Digital Forensics Tools Integration

THESIS

Alexander D.H. Kim, Captain, USAF

AFIT-ENG-MS-20-M-031

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-20-M-031

Digital Forensics Tools Integration

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Cyber Operations

Alexander D.H. Kim, B.S.

Captain, USAF

March 19, 2020

AFIT-ENG-MS-20-M-031

Digital Forensics Tools Integration

THESIS

Alexander D.H. Kim, B.S.
Captain, USAF

Committee Membership:

Gilbert L. Peterson, Ph.D
Chair

Douglas D. Hodson, Ph.D
Member

Robert F. Mills, Ph.D
Member

AFIT-ENG-MS-20-M-031

Abstract

As technology has become pervasive in our lives we record our daily activities both intentionally and unintentionally. Because of this, the amount of potential evidence found on digital media is staggering. Investigators have had to adapt and change their methods of conducting investigations to address the data volume. Digital forensics examiners current process consists of performing string searches to identify potential evidentiary items. Items of interest must then go through association, target comparison, and event reconstruction processes. These are manual and time consuming tasks for an examiner. This thesis presents a user interface that combines both the string searching capabilities that begin an investigation with automated correlation and abstraction into a single timeline visualization. The capability to improve an examiner's process is evaluated on the tools ability to reduce the number of results to sort through while accurately presenting key items for three use cases.

Acknowledgements

First, I would like to thank Dr. Gilbert Peterson for all his advice and support. This would not have been possible without your guidance. I would also like to thank Capt Daniel Schelkoph and Capt Nikolai Adderley for helping me continue their research. Lastly, I would like to thank my wife. Mona Aquino, in supporting me throughout this endeavor.

Alexander D.H. Kim

Table of Contents

Page

List of Figures

List of Tables

I. Introduction

1.1 Problem Background

The digital age ushered in various forms of technology for entertainment and productivity purposes. People now own computers, laptops, tablets, mobile phones, smart TVs, and other Internet of Things devices that store the details of their activities. It is estimated by 2025, the number of connected devices would rise to 75 billion devices with the average person estimated to own nine smart devices [1]. These numbers do not include the various devices people may have in the form of virtual machines and such that are utilized in cloud environments.

With the growing number of both physical and virtual devices, the demand for investigators capable of conducting digital forensics is increasing. Law enforcement agencies have been attempting to catch-up by using digital forensics tools that provide high-level abstraction and hide the intricate details, increasing the user base capable of collecting digital evidence. But even with this increase, case backlog is an on-going issue because of the sheer number of devices a single person may own.

1.2 Motivation

The first 48 hours of an investigation are critical [2]. After an allegation is received and an investigation is launched, the first few hours can determine the outcome. Perishable evidence such as DNA or testimony, can be permanently lost if the correct actions are not taken immediately. Investigators must utilize the initial hours effectively to determine the best course of action based on the limited information they gathered. Because of this, digital media is typically not utilized to create investigative leads but rather used to confirm hypotheses developed later. The time it takes to process and analyze digital evidence is not conducive to quickly generating a favorable course. Many factors contribute to the lengthy time it takes to process digital evidence; not having an investigator with the skills to conduct the examination readily available, the speed of the tools, the number of devices, and the locations to search on said devices. But as technology becomes more pervasive in our lives, the information stored on these devices have become increasingly necessary to produce investigative leads within this 48-hour window.

Even outside of this window, the number of devices gathered throughout investigations has created a backlog for the digital forensics examiners who are in short supply.

> ***"The requirement for digital forensics investigation has ballooned, and law enforcement agencies throughout the world are scrambling to address this demand. While more and more members of law enforcement are being trained to perform the required investigations, the supply is not keeping up with the demand. Current digital forensics techniques are arduously time consuming and require a significant amount of man power to execute [3].***

There is a need to create digital forensics tools that can triage the devices for important information necessary during the initial phase of the investigation. At the same time, there is a need to be able to process and analyze devices in a timely manner to keep up with the demand and prevent delays in investigations.

1.3 Research Objectives

The methods of analysis require change as the amount of data produced and stored on disk media continues to grow. Some of the challenges that require address are the correlation of data from heterogeneous event sources [4]. New applications and uses of technology are being developed daily. With this comes a variety of different methods of storage and data is no longer confined to the local single workstations. In fact important pieces of information can be stored in “the Cloud” where only remnants of the original data remain in the local cache. Because of this, digital forensics analysis will have to address the challenge of taking data from different sources and combining them into some base-level commonality for analysis.

The overarching research goal is to reduce the analysis time spent by examiners during the digital forensics process. The research objective was to improve upon a visualization method Temporal Analysis Integration Management Application (TAIMA) employed, which allowed examiners to quickly digest the results of disk media analysis, and its back-end database, Property Graph Event Reconstruction (PGER), that provided a method for efficient data queries and enabled automated event correlation. This research first examined how the tools fit into a common digital forensics examination process and adding the string searching capabilities they require to improve an examiner’s process.

The resulting application Event Correlation and Abstraction Timeline Visualization (ECATV) creates a keyword filterable timeline to be used in conjunction with Autopsy, an open source digital forensics tool. The filterable timeline visualization was created to make analysis easier by combining various tools and sources of data.

1.4 Results

ECATV was tested on three different scenarios and evaluated based on its ability to reduce the number of results an examiner must sort through and its ability to still locate the key pieces of digital evidence. The tool was able to reduce the number of hits by upwards of 90% and located over 70% of the key pieces of digital evidence. Overall, ECATV significantly reduces the amount of data to sort through while maintaining the ability to locate the necessary information.

1.5 Conclusion

Humans produce data at an increasing rate and digital forensics must develop new tools and methods to keep up with technology. Native graph databases have proven to provide an efficient means for data queries and abstraction creation through relationships created between events. Visualization methods have proven to allow digital forensics examiners to spend less time during the human-analysis portion of the digital forensics process. ECATV was made to combine the benefits of using a native graph database through PGER and a visualization method through TAIMA with a new keyword search filter option. This research examines the benefits of combining multiple digital forensics results from multiple sources and tools leading to a more efficient examination.

II. Background and Literature Review

There are many facets that need to be taken into account to create a forensically sound tool. Digital forensics examinations are a complicated process that involves not only technical aspects but also legal considerations. Development of Event Correlation and Abstraction Timeline Visualization (ECATV) requires first closely examining what the digital forensics process entails and where in that process the tool can improve the efficiency of examinations. It also needs to ensure it follows the criteria for forensic soundness for a viable tool. Lastly, it is important to review tools and the features they already employ to both warrant the development of a new tool and for inspiration.

The first section goes over the aspects of the digital forensics process and the common phases associated to an examination. The next section explains information visualization and how it applies to the forensic analysis phase. The third section explains the rules and criteria for a forensically sound process. The last three sections provide an overview of the currently available and widely used digital forensics tools.

2.1 Forensic Process

Before looking into building a tool for digital forensics, we present the current process used for digital investigations. There are many models out there with one of the first outlined in 1984, called the Computer Forensic Investigative Process (CFIP). It was a simple model that helped establish a basic outline. CFIP consisted of four phases: acquisition, identification, evaluation, and admission. The acquisition phase involved acquiring the physical evidence in an acceptable manner with proper authority, just like any evidence in an investigation. The identification phases involved tasks to identify the digital components from the evidence and converting it into a format understood by humans. The evaluation phase was to determine which digital pieces of evidence were relevant to the investigation and final admission phase involved presenting said evidence in the court of law.

2.1.1 Common Phases of Digital Forensics

Yousef, et al. [5] reviewed a total of 15 different models suggested by various authors from 1984 to 2010. Through a systematic process, they identified the common digital investigation phases. In total, the authors extracted 46 phases associated to the 15 digital forensics investigation models. Then the authors sorted the phases and placed them into 5 generic groups based on tasks performed in each of the phases.

The resulting Generic Computer Forensic Investigation Model (GCFIM) [5] involves five different phases. The first phase, pre-process, involves all the tasks needed to be done prior to obtaining the data. These tasks include obtaining the necessary approval from authority, setting up the tools needed, and other preparatory tasks. The second phase, acquisition and preservation, involves identifying, acquiring, collecting, transporting, storing, and preserving of data [5]. The third phase, analysis, involves the various types of analysis on the acquired evidence to identify the relevant evidence for the crime being investigated. Phase 4, presentation, involves the proper documentation from the analysis phase and presenting evidence in a manner understood by all. The last phase, post-process, involves proper return of evidence, review of the investigative process/lessons learned to improve future investigations.

GCFIM, illustrated in Figure 1, is also cyclic. Because the investigative tools, technology, skills of the digital forensics examiner, and crime scenes are always changing, the authors wanted to ensure that the ability to go back to previous phases was always present. Lastly, the authors placed the phase, Incident Response, into the post-process phase of their model because they believe any action or response taken

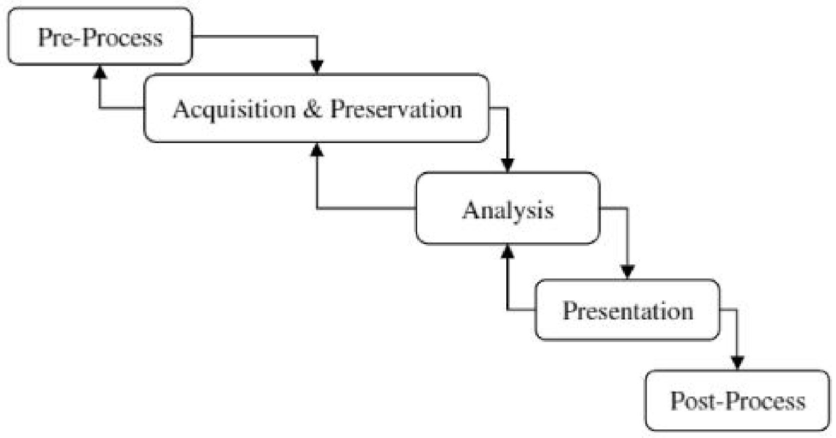


Figure 1: Generic Computer Forensic Investigation Model [5].

should be done after the evidence has been properly analyzed and presented. But there may be cases where immediate response is required, and the authors leave that prerogative up to the investigator. Most investigators prefer to delay action until all possible evidence has been gathered and analyzed. But there are instances when time is the dominating factor and investigators must act on the information immediately, for safety concerns or securing perishable evidence. It is important to recognize the digital forensics investigative process requires a flexible framework.

The authors present a model that accounts for all the different phases represented by various digital forensics models published throughout the years. But they do not consider the different phases for specific investigations. For example, the phases for an arson investigation will be different from the phases for a sexual assault case. Though the authors main purpose is to create a generic model for digital forensics investigations, it seems a little oversimplified. Nonetheless, there is still merit in establishing a standard model that aggregates all the various models presented throughout the years to base off of for future work.

The GCFIM is a starting point to building ECATV. The two phases ECATV should focus on are the analysis and presentation phase. Analysis should process large amounts of events and artifacts simultaneously and triage the relevant hits. This will help identify areas containing potential evidence and a digital forensics examiner can do a deeper dive into the relevant locations instead of having to conduct a full examination on the entire device. The presentation is an output in a presentable format to authorities. A solution is a timeline sorted by relevant events and timestamps presented in an easy-to-read format.

2.1.2 Forensic Computing

McKemmish [6] provides a similar framework to the GCFIM. His description of the digital forensics process identifies four phases: identification, preservation, analysis, and presentation, shown in Figure 2.

Identification allows examiners to determine what evidence is present and the tools necessary to extract the relevant evidentiary data from electronic media. Once examiners have identified the tools and methods necessary to conduct examinations, proper preservation of physical items is necessary to ensure the data obtained is admissible in court. This requires preserving the original data and conducting examinations that minimally impact the original data, usually done by taking either logical or physical extractions of the digital media. The main bulk of digital forensics centers around the analysis process, where examiners extract, process, and interpret digital data [6]. The last piece is presentation, where examiners must prepare a report for court that is easily digestible for all present, especially the jury and the judge. The first three phases are centered around technical aspects while the final phase is based entirely on legal issues, such as policy and law.

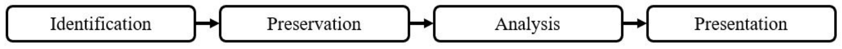


Figure 2: Forensic Process.

2.1.3 Analysis

The Analysis phase is considered the primary element of digital investigations and involves the process of "extraction, processing, and interpretation of digital data [6]." This phase takes a significant portion of the time to complete a digital forensics investigation. Thus, digital forensics tools developed to reduce the time consumed in this phase will have a larger impact in decreasing case back-logs than tools focused on reducing time spent on the other phases.

The period of analysis can be reduced in two main ways. One path involves increasing the efficiency of the digital forensics tool's extraction and processing of data through hardware and/or software upgrades. The overall goal would be to reduce the time forensic examiners wait for a tool to create an image of the data and to process the image into human readable format. The other method decreases the time an examiner spends during the interpretation portion of the analysis phase. Examiners are faced with the issue of sorting through tons of information irrelevant to the investigation. The reduction in human analysis can be accomplished by presenting the processed data in a way that allows examiners to quickly analyze the information or by providing only data that is relevant to the investigation. But the latter is difficult because each investigation is unique and developing a tool that is able to determine what information is relevant to a case is a non-trivial task. Instead this study focuses on presenting the information provided by digital forensics tools in a more manageable format, thereby reducing the human analysis time spent by examiners.

2.2 Information Visualization

Information visualization has been shown to provide examiners and analysts with the ability to process large amounts of data in a meaningful way [7] [8]. The Department of Homeland Security (DHS) established the National Visualization and

Analytics Center (NVAC) to "advance the state of the science to enable analysts to detect the expected and discover the unexpected from massive and dynamic information streams and databases consisting of data of multiple types and from multiple sources, even though the data are often conflicting and incomplete [9]." Though NVAC was originally established to combat terrorism issues, its goals closely align with the needs for the digital forensics field [4].

Visual analytics is defined as the science of analytical reasoning facilitated by interactive visual interfaces. It is utilized to develop meaning from massive and dynamic data [9]. The insights from this field of research [10] can be directly applied to developing better digital forensics tools to reduce the human analysis time of a digital investigation. There are many visualization techniques that can be used to display data for digital forensics, ranging from simple x-y plots and line graphs to more complex methods such as dense pixel visualization [11]. There is no one method that is considered the best technique to visualize data. The performance of each visualization technique is data dependent and purpose driven.

This is further reinforced by the taxonomy of visualization, which has been historically categorized into scientific and information visualization [12]. Scientific visualization often involves scientific data with a spatial component such as wind tunnel data whereas information visualization involves discrete and non-spatial data such as financial information. Typically continuous visualization models are utilized to render scientific visualizations while discrete visualization models are used for information visualization. [12] illustrates though digital forensics data lies under the information visualization umbrella, because not all data sets contain only one type of data, both discrete and continuous visualization methods can be used. What is more important is the conceptualization of the data drives the most appropriate visualization technique.

2.2.1 Forensic Analysis through Visualization

Teerlink [13] argues there is a need for software tools that reduce efforts spent by examiners, especially when it comes to handling large amounts of data. Analysts squander a lot of time trying to sort through massive amounts of data that are uncorrelated to the investigation. Teerlink proposes visualization techniques can reduce time wasted and direct examiners to the "suspicious files". His tool differs from the traditional forensic tools such as EnCase by providing advanced visualization techniques that aid data correlation and analysis beyond a simple front-end GUI [13]. The tool utilizes two different techniques: non-hierarchical and hierarchical visualization techniques.

Non-hierarchical visualization displays files and their statistics without taking into consideration the relationships between files and directories. For example, the tool highlights larger files with light colored blocks and smaller files with darker colored blocks. This helps the examiner quickly identify anomalous files. The tool can also be filtered based on a different attribute, such as time where the lighter colored blocks represent files with more recent activity. Ultimately Teerlink allows the tool to filter accordingly to any available file attributes because different investigations require examinations of different file attributes. The hierarchical visualization technique employed allows the tool to show the relationships between files and its directory structures. Teerlink's tool utilizes filtered tree maps that are filterable based on file attributes, not just the traditional method of display based on file size. Again this allows flexibility for the examiners to filter accordingly to the different scenarios of their investigation. For example, the tool can display a tree map of the files and use red to represent image files while yellow represents system binaries. This allows examiners to easily spot out-of-place files, such as a red colored image file contained in a file structure full of yellow colored system binaries.

Teerlink conducted a user test to determine the effectiveness of the tool's visualisation environment. He instructed users to locate an unknown number of files related to drug trafficking using two different methods. The first method involved using traditional Linux commands such as ls, cd, grep, file, md5sum, state, and find [13]. Teerlink argued these commands provided similar capabilities to those of EnCase's analysis tools, minus the front-end GUI. The second method involved analysis through Teerlink's developed visualization techniques. Users were also asked to record the time their examination began, the discovery time and name of each suspect file, and when their examination ended. Based on this data, Teerlink discovered the visualization techniques helped examiners discover on average 53% more files than using the more traditional method. More importantly to this study, the test showed there was a 35% reduction in time and examiners were 57% faster in locating the first suspicious file. In digital forensics, it is nearly impossible for examiners to say with a 100% certainty they found every piece of digital evidence. Instead they conclude their examinations when they have enough evidence to present in court, based on legal advice and law enforcement guidance. Sometimes all an examiner needs is to find the first piece of the puzzle in the sea of ambiguous data, which leads to various pieces of evidence until eventually the necessary amount to prove criminality is collected. Teerlink concludes the goal was not to develop a complete tool to compete with EnCase, but rather to integrate EnCase's outputs of its search capabilities as an input for the visualization environment.

2.3 Forensically Sound

Defining what it means for a process to be forensically sound is essential to ensuring the admissibility of digital evidence [6]. From a legal perspective, admissible evidence must be reliable; key factors include whether the evidence is generally accepted in its respective scientific community as well as whether the methods used to derive the evidence are repeatable and produce the same results. There are many companies and organizations that state their digital forensics tool utilizes a forensically sound process to produce its data. Because of this, many different definitions of a digitally forensic process float around.

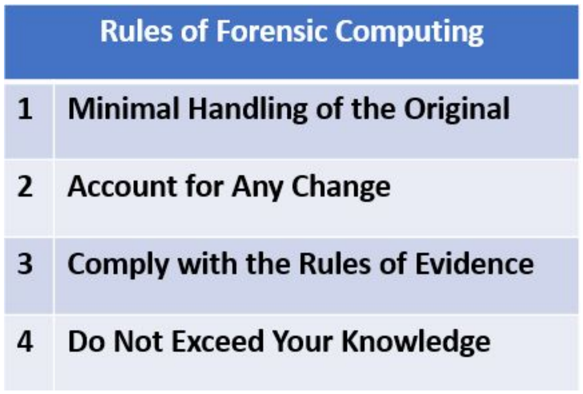
McKemmish [14] provides a more stable framework for the definition of "forensically sound" in regards to digital forensics. The paper considers a disk imaging process to be forensically sound when the process produces an exact representation of the original, the duplicated data is independently authenticated as being a true copy, and the process produces an audit trail. Overall the goal is to produce evidence that preserves the data in the state it was first discovered while the process does not diminish the evidentiary value of the data "through technical, procedural, or interpretive errors."

2.3.1 Four Rules of a Forensically Sound Process

The overall objective of digital forensics is to provide evidence in the court of law. Along those lines, McKemmish [6] provides four rules aimed at maximizing the admissibility of evidence produced by a digital forensics process, outlined in Figure 3. Rule 1 is considered the absolute important rule as examination in any forensic process (digital or physical) should minimize the likelihood of alteration of the evidentiary item. If alteration is necessary, then the examiners should do their best to duplicate the original and examine the duplicate data. Rule 2 follows the same vein as rule 1 where if changes to the item are unavoidable, these changes must be properly documented with proper reasoning for doing so. This is part of the reason why being a digital forensics examiner is difficult as the ability to identify the extent of alterations is directly correlated to the examiner's skill and knowledge. The examiner must be able to provide testimony as to why the changes were unavoidable as well as its extent and at the same time in a way the common person in a jury can understand.

Rule 3 states the application and development of forensic tools should be done with the rules of evidence in mind. This means whatever is done to the data, it should not lessen the admissibility of the final product. Lastly, rule 4 is similar to rule 2 in that an examiner should not conduct an examination that is beyond his or her knowledge. As stated in rule 2, it is essential an examiner has the ability to explain in detail the process of the examination and the extent of unavoidable changes done on the evidentiary item. It does not matter how forensically sound a tool is if the examiner is unable to properly explain the process in court.

Rule 2 and 4 are relevant but are cursory to goals of this research. It is important the tool has some level of granularity, such as providing file paths for artifacts discovered. But the two rules correlate more to the abilities of the examiner. Rule 3 relates to the forensic process and policies of how the examiners secure evidence. Generally most examiners and investigatory agencies will image drives and operate on the duplicates [15], which also relates to rule 1. Rule 1 is met by importing the cloned data into a visualization software. It introduces additional steps and time spent to analyze the image, but [6] states it is justified if it allows examiners to identify key



Rules of Forensic Computing	
1	Minimal Handling of the Original
2	Account for Any Change
3	Comply with the Rules of Evidence
4	Do Not Exceed Your Knowledge

Figure 3: Rules for Forensic Computing [6].

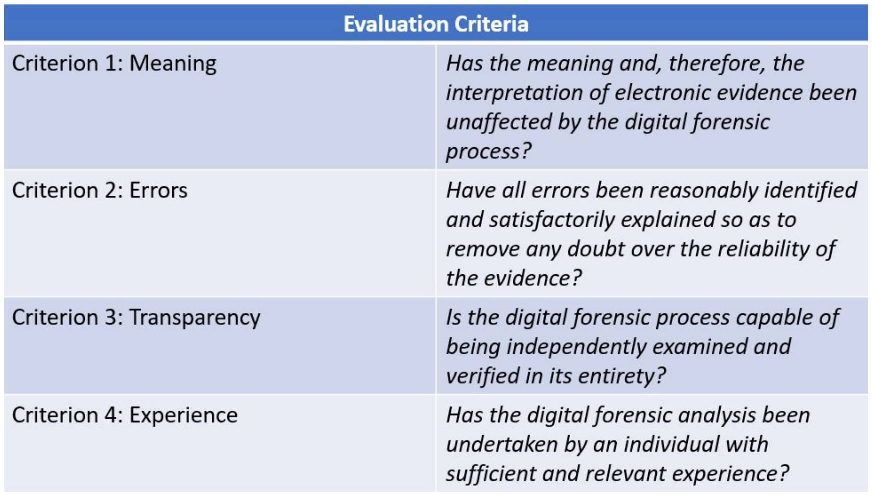
items in a shorter time.

2.3.2 Evaluation Criteria for Forensic Soundness

McKemmish [14] provides four criteria for determining forensic soundness of the process, which are outlined in figure 4. Tassone, et al. [16] further breaks down the evaluation criteria into additional steps for consideration with regards to visualization methods. Criterion one, meaning, needs to consider repeatability, allowing data to be visualized without altering or changing the actual artifacts. This means if the same data is revisualized at a different time or location, it produces the same results. It also needs to consider lucidity, where the visualization generated by the tool is easily understood by the forensic examiner. Criterion two requires anomaly detection where it provides the examiner with the ability to identify unusual data sets.

Criterion three is the most important area for this research. Tassone, et al. [16] further breaks down the transparency criterion for forensic visualization into granularity and filterable. Granularity with respect to the visualization method means it provides an overview of the data while allowing the examiner to delve into greater details of the artifacts if necessary. And filterable allows examiners to refine the visualization to key items either through text searches or time stamps. Filtering allows examiners to identify key items in less time, which is relevant to rule one mentioned in Section 2.3.1.

Criterion four does not directly correlate to developing a forensically sound visualization tool. But Tassone, et al. [16] states that it can be aided by the use of a timeline visualization method. Timelines provide the examiners with how the data was utilized over time, allowing them to analyze large data sets. Usually investigators are focused on a specific time period and timelines allow examiners to focus on these periods and expanding the scope if needed.



Evaluation Criteria	
Criterion 1: Meaning	*Has the meaning and, therefore, the interpretation of electronic evidence been unaffected by the digital forensic process?*
Criterion 2: Errors	*Have all errors been reasonably identified and satisfactorily explained so as to remove any doubt over the reliability of the evidence?*
Criterion 3: Transparency	*Is the digital forensic process capable of being independently examined and verified in its entirety?*
Criterion 4: Experience	*Has the digital forensic analysis been undertaken by an individual with sufficient and relevant experience?*

Figure 4: Evaluation Criteria for Forensic Soundness [14].

2.4 Forensic Tools

There are many different open source and commercial forensic tools available. Each tool processes and analyzes drive images using different methods. The tools also utilize varying visualization methods depending on what its developers determined is suitable to forensic examiners. Because these tools were developed with different interpretations of what data is relevant and how it should be displayed, it is important to keep in mind rule four of a forensically sound process, as mentioned in Section 2.3.1. These visualization methods can provide an overview of all the data for convenient navigation, but it is necessary for these methods to be provide some granularity and filters based on a property.

2.4.1 Commercial Tools

Tassone, et al. [16] provides a list of popular commercial forensic tools. The study selected tools based on the following criteria:

1. Commercial because open source software are not liable for any inconsistencies.

2. Used in at least one court transcript and/or officially provided to a government

agency for use in criminal investigations.

3. Provide a visualization method.

4. Its forms of visualization are public knowledge.

Based on these criteria, Tassone, et al. [16] identified the following forensic tools:

- Nuix: Visual Analytics (Nuix)
- Micro Systemation: XAMN/XRY (XAMN)
- Cellbrite's UFED Link Analysis (UFED)
- EnCase Analytics (EnCase)
- Oxygen Forensic Suite 2015 Analyst (Oxygen)
- Katana Forensics Lantern (Katana)
- Susteen Secure View 3 (Susteen)
- Forensic Toolkit FTK AccessData (FTK)
- Internet Evidence Finder IEF Magnet Forensics (IEF)
- Intalla Vound (Intella)
- i2 Analysts Notebook by IBM (i2)

XAMN, UFED, EnCase, Susteen, FTK, IEF, and Intella all provide timelines as a method of visualization. Timelines can fulfill all the evaluation criteria mentioned in Section 2.3.2, which include repeatablity, lucidity, anomaly detection, filterable, and granularity. But these tools do not provide high-level abstraction of events and instead provide artifacts as is and with limited temporal information, which can lead to information overload. They utilize various search capabilities such as time range or keyword searches to help combat this issue, but without the correlation of related events, these searches still suffer from having to sort through a quagmire of artifacts. As a base, this study looks to utilize a timeline as its visualization method because many popular commercial forensic tools employ this technique and has the potential to meet the requirements for a forensically sound process.

2.4.2 Open Source Tools

There are debates about whether open source tools provides evidence that is admissible in court. Tassone, et al. [16] focused its study on commercial tools because open source software was not liable for inconsistencies, which runs into the issue of evidence produced by the open source tools being reliable. Section 2.1.2 explains the presentation phase is based on the legal environment of where the evidence is being presented. Before looking at the visualization methods of various open source tools, this study looks at the legal admissibility of evidence produced by such tools. Though the presentation phase is not the technical piece of digital forensics, it is still necessary to consider because all the effort placed into producing the evidence is null if it cannot be admissible in trial.

2.4.2.1 Admissibility

Scientific evidence must pass the "Daubert Test" in order to be admissible in a United States legal proceeding. This means the evidence must be both relevant and reliable. The judge determines "whether the underlying methodology and technique used to identify the evidence was sound and the evidence is reliable [17]." The test uses the following categories when assessing the reliability of a scientific procedure:

- **Testing**: Can and has the procedure been tested?

- **Error Rate**: Is there a known error rate of the procedure?
- **Publication**: Has the procedure been published and subject to peer review?
- **Acceptance**: Is the procedure generally accepted in the relevant scientific community? [17]

Based on these categories, Carrier [17] states open source tools are actually more reliable than closed source commercial tools. The open source forensic tools provides its source code to all, allowing for increased scrutiny in all categories. Carrier argues that for the acceptance guideline, closed source tools were accepted based on non-procedural factors such as interface and support because it did not disclose its procedures. The developers of the commercial tools cited the large number of user they had as a way to get around this category of the admissibility test [18]. Open source tools on the other hand provide its procedures allowing for the digital forensics community to evaluate it and choose whether or not to accept or reject them.

Carrier offers a balanced solution that takes into account the commercial interests of forensic tool development. Carrier [19] and Carrier, et al. [20] splits tools into two main categories, extraction and presentation. Extraction tools process data and extract a subset, while presentation tools arrange the data from an extraction tool into a useful format [17]. There are tools that provide both capabilities while others provide one role. Carrier presents a solution where extraction tools are open source so investigators can verify the output while the presentation tools remain closed source. The source code for the open source extraction tools allow the digital forensics community to properly validate the procedures used to produce digital evidence.

2.4.2.2 Popular Open Source/Free Forensic Media Imaging Tools

There are many open source forensic tools available as of 2019. As mentioned in Section 2.4.2.1, most of these tools are split into extraction tools and presentation tools.

The popular forensic imaging tools are FTK Imager, Linux “dd” and IXImager. FTK Imager allows the examiner to create an exact replica of the drive, allowing them to view deleted files. It also brings the capability to mount forensic images to view its contents and file structure in the FTK Imager browser [21]. Linux “dd” is a tool that comes with most Linux distributions. The tool creates drive duplications and can also send data streams over the network. But examiners need to be careful with this tool because it can also provides other capabilities, such as filling a drive with random data, and has the potential to destroy evidence[22]. IXImager is a tool developed by the U.S. Treasury Department IRS Criminal Investigation Electronic Crimes Program, tested by National Institute of Standards and Technology (NIST), and made to meet Law Enforcement requirements for digital media acquisition. “IX-imager supports imaging devices which cannot otherwise be imaged in a Windows environment, including notebooks and server RAID systems. It supports hot swappable and plug-n-play devices such as, tape drives, USB and Firewire devices, as well as SCSI, IDE and fiber channel [23]”

Popular open source analysis tools include SANS Investigative Forensic Toolkit (SIFT) and Sleuth Kit/Autopsy. SIFT is an ”open source incident response and forensic tool suite” created off an Ubuntu LTS 16.04 Base. It supports a large variety of file systems, from standards such as NTFS, HFS+, and FAT16/32 all the way to vmdk. The suite includes various open source tools such as log2timeline (for timeline visualization), Plaso, Autopsy and Sleuth Kit, and Volatility (a memory analysis tool) [24]. Oxygen (the free version) and Digital Evidence and Forensics Toolkit

Zero (DEFT) are popular free forensic tools as well. DEFT is very similar to SIFT which offers various tools in its suite and is a Linux Distribution. It includes Digital Advanced Response Toolkit (DART) which contains various open source and closed source Windows applications for live forensic analysis and incident response [25].

2.5 The Sleuth Kit and Autopsy

The Sleuth Kit (TSK) is a collection of command line tools for forensic analysis of disk images. It's core functionality is to "analyze volume and file system data [26]." Autopsy provides a graphical interface to TSK and various other forensic tools.

2.5.1 TSK

TSK's interesting feature is its plug-in framework that allows examiners to incorporate additional modules created by the open source community. TSK focuses on volume and file systems and produces information about files. The examiner must then use various tools with different interfaces to analyze the data at the application layer [26]. Because there are so many different file types and analysis techniques, with new ones developed everyday, it is impossible to have one tool that is a solution to all. TSK provides an open platform for modules to operate.

"The Sleuth Kit provides a plug-in framework that makes it easier to build end-to-end digital forensics solutions." [26]

TSK's framework documents provide detail specifics of how it operates, but the basics start with the design philosophy of the framework. The analysis process is split into the following three phases and illustrated in Figure 5:

- **File Extraction**: Files identified through file system data, carving, and other data recovery techniques.

- **File Analysis**: Each file analyzed individually through the pipeline
- **Post-Processing**: Results of analyzed individual files combined with analysis of entire image.

