logicube</td>
<td>Call logs, phonebook, SMS messages, deleted SMS messages from SIM, Multimedia (MMS) messages, calendar, memos, to – do lists,</td>
<td>Uses MD5 hash algorithm to ensure data integrity. USB connection &amp; flash cards for data uploads. Data format: XML and</td>
</tr>
<tr>
<td>Tool</td>
<td>Features</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MacLock Pick</td>
<td>Pictures, video, audio and other files. Data can be extracted from CDMA,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GSM, TDMA, and iDen cell phone technologies, and time stamped documentation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MacForensic sLab</td>
<td>Call log with duration and time. Text messages (content, date and time and parties involved). Deleted data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data integrity testing. Built in searching and bookmarking. Data carving.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HTML &amp; Text reporting. Viewing acquired data with external viewer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XRY</td>
<td>Successful in retrieving SMS (allocated), contacts, Calls, Calendar, Notes, MMS, Pictures, Videos, Audio, Documents, Files and Log. For SMS, Calls, Voicemail and Notes the toll equipped with deleted directory.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This tool supports iPhone cable connection only, does not support Bluetooth.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This tool can be used to get complete summary of the iPhone status. This tool can detect the jailbroken iPhones. The only tool that can recover “Notes” from iPhone.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mac Marshal</td>
<td>Disk partition images in multiple formats, configuration files, log.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATC-NY</td>
<td>Spotlight search, supports dd, EnCase, FTK, AFF and Apple disk images.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zdziarski technique</td>
<td>No vendor</td>
<td>Can retrieve the all the call logs, SMS, contacts, email, calendar, notes, pictures, web history, cookies, iPhone applications, office documents, and HTML pages. Also can retrieve information related to GPS, Bluetooth and music.</td>
<td>Physical access to the iPhone memory</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
</tbody>
</table>

Table 6.1 clearly describes what each forensic tool can retrieve from the iPhone, and additional features also described in the table. One of the fastest data acquisition tool among all is Wolf by Sixth legion. This tool can extract the contacts from iPhone, even though the contacts are modified or accessed or changed the tool can extract the time of action. Because of direct access to the SQLite files increase in chance of retrieving deleted information [3]. UFED by Cellebrite is one of the popular tools among all. This tool can perform analysis on contacts, SMS, deleted text messages, call history including received, dialed and missed, music files, video files, pictures, ringtones and phone details. UFED can be differentiated from other tools by its additional features such as supporting both USB and Bluetooth connectivity and the tool generates reports in different formats such as HTML, XML, CSV, and XLS. Significant advantage of the tool is that, the tool works with most of major carriers.
CellDEK is the most reliable tool developed by Logicube, which uses MD5 hash algorithm to check the data integrity. Unlike other tools, CellDEK supports different types of flash cards along with USB for connectivity. CellDEK can successfully extract evidence from the areas like Call logs, phonebook, SMS messages, deleted SMS messages from SIM, Multimedia (MMS) messages, calendar, memos, to do lists, pictures, video, audio and other files. MacForensics Lab developed a forensic tool named Maclock Pick, which can be differentiated by its extensive features like built in searching, data carving, reports from the tool can be accessed using external viewers. One more extraction feature that available with MacLock Pick is that it can report the call duration and time of call. XRY tool that can be used retrieve the information from the areas such SMS (allocated), contacts, Calls, Calendar, Notes, MMS, Pictures, Videos, Audio, Documents, Files and Log in iPhone. This tool exhibits features like detection of jailbroken iPhones and gives the complete report on status of the iPhone that connected with the tool. The only tool that can recover deleted Notes from iPhone is XRY. MacMarshal is one of the finest tools, which supports different formats of evidence files for investigation unlike other forensic tools for iPhone forensic investigation. It supports dd, EnCase, FTK, AFF and Apple disk images. This tool was developed by ATC-NY, can extract the information about the disk partitioning in multiple formats, can extract the configuration information, data from OS X common applications like mail, safari, iChat and address book. One of the most powerful techniques that can be used to investigate is Zdziarski technique. This technique uses concept of jailbreaking, which intern provides the access to the iPhone as physical device rather
logical device. This gives the scope to retrieve at most information from the iPhone.

But the problem with technique is reliability. This tool uses third party applications like Cydia packages for jailbreaking, which is not approved by Apple, Inc. [12].

