I hereby affirm that I have followed the directions as published in the program Web page at

http://cs.tamucc.edu/docs/GraduateProjectAgreementForm.pdf

And I confirm that this project and documentation is my own personal work and all the material other than my own is properly referenced.

Student’s Name: ___________________________________________

Student’s Signature: _________________________________________

Date: _______________________________________________________
ABSTRACT

Forensics is the science of investigative study of data or information from surroundings in a scene of crime. Computer forensics deals with computer-related crimes like hacking, spoofing, phishing, etc.. Traditional forensics would investigate the system in its dead state. That is, the system would be disconnected, brought to a forensics lab and investigative analysis on the data would be performed. It is very safe, because the system would be frozen and would not be able to act on anything, thus providing an ideal set up for investigation. However, once the power cord is pulled out, the live data is lost forever. And live data is of prime importance because they contain processes running, memory usage, open ports, communications happening, objects being altered, malicious pieces of code running, information strings passed over the network, system passwords and many such valuable pieces of information which can be counted as vital evidence. Hence, live forensics is the methodology to extract all the live information from the compromised machine. In order to perform the live acquisition without contaminating the data, a forensic expert is generally preferred; however the availability of a forensic expert may or may not be possible at all times. In this project a new tool, compatible with Windows 7 operating system, is implemented that can automate the process of carrying out the live forensics acquisition.
# TABLE OF CONTENTS

Abstract ........................................................................................................................................ iii

Table of Contents ....................................................................................................................... iv

List of Figures ............................................................................................................................ vi

1. Background and Rationale ........................................................................................................ 1

2. Narrative ................................................................................................................................... 8

3. Proposed System Design .......................................................................................................... 12

3.1. Application of tools ............................................................................................................. 14

  3.1.1. Hashing .......................................................................................................................... 15

  3.1.2. Hostname ....................................................................................................................... 15

  3.1.3. Date and time ................................................................................................................. 16

  3.1.4. IPconfig ......................................................................................................................... 16

  3.1.5. Netstat .......................................................................................................................... 17

  3.1.6. Arp ................................................................................................................................. 20

  3.1.7. At .................................................................................................................................... 21

  3.1.8. Tasklist .......................................................................................................................... 22

  3.1.9. Auditpol ......................................................................................................................... 23

  3.1.10. Drivers ......................................................................................................................... 23

  3.1.11. Dumpel ......................................................................................................................... 24

  3.1.12. Freespace ..................................................................................................................... 25

  3.1.13. GPList ......................................................................................................................... 26

  3.1.14. HFind ........................................................................................................................... 27

  3.1.15. Net ................................................................................................................................ 27
3.1.16. Now ................................................................. 28
3.1.17. Procinterrogate .................................................. 29
3.1.18. Promiscdetect .................................................... 30
3.1.19. Pstat ............................................................... 30
3.1.20. Sc ................................................................. 31
3.1.21. Schtasks ......................................................... 32
3.1.22. Uptime ........................................................... 32
3.1.23. Iehv ............................................................... 33
3.1.24. Uname ........................................................... 33
3.1.25. Whoami .......................................................... 34
3.1.26. Md5sums ......................................................... 34
3.1.27. Memdump ....................................................... 35

4. Evaluation and Results ................................................. 36
4.1. Test case 1 ........................................................... 37
   4.1.1. Results of test case 1 ........................................... 38
4.2. Test case 2 ........................................................... 38
   4.2.1. Results of test case 2 .......................................... 39
4.3. Test case 3 ........................................................... 41
4.4. Test case 4 ........................................................... 41
4.5. Test case 5 ........................................................... 43
4.6. Test case 6 ........................................................... 44
4.7. Test case 7 ........................................................... 45
4.8. Test case 8 ........................................................... 46
4.9. Test case 9 ........................................................... 46
4.10. Test case 10 ......................................................... 47
LIST OF FIGURES