Communication between the modules within the framework is handled by the blackboard. It is a collection of *artifacts* which are associated to a file. And each of these artifacts contain a collection of name-value pairs called *attributes* [28]. Each file has various artifact types associated to it. TSK's framework provides a set of standard artifact types, such as TSK_GEN_INFO (which groups attributes that are related to the file but not to each other in any way). Module developers are able to create new artifact types if necessary, which require a unique type name and display name that does not have to be unique. TSK's framework also provides standard attribute types, which consists of a unique type name and a display name that does not have to be unique. Module developers are also able to create new attribute types as needed. These name-value pairs are closely related to its associated artifact. For

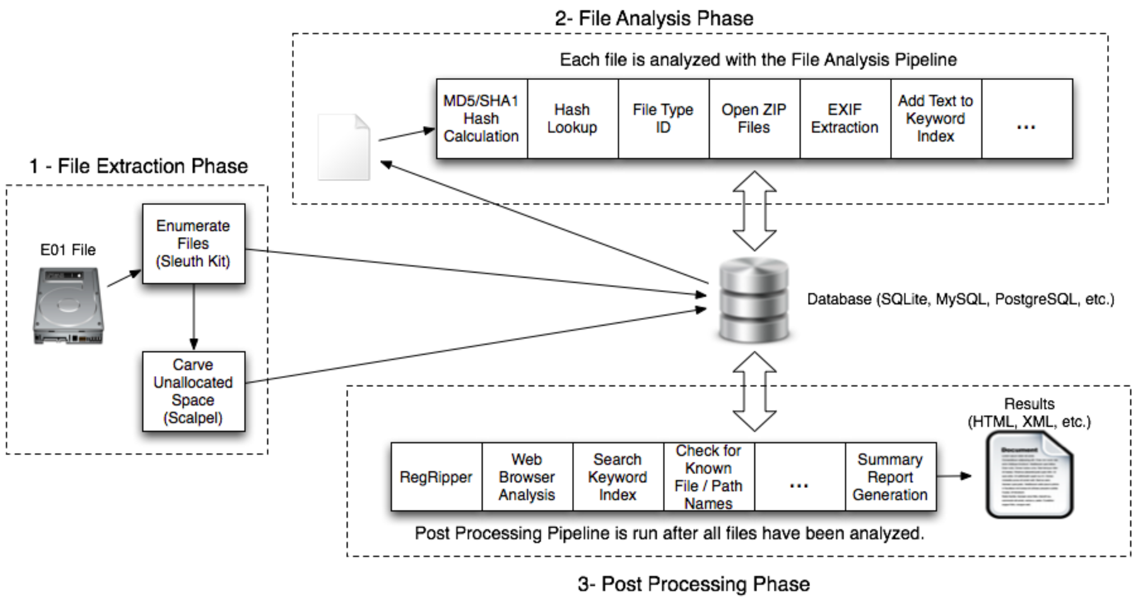

2- File Analysis Phase
Each file is analyzed with the File Analysis Pipeline
MD5/SHA1 Hash Calculation
Hash Lookup
File Type ID
Open ZIP Files
EXIF Extraction
Add Text to Keyword Index
...
1 - File Extraction Phase
E01 File
Enumerate Files (Sleuth Kit)
Carve Unallocated Space (Scalpel)
Database (SQLite, MySQL, PostgreSQL, etc.)
Results (HTML, XML, etc.)
RegRipper
Web Browser Analysis
Search Keyword Index
Check for Known File / Path Names
...
Summary Report Generation
Post Processing Pipeline is run after all files have been analyzed.
3- Post Processing Phase

Figure 5: The Sleuth Kit's Analysis Process Framework [27].

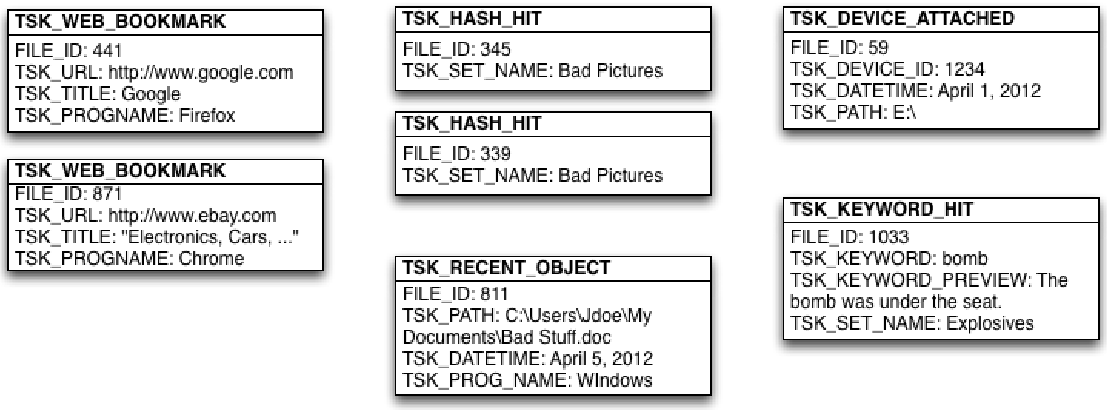
example, each individual log entry in a log file would be its own unique artifact, with the same artifact type, and each of these entries would have a collection of attributes such as date/time created associated to the artifacts. Figure 6 provides an illustration of the blackboard concept.

TSK's framework comes with several standard modules such as HashCalcModule, which calculates MD5 and SHA1 hashes for files. The standard tool TSK uses currently is tsk_analyseimg, which is a command line tool that loads a disk image into SQLite and runs various pipelines on each file [27]. The standard modules that come with TSK are rudimentary to what forensic tools are capable of because TSK is meant to be further developed through the open-source community, which leads us to Autopsy.

2.5.2 Autopsy

Autopsy is an open source digital forensics tool that uses TSK's framework and additional modules. The following are Autopsy's current features, taken from their documentation web page [29]:

- **Multi-User Cases**: Collaborate with fellow examiners on large cases.



TSK_WEB_BOOKMARK
FILE_ID: 441
TSK_URL: http://www.google.com
TSK_TITLE: Google
TSK_PROGNAME: Firefox

TSK_WEB_BOOKMARK
FILE_ID: 871
TSK_URL: http://www.ebay.com
TSK_TITLE: "Electronics, Cars, ..."
TSK_PROGNAME: Chrome

TSK_HASH_HIT
FILE_ID: 345
TSK_SET_NAME: Bad Pictures

TSK_HASH_HIT
FILE_ID: 339
TSK_SET_NAME: Bad Pictures

TSK_RECENT_OBJECT
FILE_ID: 811
TSK_PATH: C:\Users\Jdoe\My Documents\Bad Stuff.doc
TSK_DATETIME: April 5, 2012
TSK_PROG_NAME: WIndows

TSK_DEVICE_ATTACHED
FILE_ID: 59
TSK_DEVICE_ID: 1234
TSK_DATETIME: April 1, 2012
TSK_PATH: E:\

TSK_KEYWORD_HIT
FILE_ID: 1033
TSK_KEYWORD: bomb
TSK_KEYWORD_PREVIEW: The bomb was under the seat.
TSK_SET_NAME: Explosives

Figure 6: The Sleuth Kit's Blackboard [28].

- **Timeline Analysis**: Displays system events in a graphical interface to help identify activity.
- **Keyword Search**: Text extraction and index searched modules enable you to find files that mention specific terms and find regular expression patterns.
- **Web Artifacts**: Extracts web activity from common browsers to help identify user activity.
- **Registry Analysis**: Uses RegRipper to identify recently accessed documents and USB devices.
- **LNK File Analysis**: Identifies short cuts and accessed documents
- **Email Analysis**: Parses MBOX format messages, such as Thunderbird.
- **EXIF**: Extracts geo location and camera information from JPEG files.
- **File Type Sorting**: Group files by their type to find all images or documents.
- **Media Playback**: View videos and images in the application and not require an external viewer.
- **Thumbnail Viewer**: Displays thumbnail of images to help quick view pictures.
- **Robust File System Analysis**: Support for common file systems, including NTFS, FAT12/FAT16/FAT32/ExFAT, HFS+, ISO9660 (CD-ROM), Ext2/Ext3/Ext4, Yaffs2, and UFS from The Sleuth Kit.
- **Hash Set Filtering**: Filter out known good files using NSRL and flag known bad files using custom hashsets in HashKeeper, md5sum, and EnCase formats.
- **Tags**: Tag files with arbitrary tag names, such as 'bookmark' or 'suspicious', and add comments.

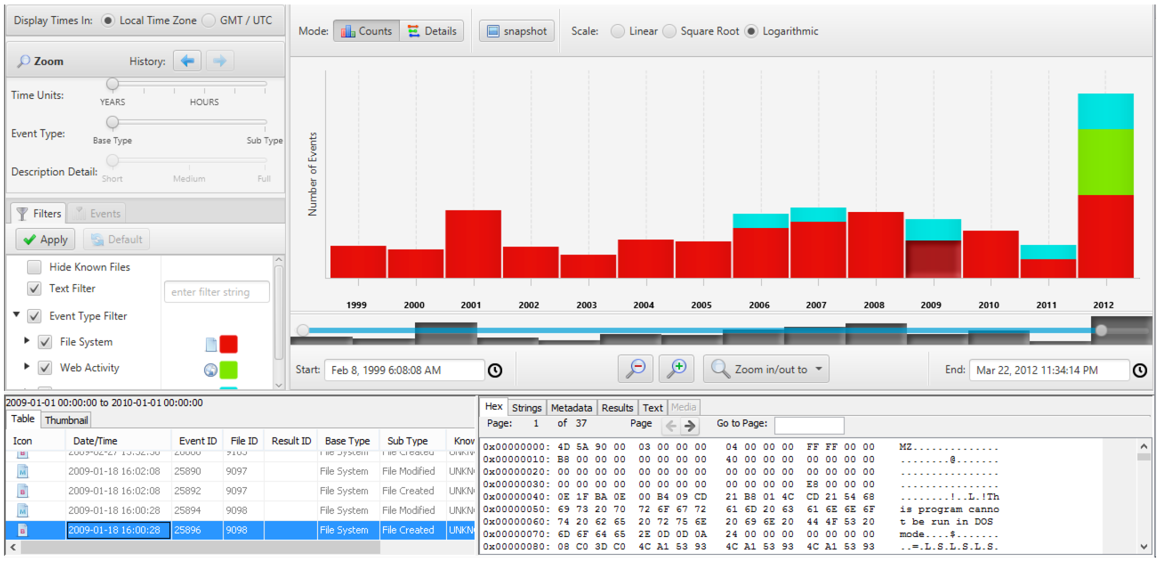
- **Unicode Strings Extraction**: Extracts strings from unallocated space and unknown file types in many languages (Arabic, Chinese, Japanese, etc.).
- **File Type Detection**: based on signatures and extension mismatch detection.
- **Interesting Files Module**: Flags files and folders based on name and path.
- **Android Support**: Extracts data from SMS, call logs, contacts, Tango, Words with Friends, and more.

Autopsy is able to support either raw/dd images or E01 format. Many of Autopsy's features stem from TSK's modules, such as its registry analysis and hash set filtering.

2.5.2.1 Timeline Analysis

Its feature of interest for this research is the timeline analysis, which utilizes timeline visualization to display results from the various modules used to analyze the disk image. The timeline pulls timestamps from files, web artifacts, and other data sets such as EXIF and GPS [30]. It provides two different timeline visualizations, a bar chart that displays the amount of data occurring in a given time frame, similar to a histogram and shown in Figure 7, and a timeline with detailed events, shown in Figure 8. Figure 8 includes criterion three mentioned in Section 2.3.2 and implements filterable attributes.

digital forensics timelines can overwhelm examiners with the amount of data they display. Autopsy's timeline visualization aims to alleviate the issue by allowing the events to be filtered not just by time, but also based on type, such as file system, web activity, or miscellaneous, which includes messages, GPS routes, location history, email, and more. Figure 9 shows the complete list.



Display Times In: ● Local Time Zone ○ GMT / UTC

Mode: Counts | Details — Scale: ○ Linear ○ Square Root ● Logarithmic

Start: Feb 8, 1999 6:08:08 AM — End: Mar 22, 2012 11:34:14 PM

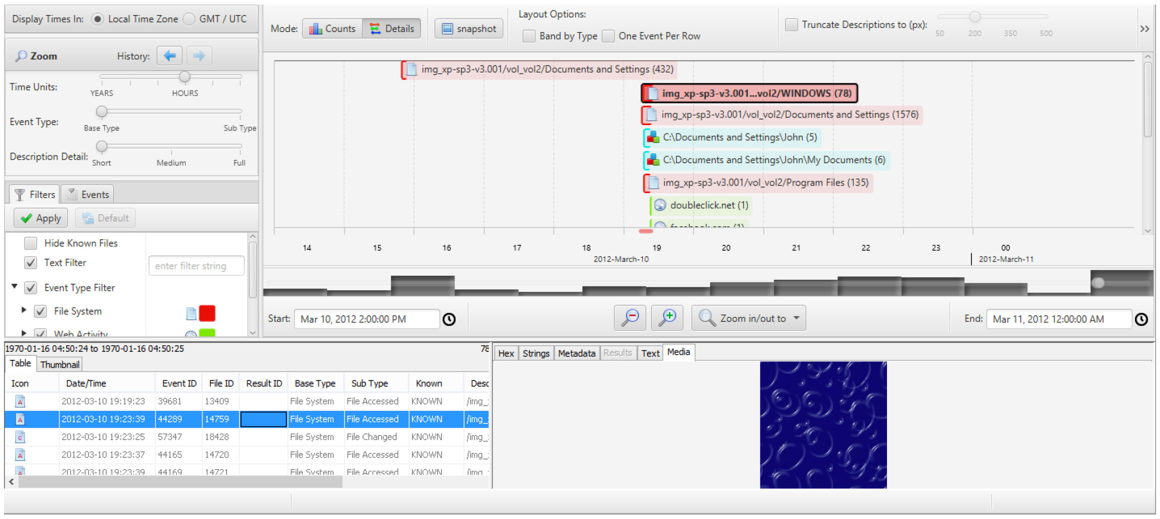
2009-01-01 00:00:00 to 2010-01-01 00:00:00

Icon	Date/Time	Event ID	File ID	Result ID	Base Type	Sub Type	Know
M	2009-01-18 16:02:08	25890	9097		File System	File Modified	UNKN
B	2009-01-18 16:02:08	25892	9097		File System	File Created	UNKN
M	2009-01-18 16:00:28	25894	9098		File System	File Modified	UNKN
B	2009-01-18 16:00:28	25896	9098		File System	File Created	UNKN

Page: 1 of 37

```
0x00000000: 4D 5A 90 00  03 00 00 00  04 00 00 00  FF FF 00 00   MZ..............
0x00000010: B8 00 00 00  00 00 00 00  40 00 00 00  00 00 00 00   ........@.......
0x00000020: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00   ................
0x00000030: 00 00 00 00  00 00 00 00  00 00 00 00  E8 00 00 00   ................
0x00000040: 0E 1F BA 0E  00 B4 09 CD  21 B8 01 4C  CD 21 54 68   ........!..L.!Th
0x00000050: 69 73 20 70  72 6F 67 72  61 6D 20 63  61 6E 6E 6F   is program canno
0x00000060: 74 20 62 65  20 72 75 6E  20 69 6E 20  44 4F 53 20   t be run in DOS 
0x00000070: 6D 6F 64 65  2E 0D 0D 0A  24 00 00 00  00 00 00 00   mode....$.......
0x00000080: 08 C0 3D C0  4C A1 53 93  4C A1 53 93  4C A1 53 93   ..=.L.S.L.S.L.S.
```

Figure 7: Autopsy's Timeline [30].



Display Times In: ● Local Time Zone ○ GMT / UTC

Mode: Counts | Details

img_xp-sp3-v3.001/vol_vol2/Documents and Settings (432)
img_xp-sp3-v3.001...vol2/WINDOWS (78)
img_xp-sp3-v3.001/vol_vol2/Documents and Settings (1576)
C:\Documents and Settings\John (5)
C:\Documents and Settings\John\My Documents (6)
img_xp-sp3-v3.001/vol_vol2/Program Files (135)
doubleclick.net (1)

Start: Mar 10, 2012 2:00:00 PM — End: Mar 11, 2012 12:00:00 AM

1970-01-16 04:50:24 to 1970-01-16 04:50:25

Icon	Date/Time	Event ID	File ID	Result ID	Base Type	Sub Type	Known	Desc
A	2012-03-10 19:19:23	39681	13409		File System	File Accessed	KNOWN	/img_
A	2012-03-10 19:23:39	44289	14759		File System	File Accessed	KNOWN	/img_
C	2012-03-10 19:23:25	57347	18428		File System	File Changed	KNOWN	/img_
A	2012-03-10 19:23:37	44165	14720		File System	File Accessed	KNOWN	/img_
A	2012-03-10 19:23:39	44169	14721		File System	File Accessed	KNOWN	/img_

Figure 8: Autopsy's Timeline [30].

Autopsy also groups similar events together to help prevent data overload. For example, it will cluster events from the same folder into a single event or URLS from the same domain are shown as a single event [30]. The examiner can zoom in to learn more details about the events, which covers the granular aspect mentioned in Section 2.3.2. The timeline is also filterable using keywords, which is detailed in the next section.

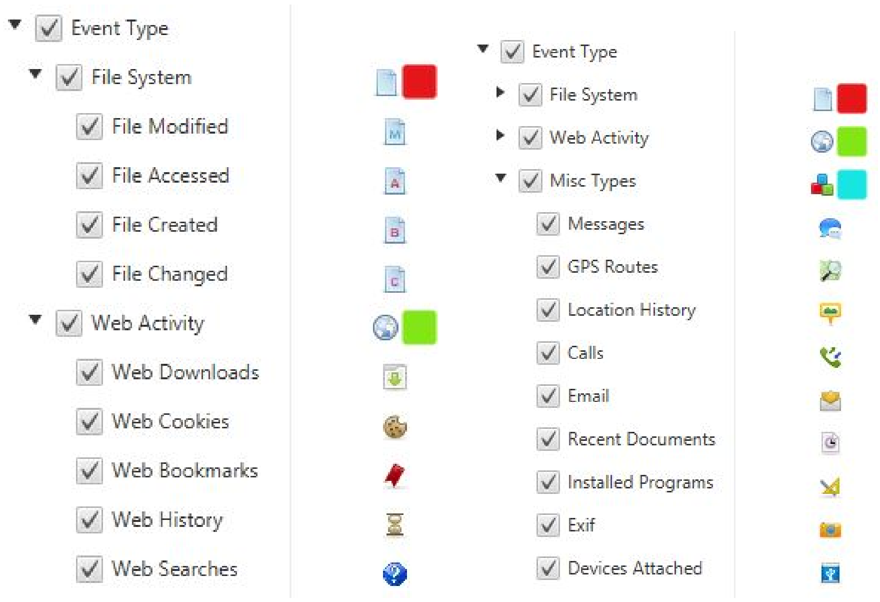


Figure 9: Timeline Filters.

Autopsy eventually hopes to integrate Plaso, another open source digital forensics tool for timeline generation. But currently, its documentation states more research is required because their method of clustering to prevent data overload does not work for possible arbitrary input types from Plaso. This is because the Autopsy's timeline generation is hard coded for the input types Autopsy produces, but eventually hopes to develop " more advanced clustering approach though so that we can leverage the parsing support from Plaso [30]." More about Plaso and its uses are explained in Section 2.6.1.

2.5.2.2 Keyword Search and Indexing

Autopsy uses Apache SOLR as the engine behind the keyword search features. Examiners can use pre-defined keyword lists or ad-hoc keyword searches and can be run directly either during or after the ingest process of the image. Apache SOLR also allows for regular expressions. All files containing text that Autopsy discovers is indexed into a SOLR database, where its text is extracted using Tika and other libraries. The actual keyword and regular expression searches are done over this text index [31]. Apache SOLR allows files to be indexed via JSON, XML, CVS, or binary over HTTP and users can retrieve the results via HTTP GET. The results are returned in either JSON, XML, CSV, or binary format [32].

2.6 Custom Forensic Tool

Two tools developed to address the timeline clustering issue are Property Graph Event Reconstruction (PGER) and Temporal Analysis Integration Management Application (TAIMA).

2.6.1 Property Graph Event Reconstruction

PGER abstracts user actions on digital media. "Ontological data representation and data normalization can provide a structured way to correlate digital artifacts [33]." Though this reduces the amount of data an examiner sorts through, the traditional method of ontology data processing requires large amounts of disk space and incurs high computational costs. PGER provides a solution by reducing the computational costs of processing events from a disk image by storing the data extracted into a native graph database, Neo4j, that improves query speeds.

2.6.1.1 Data Extraction

From the digital forensics process, it first extracts data using two different tools, Plaso or Temporal Event Abstraction and Reconstruction (TEAR). TEAR uses algorithms and pattern matching to help identify high-level events [34]. It is a C++ program that creates various CSV files by extracting artifacts from a disk image. The artifacts' data sources are from the file table, registry, and Windows events as well as Chrome and Firefox history [33].

Plaso on the other hand is a Python-based engine for log2timeline [35]. log2timeline extracts timestamps from the disk image, its partitions and volume shadow copies, and combines them into a unique data storage. Plaso then adds psort, which converts the unique Plaso data storage into an Elasticsearch database.

2.6.1.2 Graph Conversion

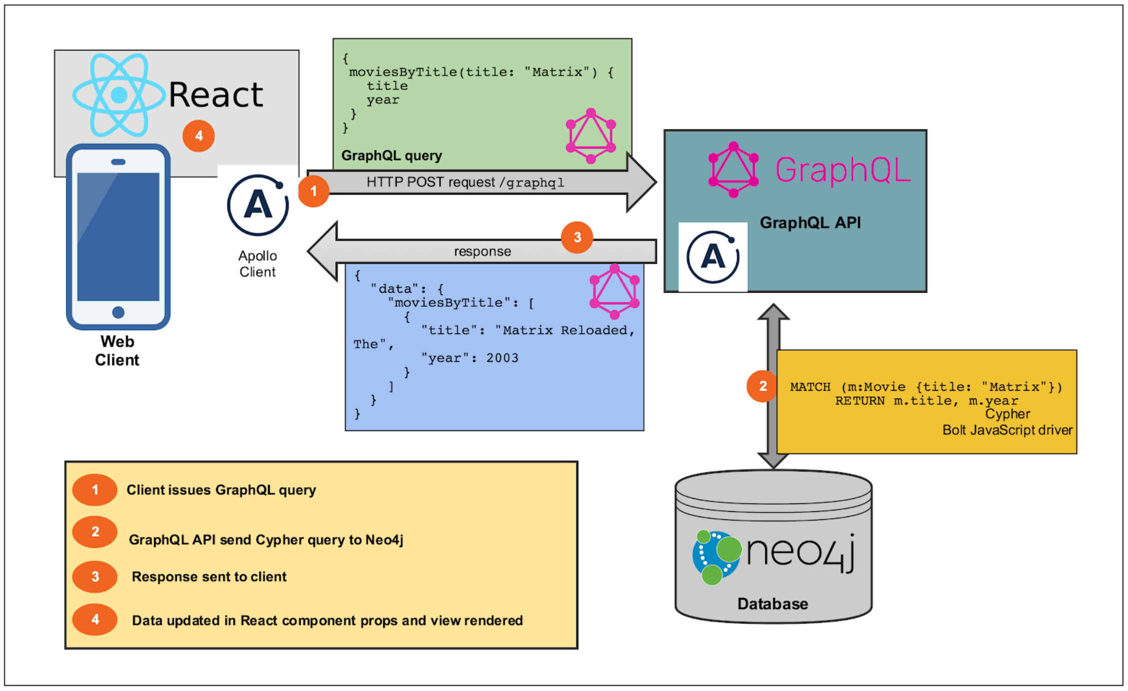
PGER converts the extracted data into Neo4j, a native graph database. Depending on what tool was utilized for the data extraction step, it uses two different tools for graph conversions. If the data was extracted using Plaso and the events are placed in an Elastic database, PGER requires logstash to convert the events into various subgraphs stored in Neo4j. If TEAR was utilized, PGER uses a custom Python script to convert the events. [33] explains the graph conversion process is dictated by the event type. Plaso stores its event type in parser field while TEAR identifies its event type in the filename. Based on these, different profiles and filters can be applied to properly parse the event data, allowing the user to pick which events appear in Neo4j. This allows the examiner to focus on a small set of relevant events rather than sorting through the entire image. PGER creates various object nodes for filenames of extracted artifacts, action nodes for specific actions taken at the timestamps, and parser nodes that explain the data's source of information. All of these nodes are related to each other through timestamps. This saves on the computational costs; for example, there is one parser node created for all the events extracted from the Windows Registry, instead of a parser node being created for each extracted registry event. Each of the registry events will have its own unique object node connected to the one parser node. But these artifacts are differentiated by the different action nodes (with a timestamp property) created that are connected to the object and parser nodes. The action nodes are also related to each other through a time tree, to make it easier to find relationships occurring in a specified time frame.

2.6.2 Temporal Analysis Integration Management Application

TAIMA [36] is an application built using a full stack development platform called GRANDstack. GRANDstack includes GraphQL, React, Apollo, and Neo4j [37]. GraphQL is a language for building Application Program Interface (API)s to query application data as a graph. It allows the developers to define types and available queries, allowing the user to only request data necessary. React, developed by Facebook, is a JavaScript library for building an interactive user interface (UI). The back-end database GRANDstack uses is Neo4j, a NoSQL, native open source graph database that allows developers to store and query data as a graph. Neo4j allows for complex graph travels using its Cypher query language, rather than an index search that comes with a traditional SQL database. Lastly, GRANDstack uses Apollo to interact with GraphQL. Apollo Client is used on the frontend to package queries into a GraphQL query and the server side GraphQL API is able to translate such requests. Figure 10 provides an illustrative overview of the full stack.

2.6.2.1 Data Abstraction

TAIMA creates high-level events based on specific predetermined logic using Cypher query searches. It creates abstracted events for program installation, power events for startup and shutdown, program executions, file downloads, and web history [36]. Each of these abstraction events are created from the events extracted using PGER. It searches the graph for specific object and parser node relationships connected to each other by action nodes. When all three are found, TAIMA creates a new high-level abstraction node describing the event in more human readable format, such as "Program Installation" or "System Startup".


React
4
Web
Client
Apollo
Client
1
HTTP POST request /graphql
GraphQL query
response
3
GraphQL
GraphQL API
2
Cypher
Bolt JavaScript driver
neo4j
Database
1 Client issues GraphQL query
2 GraphQL API send Cypher query to Neo4j
3 Response sent to client
4 Data updated in React component props and view rendered

```
{
 moviesByTitle(title: "Matrix") {
   title
   year
 }
}
```

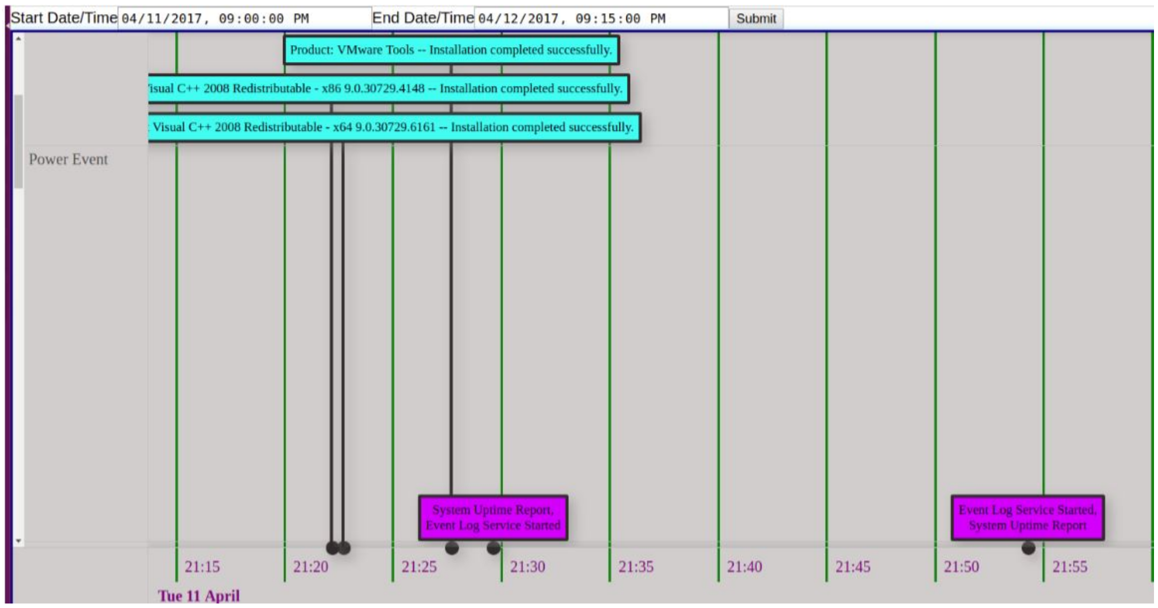
```
{
  "data": {
    "moviesByTitle": [
      {
        "title": "Matrix Reloaded,
The",
        "year": 2003
      }
    ]
  }
}
```

```
MATCH (m:Movie {title: "Matrix"})
  RETURN m.title, m.year
```

Figure 10: GRANDstack Architecture [38].

2.6.2.2 Timeline Visualization

TAIMA uses React to develop a UI for the examiners to interact with. Currently it allows the high-level abstraction nodes to be filtered based on a time range as shown in Figure 11. React allows TAIMA to enable tooltips, accessed by mousing over an event, which displays the abstracted events source. For example, for a “System Uptime Report, Event Log Service Started” abstraction event, mousing over it displays its source as “EventLog/6013 and EventLog/6005”. TAIMA is also able to interactively filter the time range displayed depending on the level zoom. This allows the examiners to search a time frame of interest and zoom in on areas with clusters of activities for increased scrutiny.


Start Date/Time 04/11/2017, 09:00:00 PM End Date/Time 04/12/2017, 09:15:00 PM

Product: VMware Tools -- Installation completed successfully.

isual C++ 2008 Redistributable - x86 9.0.30729.4148 -- Installation completed successfully.

Visual C++ 2008 Redistributable - x64 9.0.30729.6161 -- Installation completed successfully.

System Uptime Report. Event Log Service Started

Event Log Service Started. System Uptime Report

Figure 11: TAIMA Timeline [36].

2.6.3 Neo4j Full-Text Index

Both [36] and [33] explain Neo4j and its capabilities well but do not talk about Neo4j's full-text search index capabilities as it is a newer feature introduced with Neo4j version 3.5. Neo4j's full-text index is powered by Apache Lucene and supports various features such as "indexing of both nodes and relationships, can be queried using Lucene query language or Cypher procedures, kept up to date automatically as nodes and relationships are added, removed, or updated, and can support any number of documents in a single index." [39]. Neo4j will tokenize the indexed string values, allowing the database to match within the contents of indexed string properties, and does not require a literal string match [39]. The full-text search index can be created using a simple Cypher procedure such as "CALL db.index.fulltext.createNodeIndex("nameOfIndex", ["nodeLabel", "anotherNodeLabel"], ["indexedProperty", "anotherIndexedProperty"]". In this example, Neo4j will consider any nodes labeled as "nodeLabel" and/or "anotherNodeLabel" and search them for the two properties mentioned in the next argument. Those properties and the strings it contains would be indexed. It is important to note Neo4j will index any nodes with the one or both labels and if they have values in one or both of the properties specified. Because this index is powered by Lucene, it supports wildcard, regular expression, fuzzy, and proximity searches as well as boolean terms.

2.7 Summary

This chapter went over the various common phases involved in digital forensics, which were pre-process, acquisition and preservation, analysis, presentation, and post-process. It then went over information visualization and how it improves examiners efficiency with conducting digital forensics. The chapter looked into the four rules of a forensic computing as well as the various evaluation criteria for forensic soundness. It also went over the different popular forensic tools and how their employment of visualization timelines were lacking. Lastly, the chapter reviewed the features within in Neo4j's full-text index, the data abstraction done with PGER, and TAIMA's information visualization methods.