6.2) New Technique

This method focuses on iPhone data partitions. iPhone has been divided into two partitions. The first partition is used by Apple to load the Operating system and the second partition is user data partition, which stores all the user activities. First partition that is OS partition does not contain any forensic relevance information. Second partition consists the entire user activities and is the rich source of evidence.

User data partition is constructed as directory based structure. It consists five directories.

Table 6.2: Directory list on iPhone user data partition

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ Var</td>
<td>Stores encrypted passwords</td>
</tr>
<tr>
<td>/ Private</td>
<td>Stores third party application data</td>
</tr>
<tr>
<td>/ Library</td>
<td>Telephony information and built in application data will be stored</td>
</tr>
<tr>
<td>/ Media</td>
<td>Stores videos and pictures</td>
</tr>
<tr>
<td>/ iTunes_Control</td>
<td>Stores audio and video files synched through iTunes</td>
</tr>
</tbody>
</table>

Table 6.2 is the list of directories present in the iPhone user data partition. iPhone user data partition can be obtained with the help of Oxygen software (commercial) [21]. The directory / Var consists all user entered passwords in encrypted format. / Private directory stores all the user accessed data through third party applications
along with time lines. /Library stores the telephony information like call logs, SMS, Address book entries, and built in application data like Email, Safari, Notes, YouTube, Maps, and etc. /Media stores pictures and videos on iPhone. These pictures and videos include the files downloaded through Safari, Email, and third party applications. /iTunes_Control stores all the audio and video files synched through iTunes, it also stores the music that downloaded through iTunes Store [2]. Information in the directories is stored in plist format. Plist is the short form of property list. Plist files store the serialized objects in the iOS programming. This plists stores the information accessed on the iPhone and information from the third party applications. Plists are in binary format and can be made readable upon converting them to readable xml. Following two methods can be applied convert plist into xml format.

Method 1: The following piece of code converts plist into xml format.

```plaintext
xmlData = [NSPropertyListSerialization dataFromPropertyList:plist format:NSPropertyListXMLFormat_v1_0 errorDescription:&error]
```

Method 2: Command line utility `plutil` can be used to convert plist into xml format.

```
plutil -convert xml1 ${1} nano -w ${1} plutil -convert binary1 ${1}
```

Analyzing the content in plists:
After converting the plist file into xml format, the xml file can be opened with any text editor. Here text-wrangler is used as text editor to analyze the content in the plist files. Due to limited access to the iPhone user data partition this research able to present only few applications on iPhone including third party applications. Some of the plists were simulated through xcode* to show the result.

Sample result from text-free application:

Figure 6.1: TextFree.plist in xml format

Figure 6.1 shows the content from text free application. TextFree is third party application downloaded from App Store for free. Text messages have been exchanged using the application downloaded and text messages exchanged have been deleted to test the new technique to investigate iPhone. Figure 6.1 shows messages has been exchanged with number “3612485401”. This method of reading plist also gives scope of analyzing the time line for application usage. Here in this plist for application TextFree, clear key word indicates the deletion of all messages exchanged. Even
though clear operation performed, messages exchanged before clear operation have
been recovered.

Figure 6.2: Where.plist in xml format

“Where” is a third party iPhone application for free on App Store. This application
needs login setup which can be done only once after the installation of the
application. Figure 6.2 shows the part of where.plist in xml format. This application
does not support any encryption standard to hide the password. Where.plist clearly
displays the username and password in plaintext. This plist also presents the email id
associated with “Where” application.
SimCam.plist is a camera that simulated on Apple’s developer kit Xcode [25]. Simulated camera uses the built-in camera in of Macintosh Computer and on click captures the pictures, and behaves like picture captured in iPhone. There is option to send the picture via email for this camera. Then extracted the plist from the /private directory of iPhone simulator. This plist gave the result as shown in figure 6.3. It shows the time, the picture captured, location of the picture taken (only city has been displayed here) time stamp of picture taken, along with further action of emailing the photo.

7) Anti-Forensic Techniques

There are different techniques to hide data on iPhone. These techniques make the secret data invisible to the users. Only intended users can see the hidden information. Varity of information can be hidden on iPhone like audio files, video files, text files, images, notes,
and etc. Hiding techniques can be categorized in two subsections: Hiding data on Jailbroken iPhones and non-Jailbroken iPhones.