Figure 1.1 Different investigation phases ................................................................. 7
Figure 3.1 Design of the tool ...................................................................................... 14
Figure 3.2 Date and Time support .............................................................................. 16
Figure 3.3 Ipconfig report ......................................................................................... 17
Figure 3.4 Network statistics ..................................................................................... 18
Figure 3.5 Network statistics in numeral form ......................................................... 18
Figure 3.6 Network statistics with process ID and application .................................... 18
Figure 3.7 Networks statistics with interface statistics ............................................. 19
Figure 3.8 Network statistics with routing tables ..................................................... 20
Figure 3.9 Address resolution protocol (ARP) ......................................................... 21
Figure 3.10 Scheduled task listed by At .................................................................... 22
Figure 3.11 Tasklist .................................................................................................... 22
Figure 3.12 Auditpool ................................................................................................. 23
Figure 3.13 List of loaded drivers ............................................................................. 24
Figure 3.14 Dumpel system log ................................................................................ 24
Figure 3.15 Dumpel security log .............................................................................. 25
Figure 3.16 Dumpel application log ......................................................................... 25
Figure 3.17 Freespace ............................................................................................... 26
Figure 3.18 GPList ..................................................................................................... 26
Figure 3.19 Hfind ....................................................................................................... 27
Figure 3.20 Net user .................................................................................................. 28
Figure 3.21 Net started services ............................................................................... 28
Figure 3.22 Now timestamp ................................................................. 29
Figure 3.23 Procinterrogate ................................................................. 29
Figure 3.24 Promiscdetect ................................................................. 30
Figure 3.25 Pstat ................................................................. 31
Figure 3.26 Scquery ................................................................. 31
Figure 3.27 Schtasks ................................................................. 32
Figure 3.28 Uptime ................................................................. 32
Figure 3.29 Iehv ................................................................. 33
Figure 3.30 Uname ................................................................. 33
Figure 3.31 Whoami ................................................................. 34
Figure 3.32 Md5sums ................................................................. 34
Figure 4.1 Output screen ................................................................. 37
Figure 4.2 Administrator privileges settings ................................................................. 38
Figure 4.3 Restricted privileges ................................................................. 39
Figure 4.4 Dumpel error ................................................................. 40
Figure 4.5 Output screen ................................................................. 40
Figure 4.6 Output screen ................................................................. 41
Figure 4.7 Computer name ................................................................. 42
Figure 4.8 Host name ................................................................. 42
Figure 4.9 Connection – specific DNS suffix ................................................................. 43
Figure 4.10 Network connections ................................................................. 44
Figure 4.11 Tasklist ................................................................. 45
Figure 4.12 At ................................................................. 46
Figure 4.13 Hfind................................................................. 47
Figure 4.14 Net user............................................................... 48
Figure 4.15 Net start ............................................................. 48
Figure 4.16 Whoami .............................................................. 49
LIST OF TABLES

Table 3.1 List of tools ................................................................................................................. 13
1. INTRODUCTION AND BACKGROUND

Computers have become a very integral part of our culture. Our daily activities consist of trivial and non-trivial jobs performed on computers like paying bills, entertainment, performing bank transactions, storing personal information, emails, etc. There have been an increasing number of crimes related to information stored on computers and such electronic media [11]. These crimes include a wide variety of phishing, hacking, illegal activities, etc. Forensics is a branch of education that deals with scientific investigation in crimes. Computer forensics is a very distinct field of study that involves scrutinizing electronic records, data and other files which have been either involved in or used by someone to commit a criminal offense. It also deals with the analysis, interpretation and preservation of data found in electronic mediums like hard disk and other such digital media. Live forensics is a more specific field of study, where the current system state and physical memory, i.e. the random access memory (RAM), are scrutinized and analyzed for investigative study. Traditional computer forensic techniques dealt with recovering data on an electronic medium or preserving and extracting them. But these days, with extensive storage capability, internet usage, networking, traditional methods are not adequate. Imaging a large storage device is a time consuming activity which is followed by analysis and extraction of the useful case related data. Live forensics, as the name suggests, is the collecting of data at run time when all the systems, like the desktop, server and network connection, are running [1]. Hence, in live forensics, the physical memory is a good vital resource. It contains a lot of valuable information like, the processes running at a particular instant, the network associated, the number of users logged in, the files being accessed, the programs working
in the background and much more. This information can be very critical to any crime scene investigation.

The traditional forensics’ primary approach is to initiate an investigation by making an image of the storage device, mostly hard disk drives of the system. This image is analyzed to extract any useful case related information. The phases of the traditional forensics investigation are acquiring data, performing an analysis of the data acquired and extracting the data relevant to the crime. The image of the evidence from the computer system which is source or the target of the criminal activity is acquired in order to keep the evidence intact and prevent contamination. Prodiscover is one such tool capable of making a bitstream copy of a hard disk drive and creating an image. Storage devices mainly are hard disk drives and performing the data acquisition on the hard disk drive has become challenging due to the increasing capacity of the hard disk drives. In the cases where the system has network drives mapped to it, the collection of evidence is far more challenging rendering the traditional approach ineffective. Finally, if the storage is fault tolerant such as the RAID array of disks then imaging is quite impossible. The system should be offline or in dead state in order to make an image of its hard disk drive, so the impact on revenue caused by the downtime of the system is a major issue in traditional forensics. The information about the state of the system is also lost when the system is unplugged.

Live forensics provide a snapshot of what is going on in the system; it aids the collection of the volatile information such as list of running processes, all the network
connections that are active, open ports that are in place, current user who is logged on, memory dump, etc. The volatile state information is acquired by using certain tools on the machine. Gathering data on a machine by running tools may, in turn modify the volatile data on that machine, so a set of best practices should be followed to make sure the quality of the evidence is good [12]. The quality of information acquired depends on the running of known good tools, encrypting all the evidence and gathering the data in the order of volatility [12].