III. Integration

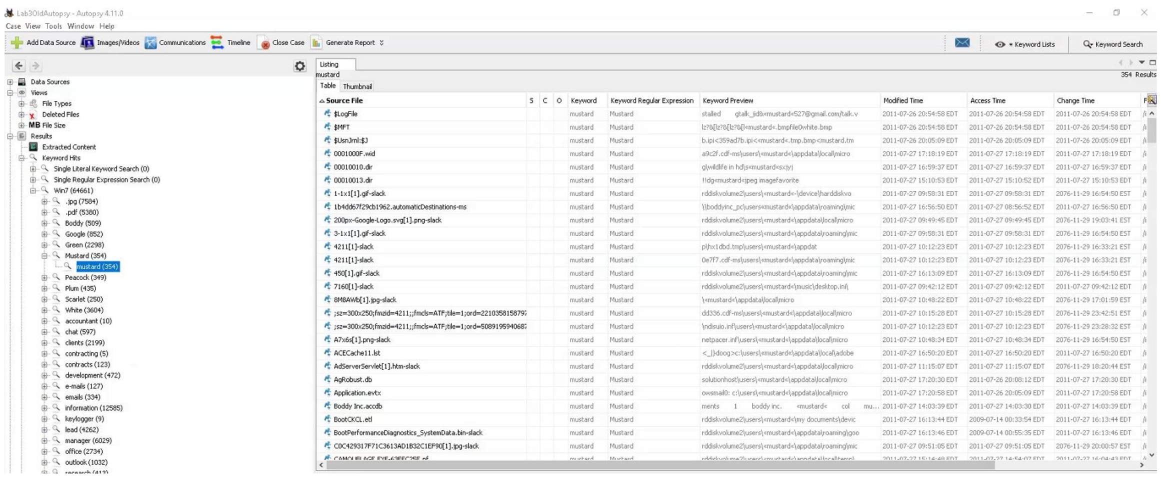
3.1 Overview

The purpose of this study is to reduce the human analysis portion of a digital forensics examination. Current practices involve string searches to identify potential evidentiary items, then taking them through various steps to determine its evidentiary value. The research looks to combine the string searching capabilities that initiate an investigation with automated correlation and abstraction logic to further streamline this process. Event Correlation and Abstraction Timeline Visualization (ECATV) uses the output of Autopy's keyword search ingest and combines it with the automated correlation capabilities of Property Graph Event Reconstruction (PGER) and the user interface of Temporal Analysis Integration Management Application (TAIMA) to provide a more complete tool. Autopsy has a timeline, as mentioned in Section 2.5.2.1 but its features are essentially a histogram or the same information generated by various Autopsy modules placed on a timeline. It provides an easier view of which artifacts occurred when but still suffers from information overload issues. The automated correlation helps reduce the number of events on a timeline while maintaining accuracy and the abstraction logic takes the concept a step further by collecting similar events that occurred at the same timestamp.

The overall design of ECATV is described below and is further broken down into the details of its ingest process as well as its user interface. It is assumed that the evidence and data was obtained through a forensically sound process described in Chapter 2. This study is more focused on the analysis phase of the process. The study also processed and analyzed Windows images as this is the more prevalent operating system and continues on the research of [36] and [33].

3.2 Design

ECATV includes the capabilities of many tools using the benefits discovered by Adderley [36] and Schelkoph [33] to create a granular and filterable visual timeline. The full design concept can be split between the ingest phase and the user interface (UI). The ingest phase contains the bulk of the behind-the-scenes processing of taking a raw, physical image and converting its artifacts into a form that provides examiners with relevant information faster than a traditional database with key-value pairs, such as Figure 12. The traditional database is a double-edged sword that provides a plethora of information which will eventually be necessary during an examination but at the same time overloads the examiner making it difficult to find a place to start. It also provides a poor overview of user(s) activity within the image. ECATV aims to resolve these issues by using the abstraction methods in [33] and the visualization method of a timeline used in [36] but also maintaining the value traditional databases provide.



Source File	S	C	O	Keyword	Keyword Regular Expression	Keyword Preview	Modified Time	Access Time	Change Time	F
$LogFile				mustard	Mustard	stalled gtalk_jid&«mustard«527@gmail.com/talk.v	2011-07-26 20:54:58 EDT	2011-07-26 20:54:58 EDT	2011-07-26 20:54:58 EDT	/i
$MFT				mustard	Mustard	lz?&[lz?&[lz?&[l«mustard«.bmpfile0white.bmp	2011-07-26 20:54:58 EDT	2011-07-26 20:54:58 EDT	2011-07-26 20:54:58 EDT	/i
$UsnJrnl:$J				mustard	Mustard	b.ipi<359ad7b.ipi<«mustard«.tmp.bmp<mustard.tm	2011-07-26 20:05:09 EDT	2011-07-26 20:05:09 EDT	2011-07-26 20:05:09 EDT	/i
0001000F.wid				mustard	Mustard	a9c2f.cdf-ms\users\«mustard«\appdata\local\micro	2011-07-27 17:18:19 EDT	2011-07-27 17:18:19 EDT	2011-07-27 17:18:19 EDT	/i
00010010.dir				mustard	Mustard	g\wildlife in hd\s«mustard«sxjyj	2011-07-27 16:59:37 EDT	2011-07-27 16:59:37 EDT	2011-07-27 16:59:37 EDT	/i
00010013.dir				mustard	Mustard	!!dg«mustard«jpeg imagefavorite	2011-07-27 15:10:53 EDT	2011-07-27 15:10:52 EDT	2011-07-27 15:10:53 EDT	/i
1-1x1[1].gif-slack				mustard	Mustard	rddiskvolume2\users\«mustard«-\device\harddiskvo	2011-07-27 09:58:31 EDT	2011-07-27 09:58:31 EDT	2076-11-29 16:54:50 EST	/i
1b4dd67f29cb1962.automaticDestinations-ms				mustard	Mustard	\\boddyinc_pc\users«mustard«\appdata\roaming\mic	2011-07-27 16:56:50 EDT	2011-07-27 08:56:52 EDT	2011-07-27 16:56:50 EDT	/i
200px-Google-Logo.svg[1].png-slack				mustard	Mustard	rddiskvolume2\users\«mustard«\appdata\local\micro	2011-07-27 09:49:45 EDT	2011-07-27 09:49:45 EDT	2076-11-29 19:03:41 EST	/i
3-1x1[1].gif-slack				mustard	Mustard	rddiskvolume2\users\«mustard«\appdata\roaming\mic	2011-07-27 09:58:31 EDT	2011-07-27 09:58:31 EDT	2076-11-29 16:54:50 EST	/i
4211[1]-slack				mustard	Mustard	p\hx1dbd.tmp\users\«mustard«\appdat	2011-07-27 10:12:23 EDT	2011-07-27 10:12:23 EDT	2076-11-29 16:33:21 EST	/i
4211[1]-slack				mustard	Mustard	0e7f7.cdf-ms\users\«mustard«\appdata\roaming\mic	2011-07-27 10:12:23 EDT	2011-07-27 10:12:23 EDT	2076-11-29 16:33:21 EST	/i
450[1].gif-slack				mustard	Mustard	rddiskvolume2\users\«mustard«\appdata\roaming\mic	2011-07-27 16:13:09 EDT	2011-07-27 16:13:09 EDT	2076-11-29 16:54:50 EST	/i
7160[1]-slack				mustard	Mustard	rddiskvolume2\users\«mustard«\music\desktop.ini\	2011-07-27 09:42:12 EDT	2011-07-27 09:42:12 EDT	2011-07-27 09:42:12 EDT	/i
8M8AWb[1].jpg-slack				mustard	Mustard	\«mustard«\appdata\local\micro	2011-07-27 10:48:22 EDT	2011-07-27 10:48:22 EDT	2076-11-29 17:01:59 EST	/i
;sz=300x250;fmzid=4211;;fmcls=ATF;tile=1;ord=221035815879				mustard	Mustard	dd336.cdf-ms\users\«mustard«\appdata\local\micro	2011-07-27 10:15:28 EDT	2011-07-27 10:15:28 EDT	2076-11-29 23:42:51 EST	/i
;sz=300x250;fmzid=4211;;fmcls=ATF;tile=1;ord=508919594068				mustard	Mustard	\ndisuio.inf\users\«mustard«\appdata\local\micro	2011-07-27 10:12:23 EDT	2011-07-27 10:12:23 EDT	2076-11-29 23:28:32 EST	/i
A7x6s[1].png-slack				mustard	Mustard	netpacer.inf\users\«mustard«\appdata\local\micro	2011-07-27 10:48:34 EDT	2011-07-27 10:48:34 EDT	2076-11-29 16:54:50 EST	/i
ACECache11.lst				mustard	Mustard	<_|}doog>c:\users\«mustard«\appdata\local\adobe	2011-07-27 16:50:20 EDT	2011-07-27 16:50:20 EDT	2011-07-27 16:50:20 EDT	/i
AdServerServlet[1].htm-slack				mustard	Mustard	rddiskvolume2\users\«mustard«\appdata\local\micro	2011-07-27 11:15:07 EDT	2011-07-27 11:15:07 EDT	2076-11-29 18:20:44 EST	/i
AgRobust.db				mustard	Mustard	solutionhost\users\«mustard«\appdata\local\micro	2011-07-27 17:20:30 EDT	2011-07-26 20:08:12 EDT	2011-07-27 17:20:30 EDT	/i
Application.evtx				mustard	Mustard	owsmail0: c:\users\«mustard«\appdata\local\micro	2011-07-27 17:20:58 EDT	2011-07-26 20:05:09 EDT	2011-07-27 17:20:58 EDT	/i
Boddy Inc.accdb				mustard	Mustard	ments 1 boddy inc. «mustard« col mu...	2011-07-27 14:03:39 EDT	2011-07-27 14:03:30 EDT	2011-07-27 14:03:39 EDT	/i
BootCKCL.etl				mustard	Mustard	rddiskvolume2\users\«mustard«\my documents\devic	2011-07-27 16:13:44 EDT	2009-07-14 00:33:54 EDT	2011-07-27 16:13:44 EDT	/i
BootPerformanceDiagnostics_SystemData.bin-slack				mustard	Mustard	rddiskvolume2\users\«mustard«\appdata\roaming\goo	2011-07-27 16:13:46 EDT	2009-07-14 00:55:35 EDT	2011-07-27 16:13:46 EDT	/i
C0C429317F71C3613AD1B32C1EF90[1].jpg-slack				mustard	Mustard	rddiskvolume2\users\«mustard«\appdata\roaming\mic	2011-07-27 09:51:05 EDT	2011-07-27 09:51:05 EDT	2076-11-29 20:00:57 EST	/i
CAMOUFLAGE EYE-63FEC25F.nf				mustard	Mustard	rddiskvolume2\users\«mustard«\appdata\local\temp\	2011-07-27 15:14:48 EDT	2011-07-27 14:54:07 EDT	2011-07-27 16:04:43 EDT	/i

Figure 12: Autopsy Example.

3.2.1 Tools

The tools involved in ECATV are:

- **Disk Imager**: Specifically used AccessData's FTK Imager.
- **Temporal Event Abstraction and Reconstruction (TEAR)**: A custom tool developed by Okolica to extract artifacts [34].
- **Plaso**: Another tool to extract artifacts and timestamps.
- **The Sleuth Kit (TSK)/Autopsy**: A digital forensics software for image analysis.
- **Neo4j**: A native graph database.
- **GraphQL**: API to interface with Neo4j.
- **Apollo**: Integrates GraphQL with React.
- **React**: Used to develop front-end UI.
- **docker**: [33] developed various containers to compartmentalize its process.

3.2.2 Integration Process

ECATV used FTK Imager to create raw dd images for its Windows hard drives. As mentioned earlier, evidence acquisition is not the focus of this study and it is assumed examiners have their own method of acquiring images. ECATV will work regardless of the acquisition tool as long as the images are a raw physical copy. The next step is to process the images through TEAR and Plaso. The extracted artifacts are then placed into Neo4j using specific filters to create various sub graphs of various events within the image. These sub graphs consist of action, parser, time, and object nodes, as illustrated in Figure 13. Schelkoph [33] explains the time nodes are a unix time stamp and represents when the action occurred. The action node contains the description of the action that affects the digital object. The object node is a digital object identified by a URL, file path, registry key, log event, or some other object. Finally the parser node contains the origin of the information. Then Cypher queries, Neo4j's graph query language, are used to create high-level abstracted events, similar to ones in [36].

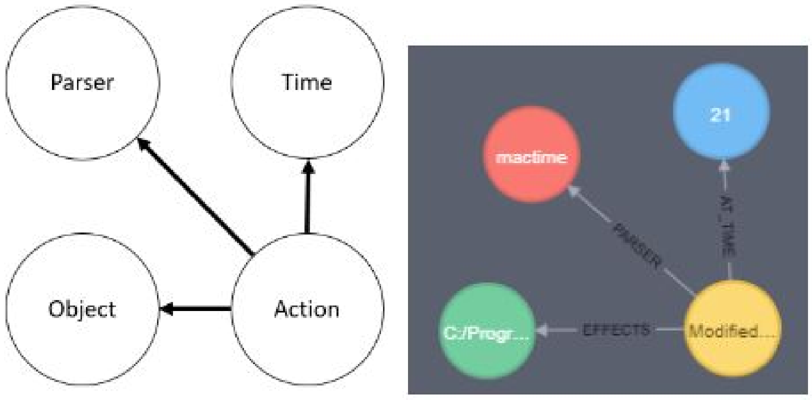


Figure 13: Neo4j Subgraph [33].

Autopsy's keyword search module is used to discover artifacts containing keywords from an examiner's pre-defined list. The high-level abstract events that contain the artifacts associated to the keywords are linked to the same keywords within Neo4j. This is done by adding a new searchKeyword property to the abstraction nodes in the Neo4j database. Finally a text index is created on the searchKeyword property to allow for text searches within the graph database. The ingest phase of ECATV creates the final back-end database the timeline visualization uses to query and display its data. ECATV uses a modified UI from [36], allowing examiners to analyze the high-level events from the image, sorted by a user specified time range. The examiner can then further filter this timeline view with desired text searches. Figure 14 provides an overview of the entire process.

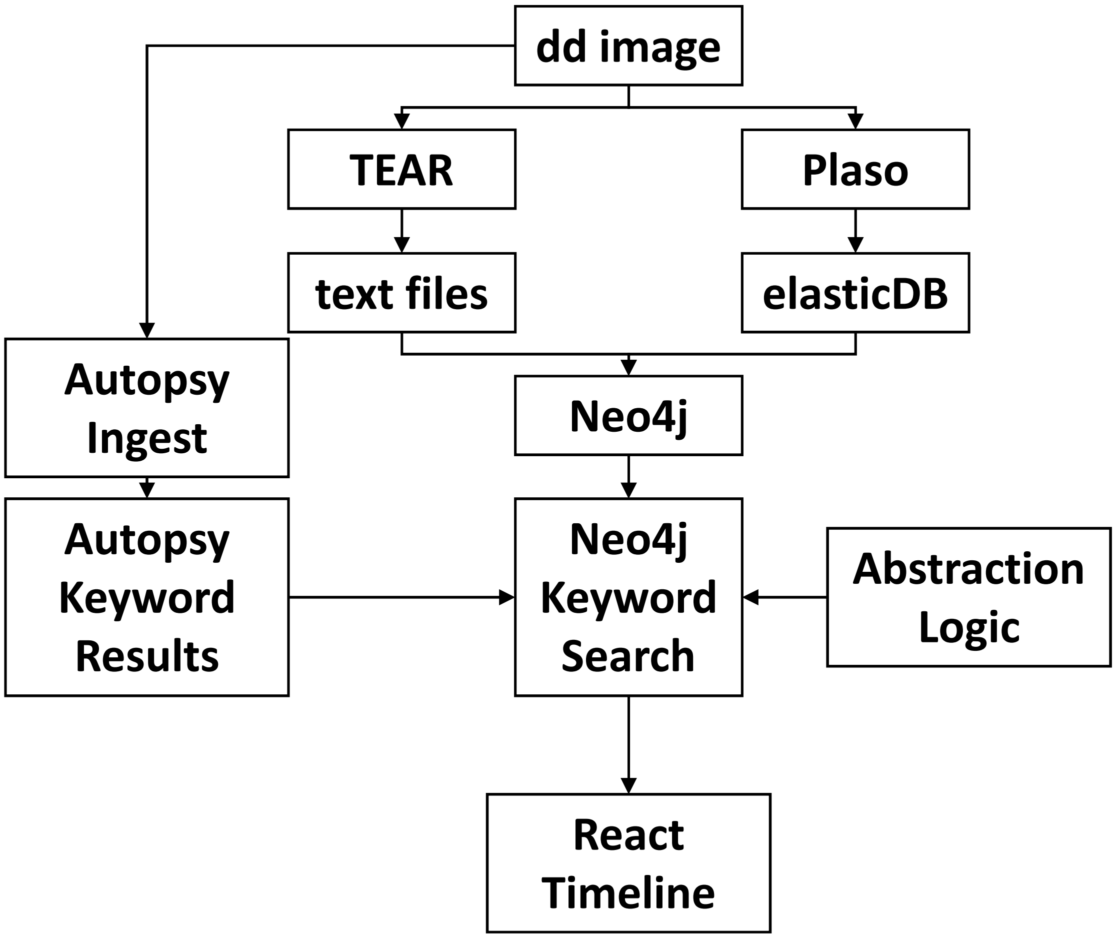


Figure 14: ECATV Flow Chart.

3.3 Ingest

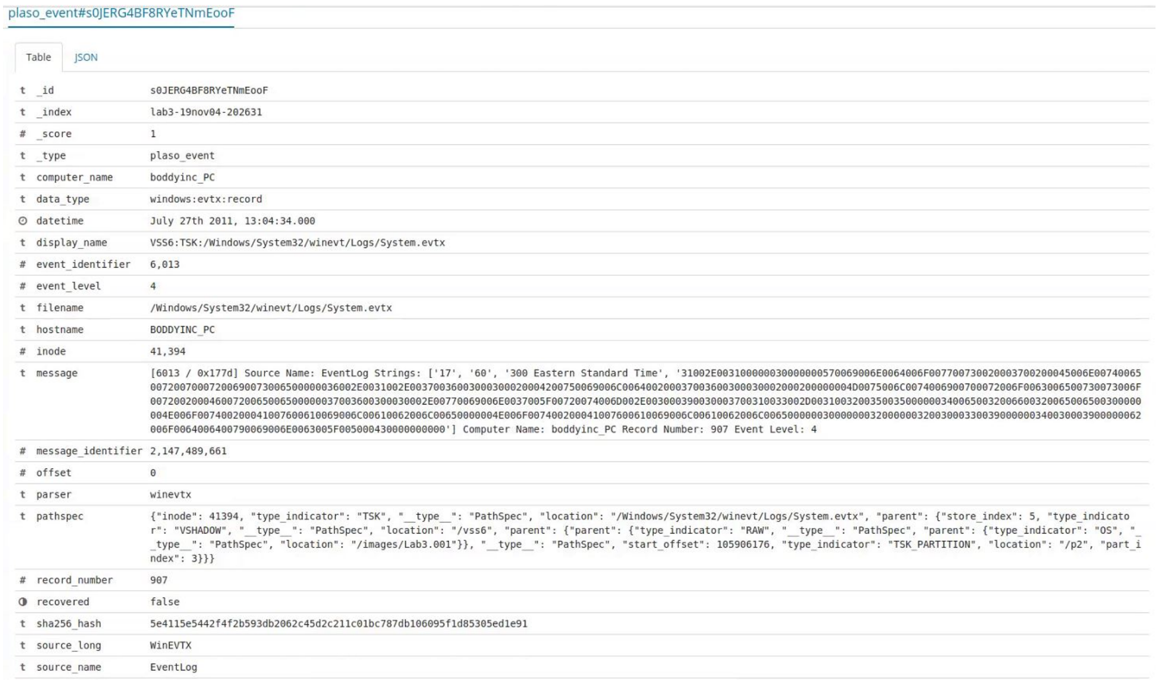
The ingest phase of the process can be split into two parts. The first part involves extracting the events of a raw physical image. Then this data is transformed and packaged into a form that makes use of Neo4j's native graph database capabilities. ECATV makes use of the research done by Schelkoph [33] and adapts its tools to do so.

The second part involves using outputs from Neo4j and applying abstraction logic to further the robustness of this database. The abstraction logic are adaptations of Adderley's research [36]. Lastly, it uses the outputs from Autopy's keyword search module to identify objects associated to specific keywords, helping set the groundwork for a keyword filterable timeline.

3.3.1 Phase One of Ingest

PGER utilizes docker containers to build its native graph database. A "Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings."[40] Docker containers are essentially packaged software that allows code to be packaged into a single unit and allows applications to be run from one computing environment to another. Because of this, ECATV was able to pick and choose the containers necessary to build the graph database and cut down on the processing time rather than processing an image through PGER's entirety. ECATV used containers "elasticDB", "plaso", and "neo4jInput".

The elasticDB container contained code setting up the ELK Stack. The ELK stack contains Elasticsearch, Kibana, Beats, and Logstash [41]. Elasticsearch is the "heart" of the ELK Stack and centrally stores the data produced by Plaso. Kibana is a front-end graphical UI that allows the users to easily view the data in the Elasticsearch database. Finally, logstash is the data-processing pipeline that ingests data from sources, transform the data through the use of filters, and outputs the new data into a database. The elasticDB container set up the environment for Plaso to input the data produced by log2timeline, mentioned in Section 2.6.1.1, into an Elasticserach database through psort. Once Plaso processed the image, this study examined the contents of Elasticsearch to determine how to configure the filters of logstash to identify the necessary data and to transform said data for Neo4j. Figure 15 shows an example of how Plaso extracted a Windows Eventlog artifact from the image and placed it into the Elasticsearch database.



```
t  _id                 s0JERG4BF8RYeTNmEooF
t  _index              lab3-19nov04-202631
#  _score              1
t  _type               plaso_event
t  computer_name       boddyinc_PC
t  data_type           windows:evtx:record
⊙  datetime            July 27th 2011, 13:04:34.000
t  display_name        VSS6:TSK:/Windows/System32/winevt/Logs/System.evtx
#  event_identifier    6,013
#  event_level         4
t  filename            /Windows/System32/winevt/Logs/System.evtx
t  hostname            BODDYINC_PC
#  inode               41,394
t  message             [6013 / 0x177d] Source Name: EventLog Strings: ['17', '60', '300 Eastern Standard Time', '31002E0031000000300000005700069006E0064006F007700730020003700200045006E00740065
                       007200700072006900730065000000036002E0031002E0037003600300030002000420075006900C006400200037003600300030002000200000004D0075006C007400690070007200 6F006300650073007300 6F
                       00720020004600720065006500000037003600300030002E00770069006E0037005F00720074006D002E0030003900300037003100 33002D003100320035003500000034006500320066003200650065003000 00
                       004E006F007400200041007600610069006C00610062006C00650000004E006F007400200041007600610069006C00610062006C006500000003000000032000000320030003300390000000340030003900000062
                       006F006400640079006 9006E0063005F005000430000000000'] Computer Name: boddyinc_PC Record Number: 907 Event Level: 4
#  message_identifier  2,147,489,661
#  offset              0
t  parser              winevtx
t  pathspec            {"inode": 41394, "type_indicator": "TSK", "__type__": "PathSpec", "location": "/Windows/System32/winevt/Logs/System.evtx", "parent": {"store_index": 5, "type_indicato
                       r": "VSHADOW", "__type__": "PathSpec", "location": "/vss6", "parent": {"parent": {"type_indicator": "RAW", "__type__": "PathSpec", "parent": {"type_indicator": "OS", "_
                       _type__": "PathSpec", "location": "/images/Lab3.001"}}, "__type__": "PathSpec", "start_offset": 105906176, "type_indicator": "TSK_PARTITION", "location": "/p2", "part_i
                       ndex": 3}}}
#  record_number       907
◑  recovered           false
t  sha256_hash         5e4115e5442f4f2b593db2062c45d2c211c01bc787db106095f1d85305ed1e91
t  source_long         WinEVTX
t  source_name         EventLog
```

Figure 15: Plaso Example.

The last container, neo4jInput, handled the process of taking the data from the Elasticsearch database, changing its format, and placing it into a Neo4j database output, all through the logstash pipeline. The filters queried the Elasticsearch database for specific instances and transformed its output into various Cypher queries utilized to create new nodes in Neoj or update existing ones with new properties and/or relationships to complete the various subgraphs like the one in Figure 13. Figure 16 is an example filter used to find various artifacts within the Elasticsearch database that Plaso extracted using the parser “lnk”. These events show when link shortcut files, pointing to an executable file, were accessed within the Windows image. Figure 17 shows an example of this filter’s end result. It created one parser node in the Neo4j database, containing the parserName property with a value of the string “lnk”. This node was associated to all the different action nodes with the properties, “action” and “timestamp”. Each action node represented a different entry within the Elasticsearch database that was extracted with Plaso’s “lnk” parser. Each of these action nodes were also associated to an object node, containing the file path of the shortcut accessed. If the same shortcut was accessed multiple instances throughout the image, the filter only created one distinct object node but associated it to various action nodes which were distinguishable by the different timestamps.

```
if [parser] == "lnk" {
  #complete standard parameters: filename, parser, and action
  translate {
    add_field => {
      "[statement1][parameters][objProps][filename]" => "%{filename}"
      "[statement1][parameters][parserProps][parserName]" => "%{parser}"
    }
    field => "timestamp_desc"
    destination => "[statement1][parameters][actProps][action]"
    dictionary => [ "Creation Time", "Link Created",
                    "Content Modification Time", "Link Modified",
                    "Last Access Time", "Link Accessed"
    ]
    fallback => "Unknown Action: %{timestamp_desc}"
  }
  #create queries and properties to link obj to link target
  if [link_target] {
    #find the target of the linkfile
    grok { match => { "link_target" => [ "%{GREEDYDATA}? %{PATH:linkTarget}",
                                         "%{GREEDYDATA:linkTarget}" ] } }
    #change link target path to match filestat (TSK)
    mutate { gsub => [ "linkTarget", "^.*:", "", "linkTarget", "[\\]", "/" ] }

    mutate {
      add_field => {
        "[statement1][parameters][lnkTgtProps][filename]" => "%{linkTarget}"
        "ss4" => "MERGE (lnkTgt:object {filename: $lnkTgtProps.filename}) ON CREATE SET
        lnkTgt=$lnkTgtProps"
        "ss5" => "MERGE (obj)-[:TARGET]->(lnkTgt)"
      }
    }
    #update statement1 to add the new queries
    mutate {
      update => { "[statement1][statement]" => "%{[statement1][statement]} %{ss4}
      %{ss5}" }
    }
  }
}
```

Figure 16: LNK Filter [33].

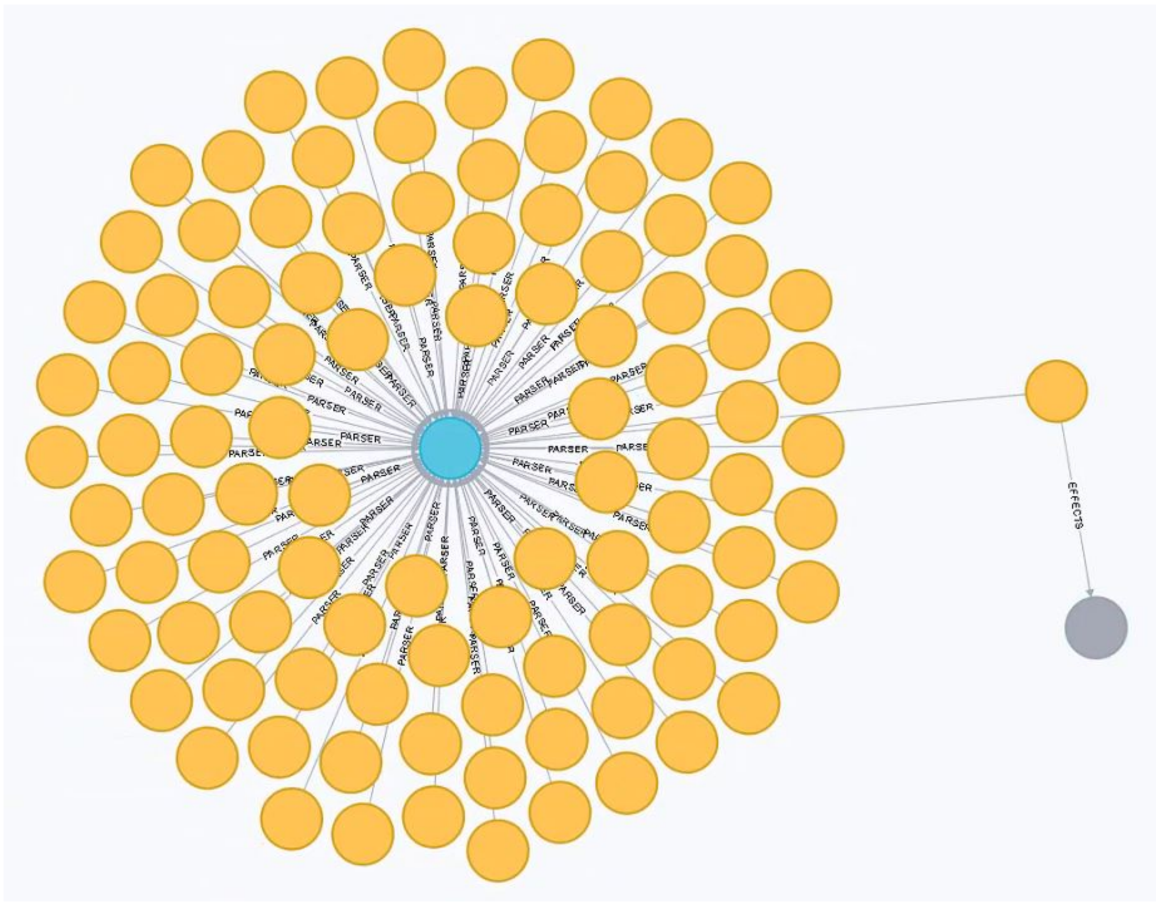
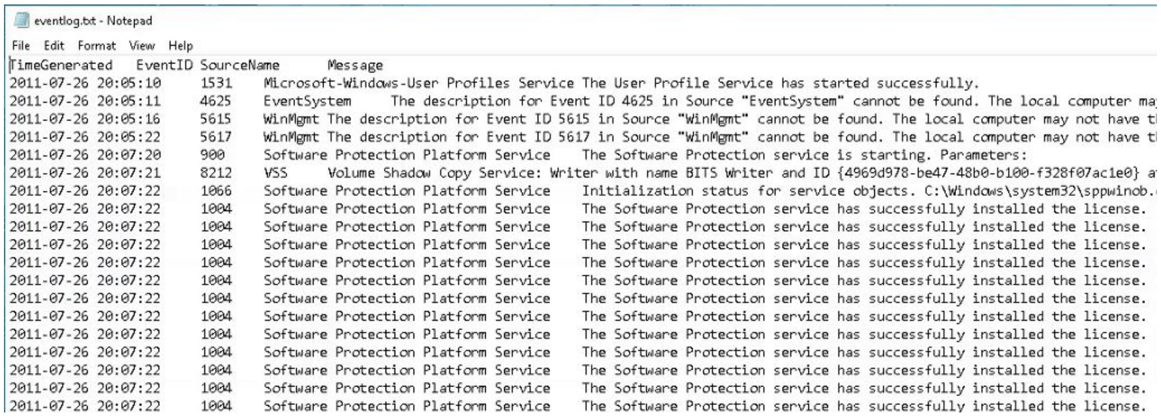


Figure 17: Neo4j LNK Example.