7.1) **Hiding data on Jailbroken iPhone**: There are number of methods exist to hide the data on Jailbroken iPhone, some of them are:

7.1.1) **Method 1**: It’s a six-step process;

- Install the Mobile Finder, Textedit Mobile, AV Player and Mobile Preview on Jailbroken iPhone.
- Install Disk Aid application on a Macintosh computer.
- From Mobile finder on Jailbroken iPhone open the settings and create new associations. And the turn off the “Start in Last Location” switch to launch the executable files. Restart the Mobile Finder application after adding each association.
- File associations to add:
  - mp4: com.system.avplayer
  - m4v: com.system.avplayer
  - 3gp: com.system.avplayer (this association is to play audio files)
  - mp3: com.system.avplayer
- Connect the iPhone to the Macintosh computer that already loaded with Disk Aid application. Open the Disk Aid application and add the files to hide on to Disk Aid application.
- To see the files hidden connect the iPhone with the computer, in which Disk Aid has been installed [19].
7.1.2) Method 2: Install the apache webserver on Jailbroken iPhone, and then move the webserver root off the 300mb part of the flash memory used by the Apple on to the user data partition by using this commands:

```bash
mv /Library/WebServer/Documents /private/var/Documents
ln -s /private/var/Documents /Library/WebServer/Documents
```

Copy the file want to hide to Documents folder using sftp,

```bash
sftp filename /Library/WebServer/Documents
```

Create an .html page to get the web view for the file wants to hide and use the link http://127.0.0.1/filename.html, clear the browsers history to hide from others [16].

7.1.3) Method 3: On Jailbroken iPhone using the installaous application download “folders” application. From PC drag and drop the files on to the “folders” application and set the password to protect the files from others.

7.2) Hiding data on non-Jailbroken iPhone: The only way to hide files on non-Jailbroken iPhone is to use applications from App Store. There are different applications to hide the file from others. Some of such applications are:

- **Poof**: Is an application to hide the folders. It cannot hide the individual files.
- **Hidepod and Spycalc** are two more applications to hide the pictures and videos.
- **The way to keep SMS and call list secret**: Pysl is an application can be used to protect secret life on iPhone. It allows user to keep track of blacklist and
whitelist of contacts for SMS conversation filtering. These conversations made password protected using pysl application. For this process user need to have iClarified installer. By using this iClarified installer user can install pysl on iPhone. Then user can create a password and filtered list of contacts whose SMS conversation user wants to hide. This application provides three options to hide SMS and call history

- **SMS STOP**: hide SMS and store in SMS History.
- **SMS DELETE**: hide SMS and delete it “on the fly” (no trace).
- **CALL STOP**: hide CALL and register call in Call History.

Hidden information is only visible through pysl application.

- **Way to hide pictures**: There's an application called "PicVault" which has a fake calculator screen, and it needs enter the right PIN number, which redirects the user to hidden directory of pictures.

- **iDescrete** application allows password protected security for the files. It works as covert, which hides the files until secret password used.

### 7.3) Anti-forensic applications proposed

The applications found were the ways to hide the secret files from other users. But in reality they cannot escape from the forensic tools, because they just password protects the files from other users not really hides secret files. This search proses an application that can securely hide and erase the evidence on iPhone.

#### 7.3.1) Application 1:

Application to hide the text messages in Apple’s Password Key Chain. Key chain is the only the location which can securely store the text format messages. Even re-installation of the iOS cannot erase the text from the Key Chain.
This is because every time the iPhone synched with computer the Key Chain will be backed up. Originally Key Chain stores all the passwords accessed through the iPhone. The newly proposed application uses the Key Chain to store the secret text messages. The application stores the secret message as password in Key Chain, and the stored secret message can be recovered by using the access ID and password set by the user.

This application involves three steps: Creating a text message in Key Chain dictionary, searching for the text message and deleting the secret message entry from the Key Chain. The secret messages are stored in the encrypted format, which makes the message unreadable. Only the user who stored the text message will have the access to that text message using the user’s password.

Creating text Dictionary:

Table 7.1: Adding a secret message to Key Chain

```swift
NSMutableDictionary *textDictionary = [self newSearchDictionary:identifier];

NSData *secretMessage = [secret dataUsingEncoding:NSUTF8StringEncoding];
NSData *passwordData = [password dataUsingEncoding:NSUTF8StringEncoding];
NSData *accessID = [accessID dataUsingEncoding:nil];

[textDictionary setObject:secretMessage forKey:(id)kSecValueData];
[textDictionary setObject:secretMessage forKey:(id)kSecValueData];
[textDictionary setObject:passwordData forKey:(id)secretMessage];

OSStatus status = SecItemAdd((CFDictionaryRef)dictionary, NULL);
[dictionary release];

if (status == errSecSuccess)
    return YES;
```
The piece of code in the table 7.1 is responsible for adding a secret text message to the Key Chain. This application treats the secret message as password and encodes with Apple’s encryption techniques. This application assigns two extra parameters to the text message to separate the secret message from other passwords in the Key Chain. One parameter is accessID, the unique ID for each secret message and a password to access the secret message.