The tools available on the target system cannot be trusted to provide accurate information, so all the tools and the depending libraries used to collect the volatile data must be provided by the investigator. A library of trusted tools must be maintained to perform accurate live acquisition. The collected data must be preserved in order to make sure that it has not changed. This can be ensured by encrypting the data collected using a hashing algorithm such as MD5. A fingerprint of the files generated during the data acquisition can be created using the hash, which can then be used at any stage to demonstrate that the evidence has not been tampered. A simple way would be to generate the hash again and compare it with the original. If they are same then it can be concluded that the evidence is untouched. In case the data acquired is transmitted over a network then the hash should be generated on the both ends and compared to see that the data has not changed.

In order to avoid any damage to the evidence, the tools should not create any additional data on the system. The data should be collected in the order of the volatility
Network connections change more often than the information of the user who is logged in so network statistics should be collected first. The order of execution of the tools can affect the data and the system log files may register the entries for each command executed. And if a tool takes a lot of time to finish execution like the dump of RAM, the changes occurred in the interim cannot be captured - such as the loss of network connection, termination of some processes, and closure of open files. Analysis of the evidence is performed after all the data is acquired; however acquiring all the data in the case of live forensics is difficult due to the dynamic nature of the situation. A forensic expert might need to collect more information after reviewing the results generated by the tools being used. For example, if there is a process which has a high port number open indicating a possible network sniffer, then the network connection, IP address of the destination system must be determined. Live forensics acquisition provides data that is not available on the hard disk drive of a system. The data gathered by making a snapshot of the system during live acquisition cannot be reproduced. There is a risk of contamination of the evidence during the acquisition. Hence the best practices should be followed to gather data in a way that can be used in a court of law.

However, the data collected in live forensics cannot be treated as impeccable. The veracity of the data needs further investigation to check its integrity. There are malicious softwares which can manipulate the kernel and thereby make the study erroneous. Kernel trojans have been designed with anti-forensic technology and can manipulate the data in the kernel [2]. There are several shortcomings in this approach. Some of them are:

*Live forensics is a step-by-step process. Each further step is built and based upon the previous one. The stages are analysis,*
preservation and collection. There is no way to prove the process in a reverse order i.e. the veracity of the previous stage cannot be checked [1].

All the three phases - collection, preservation and analysis are intermixed and executed on the data. Therefore, it is tough to verify the credibility of these methods [1].

Because of kernel modifying Trojans, it is not possible to know if the forensic tool running on the suspicious computer is gathering accurate results [1].

Many recent forensic techniques have tried to deal with these shortcomings. Memory analysis is one technique which is a new approach to deal with the shortcomings [1]. This technique can give insight to hidden user mode processes [3]. In traditional investigative practices, tools were used to simply preserve the data from the computer. There are several tools like Prodiscover [10] and such, which do this job. However, in all the traditional methods, the computer from the crime scene is disconnected and brought into the lab to run it against tools and perform a forensic study. As mentioned earlier, there is critical evidence in the volatile memory too, which gets lost when, the machine is turned off. Some information is still resident until the computer is rebooted [4]. There are some malicious pieces of code like SQL Slammer worm which can only be detected in memory [3]. Among the other useful features of live forensics is that the documents, emails and other such files can be recovered even after the application or Word editors have crashed. By searching strings, valuable information or commands that the intruder might have typed could be uncovered [4]. The information about the objects that are alive or were alive in this process could also be recovered. Various commands like netstat and ipconfig can recover network information from memory, but there is a large part which is still not accessed. Carrier and Grand [5] mention a hardware based tool tribble [5] which is very effective in copying and dumping the main memory, with a drawback being, the special PCI disk has to be installed on every machine before the attack happens. There
are some vital components of memory structure or architecture that needs to be studied and understood in order to study how ‘live forensics’ is applicable. *System.map* file which contains the addresses of kernel symbols, address table, virtual address space, map of system memory, data units and meta-data units and other memory regions are some examples [5]. There have been many research arenas under this theme. By studying these mentioned memory structures, Schuster [6] developed some search patterns, with the help of which the complete dump of memory can be scanned and traces to various linked and unlinked objects can be detected. There are various tools to analyze the memory dump. Walters and Petronmi [7] use a technique to form an incident perspective and use a tool to analyze main memory dump. Carrier and Grand [5] discuss the Windows forensics memory toolkit and [8] use similar tools to analyze the dump.

When live forensics is being conducted on the machine, the data undergoing in analysis is not compromised. In other words, these forensic tool kits deployed do not change anything in the memory profile of the machine. Hence, it is safe to be studied and analyzed by other department entities or third parties too. This helps to have a non-biased interpretation of the data. This also paves the way for recursive performing of analysis by the same or different tools and still upholding the truthfulness of the data on the compromised machine. Carrier and Spafford [8] discuss how to minimize the effects of running tools on the data. Walters and Petronmi [7] talk about a process model that discusses the activity flow of a live forensics investigation. Understanding it, helps in better approaching the subject of investigation. Figure 1.1 is a model proposed by Carrier and Spaford [8], which displays a prominent process. It is divided into six phases. Live
responses are typically performed by response toolkits. These toolkits can be performed by anyone with no extensive forensics training. But care has to be taken that these tool kits do not add to the criminality of the offense, as there are drawbacks associated with these tool kits [7].