The dd image was then processed through TEAR, the custom tool developed by Okolica [34]. This C++ program extracted data such as Windows Event Logs, prefetch instances, Windows Registry information, and various Windows user data. The information was then collected and placed into various text files in CSV format. Figure 18 show an example of the Windows Event Log data collected from the image using TEAR. Logstash was used again to transform the collected events from TEAR into Cypher queries for Neo4j. Figure 19 shows the filter utilized for Windows Event Log data.



```
TimeGenerated    EventID SourceName      Message
2011-07-26 20:05:10     1531    Microsoft-Windows-User Profiles Service The User Profile Service has started successfully.
2011-07-26 20:05:11     4625    EventSystem     The description for Event ID 4625 in Source "EventSystem" cannot be found. The local computer ma
2011-07-26 20:05:16     5615    WinMgmt The description for Event ID 5615 in Source "WinMgmt" cannot be found. The local computer may not have t
2011-07-26 20:05:22     5617    WinMgmt The description for Event ID 5617 in Source "WinMgmt" cannot be found. The local computer may not have t
2011-07-26 20:07:20     900     Software Protection Platform Service    The Software Protection service is starting. Parameters:
2011-07-26 20:07:21     8212    VSS     Volume Shadow Copy Service: Writer with name BITS Writer and ID {4969d978-be47-48b0-b100-f328f07ac1e0} a
2011-07-26 20:07:22     1066    Software Protection Platform Service    Initialization status for service objects. C:\Windows\system32\sppwinob.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
2011-07-26 20:07:22     1004    Software Protection Platform Service    The Software Protection service has successfully installed the license.
```

Figure 18: EventLog Txt.

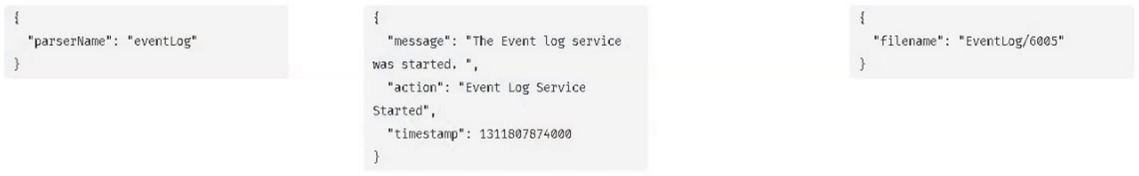
```
if [path] =~ /.*eventlog\.txt/ {
  mutate { gsub => [ "message", "\t", "|", "message", '"', "'" ] }
  csv {
    separator => "|"
    columns => [
      "TimeGenerated",
      "EventID",
      "SourceName",
      "Message"
    ]
  }
  if [TimeGenerated] {
    mutate {
      add_field => {
        "[statement1][parameters][objProps][filename]" => "%{SourceName}/%{EventID}"
        "[statement1][parameters][parserProps][parserName]" => "eventLog"
        "[statement1][parameters][actProps][message]" => "%{Message}"
      }
    }
    ruby { code => '
      timeStr = event.get("TimeGenerated")
      timeInt = (Time.strptime(timeStr, "%Y-%m-%d %H:%M:%S").to_i + 25200) * 1000
      event.set("[statement1][parameters][actProps][timestamp]", timeInt)
      '
    }
    if [SourceName] == "Microsoft-Windows-Winlogon" {
      translate {
        field => "[EventID]"
        destination => "[statement1][parameters][actProps][action]"
        dictionary => [
          "7001", "Logon CEI Notification",
          "7002", "Logoff CEI Notification"
        ]
        fallback => "Unknown Action: %{EventID}"
      }
    }
    else if [SourceName] == "EventLog" {
      translate {
        field => "[EventID]"
        destination => "[statement1][parameters][actProps][action]"
        dictionary => [
          "6006", "Event Log Service Stopped",
          "6009", "Windows Version",
          "6005", "Event Log Service Started",
          "6013", "System Uptime Report",
          "6011", "Computer Name Changed"
        ]
        fallback => "Unknown Action: %{EventID}"
      }
    }
    else if [SourceName] == "e1iexpress" {
    translate {
        field => "[EventID]"
        destination => "[statement1][parameters][actProps][action]"
        dictionary => [
          "32", "Network Interface Connected",
          "27", "Network Interface Disconnected"
        ]
        fallback => "Unknown Action: %{EventID}"
      }
    }
    else if [SourceName] == "Microsoft-Windows-Power-Troubleshooter" {
    translate {
        field => "[EventID]"
        destination => "[statement1][parameters][actProps][action]"
        dictionary => [
```

Figure 19: EventLog Filter.

```
            "1", "System Returned from Low Power State"
          ]
          fallback => "Unknown Action: %{EventID}"
        }
      }
      else if [SourceName] == "Microsoft-Windows-RestartManager" {
      translate {
          field => "[EventID]"
          destination => "[statement1][parameters][actProps][action]"
          dictionary => [
            "10000", "Starting Session",
            "10001", "Ending Session",
            "10002", "Shutting Down Application"
          ]
          fallback => "Unknown Action: %{EventID}"
        }
      }
      else if [SourceName] == "MsiInstaller" {
      translate {
          field => "[EventID]"
          destination => "[statement1][parameters][actProps][action]"
          dictionary => [
            "1040", "Beginning Install Transaction",
            "11707", "Installation Completed",
            "1033", "Install Info",
            "1042", "Install Transaction Complete",
            "1005", "Installer Initiated Restart"
          ]
          fallback => "Unknown Action: %{EventID}"
        }
      }
      else { mutate { remove_field => [ "[statement1][parameters][objProps]" ] } }
    }
  }
```

Figure 19: EventLog Filter.

The filters again created subgraphs similar to Figure 17. Specific to the event log filter, logstash created one parser node with an "eventLog" paserName property and several action nodes, again distinguishable by timestamps. The properties from the Event Log subgraphs were labeled and constructed differently because these log events did not necessarily have, for example, a file path of the shortcut accessed such as the LNK subgraphs mentioned earlier. Instead, the event log subgraphs created object nodes with specific Windows Event Log IDs and action nodes with the Event Log message along with the timestamps. Figure 20 shows an example of one entry from events produced by TEAR, placed into the Neo4j database. The object nodes for these subgraphs contained information of the log data's origin, such as Microsoft Windows Winlogon or Msi Installer logs.



```
{
  "parserName": "eventLog"
}
```

```
{
  "message": "The Event log service
was started. ",
  "action": "Event Log Service
Started",
  "timestamp": 1311807874000
}
```

```
{
  "filename": "EventLog/6005"
}
```

Figure 20: EventLog Neo4j Subgraph.

Overall, phase one of the ingest process involved using Plaso and TEAR to extract events from a Windows image and place them in their respective outputs, Elasticsearch database for Plaso and CSV format text files for TEAR. Then logstash is a pipeline to accept as inputs these two difference sources of events, transform the data into Cypher queries to create desired subgraphs, and insert the results into a Neo4j database. ECATV created the same subgraphs in [33] but modified filters to correctly convert the data from two different sources into Neo4j.

Two different sources were used to create a more robust database for the abstraction portion of the ingest phase. This was because TEAR captured events in a different manner than Plaso and provided more information for certain events. For example, Plaso extracted the Event Log artifacts from the image but was unable to pull the actual message as shown in Figure 15. TEAR on the other-hand was able to extract the source of the logs as well as the specific messages associated to each of these events. So ECATV used logstash with Event Log filters that accepted TEAR Event Log text files for its inputs. Meanwhile it used the LNK filters and used Plaso's extracted events to create the File Table subgraphs mentioned in [33]. It is important to note, as mentioned earlier, all subgraphs were also related to time nodes, which were important during the creation of high-level abstracted events in phase two of ingest.

3.3.2 Phase Two of Ingest

Phase Two involves applying the abstraction logic through Cypher queries to the Neo4j database and creating high level abstracted nodes. Adderley [36] created high level events for power events, installation information, program execution, file download, and web history. ECATV follows the same concept and created the same high-level events. These events were represented in Neo4j as new abstraction nodes. Abstraction nodes were created by identifying and collecting multiple low level events and collecting its properties to populate the abstraction nodes' properties. For example, for the installation information abstraction nodes, the query searched for various object nodes containing the log entries 11707, 1042, and 1033 from the Msi Installer logs. Msi Installer Event ID 11707 translated to "Successful installation", Event ID 1042 translated "Notification of the installation process completion" and Event ID 1033 translated "Records the end result of a program installation. Status code 0 means the installation was successful" [36].

ECATV utilized similar queries to Adderley's research, essentially looking for the same low level events. The logic for creating Program Execution abstraction nodes was overhauled due to the change in some of the subgraphs' schema of the low level events in Neo4j. Because both TEAR and Plaso were applied for ECATV, the Cypher query required a significant change. Figure 21 contains the Cypher query used to create the Program Execution abstraction nodes within ECATV's Neo4j database.

Once these abstraction nodes were created, ECATV used Autopsy's keyword search module to locate files associated to keywords an examiner was interested in. Autopsy ingested the files of the Windows images by extracting and indexing their text into a SOLR database. This allowed for quick regular expression and keyword searches across the image. The results of the searches were ultimately stored into a SQL database, called autopsy.db, for the Autopsy "case" of the image. This database

```
//Program Execution
MATCH (parser:parser)<-[:PARSER]-(act:action)-[:EFFECTS]->(obj:object)
WHERE parser.parserName = 'prefetch'
MATCH (act) -[:AT_TIME]->(sec:Second)
MATCH p = (sec) -[:NEXT *10]->()
WITH p
UNWIND nodes(p) AS secNodes
MATCH (secNodes)
<-[:AT_TIME]-(act:action)-[:EFFECTS]->(obj:object)-[:TARGET]->(obj2:object)
WITH act.timestamp as timestamp,  COLLECT(DISTINCT obj.filename) as filenames,
collect(DISTINCT act) AS collection,
    collect(DISTINCT obj) AS b, COLLECT(DISTINCT act.action ) AS acts, collect
    (DISTINCT obj.title) as titles
WITH [event in filenames | (SPLIT(event, "/"))][0][2] as extractedValues, filenames,
acts, titles, timestamp, collection, b
CREATE (a:Abstraction{ Event: [extractedValues], Trace: filenames, Description: acts +
titles , timestamp:timestamp})
    FOREACH (pair in collection | MERGE (pair)-[:LVL1_ABSTRACTION_LINK]->(a)
        FOREACH (set in b | MERGE
        (set)-[:LVL1_ABSTRACTION_LINK]->(a)))
```

Figure 21: Program Execution Cypher Query.

followed the same schema for module communication within TSK, called the blackboard (reference Section 2.5.1).

Autopsy stored the artifacts it discovered into the `blackboard_artifacts` table with attributes `artifact_id`, `obj_id`, `artifact_obj_id`, and `artifact_type_id`. The `artifact_type_id` referenced the table `blackboard_artifact_types` with attributes `artifact_type_id`, `type_name`, and `display_name`. `Artifact_type_id` nine corresponded to `type_name` of `TSK_KEYWORD_HIT` with a `display_name` of Keyword Hits. Once all the artifacts with `artifact_type_id` of nine were isolated within the `blackboard_artifacts table`, the next step was to examine the actual `artifact_id`. These values were connected to the `blackboard_attributes` table with properties `artifact_id`, `attribute_type_id`, and `value_text`. The `value_text` attribute contained the actual strings of the keywords or regular expressions the module searched for. The `attribute_type_id` attribute was related to the `blackboard_attribute_types` table which contained attributes of `attribute_type_id`, `type_name` and `display_name`. The `attribute_type_ids` of interest for this study were 10 and 11, which had `type_names` of `TSK_KEYWORD` and `TSK_KEYWORD_REGEXP` respectively. The `blackboard_attributes` table was then isolated for values 10 and 11 of the `attribute_type_id` to determine which search keywords corresponded to which artifact. The last bit of necessary information was the file paths of the actual keyword search hits. These resided in the `tsk_files` table with attributes `obj_id` and various other attributes but the ones of interest were name and `parent_path`. The `obj_id` values related to the same `obj_id` values within the `blackboard_artifacts table`. The combination of the `parent_path` value and the name value within the `tsk_files` table contained the full directory of the files within the Windows image, associated to the keyword search hits. A Python script was written to extract the necessary information of the search keywords with a hit and its associated files. Figure 22 provides a relational overview of the various tables within the Autopsy database.

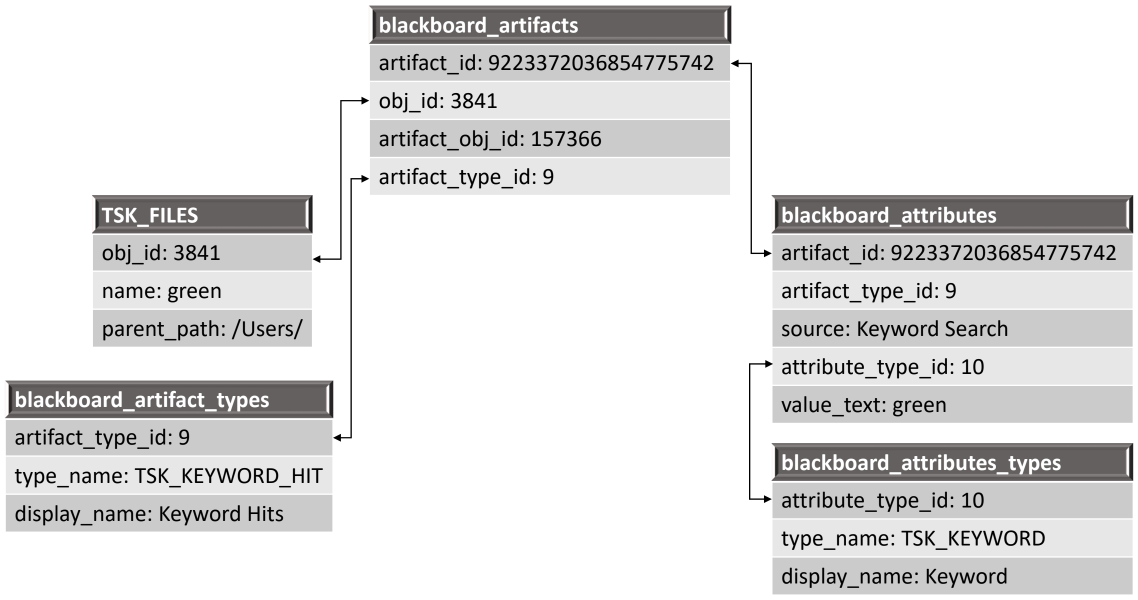


Figure 22: Autopsy Blackboard Communication.

This information then was integrated into Neo4j through a series of Cypher queries. The queries first matched the object node associated to the file path. It then searched for its corresponding abstraction node and added a searchKeyword property with a string value of the original keyword used in the Autopsy Keyword

Search Module. The base keyword was used as the value within this property and not a series of its associated regular expressions because Neo4j's text indexing features already include regular expressions as a capability.

3.4 User Interface

ECATV uses a similar REACT timeline as TAIMA with a few modifications. ECATV uses the same GraphQL API but with a modified schema. A new query type was added to take into account the possible queries for search keywords. Figure 23 shows the abstraction schema that returns the same properties from Neo4j as TAIMA did, but with the added searchKeyword property.

```
type abstraction {
  startTime: Float!
  endTime: Float!
  Event: [String]
  Trace: [String]
  Description: [String]
  timestamp: Float
  searchKeyword: String
}
```

Figure 23: Abstraction Schema.

Figure 24 illustrates the actual queries that were made to Neo4j. The first query remained the same as the one in TAIMA to account for examiners wanting the timeline to return all events within the given time frame. The second query was added for ECATV to query Neo4j's database using its full text search capabilities mentioned in Section 3.3.2. This query first pulled the nodes that successfully matched the keyword search and returned only the nodes that also matched the examiner's requested time frame. It is also important to note that the second abstraction query now requires a searchKeyword argument on top of the already required startTime and endTime arguments.

```
abstraction(startTime: Float!, endTime: Float!):
    [abstraction] @cypher(statement:
        "MATCH
        (:Second)<-[:AT_TIME]-(act:action)-[:LVL1_ABSTRACTION_LINK]->(abs:Abstraction)
        WHERE abs.timestamp >= startTime AND abs.timestamp <= endTime with
        abs.timestamp as timestamp,
        COLLECT (DISTINCT abs) AS abs
        UNWIND abs AS x
        RETURN x"
        )

abstractionSearch(startTime: Float!, endTime: Float!, searchKeyword: String):
    [abstraction] @cypher(statement:
        "CALL db.index.fulltext.queryNodes('keyword', searchKeyword)
        YIELD node as y
        MATCH
        (:Second)<-[:AT_TIME]-(act:action)-[:LVL1_ABSTRACTION_LINK]->(abs:Abstraction)
        WHERE abs.timestamp >= startTime AND abs.timestamp <= endTime AND abs.Trace =
        y.Trace
        WITH abs.timestamp as timestamp,
        COLLECT (DISTINCT abs) AS abs
        UNWIND abs AS x
        RETURN x"
        )
```

Figure 24: Abstraction Queries.

The resolver function in charge of translating GraphQL queries into a Cypher query was also modified to account for the new query, as shown in Figure 25. The other parts of the UI such as the viewing transformation, rendering, and visual display components of ECATV remained largely the same as TAIMA [36]. A text box was added to the visual display component for examiners to submit the desired search keyword.

```
export const resolvers = {
    Query: {
        object: neo4jgraphql,
        Second: neo4jgraphql,
        abstraction: neo4jgraphql,
        abstractionSearch: neo4jgraphql
    }
}
```

Figure 25: Resolver Function.

3.5 Summary

ECATV's ingest process involves multiple tools and combines their capabilities into a single entity. The process goes from a raw physical image of a Windows machine to a timeline visualization that maps out user and system activity over a specified time period. The image was processed through TEAR where its output is placed into CSV formatted text files. It was also processed through Plaso and the events were placed into an Elasticsearch database. All these events were processed through logstash where the pipeline's filters transformed the information into Cypher statements for a Neo4j database. These statements created various subgraphs consisting of action, object, parser, and time nodes. The Neo4j database was supplemented with additional abstraction logic in the form of Cypher statements to connect a collection of low-level events into a high-level event. These abstraction nodes were also supplemented with the results of Autopsy's Keyword Search Module and added a new searchKeyword property to the abstraction nodes. These properties were indexed using Neo4j's fulltext search feature and allowed the timeline to conduct string searches. ECATV allows examiner to visualize the user and system activities across a specified time period, similar to TAIMA, while also allowing the examiner to further filter to results down to specific search keywords, such as a user name. The new feature allows examiner to filter the results, for example, down to the events of a single user, a specific program, or a user's association to a specific program. It also provides examiners with the high-level events' trace, allowing them to closely examine the potential items of evidence through Autopsy or other digital forensics tool.

IV. ECATV and Digital Forensics Examination Process

This chapter presents how the Event Correlation and Abstraction Timeline Visualization (ECATV)'s effectiveness was assessed. The next section explains the different phases in the common digital forensics examination process. Then it explains the three user stories that were selected to test ECATV. Lastly, it walks through how ECATV can be used and how it fits within the digital forensics process. Scenario one was selected for the initial walkthrough because it was the same scenario used in the Adderley's user study [36] and due to the differences explained in Section 4.3.

4.1 Assessment

The previous chapter discussed the process of taking a physical image all the way to a timeline visualization with an added keyword search capability. Keyword search was added to provide examiners with a method to reduce the number of events displayed on the timeline and allow the user to control what he or she desired.

Evaluation of ECATV consist of use cases for three different images to measure its capabilities in the current digital investigation process. First it tested whether or not this process was viable in creating the desired timeline with the high level events and search keywords for a different data set. The details of the three images and the scenarios for each were outlined in the next section "User Stories". Though ECATV was researched with the data agnostic concept in mind, it is currently limited to Windows images as many of the rules, logic, and filters developed were targeted for the Windows operating system. The measurement is ECATV's accuracy in presenting the necessary pieces of information for each scenario. This was done to provide a more objective measure of its viability as an operational module. The many benefits of a timeline were difficult to measure as there are many subjective categories and its usability as an investigative tool would rely on the examiners' level of ability in conducting digital forensics. But [36] conducted a user study confirming the usability benefits of a timeline visualization. Thus it was necessary to test ECATV on a more objective level before continuing with an updated usability test.

4.2 Common Digital Forensics Examination Process

Carrier [42] breaks down a digital forensics investigation into three phases; digital crime scene preservation and documentation, digital evidence searching and documentation, and digital event reconstruction and documentation. The first phase pertains to imaging and preserving the original copy and is not the focus of this study. The focus is on finding evidence which partially overlaps into the third phase, reconstruction. Documentation standards differ based on legal requirements of where the investigation was conducted and again is not the focus.

Carrier explains the evidence searching phase, or the digital forensics examination process, is broken down into four phases. Phase one is target definition used to locate evidence, such as a specific file name or content within a file. This is the most challenging aspect and targets are "defined from either experience or existing evidence" [42]. The examiners use their experience from similar investigations to define common targets. They can also derive additional targets from the evidence they already discovered during the examination. The second phase involves data extraction and interpretation. Once the target is defined, investigators conduct a search of the digital crime scene for evidence through interpretation of different abstraction layers. Carrier explains looking at each file, or sector, or each network packets provides different levels of information and searches conducted at different abstraction layers provide different amounts of information used to locate the targets. This is akin to a physical crime scene search. For example, in a homicide, an investigator would define his/her target as the blood and would use an ordered pattern to search the physical crime scene for blood. The investigator may discover a tiny blood splot causing him/her to pay closer attention to a particular area in hopes of locating a better sample. digital forensics examinations are similar where instead of ordered patterned physical searches, examiners use visualization techniques to locate files modified at a given time, or keyword searches to find specific values in its name or content, or hash databases to find files with content of a specific value [42].

Phase three then compares the extracted data to the target and the examiner determines if the object is evidence. Referring back to the physical investigation, an investigator may locate a large pool of substance that appears to look like blood to the naked eye. He/she can then conduct a field test, such as using luminol or UV light, to determine whether or not the object containing the blood stain should be considered evidence for further examination at a forensic lab. digital forensics examination is slightly different where the processes of seizing the evidence for further analysis is unnecessary and the examination to determine if the object is evidence can be conducted immediately. Phase four updates the knowledge base of the investigation which further defines additional targets to search for. This process repeats until the examiner is satisfied with the amount of evidence necessary for the investigation.

4.3 User Stories

A Windows 98, Windows 7, and Windows 10 operating system images were used as the base for the timeline visualizations. The Windows 98 image was selected because it was the same image used in Adderley's user study in [36]. The details of the scenario are outlined in Appendix C of [36] and the following search keywords were used to develop the search index within Neo4j:

Greg Schardt; Starbucks; T-mobile; Evil; Mr. Evil; password; Cain & Abel; Ethereal; 123 Write All Stored Passwords; Anonymizer; CuteFTP; Look&Lan_1.0; NetStumbler; sniffer; cracker; access point; discovery; tool; Interception; Credit; Cards; username; hackers; hacking; hackerz; adware; back door; black hat; bot; botnet; cookies; Denial of Service; DOS; Distributed Denial of Service Attack; Dumpster Diving; Easter Egg; Firewall; Gray Hat; hackers; keylogger; logic bomb; malware; master program; payload; phishing; phreaker; rootkit; polymorphic virus; script kiddie; social engineering; spam; spoofing; spyware; time bomb; trojan; virus; wardriving; white hat; worm; zero day; exploit; zombie; install; startup

These were selected based off the scenario as well as the user study conducted in [36].

The Windows 7 and Windows 10 images were taken from Computer Science and Computer Engineering (CSCE) 527 Cyber Forensics class offered at Air Force Institute of Technology (AFIT). The details of the scenarios and assignments were outlined in Appendix A and Appendix B. The Windows 7 image consisted of a scenario where the owner of a company was murdered and the company workstation was taken as evidence. For the Windows 10 image, the scenario involved a drug investigation where a public computer at a library was the main source of digital evidence. These images and scenarios were chosen to test the capabilities of ECATV on newer operating system as well as its ability to perform when multiple users were involved on a single image. The following search keywords were used to develop the search index for the Windows 7 image:

Boddy; Mustard; Green; Plum; Scarlet; Peacock; White; clients; contracts; information; technology; support; research; development; office; manager; accountant; researcher; lead; Boddy, INC; contracting; Google; chat; cooked books; jpg; pdf; keylogger; outlook; emails; e-mails; startup; installation

The Windows 10 scenario was slightly different because most of the e-mails were not located on the actual image and were stored on an e-mail server. Because of this, the e-mails were analyzed prior to examining the image and the following keywords were selected:

Green; Forest; White; Pearl; Mustard; Golden; Scarlet; Ruby; Plum; Liseran; Iris; Peacock; Pym; Gerald; txt; log; key; keylogger; jpg; drug; pickup; locations; hackers; hack; keystroke; logging; fish; logged; drop; and Park; forest.green13@protonmail.com; proprietor pym's country store; leader copy; slots; gambling; blackmail; startup; install

These two images were unique from the first scenario because the images involved multiple users on a single workstation. The keywords were chosen based on what examiners may typically search in their initial examination based on the preliminary scenario information. The usernames as search keywords also makes it easy for the examiners to filter the events on the timeline based on the events only associated to said user. This made it easier to view the overall activities of specific users or track specific usage patterns. The final difference of interest between these two images and the first was their development process. The first Windows 98 image was made specifically to answer the questions of the scenario and in total the image was only 5 GB. The Windows 7 and Windows 10 images were developed to mimic actual user usage in a workstation environment such as having event logs enabled and were 40 GB and 120GB respectively. It also logged onto each user account and contained the activities done on that account to what the "user" would have fictionally accomplished within the scenario. For example, the user Green in both scenarios was in charge of Information Technology (IT) support and thus his activities on both images mirrored what a typical IT support staff would conduct. ECATV makes analysis easier by isolating events of these images down to a specific user. Then the examiner can look through these events and quickly determine if any activity stands out as an anomaly not only in general sense, but also more tailored to the specifics of an investigation. For example, as an IT staff member, Green would not normally access the contents of client records or email the company's customers. If these events are shown on the timeline after a search keyword of "Green" was submitted, it would raise flags and the examiner can began going down this trail of analysis.

4.4 Scenario 1

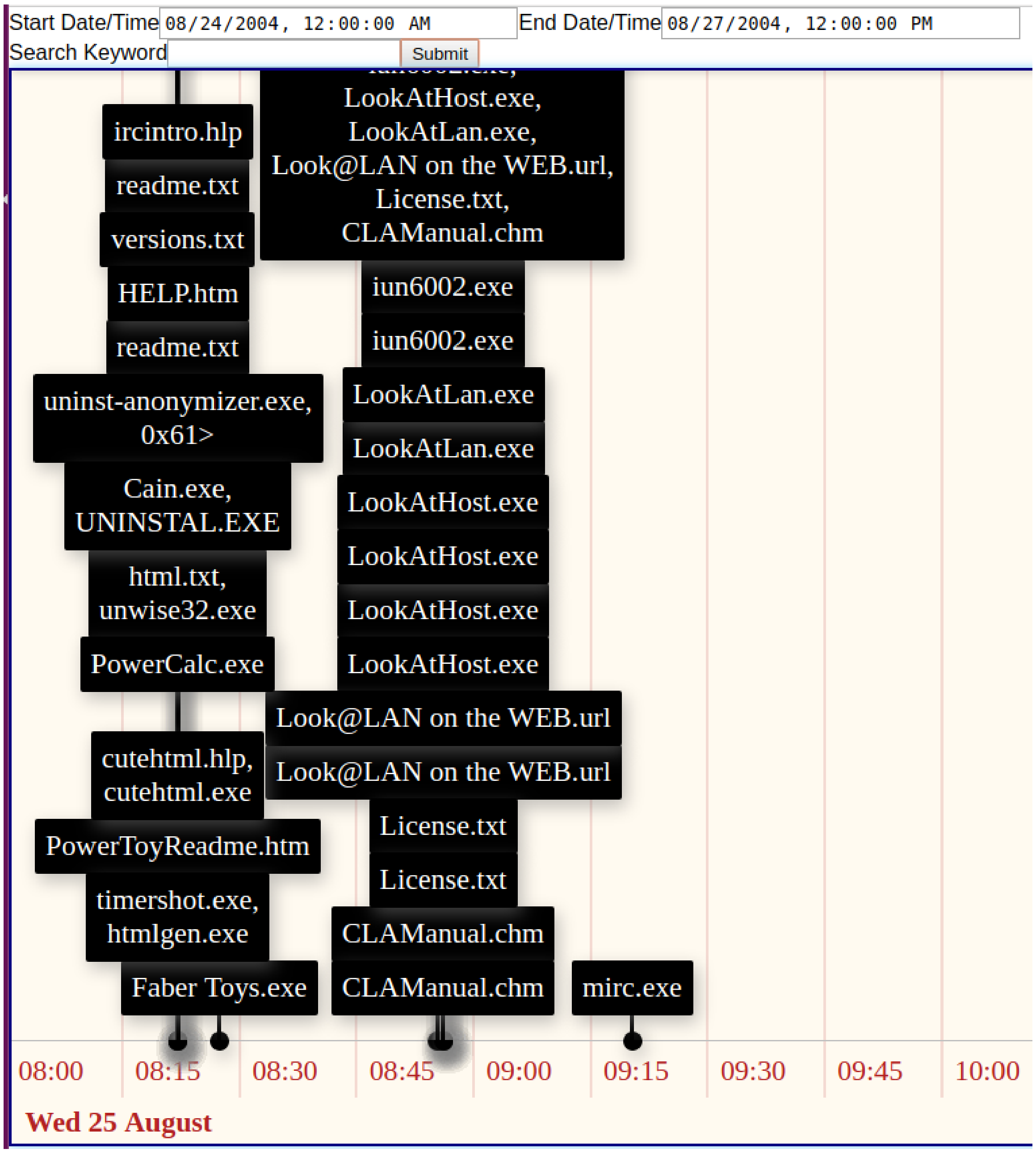
This scenario involved a hacker named Greg Schardt who utilized a notebook computer and a wireless PCMCIA card attached to a external homemade 802.11b antennae to intercept internet traffic for username & passwords and credit card numbers. Schardt also went by the online nickname of "Mr. Evil". His associates stated "Mr. Evil" would park his car within range of wireless access points and would intercept traffic in hopes of obtaining sensitive information. The goal of the scenario was to locate any hacking software Schardt utilized within the given suspect's Windows 98 image.

4.5 ECATV Walkthrough

Scenario one was selected to demonstrate how ECATV could be used within the digital forensics process. The first step was finding a suitable time range to search that would make identifying the artifacts of interest easier. In the scenario, the notebook computer was seize on 20 September 2004 so it would make sense to begin searching from January 2004 to September 2004. The examiner can then see where the clusters of activities were and begin narrowing down the scope. For this specific scenario, the time range of interest was already identified during the user study. The time range specified in [36] of 27 Aug 2004 12:00:00 AM - 27 Aug 2004 12:00:00 PM provided the results necessary to find the six important results of the hacking software, as shown in Figure 26. These nodes were mostly program execution nodes and were missing the power event and installation information abstraction nodes. This was due to the image being a Windows 98 operating system which at the time did not have Windows Event Log features. Since it was missing these low level events, phase two of the ingest process failed to produce those two abstraction node types. Regardless, the timeline was able to produce the necessary results to find the six hacking programs.

Another useful feature of a keyword search timeline is its ability to discover patterns and correlation between these events, as mentioned in Section 4.3. For example when the timeline in Figure 26 was filtered based on “ethereal” keyword search, it produced Figure 27. This kept the program execution nodes of the packet sniffer, Ethereal, as expected. But it also kept the nodes for NetStumbler which was a wireless access point discovery tool. This view also showed the user executed the access point discovery tool prior to executing a packet sniffer. Scenario one involved the hacker used the notebook computer and a wireless PCMCIA card to intercept internet traffic for username and passwords and credit card numbers. Based on this, the examiner can then search for “ethereal” and “password” to discover that Cain, a password cracker, was executed soon after, as shown in Figure 28.