Table 7.2: Searching for the secret message in the dictionary

```swift
NSMUTableDictionary *text1Dictionary = [self newSearchDictionary:identifier];
[text1Dictionary setObject:(id)kSecMatchLimitOne forKey:(id)kSecMatchLimit]
[text1Dictionary setObject:(id)kCFBooleanTrue forKey:(id)kSecReturnData];
NSData *result = nil;
OSStatus status = SecItemCopyMatching((CFDictionaryRef)searchDictionary,
                                        (CFTYPERef *)&result);
[searchDictionary release];
return result;
```

Table 7.2 is the code block for searching the Key Chain dictionary. The application uses the access ID and password to search the secret text message in the Key Chain. Deletion operation involves two steps. First step the application searches for the secret text based on the access Id and password. Once the secret text found then it simply overwrites the secret text message, access Id and password with empty strings.
Testing the Application:

Figure 7.1: Adding secret text to Key Chain.
Figure 7.1 shows the interface of the application developed and the process of adding secret text to the key chain. Secret text message has been entered along with access Id and the password. Access Id and password together separates secret text message from other passwords in the key chain. Secret text message is saved in the key chain in the form password, and in encrypted format.

To retrieve the secret text hidden in key chain, access Id and the password associated with that secret text need to be entered. Up on entering the access Id and password application starts searching for the match it will return the secret message and displays in unencrypted format. The secret exists in the key chain until delete operation performed.
Figure 7.2: Retrieving the secret text from key chain
Figure 7.3: Deleting the secret message from key chain
As shown in figure 7.3 deleting secret text from the key chain needs access Id and password associated with that secret text. Deleting secret text goes through two steps; first application searches for the secret text associated with access Id and password entered. If the match found it removes the secret text form the key chain.

Figure 7.4 shows that if there is no match found for the access Id and password entered. In such case application will notify the user about wrong password or wrong access Id entered while searching for the secret text using the access Id and password. Figure 7.5 shows the similar kind error message displayed while trying to delete the secret text from key chain with wrong access Id or wrong password.
Figure 7.4: Error message for wrong id or password
Figure 7.5: Delete failed because of wrong access Id or password
Figure 7.6: Xml format of HideSecret.plist

Figure 7.6 is the part of the plist from the application HideSecret developed. From the plist recovered for the application HideSecret gives the information about application access time and access Id. From the content of plist it is clear that secret text and password are in the encrypted format; secret text and password were encrypted because of password key chain property applied. But the plist failed to give the information about the activity whether it is adding secret text to key chain or deleting secret text or retrieving secret text from key chain.

7.3.2) Application 2: Application 2 is a complete eraser of deleted evidence. This application deletes any residues left over after deleting the files on iPhone. Most of the iPhone forensic tools presented in section 6 have the ability to recover the deleted files from the iPhone. These forensic tools able to recover deleted files because the simple file deletion cannot delete the file completely from the iPhone rather it deletes the
access pointer to that particular file. Proposed application deletes the file completely from iPhone memory. Logic is simple, by default any application periodically monitors for the warning on low memory. As soon as an application receives a low memory warning it releases the unused memory that is an application completely releases the memory units that are not in use by that application. The application proposed in this research periodically requests for the memory allocation, which leads to the low memory situation. Then iOS sends the low memory warning to all other applications to release their unused memory units.

Proposed application involves an infinite loop that keeps on assigning the memory units to the application, and periodic requests for the memory units.

Table 7.3: Code block to send periodic memory allocation requests

```c
NSArray *tempData[] = [[[NSString alloc] initWithFormat:@""]];
int i=0, j=0;
while (1) {
    tempData[i] = [[[NSString alloc] initWithFormat:@"Garbage values here"]];
    NSMessagePort *message = [[[NSMessagePort alloc] initWithCoder:@"Request"]];
    NSDataReadingOptions = [[[NSDataReadingOptions alloc] allKeys alloc]Request:message new rewind(nil);
    i++;
}
```

Table 7.3 is a piece of code, which is responsible for sending the periodic requests to the iOS to allocate memory and thereby triggers iOS to send the low memory requests to other applications on iPhone. This code runs infinite number of times to completely
erase the deleted information by sending the low memory requests, and making the other applications to release the unused memory blocks.