Figure 1.1 Digital Investigation Phases [7]

Although an automated tool has its limitation it serves the purpose of performing live acquisition if a forensics expert is unavailable at the scene of crime. Once the live data is collected it can be analyzed just as in traditional forensics to find and extract case related relevant information. In the following sections a design and implementation of forensically sound automated live acquisition tool which follows the best practices of live forensic acquisition is discussed. This tool allows any authorized person at the scene of crime to perform live acquisition.
2. NARRATIVE

Acquisition tool

Live forensics deals with volatile data collection and analysis. When the current state of a system is collected, the raw data is open for analysis. The tools and mechanisms used to perform this raw image analysis can be sent to an external party, who has the liberty to understand and interpret the information in their own way, thereby supporting an unbiased approach to the investigation. Furthermore, the analysis and operations performed by these tools can be conducted again by these external third parties and results can be checked and corroborated. That is to say, they can have their own procedure of analysis.

There are certain memory images, patterns, linkings, threads, processes and objects in the memory which can be used for investigation. It is evident that these rich information sources are not accessed during the initial recovery of the data in traditional forensics, and hence not available for analysis. To interpret this information for more superior understanding of memory, forensics expertise and additional tools and mechanisms are required. A case in point would be that in an investigation scene during the live response analysis, some malware would go undetected. This can be possible because of its residing in the low-level memory area. Subsequently, during the deeper investigation, it can be revealed. Hence, there is a huge possibility of such malicious code of software being revealed at a later part of investigative analysis purely owing to low-level memory access.
Here, it is also important to note that there are two phases in volatile memory access. They can be classified as data collection and data analysis. Volatile memory access tries to clearly differentiate and distinguish between the methods of data capturing and data analysis. Data capturing phase strictly adheres to the aspect of collecting data and nothing more. Data analysis is to be merely understood as performing analysis on the suspect machine, where several tools are run to generate and dissect the information into meaningful sections. There is a risk that this action can alter and damage the potential evidence still resident but undetected, on the compromised machine [5].

Hence a tool, aimed at making a credible extraction of live data from the compromised machine and which has overcome the shortcomings thereby reducing the intrusiveness to the machine under study, is needed for live response. A tool that can perform data analysis and data collection easily without disrupting the undetected evidence under a condition that the compromised system is free of data manipulating malware and trojan is the scope of this project.

This acquisition tool is targeted to perform on Windows 7 operating system. There are no intensive requirements for the tool to run. This is a light weight tool and it can perform the acquisition with very minimal hardware and software requirements. All the trusted data collection applications like ipconfig.exe and netstat.exe are integrated into this acquisition tool to execute and collect data and save it onto a log file. Most of the data collection applications that are integrated as part of this tool are the utilities of the Windows 7 operating system. The rest of the applications are gathered from various
trusted software vendors like Microsoft, MacAfee, Sysinternals, NTSecurity, Nirsoft, PathSolutions and Open Source Lab (Osuosl).

The tool does not prompt for input. In order to perform the acquisition it can be invoked from its location through windows explorer. In case of output all the collected applications are integrated to perform the acquisition and the output is saved as a text file. Since all the tools are utilities of the operating system which access the system information, the current user will need administrator privileges to execute the tool. Since the tool is expected to be invoked from the windows explorer, it is assumed that the system is unlocked and current user has the administrator privileges.

In the hands of a forensic expert the tool accelerates the data acquisition process. Unlike the traditional forensics where the data acquisition phase and data analysis phase are separate and occur linearly one after the other, these phases work together in live forensics. It is quite impossible to gather all the live data available on a system and then perform the analysis of that data in the case of live forensics. An expert would essentially gather data which is most relevant to the crime, examine it and decide what other data is required for further acquisition.

Consider for example, the case of a report of a slow web server. An investigator obtains the list of all the processes and the list of all the network ports that are open. Assume that a process with high port number is identified by examining the above results, further information of the process is retrieved and the investigator identifies that there is a file open for writing for this process. Further information retrieved about the file lets the investigator conclude that the file is a log file for the network traffic,
confirming that the process attached to the file is a network sniffer. Quick analysis of information leads to acquisition of more relevant information.

The IP address and port number in the above attack could not have been determined without live forensics. Live forensics gives the investigator an opportunity to respond faster. Live forensics aids the analysis phase of traditional forensics. As part of the evidence collection of the file in the above example, it must have been written to a location on the hard disk drive. Hence the file can be found in a short amount of time on the hard disk drive by using the file write location associated with the process. Procedure for the live acquisition:

1. Document all the steps for evidentiary reasons and later references.
2. Check to see if the system is locked. If it is locked, then obtain the credentials for account with administrator privileges as administrator privileges are necessary to image the physical memory.
3. All the windows should be left running, as closing them would kill the respective process.
4. Limiting the interaction with system to as few steps as possible, as higher interaction might cause the evidence to contaminate.
5. USB port must be enabled on the system.
3. SYSTEM DESIGN

A crime ordinarily, can occur anywhere and the availability of a computer forensic expert who can perform a forensic investigation is doubtful. This may result in the possibility of loss of valuable evidence which is in the form of volatile data in the compromised system. Hence, a tool which can automate the activity of data analysis and collection process is needed. Such a tool will be required to recognize the system and its properties and thereby would apply the right tools suitable for that specific system. Optimum user friendliness is required to ensure that it can be operated by any law enforcement personnel in the absence of a computer forensics expert. Finally, the collected volatile data need to be presented in a most comprehensible manner which can easily be handled by a forensic expert.