Searching for “credit cards” during the same time period produced a node for “channels.txt; keys.txt; Receipt.rtf” which was different from the normal NetStumbler program execution nodes. The tooltip showed the traces for the node as shown in Figure 29. Because the scenario involved stealing credit card numbers as well, the examiner can use this as an initial point to start digging deeper into these files for the possibility of finding what numbers were stolen. It was also interesting that a internet relay chat program’s channel text files were executed around the same time these tools were used. It raises the flag that the hacker was potentially part of a group



Start Date/Time 08/24/2004, 12:00:00 AM End Date/Time 08/27/2004, 12:00:00 PM

Search Keyword Submit

ircintro.hlp

readme.txt

versions.txt

HELP.htm

readme.txt

uninst-anonymizer.exe, 0x61>

Cain.exe, UNINSTAL.EXE

html.txt, unwise32.exe

PowerCalc.exe

cutehtml.hlp, cutehtml.exe

PowerToyReadme.htm

timershot.exe, htmlgen.exe

Faber Toys.exe

LookAtHost.exe, LookAtLan.exe, Look@LAN on the WEB.url, License.txt, CLAManual.chm

iun6002.exe

iun6002.exe

LookAtLan.exe

LookAtLan.exe

LookAtHost.exe

LookAtHost.exe

LookAtHost.exe

LookAtHost.exe

Look@LAN on the WEB.url

Look@LAN on the WEB.url

License.txt

License.txt

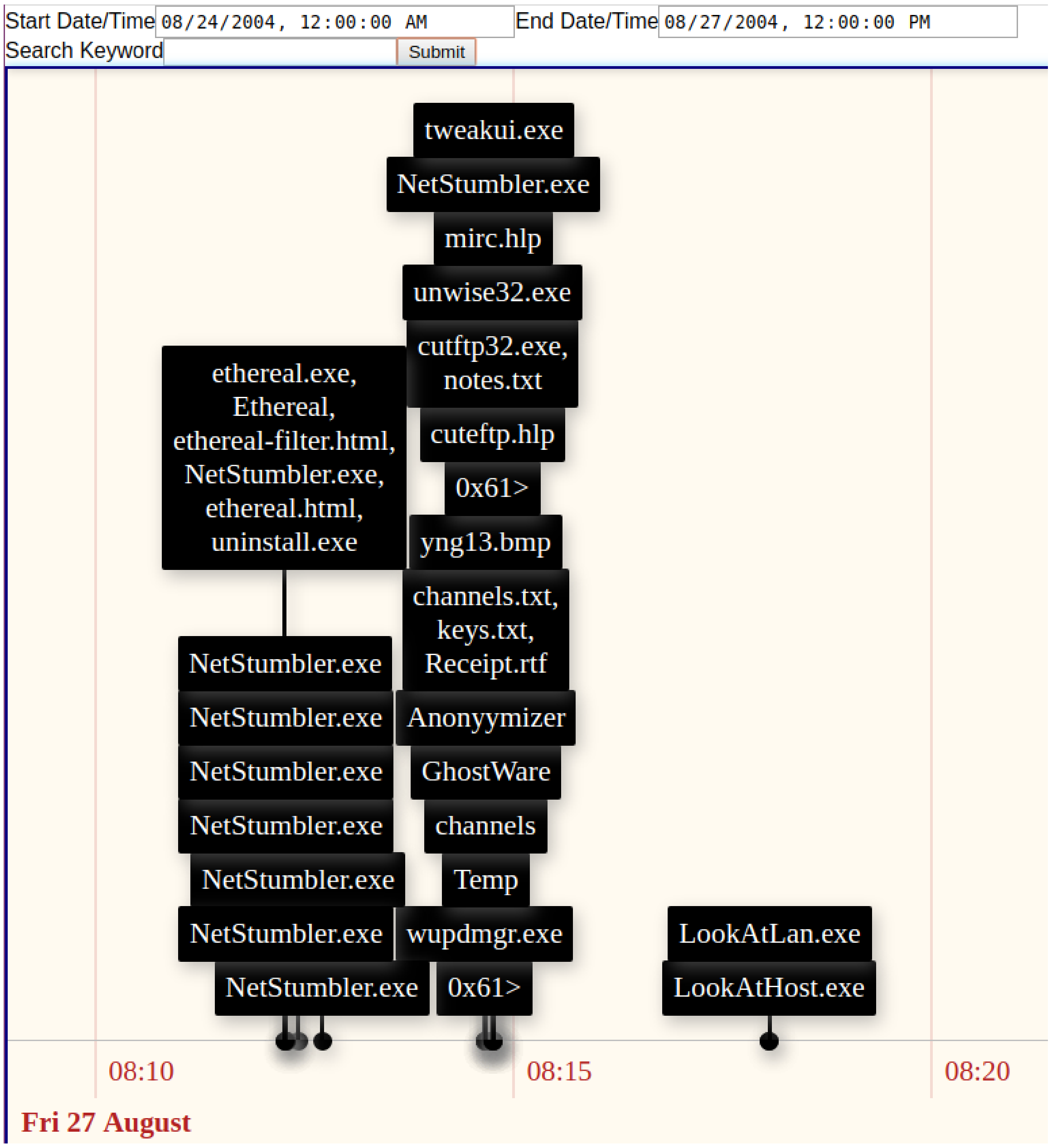
CLAManual.chm

CLAManual.chm

mirc.exe

08:00 08:15 08:30 08:45 09:00 09:15 09:30 09:45 10:00

Wed 25 August



Start Date/Time 08/24/2004, 12:00:00 AM End Date/Time 08/27/2004, 12:00:00 PM
Search Keyword Submit

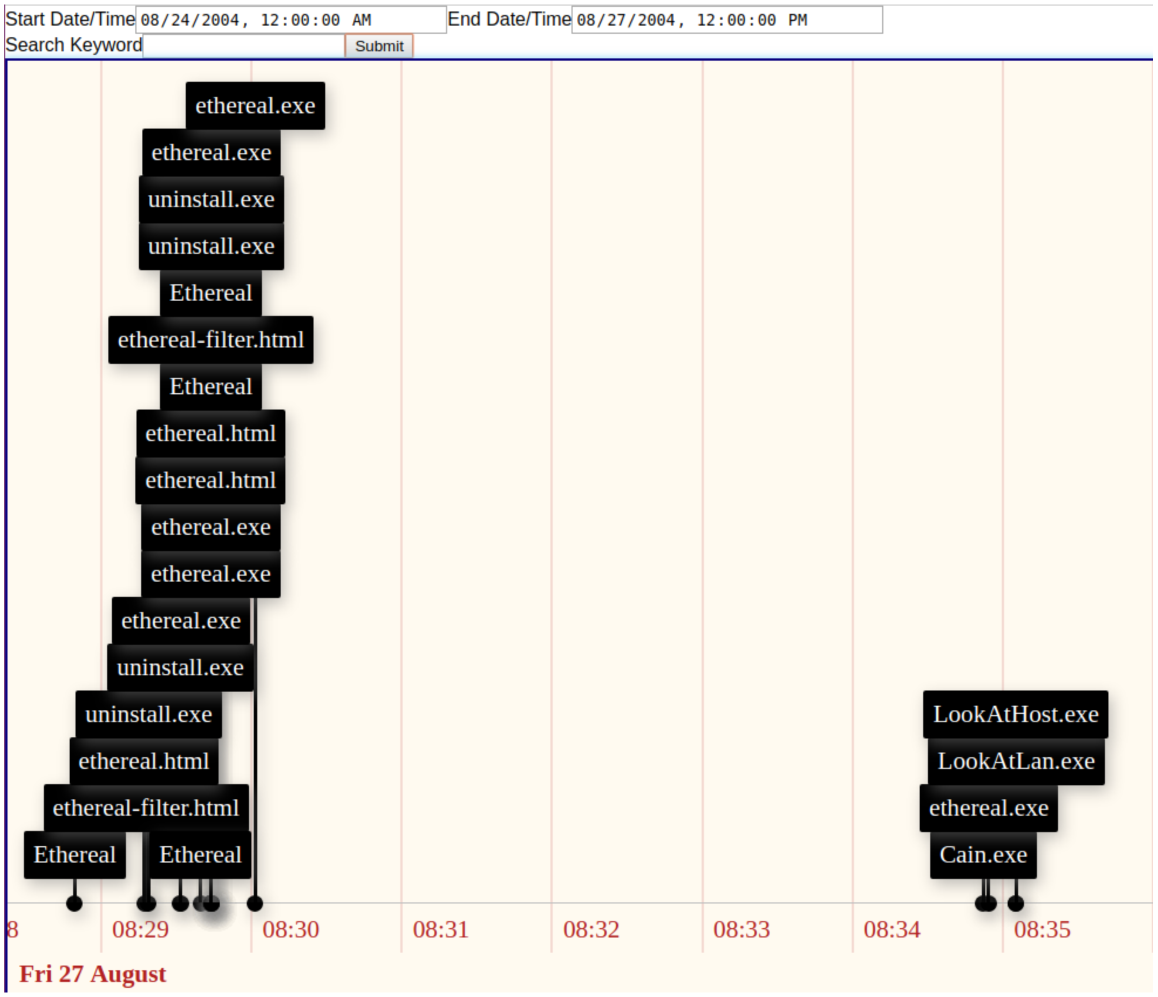


Figure 26: Scenario 1 Timeline.

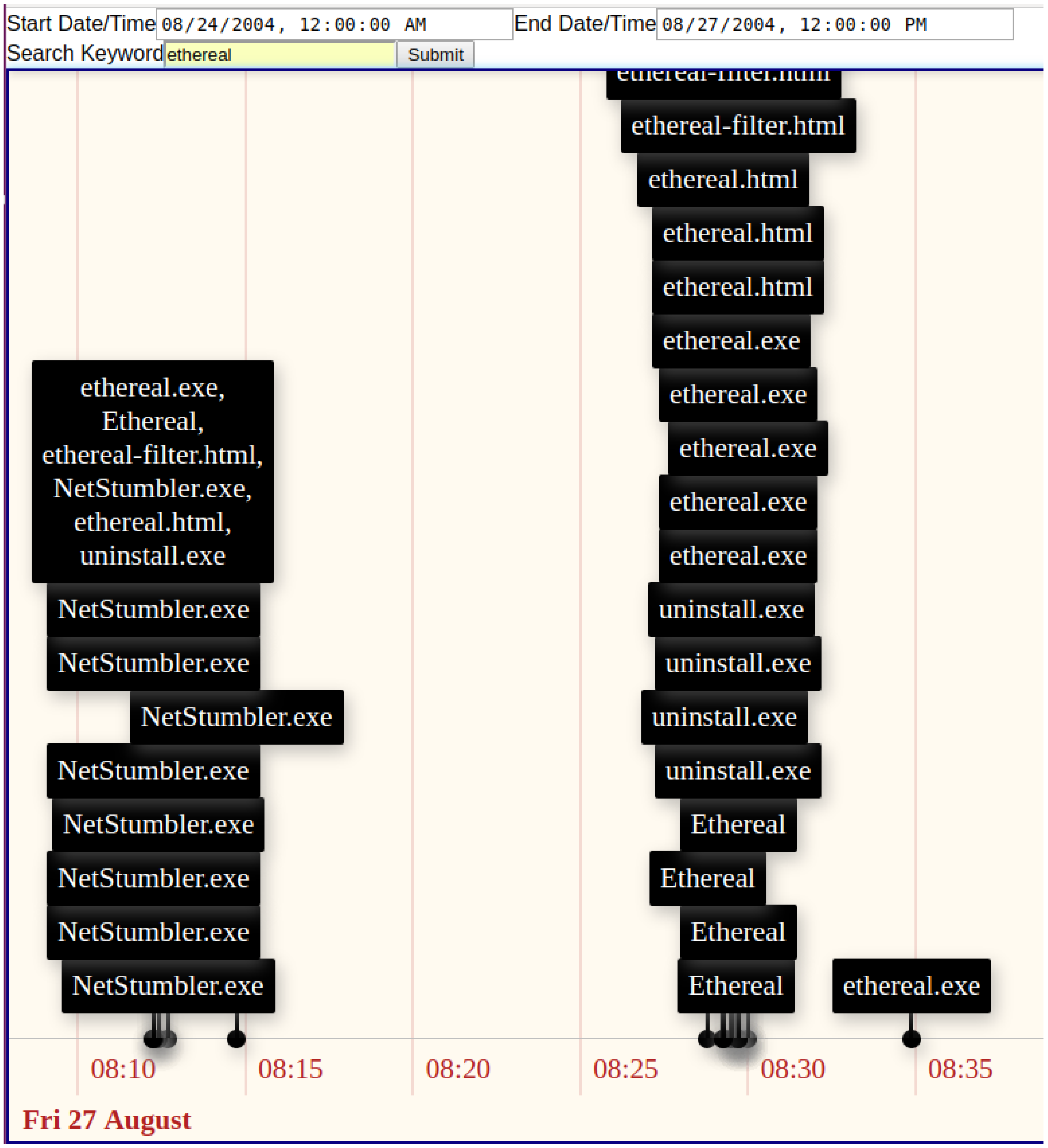


Figure 27: Scenario 1 Timeline Ethereal Search.

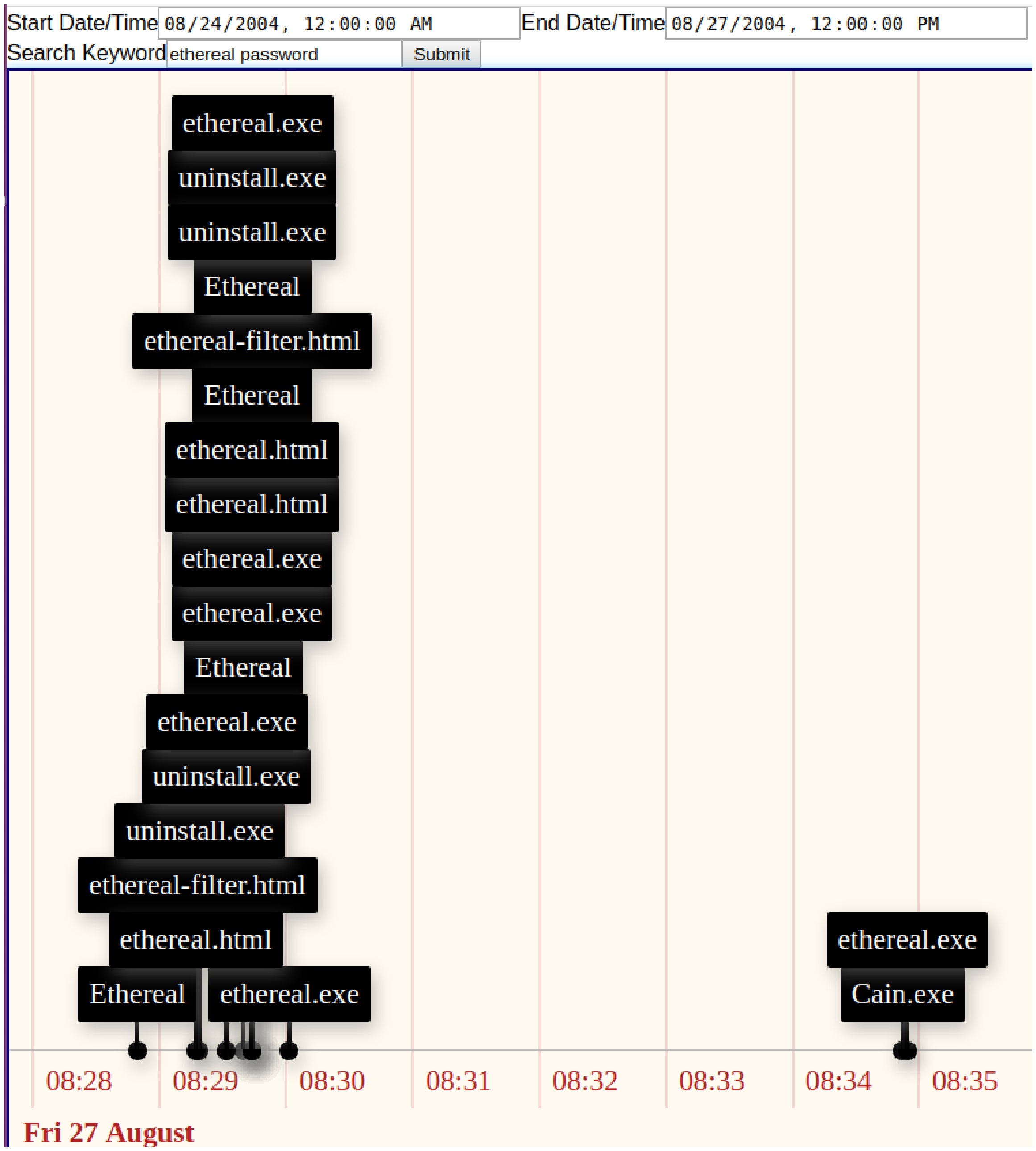

Start Date/Time 08/24/2004, 12:00:00 AM End Date/Time 08/27/2004, 12:00:00 PM
Search Keyword ethereal password Submit

Figure 28: Scenario 1 Timeline Ethereal & Password Search.

that discussed these techniques for intercepting wireless traffic or he was selling these usernames & passwords and/or credit card numbers via mIRC.

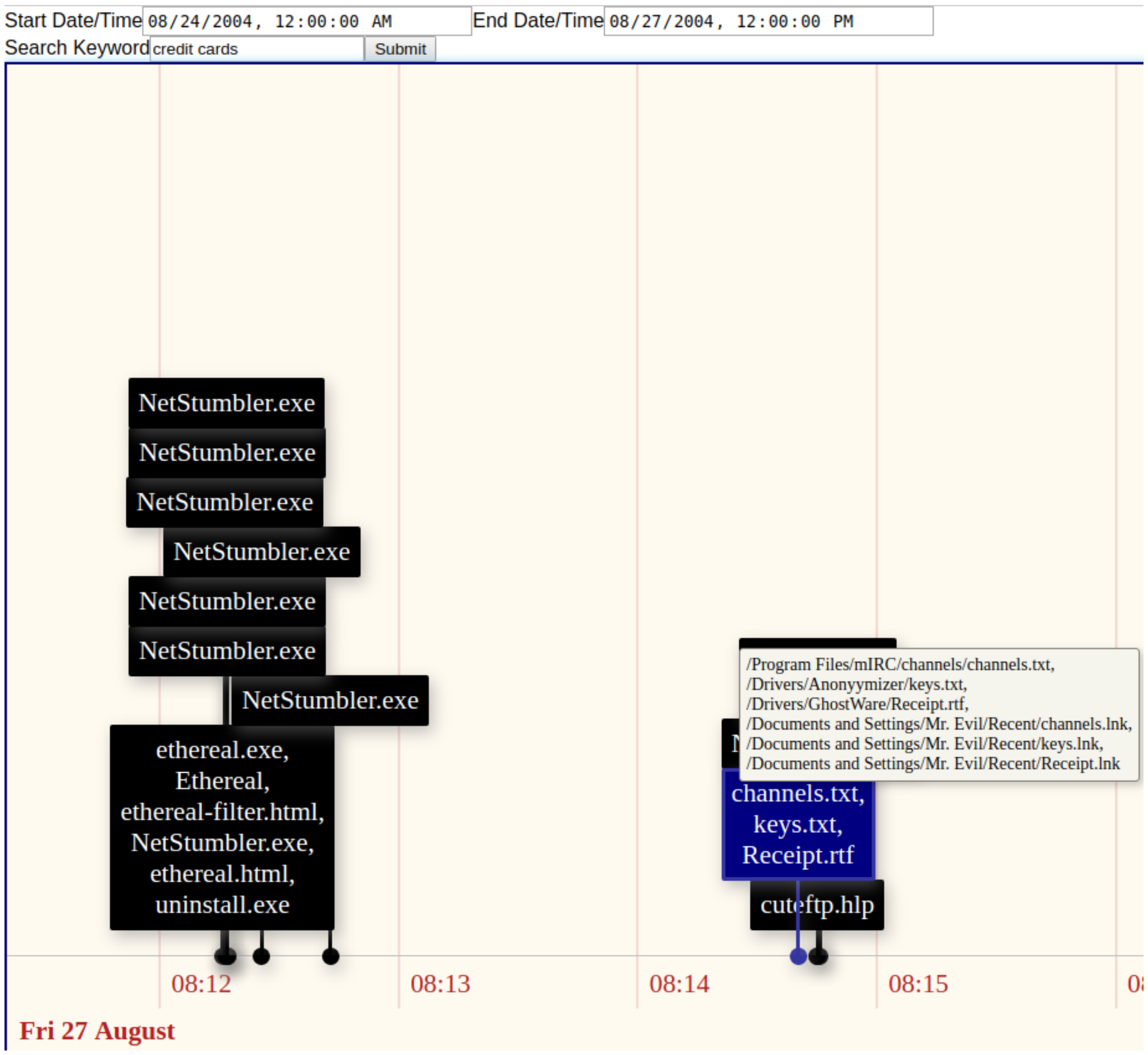

Start Date/Time 08/24/2004, 12:00:00 AM
End Date/Time 08/27/2004, 12:00:00 PM
Search Keyword credit cards
Submit
NetStumbler.exe
NetStumbler.exe
NetStumbler.exe
NetStumbler.exe
NetStumbler.exe
NetStumbler.exe
NetStumbler.exe
ethereal.exe, Ethereal, ethereal-filter.html, NetStumbler.exe, ethereal.html, uninstall.exe
/Program Files/mIRC/channels/channels.txt,
/Drivers/Anonyymizer/keys.txt,
/Drivers/GhostWare/Receipt.rtf,
/Documents and Settings/Mr. Evil/Recent/channels.lnk,
/Documents and Settings/Mr. Evil/Recent/keys.lnk,
/Documents and Settings/Mr. Evil/Recent/Receipt.lnk
channels.txt, keys.txt, Receipt.rtf
cuteftp.hlp
08:12
08:13
08:14
08:15
Fri 27 August

Figure 29: Scenario 1 Timeline Credit Cards Search.

Overall, ECATV helped locate the six hacking software as originally intended. But it also helped narrow down files of interest through the use of keyword searches and automated correlation of events organized by time. The tool helps examiners create multiple hypotheses to test and locate evidence. The traces associated to the nodes and the timestamps also help with the reconstruction portion of the digital forensics examination process.

V. Results and Analysis

5.1 Overview

This chapter goes over the results and analysis of the three scenarios. Scenarios two and three also include an example of how Event Correlation and Abstraction Timeline Visualization (ECATV) can benefit the examination process.

ECATV was evaluated based on its ability to reduce the number of results to sort through and also its completeness in presenting the key items for the three different user scenarios. The study also examined the number of hits the keyword search of the timeline visualization produced compared to what an Autopsy keyword search module would produce for each of the search terms. Most examiners begin their analysis using keyword searches to find a good place to start and to establish an idea of what evidence can be found. ECATV aims to reduce the number of hits an examiner must go through to identify a starting point while still having the ability to discover the necessary information. The reduction in the number of results presented can potentially cause the examiners to miss necessary pieces so it was important to evaluate ECATV's ability on result reduction and completeness.

5.2 Scenario 1

The summary of the scenario and the walkthrough of using ECATV for scenario were outlined in Section 4.4 and Section 4.5 respectively.

5.2.1 Scenario 1 Analysis and Results

Autopsy's Keyword Search Module produced 13,107 hits across all the terms mentioned in Section 4.3. On the other hand, the timeline visualization produced 397 hits to sort through, which reduced the number of hits by 97%.

As expected, the new timeline included all six hacking programs. ECATV provided the added benefit of being able to determine and find usage patterns compared to the previous version from Adderly's study [36].

5.3 Scenario 2

This fictitious scenario involved a millionaire and philanthropist, Boddy having been murdered and the investigators determined whoever murdered him worked in his contracting company, Boddy, Inc. This company consisted of Col Mustard, Mr. Green, Prof Plum, Miss Scarlet, Mrs. Peacock, and Mrs. White. Mustard was the vice presiden who was in charge of garnering contracts and liasing with clients. Plum was in charge of research and development while White worked with him as the lead researcher. Lastly, Scarlet was the office manager and Peacock was the company's accountant. The goals of the scenario were to reconstruct the crime that took place based on the information found on the image and to build a narrative for the lead investigator.

5.3.1 Scenario 2 ECATV Walkthrough

The first step for this scenario was to narrow down the time range of interest. Figure 30 shows the initial time range of 1 Jan 1990 1:01:01 AM - 1 Jan 2019 1:01:01 AM and "startup" keyword used to see when the system was started. These power events were generated from Windows Event Logs, specifically from entries of 6013 and 6005, as the tooltip in the figure shows. Log entries 6013 identified system uptime reports while 6005 identified that the event logging service started. The results showed the time range could be narrowed down to 26 Jul 2011 - 27 Jul 2011. The time range of 25 Jul 2011 12:00:00 AM - 28 Jul 2011 12:00:00 PM was used for each subsequent searches to ensure all activity between these dates were captured.

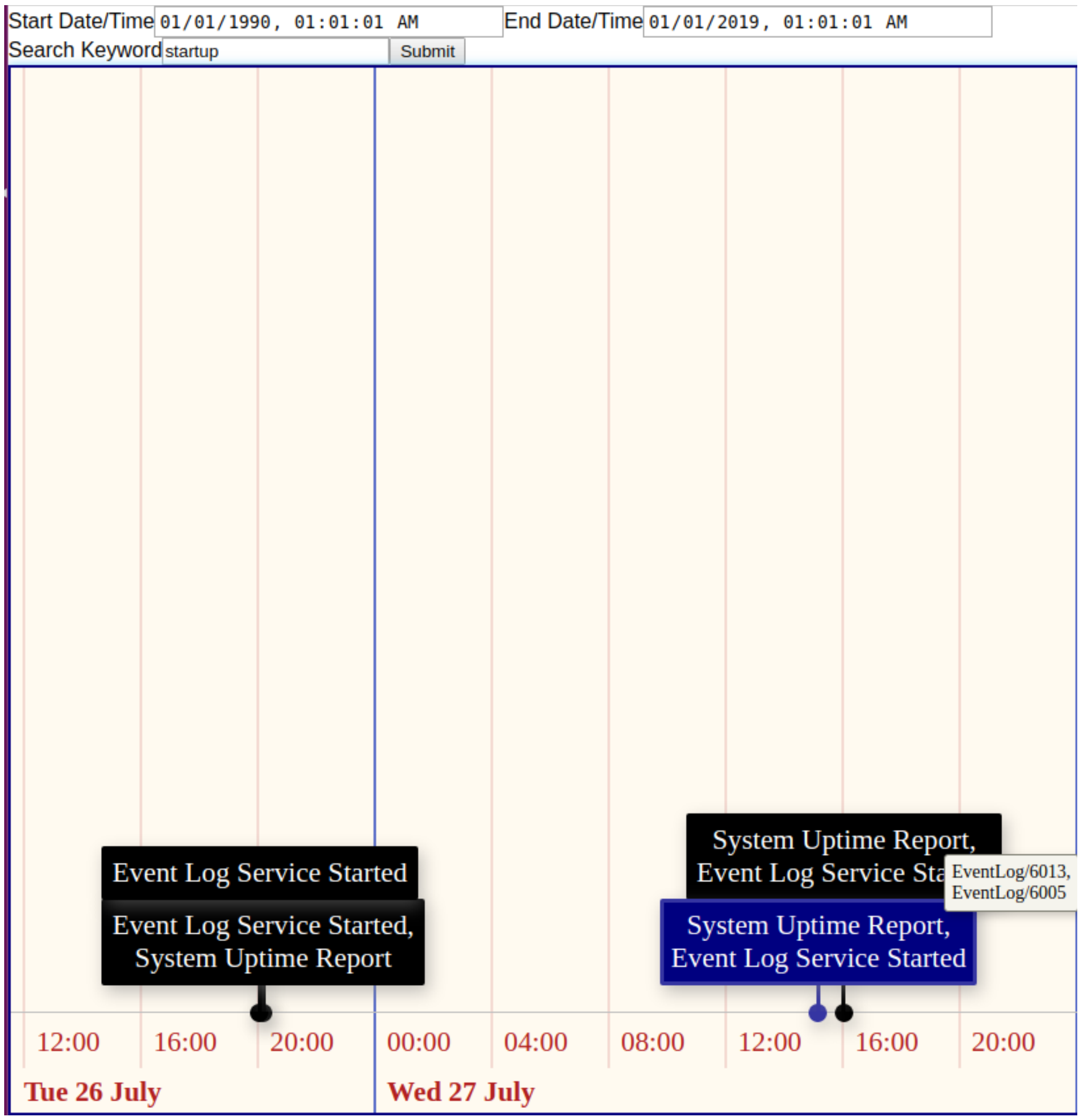
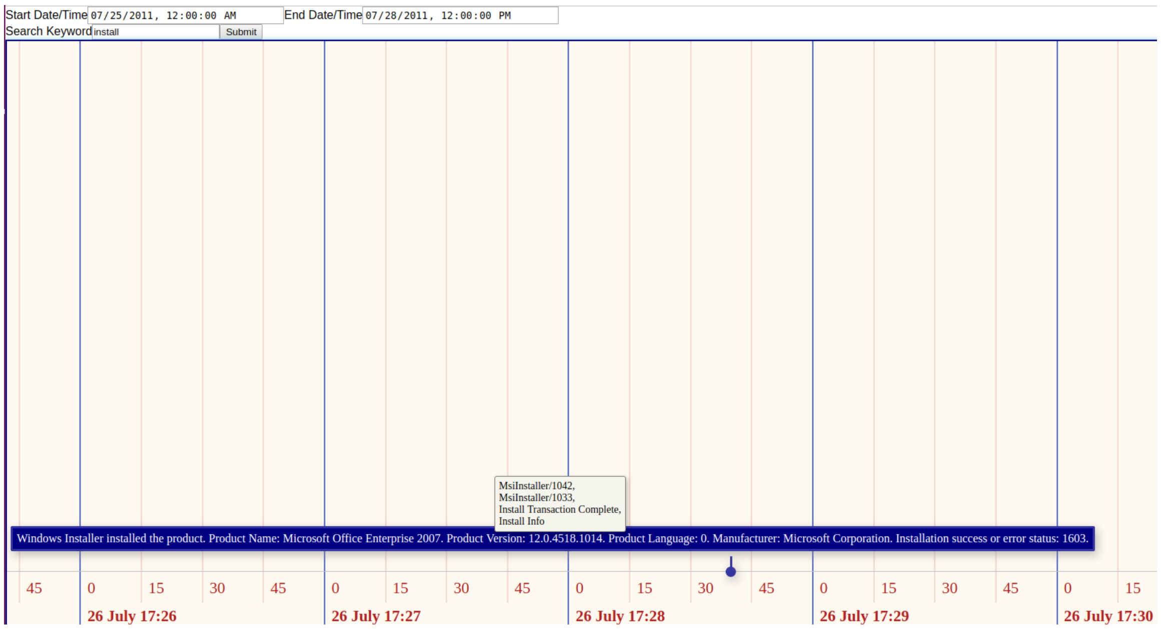

Start Date/Time 01/01/1990, 01:01:01 AM
End Date/Time 01/01/2019, 01:01:01 AM
Search Keyword startup
Submit
Event Log Service Started
Event Log Service Started, System Uptime Report
System Uptime Report, Event Log Service Sta
EventLog/6013, EventLog/6005
System Uptime Report, Event Log Service Started
12:00
16:00
20:00
00:00
04:00
08:00
12:00
16:00
20:00
Tue 26 July
Wed 27 July

Figure 30: Scenario 2 Timeline “startup” Search.