**Testing the Application:**

![KillEvidence application on iPad simulator](image)

Figure 7.7: KillEvidence application on iPad simulator
Figure 7.7 shows the first look of the application KillEvidence, which cleans up the deleted content on the iPhone. This application runs an open GL functions to create a rectangular object, which moves like it erasing the iPhone. In the background an infinite loop of memory allocating function runs to make periodic calls to iOS to assign memory to the current application and responsible iOS to send low memory warnings to other applications to release their unused memory blocks.

8) Data Integrity Solution

In any forensic investigation data integrity means that correctness of data starting from evidence collection to reporting in court of law. Data integrity check proves correctness of data that is there is no alteration to the evidence at any point of situation in forensic investigation. Data integrity can be achieved in different ways for example backing up data, implementing secure mechanisms to access the data, and etc. But in case iPhone backing up entire iPhone is difficult thing and only can be achieved through Jailbreaking, which gives the complete access to iPhone memory according to Zidzarski [12]. But Jailbreaking it self is a threat to data integrity on iPhone, because Jailbreaking involves installing applications unauthorized by Apple, Inc. These additional applications installed may alter the data on iPhone. According to the research carried out on different forensic tools and techniques presented in Section 6, lacks in data integrity check. Lack of data integrity check makes the evidence report generated by these tools and techniques unreliable in court of law. This research proposes a simple application to prove the data integrity in iPhone forensic investigation using the plists from user data partition on iPhone. User data partition consists plists belongs to all the third party applications and built-in applications in five different directories, discussed in Section 6.2. The simple
application developed to create the hash of the plists for all the applications. This application up on applying before and after extracting and analyzing the evidence on iPhone can result in data integrity check.

The application in this research is developed in Java, uses the Java security Library from Java API. This application also uses Java Swings for interface design.

Figure 8.1: Hashing application for data integrity

Figure 8.1 shows the application interface for Hashing method can be applied on plists to provide data integrity check. Three different hashing techniques have been implemented with this application MD5, SHA1 and SHA-512. Plist file can be chosen from the directory of user data partition of iPhone.

Table 8.1: Code block for hashing the content of plist

```java
MessageDigest md = null;
try {
    md = MessageDigest.getInstance("MD5");
}
try {
    md = MessageDigest.getInstance("SHA1");
}
```
try {
    md = MessageDigest.getInstance("SHA512");
}
md.update(data);

byte[] digest = md.digest();

hash = gethash(digest);

Table 8.1 is the piece of code that used to convert the plist content in to hash using one-way hash functions like MD5, SHA1, and SHA-512. If there is single bit change in plist the application outputs hash value with greater change. Comparing the hash values before and after the analysis of evidence on iPhone data integrity can be ensured.

9) Conclusion and Future work

The research in this project signifies the importance evidence analysis on iPhone as the number of third party applications increasing. This research report creates base line to asses the challenges that forensic investigator can face while investigating the suspect’s iPhone. New technique has been proposed to deal with increasing number of third party applications, which uses the plist files from the user data partition on iPhone. Converting the binary form plist into readable xml gives the scope to analyze the activities performed on iPhone using different applications. Data integrity check application has been proposed to prove the correctness of evidence by different techniques and tools. This application uses one-way hash functions MD5, SHA1, and SHA 512 to create hash values of plist files. Anti forensic techniques have been analyzed to foreshadow the bad
guys perspective in escaping from the forensic investigation. Two anti forensic applications have been proposed with this research; one to hide the text information password key chain and the other is to clean up the residues of deleted files on iPhone by using the concept of periodic memory allocation requests. Further more this research can be extended to analyze the encrypted content from third party applications. Also the anti forensic method of hiding the text information can be extended to media files like audio, video and image files.
References:


[14] MacForensics Lab MacLockPick,  


[16] MacRumors, “How to hide porn on an iPod touch/iPhone”.  


[18] MicroSystemation, XRY Logical Software,  


http://www.optimumrecovery.com/data/macintosh-hfsplus.html


[22] Paraben corporation, “Device Seizure v4.3”,  
http://www.paraben.com/device-seizure.html


http://stackoverflow.com/questions/3190034/turn-on-torch-flash-on-iphone-4/3367424#3367424