The proposed tool in this project is to develop one such tool which can apply necessary compatible tools to analyze and extract the volatile data, in the most convenient manner, with little or no disruption to the evidence in the compromised system. Table 3.1 contains the list of some tools which are capable of collecting the volatile data. Tools such as *arp.exe, ipconfig.exe, netstat.exe, net.exe, openports.exe, nbstat.exe* are capable of collecting the network information of the system [9]. Although each tool has its own capability, not every tool is compatible with all the versions of Windows operating system and without the right combination of all these tools, the data generated about the network connections may not be in a desirable or beneficial manner. Figure 3.1 depicts the proposed design of the tool.
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hostname</td>
</tr>
<tr>
<td>2</td>
<td><code>ipconfig /all</code></td>
</tr>
<tr>
<td>3</td>
<td><code>netstat -a</code></td>
</tr>
<tr>
<td>4</td>
<td><code>netstat -an</code></td>
</tr>
<tr>
<td>5</td>
<td><code>netstat -o -b</code></td>
</tr>
<tr>
<td>6</td>
<td><code>netstat -e</code></td>
</tr>
<tr>
<td>7</td>
<td><code>netstat -r</code></td>
</tr>
<tr>
<td>8</td>
<td><code>Arp</code></td>
</tr>
<tr>
<td>9</td>
<td><code>At</code></td>
</tr>
<tr>
<td>10</td>
<td><code>Tasklist</code></td>
</tr>
<tr>
<td>11</td>
<td><code>Auditpol</code></td>
</tr>
<tr>
<td>12</td>
<td><code>Drivers</code></td>
</tr>
<tr>
<td>13</td>
<td><code>dumpel -l system</code></td>
</tr>
<tr>
<td>14</td>
<td><code>dumpel -l security</code></td>
</tr>
<tr>
<td>15</td>
<td><code>dumpel -l application</code></td>
</tr>
<tr>
<td>16</td>
<td><code>Freespace</code></td>
</tr>
<tr>
<td>17</td>
<td><code>Gplist</code></td>
</tr>
<tr>
<td>18</td>
<td><code>Hfind</code></td>
</tr>
<tr>
<td>19</td>
<td><code>net user</code></td>
</tr>
<tr>
<td>20</td>
<td><code>net start</code></td>
</tr>
<tr>
<td>21</td>
<td><code>Now</code></td>
</tr>
<tr>
<td>22</td>
<td><code>Procinterrogate</code></td>
</tr>
<tr>
<td>23</td>
<td><code>Promiscdetect</code></td>
</tr>
<tr>
<td>24</td>
<td><code>Pstat</code></td>
</tr>
<tr>
<td>25</td>
<td><code>sc query</code></td>
</tr>
<tr>
<td>26</td>
<td><code>Schtasks</code></td>
</tr>
<tr>
<td>27</td>
<td><code>Uptime</code></td>
</tr>
<tr>
<td>28</td>
<td><code>Iehv</code></td>
</tr>
<tr>
<td>29</td>
<td><code>Uname</code></td>
</tr>
<tr>
<td>30</td>
<td><code>Whoami</code></td>
</tr>
<tr>
<td>31</td>
<td><code>Md5sum</code></td>
</tr>
</tbody>
</table>

Table 3.1 List of Tools used for live acquisition
There are three components in this design. The first component is going to probe and check the underlying host machine and verify if it is a 32 bit or a 64 bit machine. The tool automatically selects the compatible tools that are capable of performing volatile data extraction on the target system. The second component is going to perform a physical memory dump which is a vital step in live forensics. Finally the third component is responsible for representing the data collected in a comprehensible manner.

3.1 Application of tools

All the trusted applications needed for acquisition are collected and placed under the folder ‘library’. Some of these applications can be found on the target system as part of the Windows 7 collection of utilities but since the target system could have been
compromised, the investigator must always carry the library of the tools which are needed in order to perform the acquisition. The design of the tool is a fire-and-forget model. After the invocation of the tool, it is going to generate the nine reports and save them under the folder named ‘log’. Naming convention followed for the reports is a combination of current date, current time, application name and switch used. For example the consolidated report generated is named as “04-10-2012_22-35-32.txt”. There is no application name specified in the file name as there are multiple application reports logged in the consolidated file, and the report for the application Dumpel with the Switch security is named as ’04-10-2012_22-35-32_DUMPEL_SECURITY.txt’. Once all the reports have been generated the memory dump is collected with the name ‘memory.bin’

3.1.1 Hashing

The generated reports are to be carried from the site to the lab for further analysis and as a starting step to the next phases of the forensic analysis. Since this evidence is not carried by a forensic expert, it is important for the forensic expert to be able prove that the evidence collected from the target system is not tampered with or modified. Hence, after all the reports are generated, the files are cryptographically hashed using the MD5 algorithm. The forensic expert just has to generate the new hash for each file and compare it to the generated hash to see if they match.