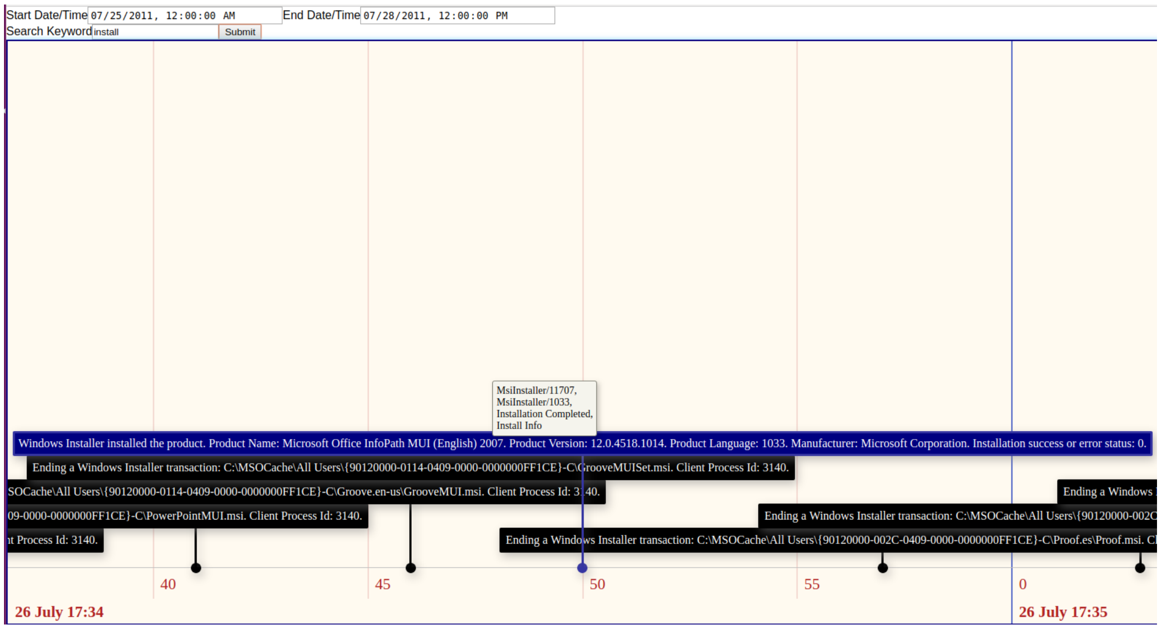
The next step involved looking for installation information to develop an idea of what sorts of software the users interacted with. A search of 25 Jul 2011 12:00:00 AM - 28 Jul 2011 12:00:00 PM and “install” keyword revealed various programs installed within this time period. Figure 31 shows an attempted installation of Microsoft Office Enterprise 2007 with an installation error status of 1603. A successful installation would contained an installation success code of 0, as shown in Figure 32.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword install Submit

MsiInstaller/1042,
MsiInstaller/1033,
Install Transaction Complete,
Install Info

Windows Installer installed the product. Product Name: Microsoft Office Enterprise 2007. Product Version: 12.0.4518.1014. Product Language: 0. Manufacturer: Microsoft Corporation. Installation success or error status: 1603.

Figure 31: Scenario 2 Timeline "install" Search.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword install Submit

MsiInstaller/11707,
MsiInstaller/1033,
Installation Completed,
Install Info

Windows Installer installed the product. Product Name: Microsoft Office InfoPath MUI (English) 2007. Product Version: 12.0.4518.1014. Product Language: 1033. Manufacturer: Microsoft Corporation. Installation success or error status: 0.

Ending a Windows Installer transaction: C:\MSOCache\All Users\{90120000-0114-0409-0000-0000000FF1CE}-C\GrooveMUISet.msi. Client Process Id: 3140.

SOCache\All Users\{90120000-0114-0409-0000-0000000FF1CE}-C\Groove.en-us\GrooveMUI.msi. Client Process Id: 3140.

09-0000-0000000FF1CE}-C\PowerPointMUI.msi. Client Process Id: 3140.

nt Process Id: 3140.

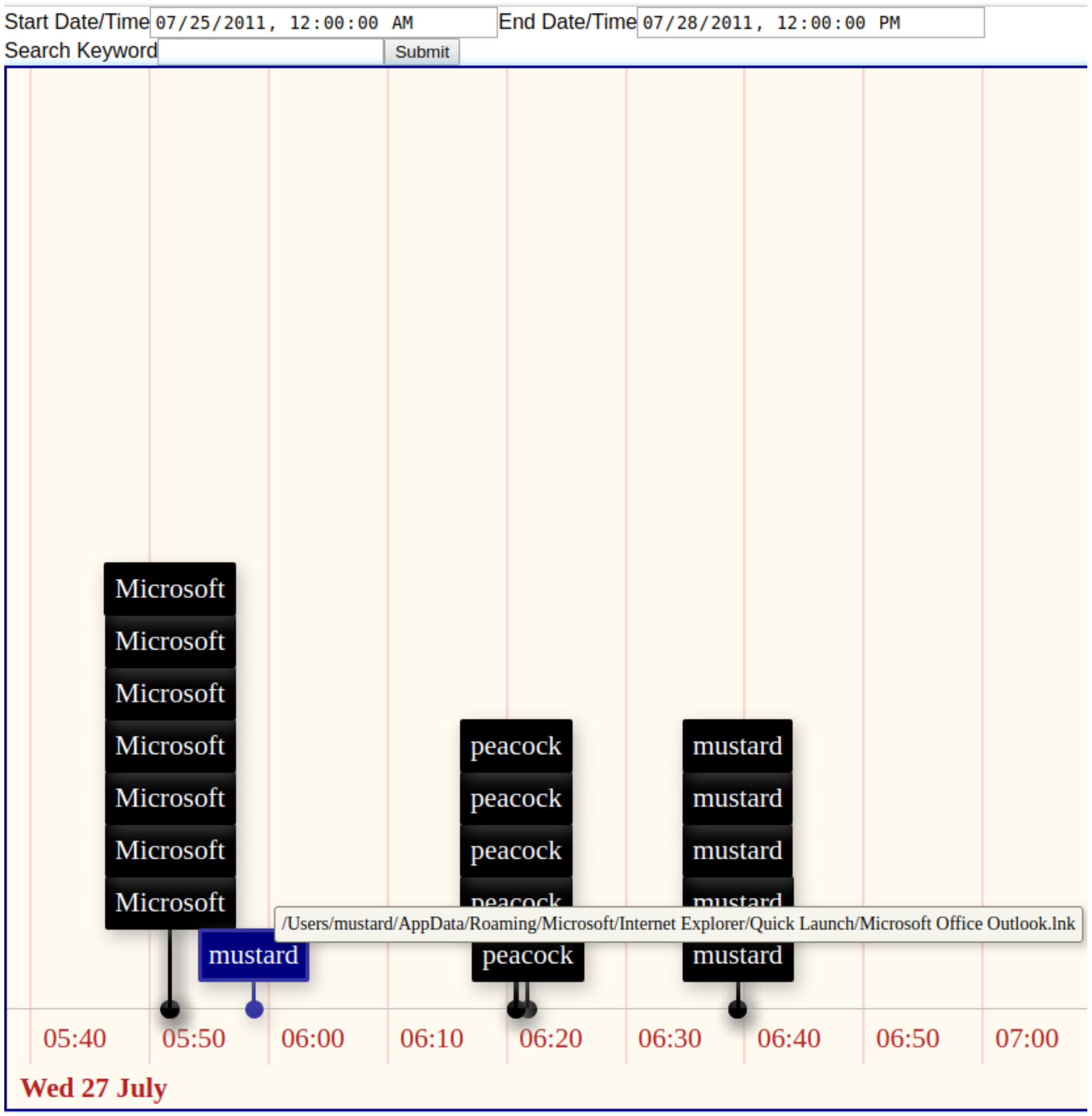
Ending a Windows

Ending a Windows Installer transaction: C:\MSOCache\All Users\{90120000-002C

Ending a Windows Installer transaction: C:\MSOCache\All Users\{90120000-002C-0409-0000-0000000FF1CE}-C\Proof.es\Proof.msi. Cl

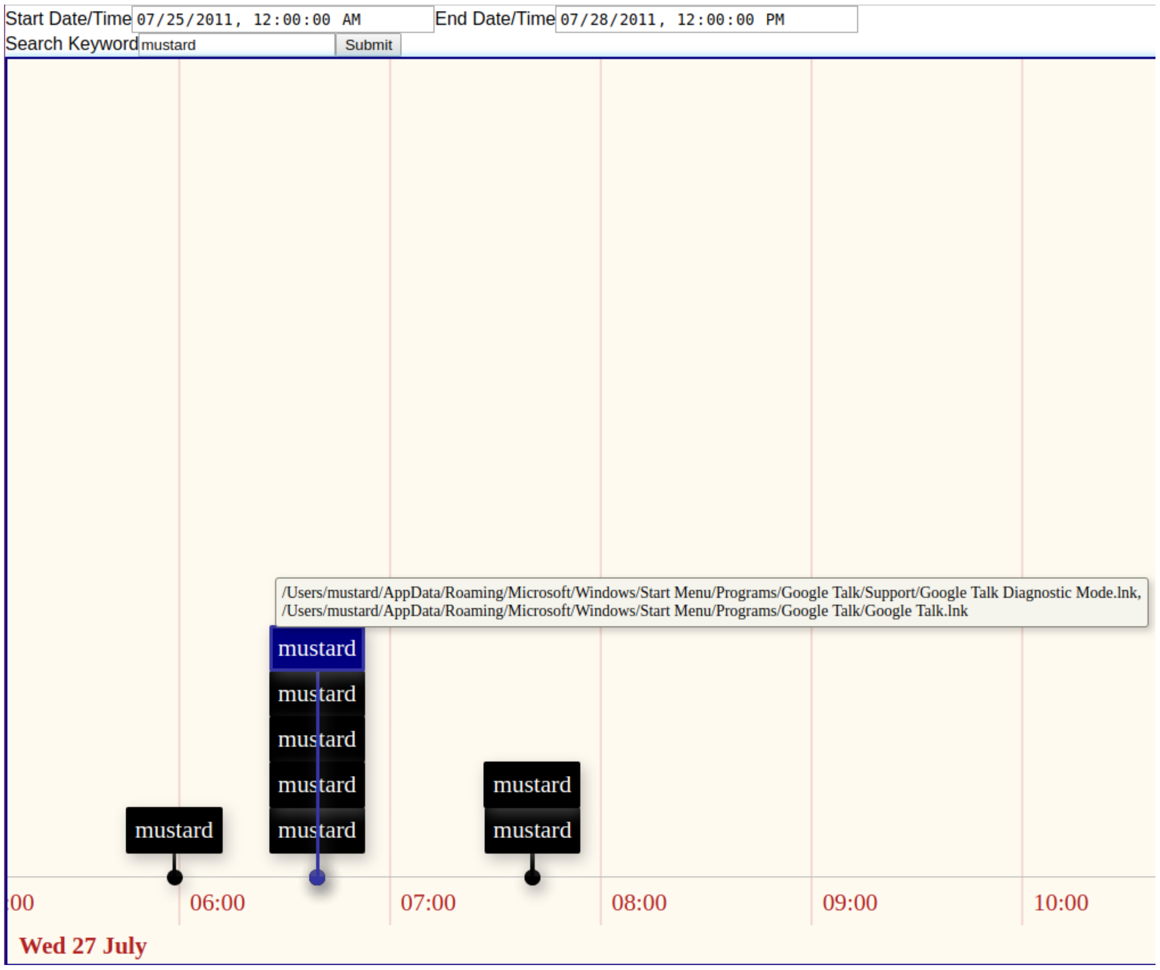
Figure 32: Scenario 2 Timeline "install" Search.

The next step was to search for clues and usage patterns to find the key pieces of evidence within this scenario. Figure 33 confirms that Microsoft Office was successfully installed because Mustard was able to executed Outlook.exe the next day. The user then narrowed down the search to Col Mustard's activity by using the "mustard" keyword. This isolated all of his program execution activities. Figure 34 shows Mustard using Google Talk. Based on the information thus far, the user can begin searching Google Talk logs and/or chat messages and outlook e-mail messages to find additional clues. The user can click through the various nodes to see Mustard also accessed various pictures/JPGs as the tooltip shows him accessing /Users/mustard/AppData/Roaming/Microsoft/Windows/Recent/pics.lnk (shown in Figure 35). Autopsy revealed this LNK file was linked to a public folder stored on the main drive as C:/pics. The timeline specifically shows Mustard opening DSCF0734 and DSCF0923 multiple times throughout the day. The user can also find that Mustard accesses readme2.htm, USB Disk (E) drive, and "too late.txt". The user takes note of these patterns and should also be on the lookout for a USB drive as an additional piece of evidence.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword Submit

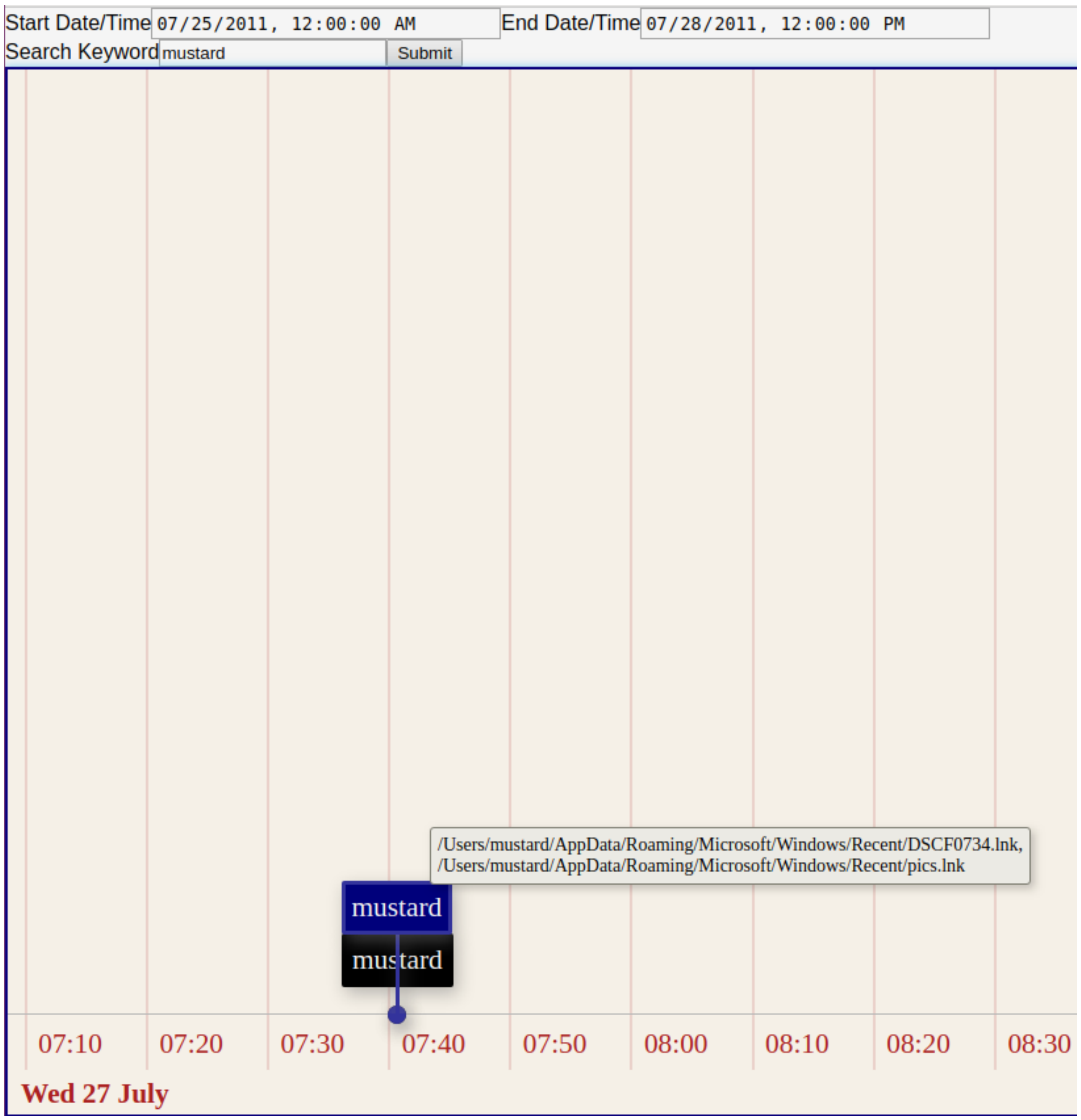
/Users/mustard/AppData/Roaming/Microsoft/Internet Explorer/Quick Launch/Microsoft Office Outlook.lnk

Figure 33: Scenario 2 Timeline Search.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword mustard

/Users/mustard/AppData/Roaming/Microsoft/Windows/Start Menu/Programs/Google Talk/Support/Google Talk Diagnostic Mode.lnk,
/Users/mustard/AppData/Roaming/Microsoft/Windows/Start Menu/Programs/Google Talk/Google Talk.lnk

Figure 34: Scenario 2 Timeline “mustard” Search.

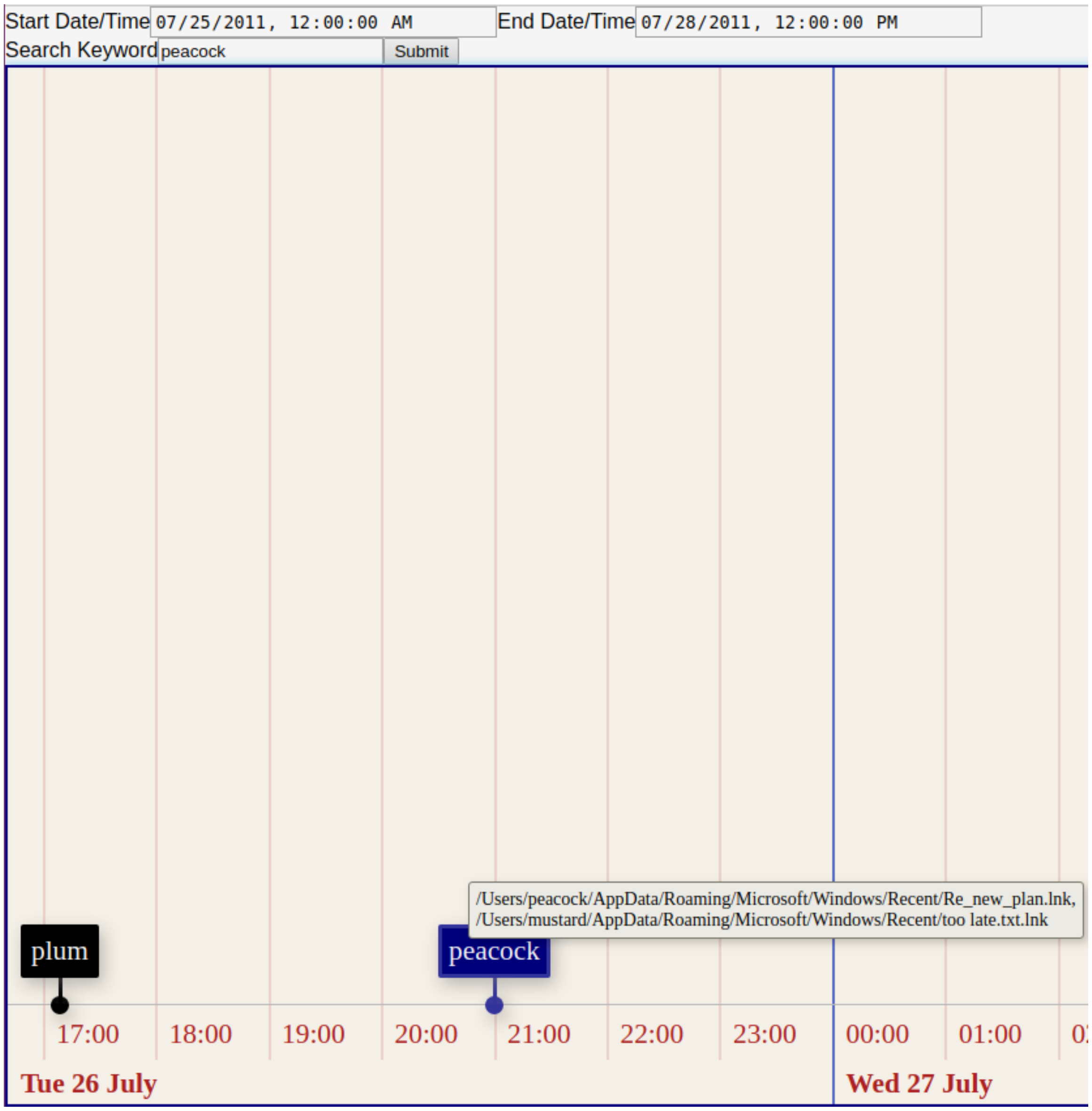


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword mustard

/Users/mustard/AppData/Roaming/Microsoft/Windows/Recent/DSCF0734.lnk,
/Users/mustard/AppData/Roaming/Microsoft/Windows/Recent/pics.lnk

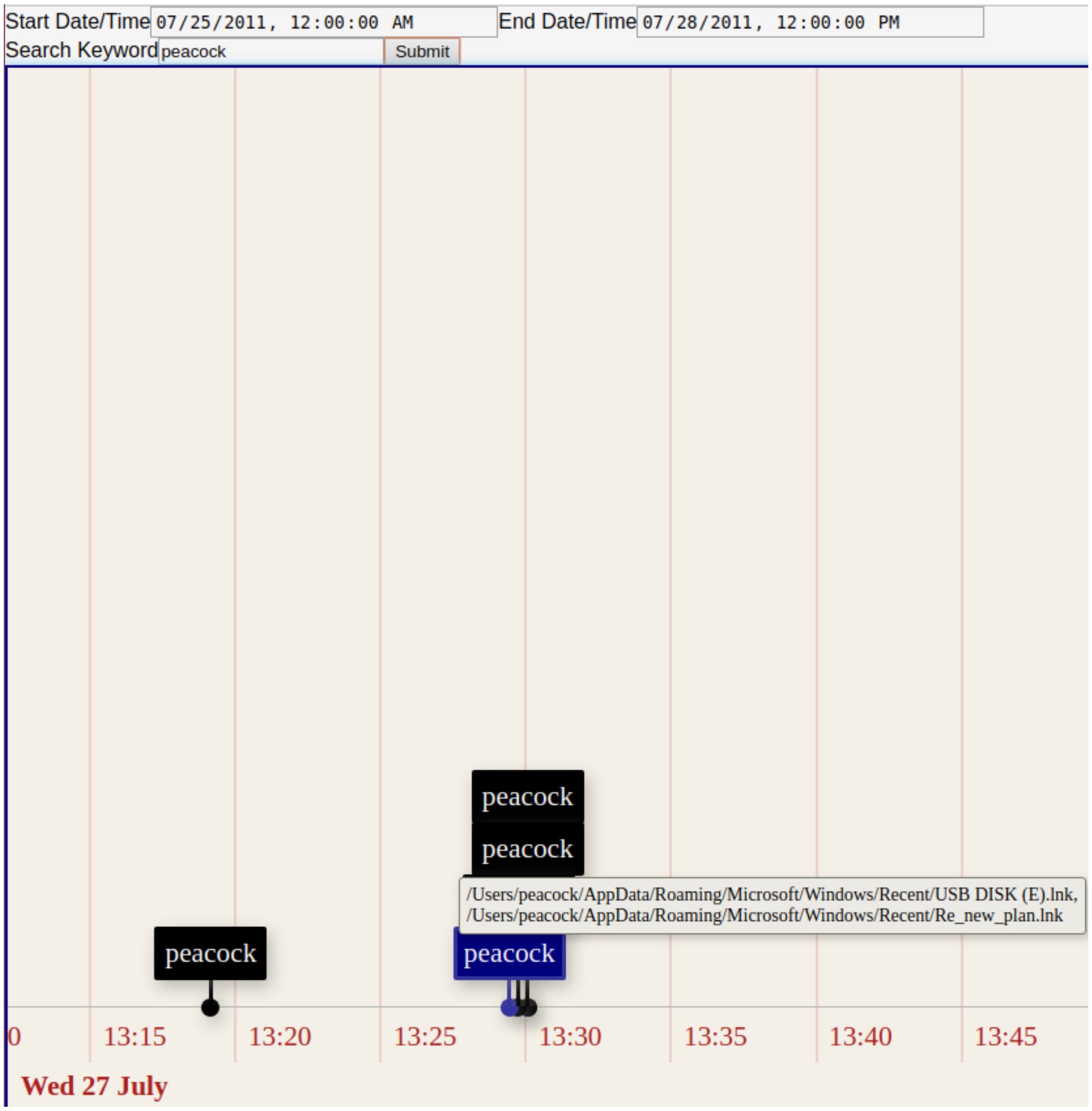
Figure 35: Scenario 2 Timeline “mustard” Search 2.

The next user that had a high amount of activity was Peacock. The timeline was filtered using the "peacock" keyword and the user is able to see Peacock accessed "Re_new_plan.txt" and "too late.txt" on the day prior to Mustard's access, as shown in Figure 36. Figure 37 shows Peacock accessing USB Disk (E) drive and "Re_new_plan.txt" the next day. Between this time, Peacock accessed Boddy Inc. with a trace of /Users/peacock/AppData/Roaming/Microsoft/Office/Recent/Boddy Inc.LNK, as shown in Figure 38. The user can then use Autopsy with that specific trace and find that the lnk file is associated to a database named Boddy Inc. Figure 39 shows a snippet of the database which contains various information on the company's customers. The user can take note that Peacock was the company's accountant but should raise a flag that the database was specifically accessed between the two times shown in Figure 36 and Figure 37 where Peacock accessed "Re_new_plan.txt". The timeline also revealed prior to Peacock accessing the database, Microsoft Excel and Microsoft Access were launched. Another item of interest was Peacock accessing Cisco 3550 manual PDF shortly after touching USB Disk (E) drive and "Re_new_plan.txt" (Figure 40). The scenario stated Green was the company's Information Technology support. This behaviour of an accountant going through an ethernet switch manual should raise a flag.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword peacock Submit

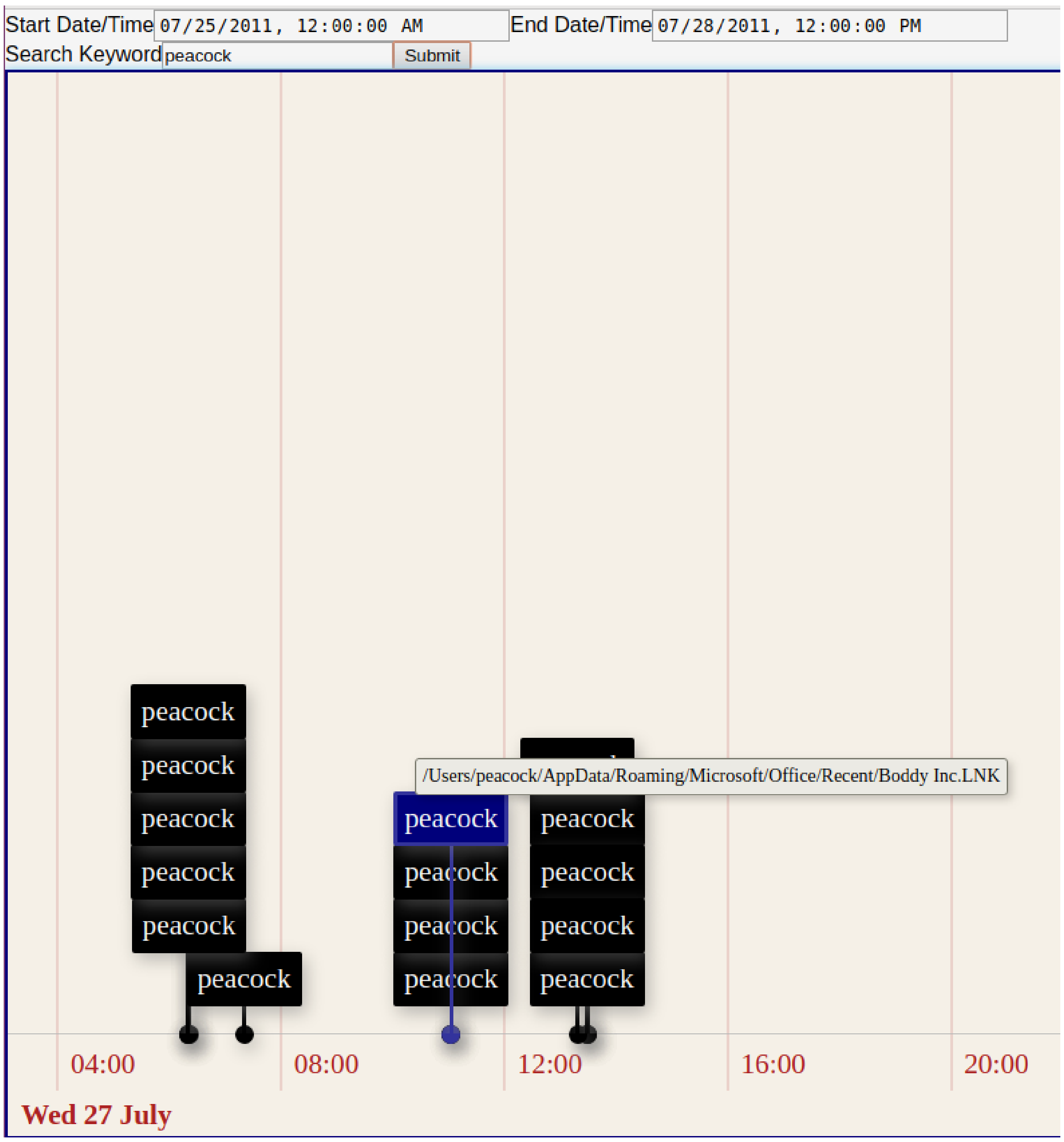
/Users/peacock/AppData/Roaming/Microsoft/Windows/Recent/Re_new_plan.lnk,
/Users/mustard/AppData/Roaming/Microsoft/Windows/Recent/too late.txt.lnk

Figure 36: Scenario 2 Timeline “peacock” Search.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword peacock Submit

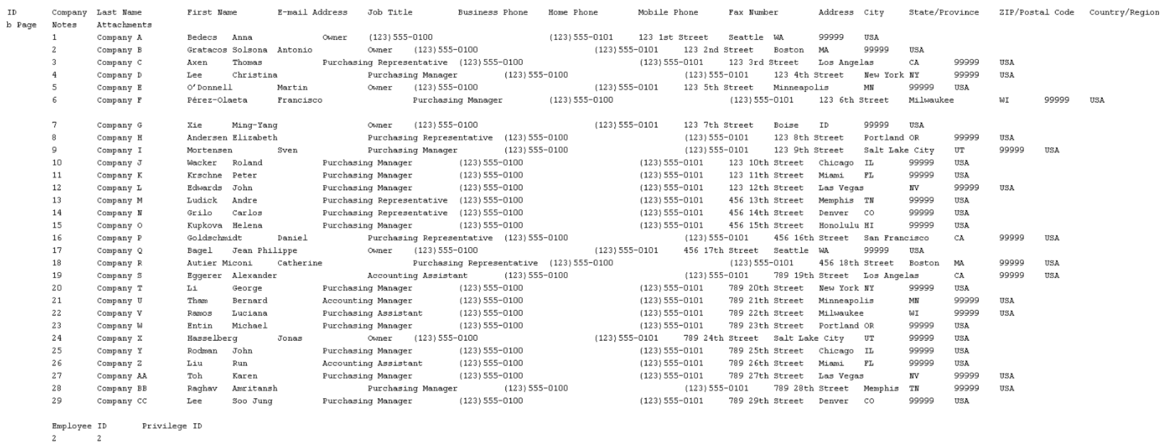
/Users/peacock/AppData/Roaming/Microsoft/Windows/Recent/USB DISK (E).lnk,
/Users/peacock/AppData/Roaming/Microsoft/Windows/Recent/Re_new_plan.lnk

Figure 37: Scenario 2 Timeline “peacock” Search 2.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword peacock

/Users/peacock/AppData/Roaming/Microsoft/Office/Recent/Boddy Inc.LNK

Figure 38: Scenario 2 Timeline “peacock” Search 3.