The list of tools and their descriptions are as follows:

3.1.2. Hostname
This utility is used to retrieve the name of the current system. This is associated with Windows 7 Operating System. It is a simple application which can run by itself with no additional switches.

### 3.1.3. Date and Time

The date and time are very important details in a case of forensic acquisition. These two details are grouped together along with the hostname application results in the final report as shown in Figure 3.2.

![Figure 3.2 Date and Time report](image)

### 3.1.4. IPconfig

The data should be gathered in the order of its volatility. Since network connections are more likely to be changed or closed anytime during the acquisition, the network data is collected first. IPconfig is a Microsoft Windows 7 utility which records the network configuration information. This application has many switches, a detailed description of the network configuration can be generated by using the switch ‘all’. The report of this application is as shown in Figure 3.3.
Figure 3.3 IPconfig report

3.1.5. Netstat

The detailed network information report can be generated by using this application. Network Statistics lists out the open network connections, port numbers and routing table information. The source, destination system name and IP address along with the port number for a connection is listed in the report. There many switches available for
different information reports about the connections. As part of the acquisition tool, the following switches have been used.

1. netstat -a: It displays all the open connections and the listening ports. The report of this application is shown in Figure 3.4.

![Figure 3.4 Network statistics](image)

2. netstat -an: It displays all the open connections and listening ports in numerical form. The report of this application is shown in Figure 3.5.

![Figure 3.5 Network statistics in numerical form](image)
3. netstat -o -b: The switch -o displays the process id of the connections that is active and -b displays the executable associated with the connection if any. The report of this application is shown in Figure 3.6

![Figure 3.6 Network statistics with process ID and application](image)

4. netstat -e: The switch -e displays the interface statistics is shown in Figure 3.7

![Figure 3.7 Network statistics with interface statistics](image)
5. `netstat -r`: The switch `-r` of netstat displays the routing table information for a connection, the connections are grouped with the internet protocol version used. The report of this application is shown in Figure 3.8.

![Figure 3.8 Network statistics with routing tables](image)

3.1.6. **Arp**

The application displays the address translation table translating from IP to physical address, as used by the address resolution protocol. This application has switches which allow to modify or delete an entry in the table, as part of the acquisition tool the switch `-a` and `-v` are used, which retrieve the translation table entries is shown in Figure 3.9.
3.1.7. At

This application lists out the scheduled jobs that are set to run on the system through the scheduler. This is an important application, if the user of the system decides to hide the data, then a schedule job can be set to delete all the temp files in regular time intervals. The report of this application is shown in Figure 3.10.
3.1.8. Tasklist

The tasklist gives out the list of process information currently running on the system. The service name, process ID and memory usage are some important details listed by this application. The report of this application is shown in Figure 3.11.
3.1.9. Auditpol

This application displays the audit policy settings configured on the system. The report of this application is shown in Figure 3.12.

![Auditpol Report](image)

Figure 3.12 Auditpol

3.1.10. Drivers

This application lists the device drivers that are installed. The report of this tool in the acquisition tool is generated to a separate file, with file name containing ‘date-time_DRIVERS.txt’. The report of this application is shown in Figure 3.13.
3.11. Dumpel

This application dumps event logs of the local system into a text file. There are three different log files retrieved using this application in the acquisition tool - system logs, security logs and application logs. These logs are generated into a separate file named with date, time and log type. The reports generated by this application are as shown in Figures 3.14, 3.15 and 3.16
3.1.12. Freespace

This application lists the storage capacity and the amount of storage utilized. The report of this application is shown in Figure 3.17.
3.1.13. GPList

This application lists out the group policies applied on the various resources of the system. The report of this application is shown in Figure 3.18.
3.1.14. HFind

This application retrieves the list of hidden files and their last access time in the directory mentioned. So an entire scan of directory is performed to identify the list of hidden files. The report of this application is shown in Figure 3.19.

![HFind Application Screenshot](image)

Figure 3.19 HFind

3.1.15. Net

This application is used to retrieve the list of user accounts created on a system and the windows services that have started and are running currently. The report of this application is shown in Figures 3.20 and 3.21.
3.1.16. Now

This application takes the current date and time stamp. The report of this application is shown in Figure 3.22.
3.1.17. Procinterrogate

This application lists out the complete information about the process. The process ID, modules that loaded for the process and the total size occupied are important details that are listed in the report is shown in Figure 3.23. This report is generated to a separate file as part of the acquisition tool with the name of the report as ‘date-time-procinterrogate.txt’
3.1.18. Promiscdetect

This application detects if the network adapters are in the promiscuous mode. The report of this tool is shown in Figure 3.24.