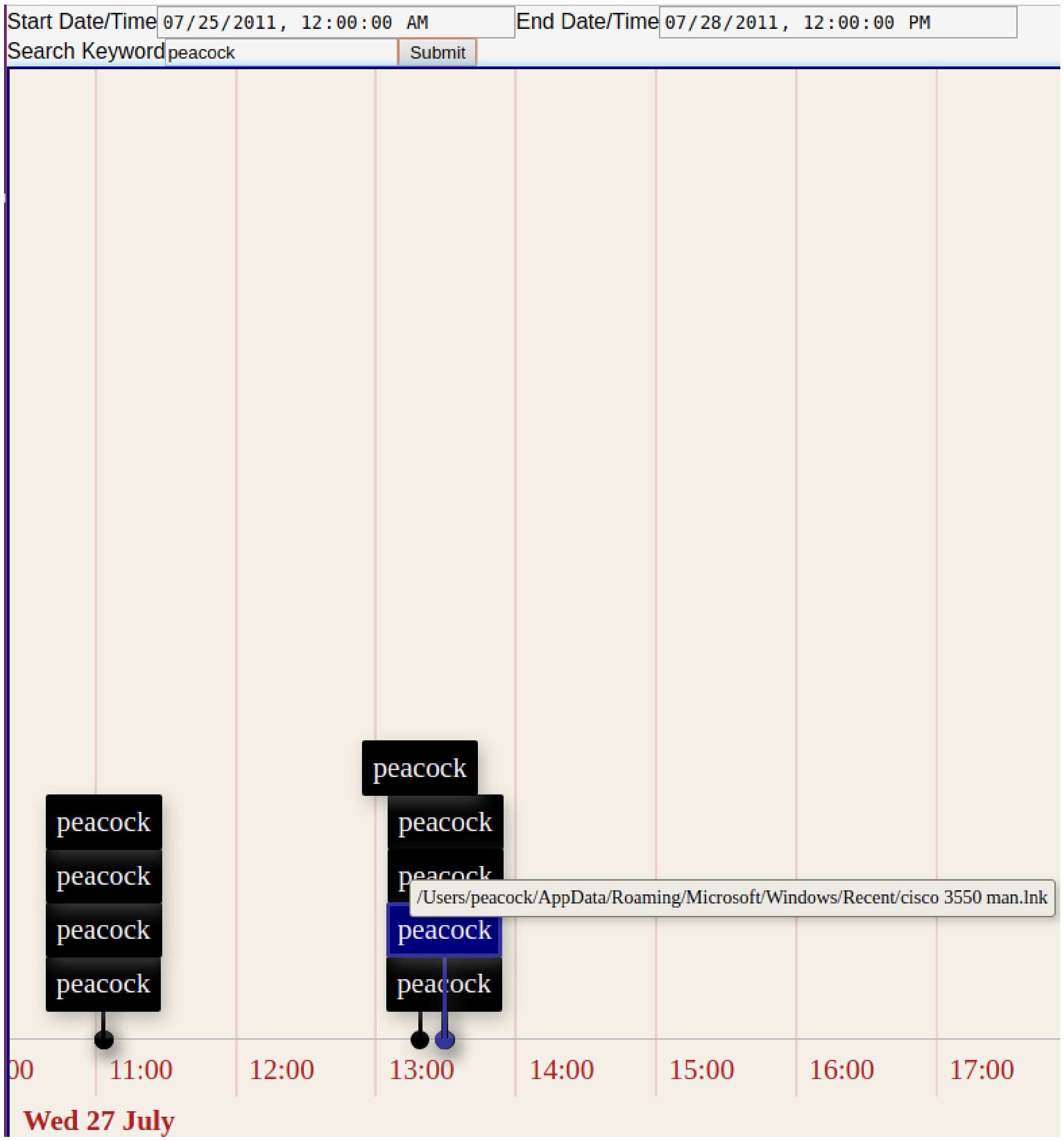


```
ID      Company  Last Name      First Name      E-mail Address    Job Title      Business Phone    Home Phone      Mobile Phone      Fax Number      Address  City    State/Province    ZIP/Postal Code   Country/Region
b Page  Notes    Attachments
        1        Company A      Bedecs   Anna               Owner    (123)555-0100                    (123)555-0101     123 1st Street    Seattle  WA      99999   USA
        2        Company B      Gratacos Solsona  Antonio          Owner    (123)555-0100                    (123)555-0101     123 2nd Street    Boston   MA      99999   USA
        3        Company C      Axen     Thomas             Purchasing Representative  (123)555-0100                    (123)555-0101     123 3rd Street    Los Angelas      CA      99999   USA
        4        Company D      Lee      Christina                 Purchasing Manager        (123)555-0100                    (123)555-0101     123 4th Street    New York NY      99999   USA
        5        Company E      O'Donnell         Martin           Owner    (123)555-0100                    (123)555-0101     123 5th Street    Minneapolis      MN      99999   USA
        6        Company F      Pérez-Olaeta      Francisco                 Purchasing Manager        (123)555-0100                    (123)555-0101     123 6th Street    Milwaukee        WI      99999   USA

        7        Company G      Xie      Ming-Yang                 Owner    (123)555-0100                    (123)555-0101     123 7th Street    Boise    ID      99999   USA
        8        Company H      Andersen Elizabeth                 Purchasing Representative  (123)555-0100                    (123)555-0101     123 8th Street    Portland OR      99999   USA
        9        Company I      Mortensen         Sven             Purchasing Manager        (123)555-0100                    (123)555-0101     123 9th Street    Salt Lake City   UT      99999   USA
        10       Company J      Wacker   Roland             Purchasing Manager        (123)555-0100                    (123)555-0101     123 10th Street   Chicago  IL      99999   USA
        11       Company K      Krschne  Peter              Purchasing Manager        (123)555-0100                    (123)555-0101     123 11th Street   Miami    FL      99999   USA
        12       Company L      Edwards  John               Purchasing Manager        (123)555-0100                    (123)555-0101     123 12th Street   Las Vegas        NV      99999   USA
        13       Company M      Ludick   Andre              Purchasing Representative  (123)555-0100                    (123)555-0101     456 13th Street   Memphis  TN      99999   USA
        14       Company N      Grilo    Carlos             Purchasing Representative  (123)555-0100                    (123)555-0101     456 14th Street   Denver   CO      99999   USA
        15       Company O      Kupkova  Helena             Purchasing Manager        (123)555-0100                    (123)555-0101     456 15th Street   Honolulu HI      99999   USA
        16       Company P      Goldschmidt       Daniel           Purchasing Representative  (123)555-0100                    (123)555-0101     456 16th Street   San Francisco    CA      99999   USA
        17       Company Q      Bagel    Jean Philippe             Owner    (123)555-0100                    (123)555-0101     456 17th Street   Seattle  WA      99999   USA
        18       Company R      Autier Miconi     Catherine                 Purchasing Representative  (123)555-0100                    (123)555-0101     456 18th Street   Boston   MA      99999   USA
        19       Company S      Eggerer  Alexander                 Accounting Assistant      (123)555-0100                    (123)555-0101     789 19th Street   Los Angelas      CA      99999   USA
        20       Company T      Li       George             Purchasing Manager        (123)555-0100                    (123)555-0101     789 20th Street   New York NY      99999   USA
        21       Company U      Tham     Bernard            Accounting Manager        (123)555-0100                    (123)555-0101     789 21th Street   Minneapolis      MN      99999   USA
        22       Company V      Ramos    Luciana            Purchasing Assistant      (123)555-0100                    (123)555-0101     789 22th Street   Milwaukee        WI      99999   USA
        23       Company W      Entin    Michael            Purchasing Manager        (123)555-0100                    (123)555-0101     789 23th Street   Portland OR      99999   USA
        24       Company X      Hasselberg        Jonas            Owner    (123)555-0100                    (123)555-0101     789 24th Street   Salt Lake City   UT      99999   USA
        25       Company Y      Rodman   John               Purchasing Manager        (123)555-0100                    (123)555-0101     789 25th Street   Chicago  IL      99999   USA
        26       Company Z      Liu      Run                Accounting Assistant      (123)555-0100                    (123)555-0101     789 26th Street   Miami    FL      99999   USA
        27       Company AA     Toh      Karen              Purchasing Manager        (123)555-0100                    (123)555-0101     789 27th Street   Las Vegas        NV      99999   USA
        28       Company BB     Raghav   Amritansh                 Purchasing Manager        (123)555-0100                    (123)555-0101     789 28th Street   Memphis  TN      99999   USA
        29       Company CC     Lee      Soo Jung           Purchasing Manager        (123)555-0100                    (123)555-0101     789 29th Street   Denver   CO      99999   USA

        Employee ID      Privilege ID
        2        2
```

Figure 39: Scenario 2 Boddy Inc. Database Snippet.


Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM
Search Keyword peacock

/Users/peacock/AppData/Roaming/Microsoft/Windows/Recent/cisco 3550 man.lnk

Figure 40: Scenario 2 “peacock” Search 4.

The examiner can then determine a connection between Mustard and Peacock based on the information thus far. Autopsy's e-mail search module produced messages between Peacock and Mustard, taken from various Outlook.pst sources. These messages, Figure 41, showed Mustard and Peacock stating they left messages but with no additional context of the actual message. A close look at the programs executed (Figure 33) between the time range of 25 Jul 2011 12:00:00 AM - 28 Jul 2011 12:00:00 PM showed "Microsoft" nodes with a trace of /ProgramData/Microsoft/Windows/Start Menu/Programs/Camouflage. A simple Google search revealed Camouflage was a program that provided steganography capabilities. A keyword search of "peacock mustard Microsoft", shown in Figure 42, revealed that this program was executed between the time Peacock accessed "Re_new_plan.txt" and reaccessed the same file as well as USB Disk (E) drive. The examiner can also determine Camouflage was accessed after Peacock touched the Boddy, Inc. database. It was evident through Autopsy that Boddy sent Mustard an e-mail needing to talk about accounting errors, as shown in Figure 41. Based on the e-mails, the activity between Mustard and Peacock, the examiner can determine some scheme was going on between Mustard and Peacock involving the company's clients and Boddy was beginning to notice the errors. Due to Camoflouge's presence, the user should also use steganography on the the various files Mustard and Peacock touched between this time period to find more information. These files include DSCF0734.JPG, DSCF0923.JPG, "too late.txt", "Re_new_plan.txt", and "cisco 3550 man.pdf".

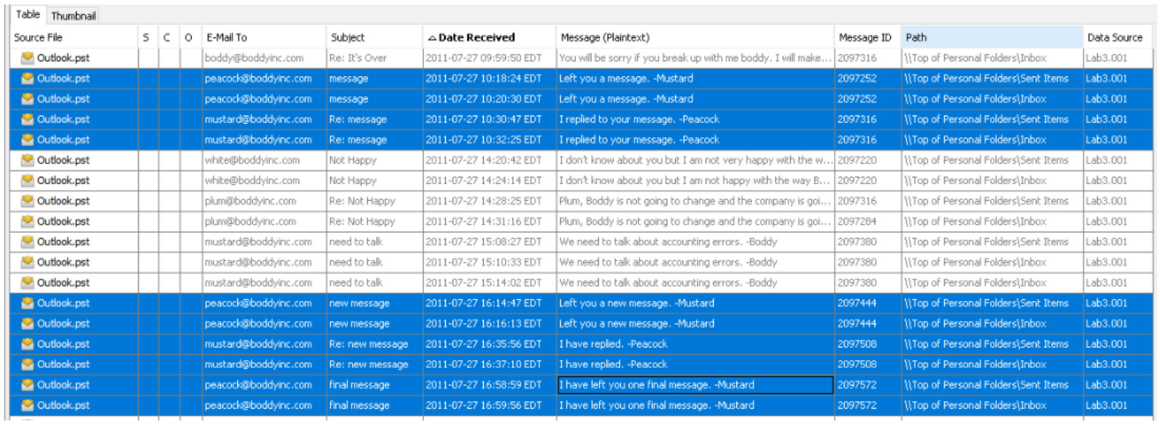
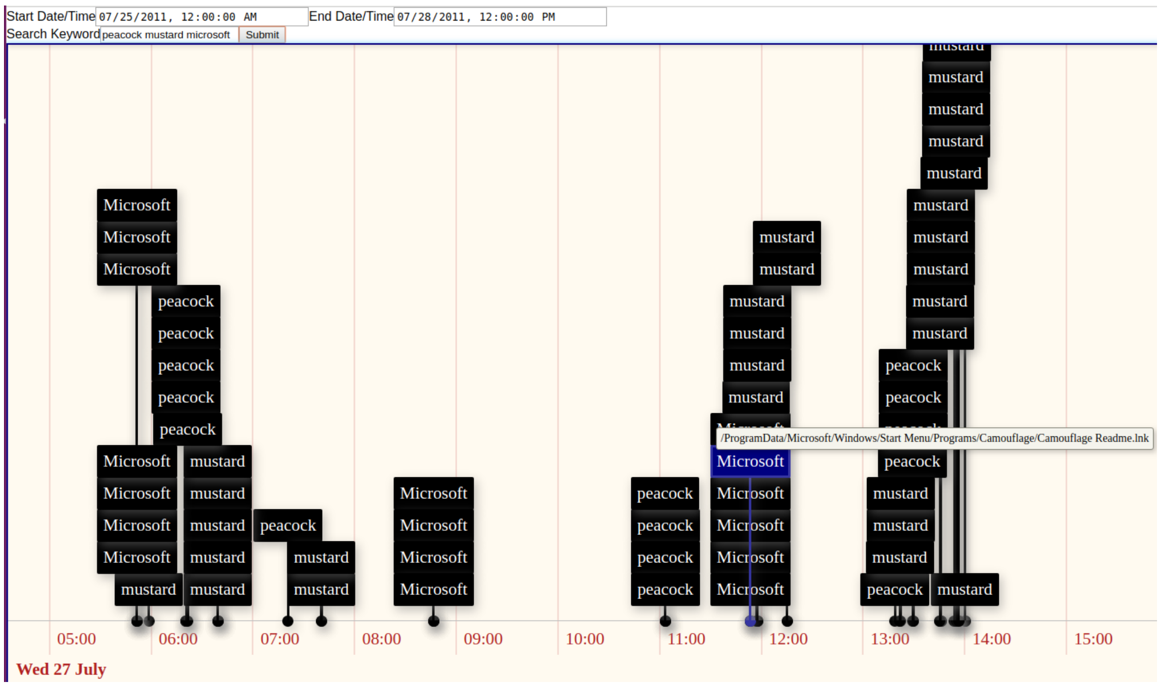


Table | Thumbnail

Source File	S	C	O	E-Mail To	Subject	Date Received	Message (Plaintext)	Message ID	Path	Data Source
Outlook.pst				boddy@boddyinc.com	Re: It's Over	2011-07-27 09:59:50 EDT	You will be sorry if you break up with me boddy. I will make…	2097316	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				peacock@boddyinc.com	message	2011-07-27 10:18:24 EDT	Left you a message. -Mustard	2097252	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				peacock@boddyinc.com	message	2011-07-27 10:20:30 EDT	Left you a message. -Mustard	2097252	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				mustard@boddyinc.com	Re: message	2011-07-27 10:30:47 EDT	I replied to your message. -Peacock	2097316	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				mustard@boddyinc.com	Re: message	2011-07-27 10:32:25 EDT	I replied to your message. -Peacock	2097316	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				white@boddyinc.com	Not Happy	2011-07-27 14:20:42 EDT	I don't know about you but I am not very happy with the w…	2097220	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				white@boddyinc.com	Not Happy	2011-07-27 14:24:14 EDT	I don't know about you but I am not happy with the way B…	2097220	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				plum@boddyinc.com	Re: Not Happy	2011-07-27 14:28:25 EDT	Plum, Boddy is not going to change and the company is goi…	2097316	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				plum@boddyinc.com	Re: Not Happy	2011-07-27 14:31:16 EDT	Plum, Boddy is not going to change and the company is goi…	2097284	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				mustard@boddyinc.com	need to talk	2011-07-27 15:08:27 EDT	We need to talk about accounting errors. -Boddy	2097380	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				mustard@boddyinc.com	need to talk	2011-07-27 15:10:33 EDT	We need to talk about accounting errors. -Boddy	2097380	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				mustard@boddyinc.com	need to talk	2011-07-27 15:14:02 EDT	We need to talk about accounting errors. -Boddy	2097380	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				peacock@boddyinc.com	new message	2011-07-27 16:14:47 EDT	Left you a new message. -Mustard	2097444	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				peacock@boddyinc.com	new message	2011-07-27 16:16:13 EDT	Left you a new message. -Mustard	2097444	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				mustard@boddyinc.com	Re: new message	2011-07-27 16:35:56 EDT	I have replied. -Peacock	2097508	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				mustard@boddyinc.com	Re: new message	2011-07-27 16:37:10 EDT	I have replied. -Peacock	2097508	\\Top of Personal Folders\Inbox	Lab3.001
Outlook.pst				peacock@boddyinc.com	final message	2011-07-27 16:58:59 EDT	I have left you one final message. -Mustard	2097572	\\Top of Personal Folders\Sent Items	Lab3.001
Outlook.pst				peacock@boddyinc.com	final message	2011-07-27 16:59:56 EDT	I have left you one final message. -Mustard	2097572	\\Top of Personal Folders\Inbox	Lab3.001

Figure 41: Scenario 2 Emails.



Start Date/Time 07/25/2011, 12:00:00 AM End Date/Time 07/28/2011, 12:00:00 PM

Search Keyword peacock mustard microsoft

/ProgramData/Microsoft/Windows/Start Menu/Programs/Camouflage/Camouflage Readme.lnk

Figure 42: Scenario 2 Timeline “peacock mustard Microsoft” Search.

5.3.2 Scenario 2 Analysis and Results

Autopsy's Keyword Search Module produced 65,513 hits across all the terms mentioned in Section 4.3. On the other hand, the timeline visualization produced 142 hits to sort through, which reduced the number of hits by 99%.

The combination of Autopsy and ECATV was able to locate most of the important evidence. The tools located 9 out of 13 pieces of digital evidence planted for the scenario. Many of the pieces of evidence were e-mail messages between the various employees, which were located using the e-mail search module of Autopsy. ECATV helped spur this search by showing the installation process of Microsoft Office 2007 as well as various employees accessing Outlook. Unfortunately, the timeline was unable to pinpoint the actual Google Talk messages, but at least notified the examiner of its existence and usage. Three pieces of digital evidence were "hidden" using stegonagraphy on the following files: "cisco 3550 man.pdf", DSCF0744.JPG, and "README2.HTM". ECATV was able to locate "cisco 3550 man.pdf" and "README2.HTM" and the employee(s) who accessed them. It was unable to find DSCF0744.JPG being executed by an employee, but it did demonstrate Mustard accessing similar JPEGs, DSCF0734.JPG and DSCF0923.JPG. These files were located in C:/pics, which was the parent directory for the key piece, DSCF0744.JPG, that ECATV was unable to display on its timeline. More importantly, Scenario 2 was able to demonstrate ECATV's capabilities in usage patterns. The tool made it easy to discover Camoflauge and led examiners down the path of stenography as a technique to look out for and even the specific software utilized. It also helped draw the connection between Mustard and Peacock. The last piece of evidence was an e-mail from Boddy that Scarlet forged using his account. This piece was found entirely using Autopsy's e-mail search module. The e-mails alone made it evident Boddy was involved in an affair with Scarlet and when Boddy attempted to end it, Scarlet threatened him, stating “You will be sorry if you break up with me boddy.” The forged e-mail sent to the entire company shows the e-mail was sent from Boddy, but the source of the message originated from Scarlet’s Outlook.pst.

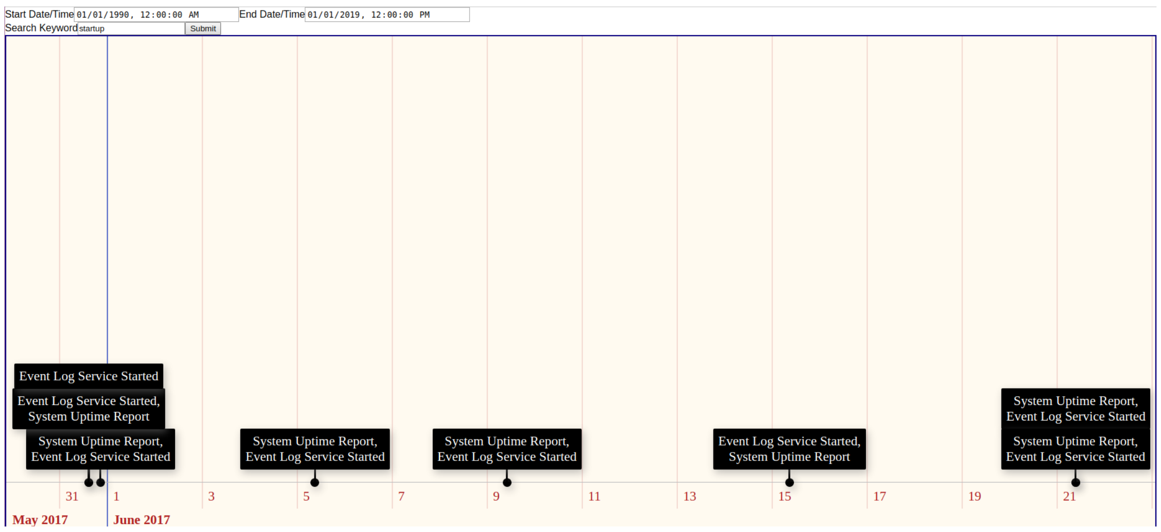
ECATV was unable to find the actual “cooked books” hidden through an alternate data stream. But Scenario 2 demonstrated that the tool and the e-mails helped establish the theory Peacock and Mustard were altering what were written in the accounts. The examiner would have eventually found the actual document through the clues and information discovered throughout this process. It is just that ECATV was unable to locate the actual file. The last two pieces of digital evidence missed were the clues in the slack space of the master boot record and the main drive. These would not have been discovered because the ingest process did not account for data located in the slack space and is a limitation of ECATV. Overall, ECATV was able to help locate 69% pieces of digital evidence.

5.4 Scenario 3

This scenario began with an altercation that took place in the Bredon Library. Police responded to a report that Golden Mustard and Ruby Scarlet were arguing over use of one of the library’s computers. During the arrest, police discovered narcotics in Mustard’s possession. The computer’s hard drive, along with a pile of USB drives near the computer were seized as well. The lead investigator submitted a request to the email providers of the local users and copies of the emails were provided to the forensic examiner. The goals were similar to scenario two and examiners were charged with building a narrative of the case for the lead investigator as well as a reconstruction of the crimes.

5.4.1 Scenario 3 ECATV Walkthrough

Again, the first step for this scenario was to narrow down the time range of interest. Figure 43 shows the initial time range of 1 Jan 1990 12:00:00 AM - 1 Jan 2019 12:00:00 PM and "startup" keyword used to see when the system was started. The results showed the time range could be narrowed down to 30 May 2017 - 22 Jun 2017. The time range of 30 May 2017 12:00:00 AM - 22 Jun 2017 12:00:00 PM was used for each subsequent searches to ensure all activity between these dates were captured.



Start Date/Time 01/01/1990, 12:00:00 AM
End Date/Time 01/01/2019, 12:00:00 PM
Search Keyword startup
Submit
Event Log Service Started
Event Log Service Started, System Uptime Report
System Uptime Report, Event Log Service Started
System Uptime Report, Event Log Service Started
System Uptime Report, Event Log Service Started
Event Log Service Started, System Uptime Report
System Uptime Report, Event Log Service Started
System Uptime Report, Event Log Service Started
31
1
3
5
7
9
11
13
15
17
19
21
May 2017
June 2017

Figure 43: Scenario 2 Timeline "startup" Search.

The second step is the same as the second step in scenario two, which entailed searching for installed programs. A search of a 30 May 2017 12:00:00 AM - 22 Jun 2017 12:00:00 PM and "install" showed a Google update finishing. The other events showed various Microsoft programs, the English versions, were installed as well as a Windows 10 update.

This scenario was different from scenario two in Section 5.3 because there were seven different USB's that potentially contained key pieces of evidence. This scenario also involved a lead investigator who submitted a request to e-mail providers to obtain the messages of the local users. Many of the actual key pieces of digital evidence were not found on the Windows image, but ECATV was still useful in identifying the remnants these pieces left behind when interacting with the workstation. The scenario essentially involved the user, Pearl White, having Golden Mustard set up drug deals. Mustard accomplished this through various e-mails he sent to Pym. During this time, White embedded a message using stegonagraphy onto fish_4.jpg. The text file contained the message “The next exchange will at 4pm on the corner of 9th and Lake.” White also embedded a message onto the Black_Moor_Fish.jpg’s alternate data stream that contained the message “Blackmail is your problem not mine - you owe me, even more now. Next exchange at 7th and Maple at 7pm.” The other pieces of digital evidence were associated to Forest Green. This user installed a keylogger Powershell script onto the system and began blackmailing Mustard. The key pieces of evidences that revealed this information were logged.txt and and e-mail from Green to Mustard which stated “I have proof that you are working for the notorious Queen White and if you don’t leave me $1,000 in the pipe by the computer tomorrow I will notify the authorities!” ECATV was evaluated based on not only its ability to find the digital evidence of the Windows image, but also its ability to provide the examiner with a solid base to work on rather than aimlessly searching the image.

The third search, Figure 44, provided an overview of the active users within the time range of 30 May 2017 12:00:00 AM - 22 Jun 2017 12:00:00 PM. We determined that Mustard, Green, White, and Peacock were the most active users based on the results. A search with the same time range but with the “white” keyword revealed White accessed various JPGs. This search was further filtered to include “jpg” which showed a Windows Powershell was launched (Figure 45) at approximately 21 Jun 2017 06:20 AM. Then between 07:40 AM and 07:50 AM, White accessed fish_4.lnk, Black_Moor_Fish.lnk, and the Public Pictures file location, as shown in Figures 46, 47, and 48.

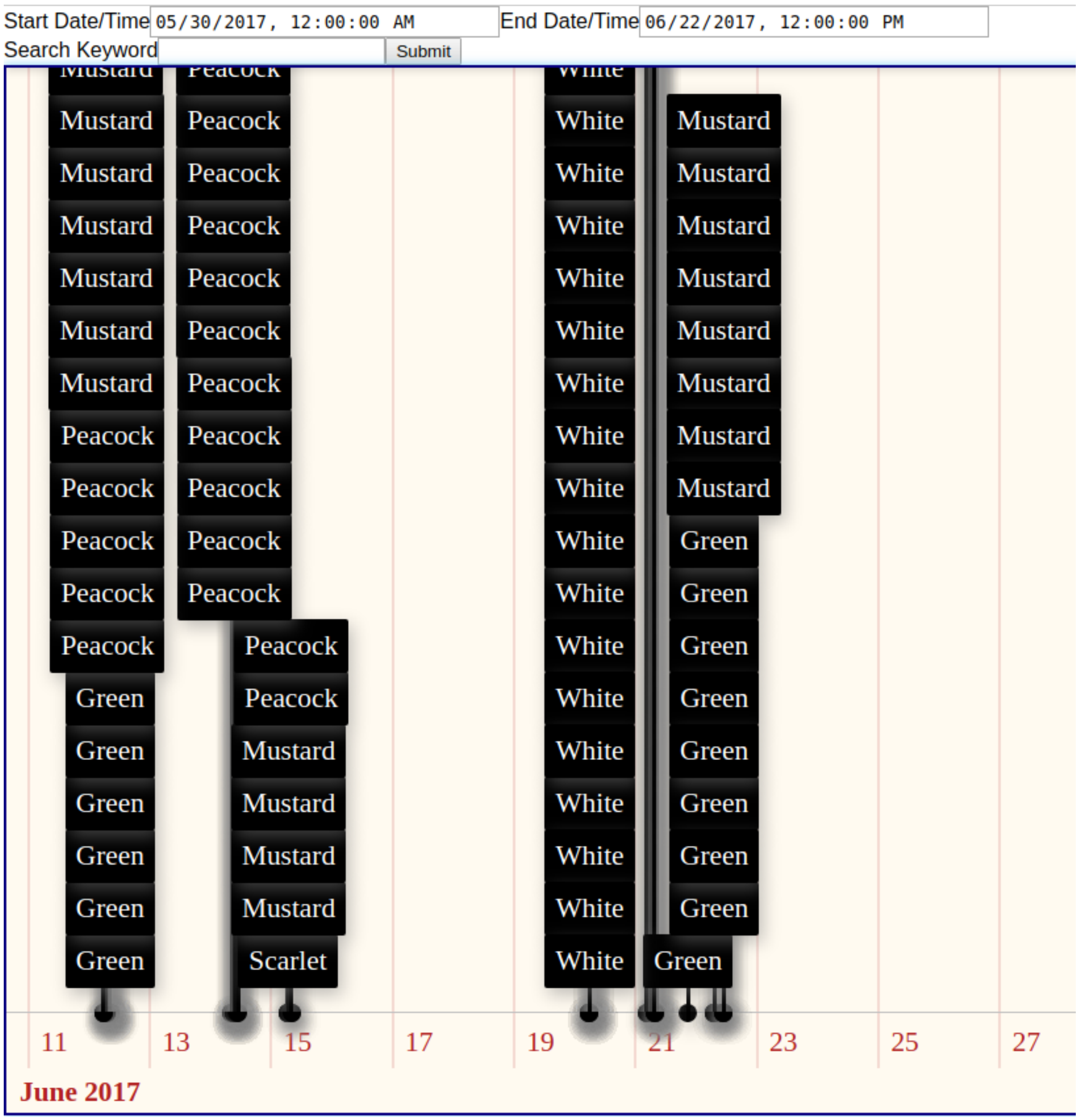
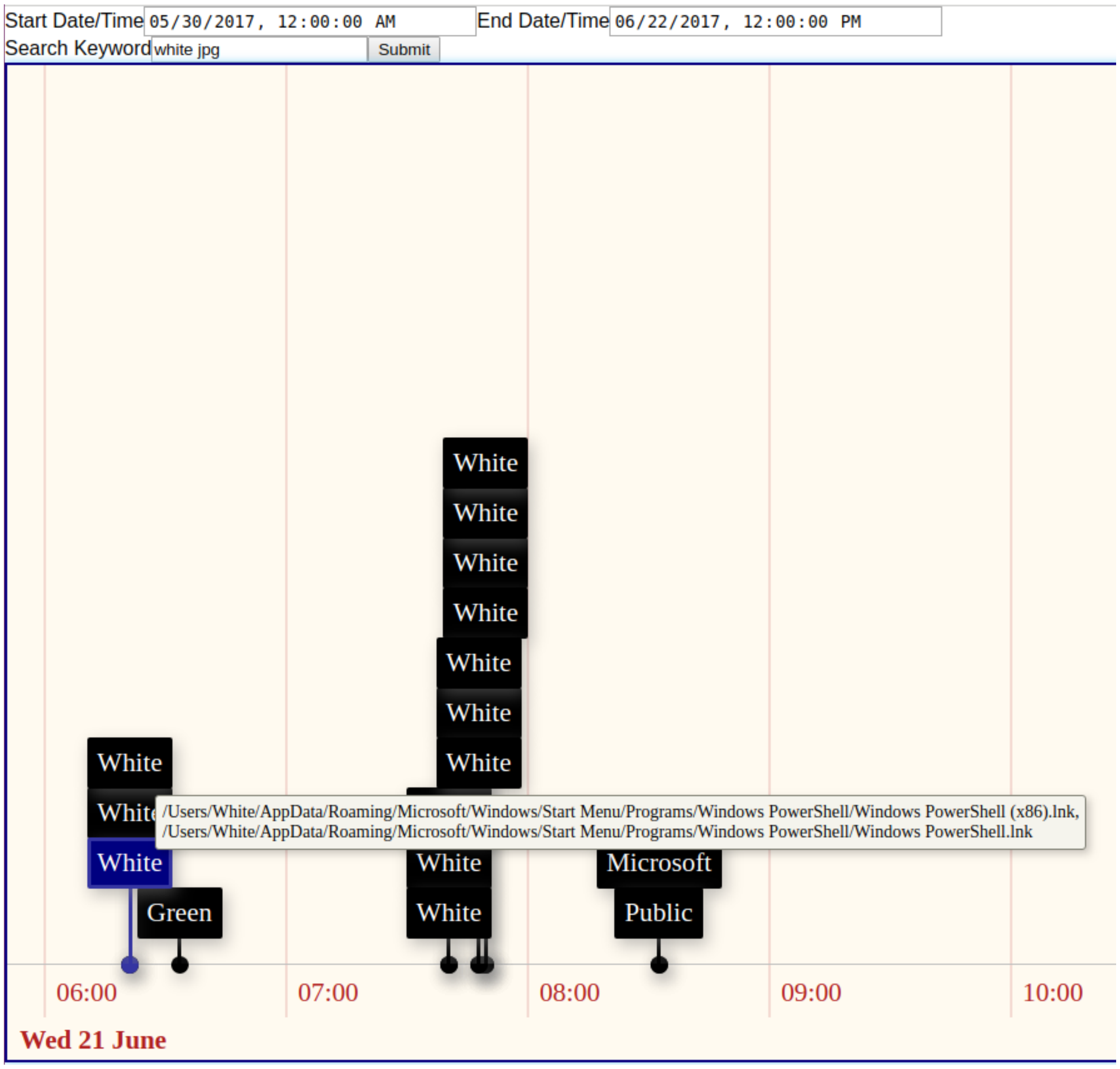