3.1.19. Pstat

This application lists all the processes and threads and their statuses. The report of this application is shown in Figure 3.25.
Figure 3.25 Pstat

3.1.20. Sc

This application is used to create or start services and generate the complete status report for the services currently running. The report of this application is generated to a separate file with name ‘date-time_SC.txt’ and it in format is shown in Figure 3.26.

Figure 3.26 Sc query
3.1.21. Schtasks

This application lists out all the system tasks that are scheduled to run. The report of this application is shown in Figure 3.27.

![Figure 3.27 Schtasks](image)

3.1.22. Uptime

This application gives the time for which the system has been up and running. The report of this application is shown in Figure 3.28.

![Figure 3.28 Uptime](image)
3.1.23. Iehv

This application stands for Internet explorer history viewer. The report of this application is generated to a separate file name ‘date_time_IEHV.txt.’ This application generates the entire history of the sites visited, files accessed on the local machine, and saves them to a text file as shown in Figure 3.29.

![Figure 3.29 Iehv](image)

3.1.24. Uname

This application lists system properties as shown in Figure 3.30.

![Figure 3.30 Uname](image)
3.1.25. Whoami

This application lists out the current user logged into the system as shown in Figure 3.31.

![Figure 3.31 Whoami](image1)

3.1.26. Md5sums

This application is a cryptographic hashing algorithm which is used to hash the logs obtained during the acquisition process. This application allows the investigator to determine that evidence collected is not tampered with. The report of this application is generated to a separate file with name ‘date_time_hash.txt’ as shown in Figure 3.32.

![Figure 3.32 Md5sums](image2)
3.1.27. Memdump

This application is used to perform the physical memory dump. The physical memory is copy bit wise into a file creating a binary file named ‘memory.bin’.
4. EVALUATION AND RESULTS

The acquisition tool works with minimal requirements, such as not having to install it on the target system in order to use it. However, the tool requires the USB port of the machine to be enabled in order to perform the acquisition. Most of the commands retrieve system data; hence the user currently logged in should have the administrative access in order to have the same access to the tool. The feature of the tool to be able to run from a USB makes it impossible for any modification of any content of storage on the target machine. It is tested against Windows 7 and works consistently to collect the data. The time taken to finish the acquisition is considerably high as the physical memory dump is an intense activity. The acquisition is being taken by the physical memory dump 90% of the time. The tool has collected the data consistently when run multiple times on the same machine and other machines. The tool is designed on the fire-and-forget model which requires the user to invoke just one file and wait until it finishes, adding to the user friendliness of the tool. Deductions from the results are:

1. Consistent
2. Minimal impact on System
3. Uses trusted applications
4. Gathers extensive system information
5. Hashes the results.
6. Organized reports.

The tool can be invoked directly from the flash drive and the screen of execution is shown in Figure 4.1
4.1. Testcase 1

The test case 1 comprises of end to end functional testing on a test machine with complete administrative access. Figure 4.2 shows the user account control settings set to “never notify”, which places almost no restriction over the applications that run in the current user account that has been logged in.
4.1.1 Results of Testcase 1

No errors have been encountered during the execution of the tool. All the files have been created with complete information. Figure 4.1 shows the output report of the tool.

4.2. Testcase 2

The test case 2 comprises of end to end functional testing on a test machine with minimal administrative privileges i.e., all settings other than complete administrative access or a
‘standard user account’. Figure 4.3 shows default setting applied for an administrative user account.

![Restricted privileges](image)

Figure 4.3 Restricted privileges

### 4.2.1 Results of Testcase 2

The applications that fail to complete execution due to the insufficient access privileges usually are the ones trying to access security information of the system. The security log from *dumpel* application and audit policy from *auditpol* application failed due to insufficient access privileges. Figure 4.4 shows the error message popping up for dumpel application, this window can be closed in order to continue execution. Figure 4.5 shows the error message.
Figure 4.4 Dumpel error

Figure 4.5 Output screen
4.3. Testcase 3

The testcase 3 comprises of testing the tool on a restricted account, for example a *guest* account. The tool stalls while trying to retrieve the process list due to insufficient privileges. Figure 4.6 shows the screen shot of the tool stalling.

![Figure 4.6 Output screen](image)

4.4. Testcase 4

Scenario: To test the hostname application. The hostname application should reflect the name assigned to the computer.

Expected output: The expected hostname entry for the given test machine is “Arun-PC” as shown in Figure 4.7
Received output: The tool captured the name of the host as “Arun-PC” as shown in the Figure 4.8.
4.5. Testcase 5

Scenario: To test the ipconfig application. The connection specific domain names are configured to the network adapter dynamically upon connecting to a network. In this scenario, the network adapter is connected to Texas A&M University Corpus Christi (TAMU CC) University network and DNS is verified.

Expected output: The connection specific DNS suffix must reflect the TAMU CC DNS name.

Received output: The tool captured the output that contains the TAMU CC DNS name as shown in Figure 4.9.
4.6. Testcase 6

Scenario: To testing the netstat application. The network connections are the most volatile information that can be retrieved from a computer. In this scenario, a new network connection is established pointing to https://www.google.com.

Expected output: The newly established network connection is expected to be captured by the tool.

Received output: The received output shows the IP address of pertaining to the established connection as shown in Figure 4.10.

![Figure 4.10 Network connections](image-url)
4.7. Testcase 7

Scenario: To test the tasklist application for consistency. Tasklist application lists out all the applications that are currently running. In this scenario a Google chrome window is opened.

Expected output: The tasklist application should list chrome in its entries of running applications.

Received output: The received output shows the entries of the chrome application as shown in Figure 4.11

Figure 4.11 Tasklist
4.8. Testcase 8

Scenario: To test the At application. At application is used to retrieve the list of scheduled tasks. Scheduling a batch file “delete.bat” to run on weekdays at 9:00am.

Expected output: A scheduled task to run “delete.bat” should be populated.

Received output: The task setup for running “delete.bat” is listed as shown in Figure 4.12.

![Figure 4.12 At](image)

4.9. Testcase 9

Scenario: To testing the Hfind application. Hfind application is used to retrieve the list of hidden files. Hidden file “sample.txt” is created in a folder on the hard disk drive for this scenario.

Expected output: The hidden file name must be captured in the logs.
Received output: The hidden file “sample.txt” has been captured in the logs as shown in Figure 4.13.

![Image of Notepad++ with HFind output]

**Figure 4.13 Hfind**

### 4.10. Testcase 10

Scenario: To test the Net application. Net application is used to retrieve the list of users created and services running. “Test” user has been created for testing the user logs and “TeamViewer” service has been started for this scenario.

Expected output: Test user must be listed in the logs as a user. Team Viewer service must be started.

Received output: “Test” user has been captured in the logs as shown in Figure 4.14. Team Viewer service has been captured in the logs as shown in Figure 4.15.
Figure 4.14 Net user

4.11. Testcase 11

Scenario: To testing the whoami application. Logging into the system as the user “Arun”.

Expected output: User “Arun” should be listed in the log file.

Received output: “Arun” is captured in the logs as shown in Figure 4.16.
Figure 4.16 Whoami
5. Conclusion

A forensically sound tool that is compatible with Windows 7 operating systems is created in this project. The motivation behind the tool was the need to perform live forensics on a Windows 7 platform. Traditional forensics is not adept in acquiring critical ‘live’ information with respect to a crime. A running system has information such as passwords, search strings, opened websites, web history, network ports working, running processes and etc. These details can help an investigator draw a sketch about the nature of the crime, the underlying motivation and also about the criminal. Extracting such information is not possible when the system is in its dead state. Hence a live forensics tool is very critical in any investigation for more results.

The main purpose and the importance of this project is to create an automated live acquisition tool which can aid the non-technical personnel in performing the live acquisition. The users of this tool are mostly police officials and law enforcement officers who are most likely not skilled in forensics knowledge and forensics acquisition tactics. Hence it was important to take this into consideration while creating the tool. The tool is advantageous to law enforcement agencies since it does not make them rely on the availability of expert forensics investigators at the crime scene. The ‘fire and forget’ nature of the tool makes the user relieve of his/her interactions with the tool and thereby making their work easy. Many applications used in this tool have been collected from trusted sources and integrated into this tool in order to automate the live acquisition process. It is expected to aid the investigation especially for Windows 7 operating system as there are no automated tools that are compatible with it. As mentioned, the tool can easily be used by any non-technical personnel. The reports generated by the tool are
targeted to a forensic expert and can be easily comprehended. The tool has been tested on various scenarios and concluded that in order to retrieve complete system information, complete administrative privilege are needed. The tool has been tested with adequate test cases and has been corroborated with the results achieved. The extensive testing of the tool has also demonstrated that it does not alter any existing data on the host machine. Any other access privileges to the system will cause the tool to partially retrieve the system information or stall due to insufficient privileges. A successful way to perform live forensics on a compromised system with Windows 7 operating system is achieved. Since the results are hashed using MD5 hash algorithm, the integrity of the data is rest assured. The tool successfully conducts a live forensics acquisition on a Windows 7 machine, extracts relevant information which is only available as long as the system is running, and generates truthful reports of the results and encrypts them; thereby perfectly complying with the computer forensics methodology and principles. This tool can greatly enhance the current research happening among the forensic community, as there is a shortage of automated tools for Windows 7 platform.

As a future work, the tool can be further enhanced for better performance and a wider functionality. Currently the tool is compatible with Windows 7 operating system it can be extended to work with other windows operating systems like Windows 8, Windows XP and Windows Vista. The number of applications integrated into this tool can be increased in order to collect more live data. The reports generated by the tool can be further enhanced with better presentation.
BIBLIOGRAPHY AND REFERENCES