Start Date/Time 05/30/2017, 12:00:00 AM End Date/Time 06/22/2017, 12:00:00 PM
Search Keyword
Submit
Mustard Peacock White Mustard
Mustard Peacock White Mustard
Mustard Peacock White Mustard
Mustard Peacock White Mustard
Mustard Peacock White Mustard
Mustard Peacock White Mustard
Peacock Peacock White Mustard
Peacock Peacock White Mustard
Peacock Peacock White Green
Peacock Peacock White Green
Peacock Peacock White Green
Green Peacock White Green
Green Mustard White Green
Green Mustard White Green
Green Mustard White Green
Green Mustard White Green
Green Scarlet White Green
11 13 15 17 19 21 23 25 27
June 2017

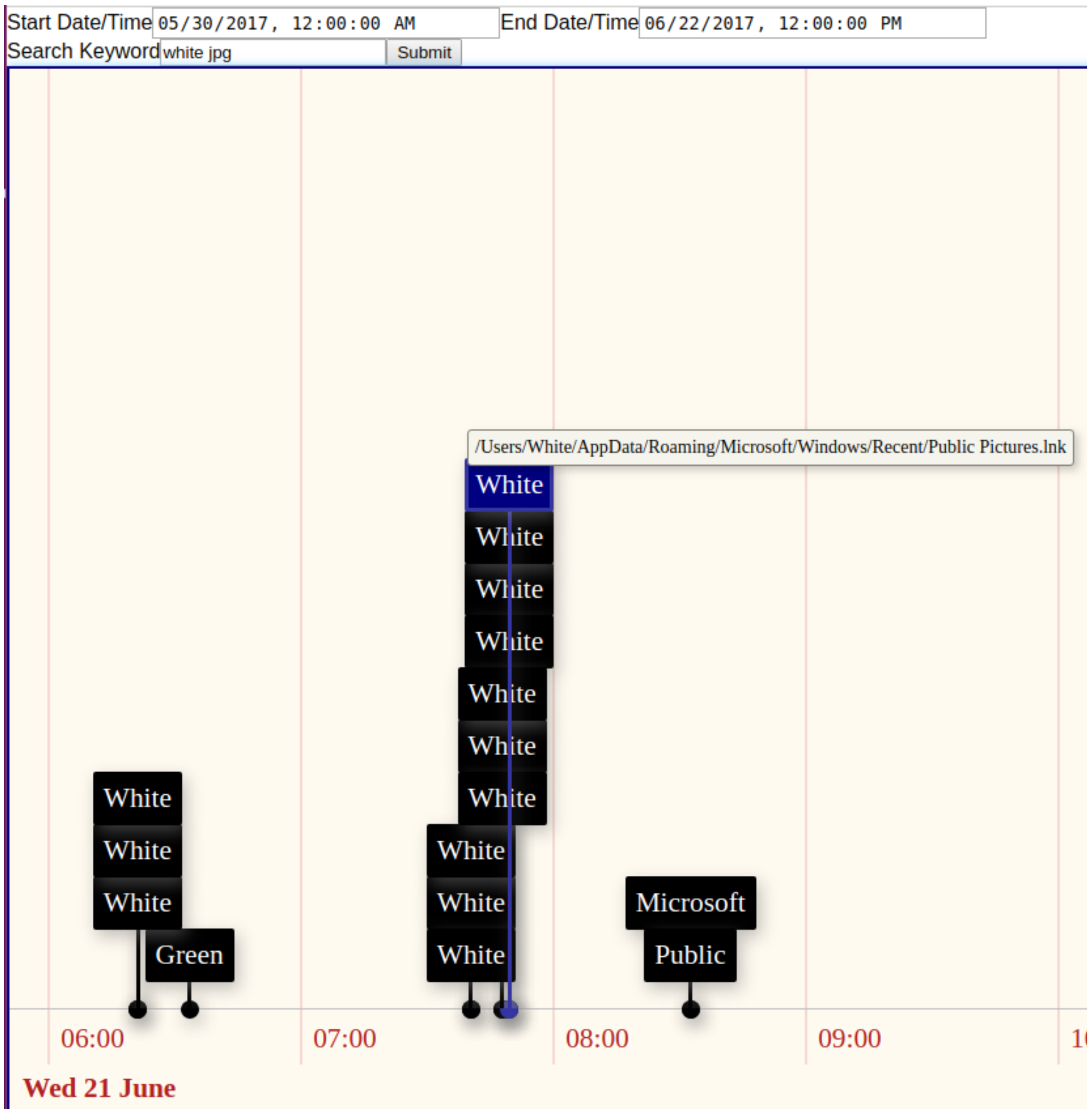
Figure 44: Scenario 3 Timeline User Activity Search.



Start Date/Time 05/30/2017, 12:00:00 AM End Date/Time 06/22/2017, 12:00:00 PM
Search Keyword white jpg Submit

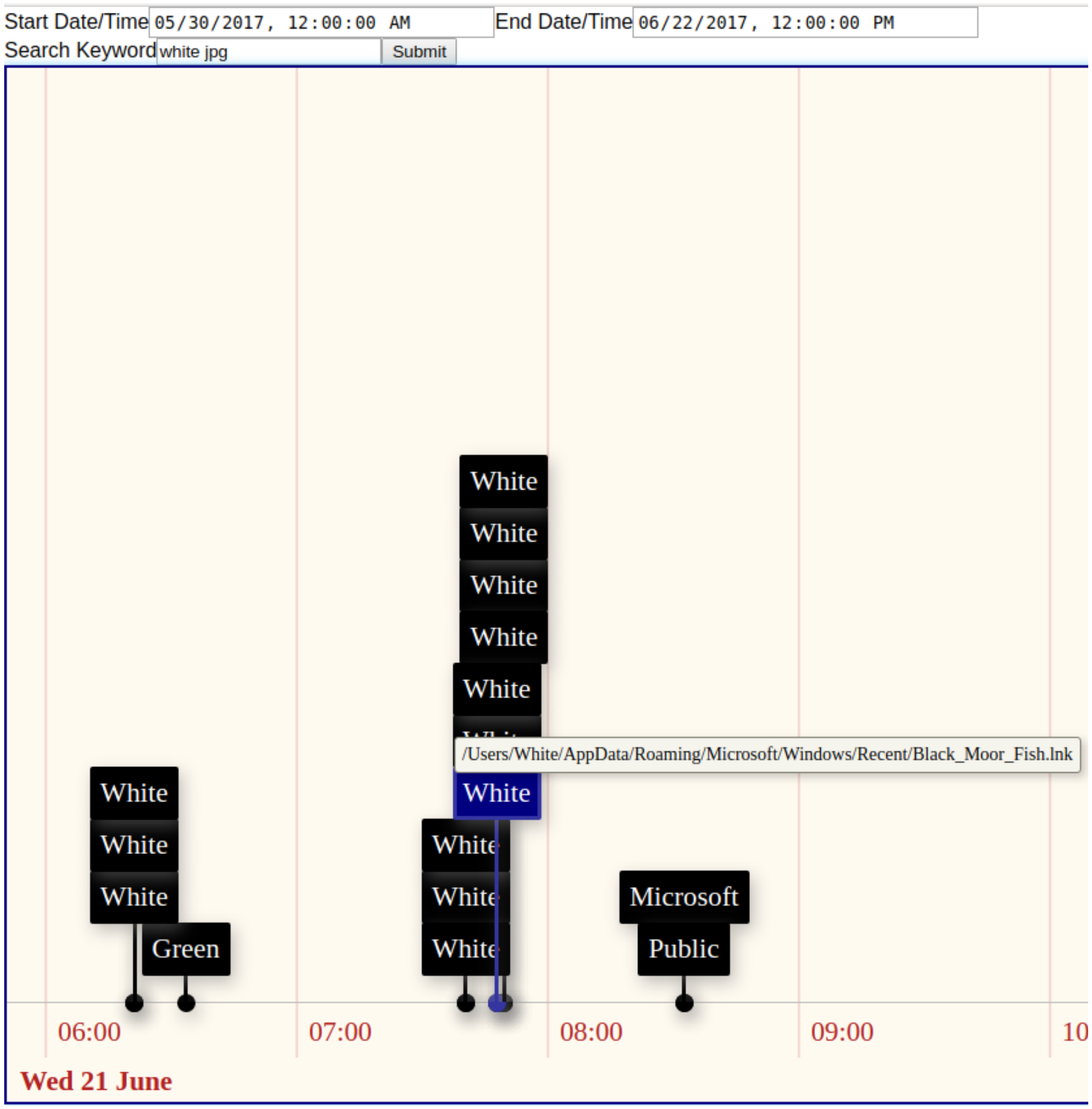
/Users/White/AppData/Roaming/Microsoft/Windows/Start Menu/Programs/Windows PowerShell/Windows PowerShell (x86).lnk,
/Users/White/AppData/Roaming/Microsoft/Windows/Start Menu/Programs/Windows PowerShell/Windows PowerShell.lnk

Figure 45: Scenario 3 Timeline "white" Search.


Start Date/Time 05/30/2017, 12:00:00 AM End Date/Time 06/22/2017, 12:00:00 PM
Search Keyword white jpg Submit

/Users/White/AppData/Roaming/Microsoft/Windows/Recent/Public Pictures.lnk

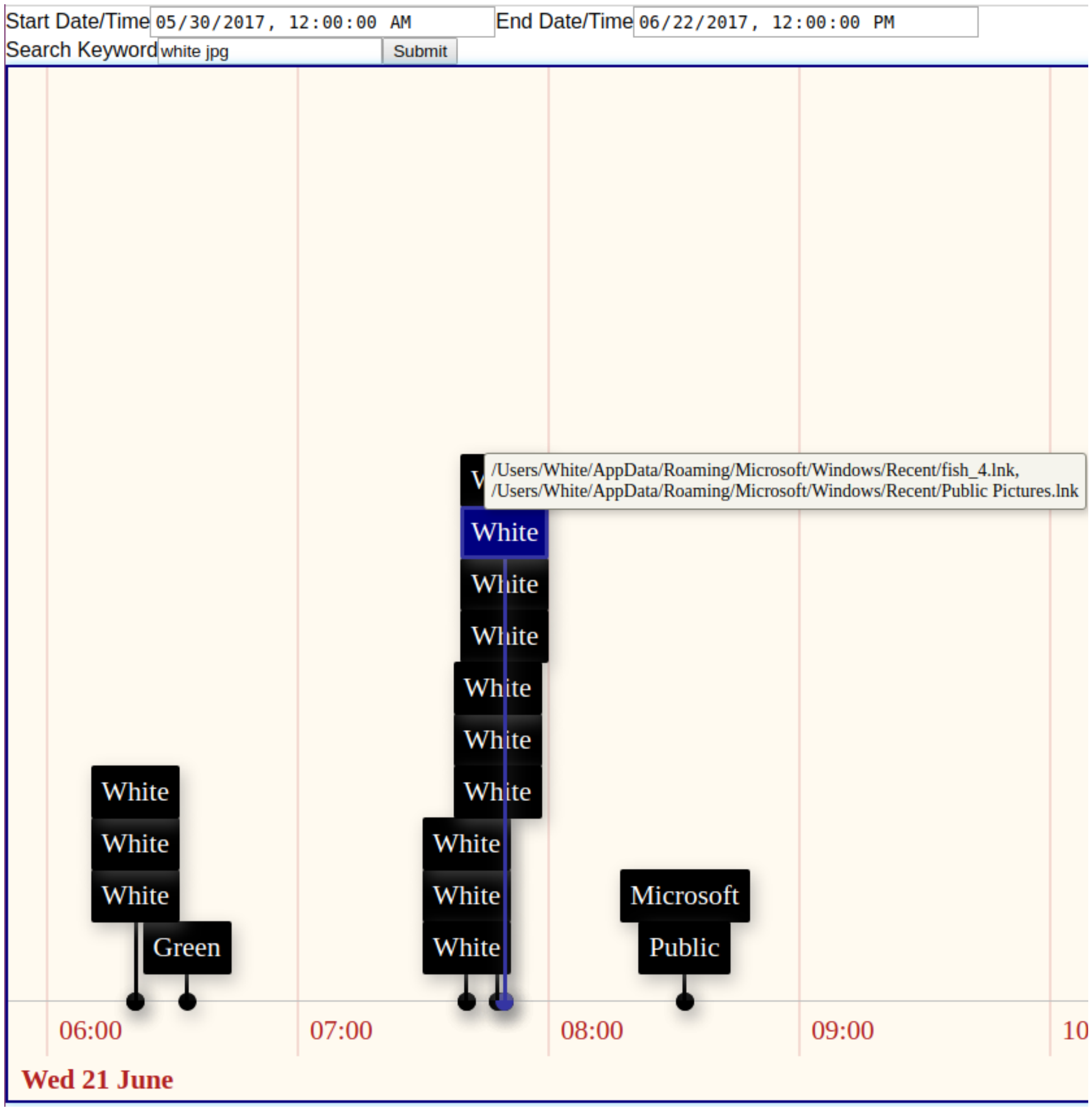
Figure 46: Scenario 3 Timeline “white jpg” Search.



Start Date/Time 05/30/2017, 12:00:00 AM End Date/Time 06/22/2017, 12:00:00 PM
Search Keyword white jpg Submit

/Users/White/AppData/Roaming/Microsoft/Windows/Recent/Black_Moor_Fish.lnk

Figure 47: Scenario 3 Timeline “white jpg” Search.



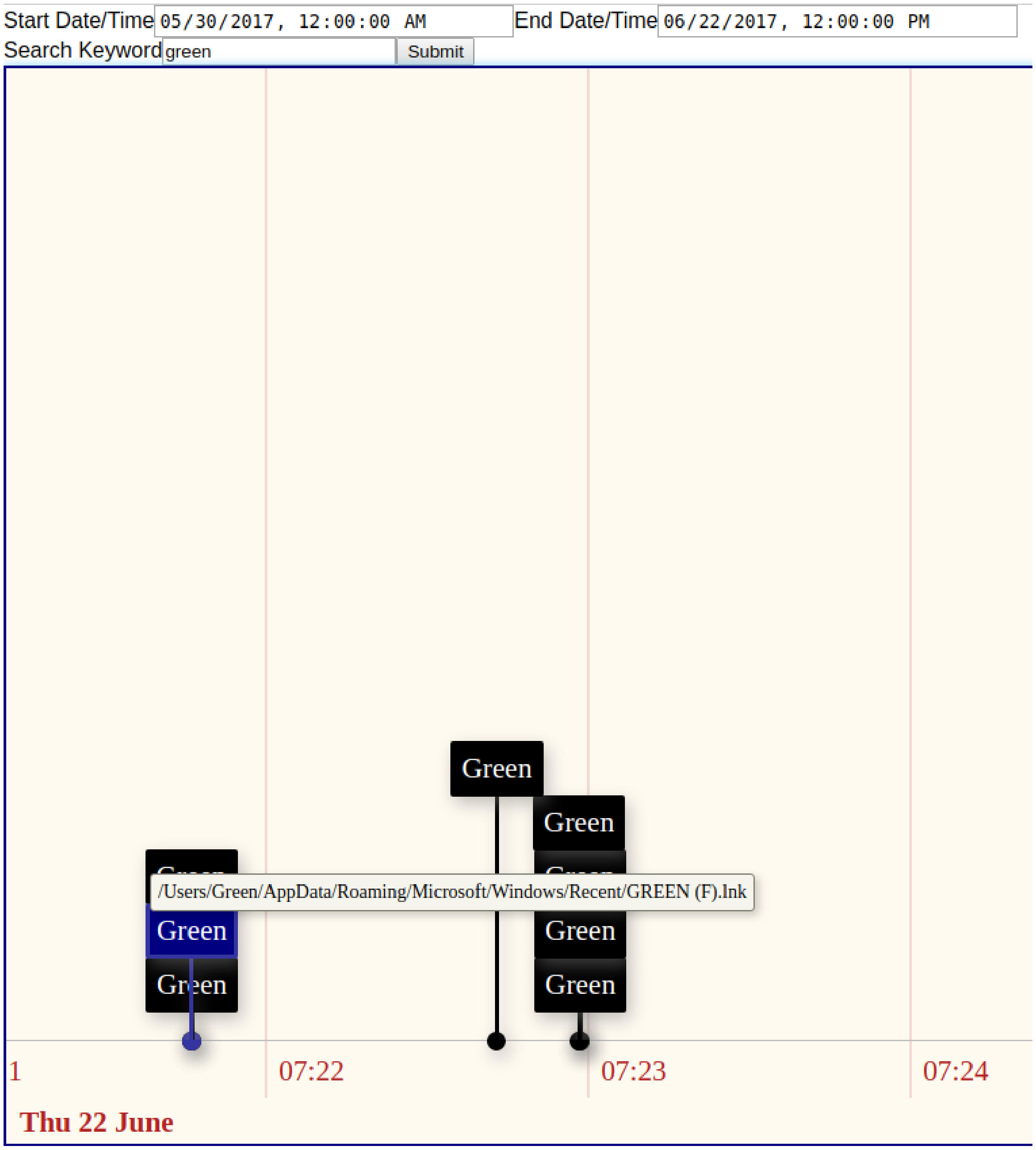
Start Date/Time 05/30/2017, 12:00:00 AM End Date/Time 06/22/2017, 12:00:00 PM

Search Keyword white jpg Submit

/Users/White/AppData/Roaming/Microsoft/Windows/Recent/fish_4.lnk,
/Users/White/AppData/Roaming/Microsoft/Windows/Recent/Public Pictures.lnk

Figure 48: Scenario 3 Timeline “white jpg” Search.

It is important to note these searches and ECATV did not provide the actual pieces of evidence, such as the message that were hidden within the alternate data stream or embedded using stenography. But it located the items of interest allowing the examiners to begin drilling down on these specific files. The same can be said when using the keyword “green”. This search revealed a USB was inserted and mounted as the F: drive and was named GREEN (Figure 49). Soon after Green accessed Parse_Keys.lnk (Figure 50). A deeper dive into Autopsy showed this link file originated from Parse_Keys.ps1 from the GREEN USB, as shown in Figure 51. This search through ECATV identified that out of the seven USB’s, the GREEN USB required further examination. The USB in fact contains evidence of a keylogger and the trace revealed in Figure 50, which was /Users/Green/AppData/Roaming/Microsoft-/Windows/Recent/Parse_Keys.lnk, helped tie in the fact the key logger originated from Green or whoever had access to Green’s user account on the Windows workstation.



Start Date/Time 05/30/2017, 12:00:00 AM End Date/Time 06/22/2017, 12:00:00 PM
Search Keyword green

/Users/Green/AppData/Roaming/Microsoft/Windows/Recent/GREEN (F).lnk

Figure 49: Scenario 3 Timeline "green" Search.

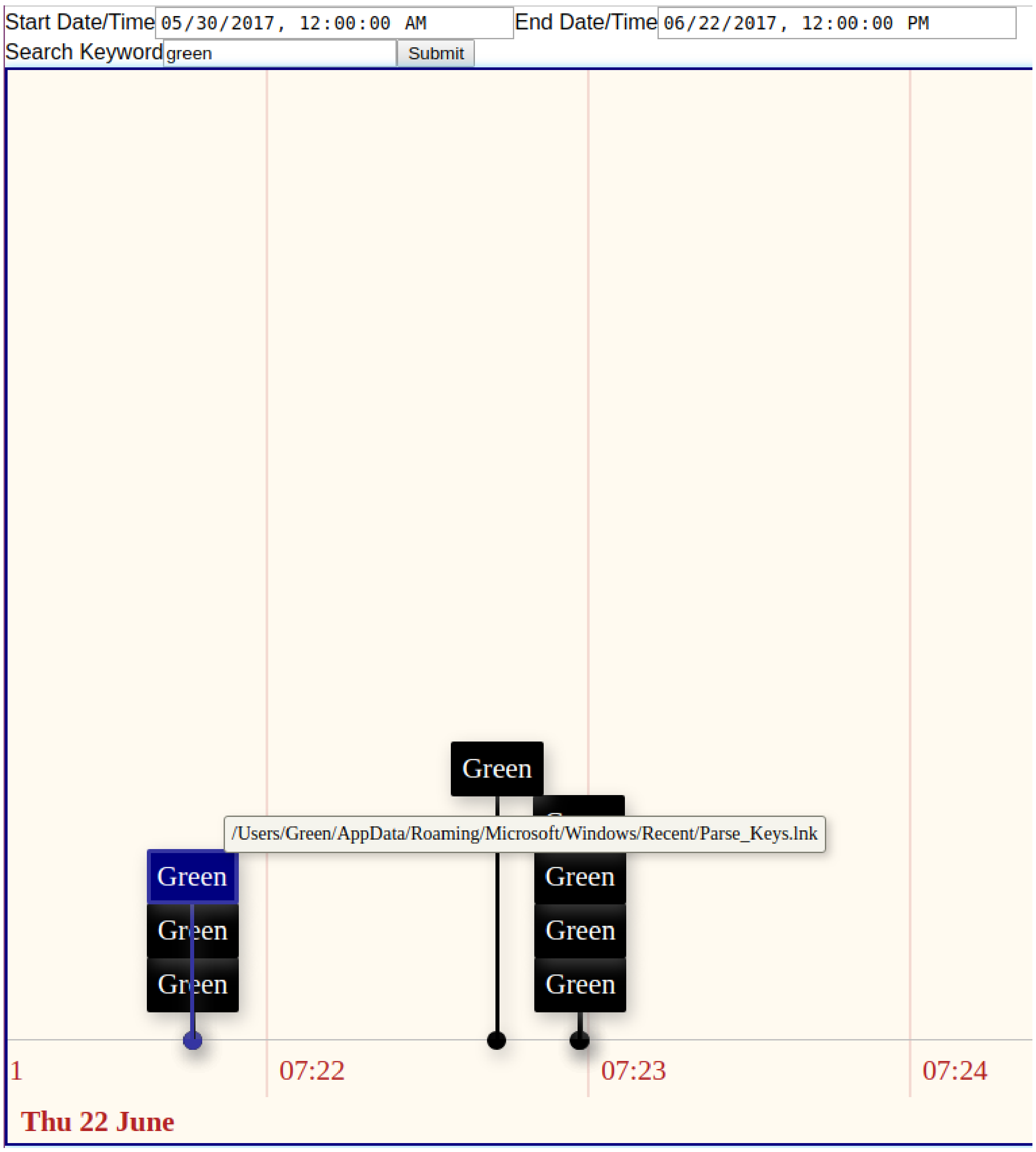
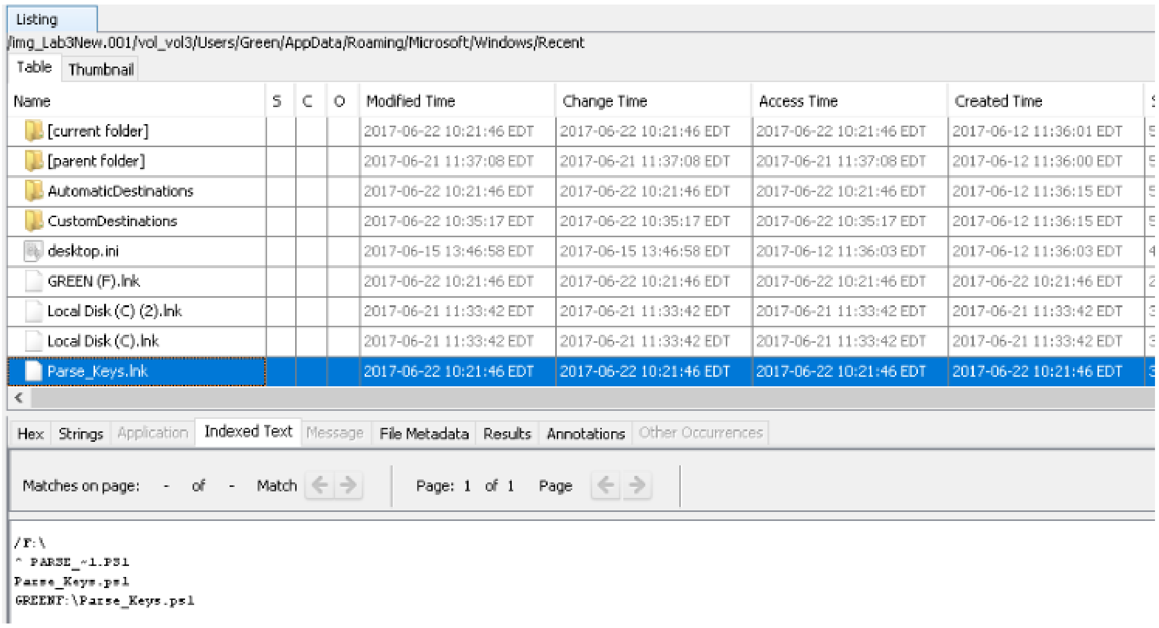


Figure 50: Scenario 3 Timeline “green” Search.



/img_Lab3New.001/vol_vol3/Users/Green/AppData/Roaming/Microsoft/Windows/Recent

Name	S	C	O	Modified Time	Change Time	Access Time	Created Time
[current folder]				2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-12 11:36:01 EDT
[parent folder]				2017-06-21 11:37:08 EDT	2017-06-21 11:37:08 EDT	2017-06-21 11:37:08 EDT	2017-06-12 11:36:00 EDT
AutomaticDestinations				2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-12 11:36:15 EDT
CustomDestinations				2017-06-22 10:35:17 EDT	2017-06-22 10:35:17 EDT	2017-06-22 10:35:17 EDT	2017-06-12 11:36:15 EDT
desktop.ini				2017-06-15 13:46:58 EDT	2017-06-15 13:46:58 EDT	2017-06-12 11:36:03 EDT	2017-06-12 11:36:03 EDT
GREEN (F).lnk				2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT
Local Disk (C) (2).lnk				2017-06-21 11:33:42 EDT	2017-06-21 11:33:42 EDT	2017-06-21 11:33:42 EDT	2017-06-21 11:33:42 EDT
Local Disk (C).lnk				2017-06-21 11:33:42 EDT	2017-06-21 11:33:42 EDT	2017-06-21 11:33:42 EDT	2017-06-21 11:33:42 EDT
Parse_Keys.lnk				2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT	2017-06-22 10:21:46 EDT

```
/F:\
^ PARSE_~1.PS1
Parse_Keys.ps1
GREENF:\Parse_Keys.ps1
```

Figure 51: Scenario 3 Parse_Keys.lnk.

5.4.2 Scenario 3 Analysis and Results

Autopsy's Keyword Search Module produced 159,918 hits across all the terms mentioned in Section 4.3. On the other hand, the timeline visualization produced 225 hits to sort through, which reduced the number of hits by 99%.

The scenario involved locating 13 pieces of key evidence but two of them were e-mails which were obtained from the email providers. Thus ECATV was evaluated based on its ability to find the other 11 pieces. The tool helped locate eight of these pieces, which included the alternate data stream from Black_Moor_Fish.jpg, the embedded message on fish_4.jpg, the keylogger, the Powershell being executed, the downloading of the fish pictures into the Public folder, and the associated activities.

ECATV was unable to locate two pieces of digital evidence associated to the execution of WinHex; one by Mustard and one by White. These pieces of evidence would tip the examiner to look for embedding in the partition slack, but ECATV alone was unable to identify this. The tool also failed to locate the log file the keylogger produced. This text file contained all the keys that were logged since Green installed the program. It showed evidence of Mustard typing an email to White stating it was "the last time I arrange a drop for you" as well as the message "My debts should be fully repaid!" The actual file, logged.txt, was on the GREEN USB and the examiner would have eventually located it after discovering the USB was important. But ECATV was unable to create an abstraction node for the execution of logged.txt associated to any of the users. This is because for some reason the Windows image did not create a prefetch or LNK associated to logged.txt. Autopsy revealed no such artifact existed. Regardless this was one of pieces of digital evidence planted for the scenario and ECATV was unable to identify it.

5.5 Summary

This research evaluated ECATV based on two objective measurements; the number of the key pieces of evidence the tool was able to locate and the number of search results it produced compared to another open source digital forensics tool, Autopsy. Overall, ECATV reduced the number of search results by at least 90% for the three scenarios, significantly reducing the amount of search terms an examiner would have to sort through. The tool also located around 70% of the key pieces of digital evidence. There were some limitations to ECATV and its ingest process that caused the tool to be unable to locate some of the evidence. Table 1 shows the breakdown of the results for each scenario.

Table 1: ECATV and Scenario Results.

	Scenario 1	**Scenario 2**	**Scenario 3**
Digital Evidence Located	6 out of 6 **(100%)**	9 out of 13 **(69%)**	8 out of 11 **(73%)**
Search Results Produced by ECATV	397	142	225
Search Results Produced by Autopsy	13,107	65,513	159,918
Percentage of Reduction of Search Results	97%	99%	99%

VI. Conclusions

6.1 Summary

This study looked into the possibility of integrating temporal abstraction [33] and interactive visualization [36] into a digital forensics examiners' current process. This document presents the entire digital forensics examination process. Event Correlation and Abstraction Timeline Visualization (ECATV) is presented as a novel examination capability that works with the existing process. Specifically, ECATV adds a keyword search feature onto the timeline to match the process, better filter results, and increase the robustness of its operational capabilities. This was done through dissecting Autopsy's communication method, called the Blackboard, between various modules and locating the necessary data to augment the timeline visualization's database.

Overall, ECATV was able to significantly reduce the number of search results by more than 90%. The tool was tested on three different scenarios and was able to locate about 70% of the key pieces of digital evidence. The objective of this study was to reduce the amount of time spent during the analysis portion of a digital forensics examination by combining different strategies aimed at the same goal. This was done by combining the research done in [33] and [36] to create a timeline visualization backed with a native graph database that was filterable with keywords. Testing ECATV through the three different scenarios revealed that the tool made it easier to detect patterns and user behavior. When the image contained multiple users, the keyword filterable timeline made it easier to isolate the activities of a single user. It also had the ability to filter results down to multiple specific users making it easier to detect interactions between various users, a feature useful on enterprise workstations.

6.2 Limitations

Currently there are three main limitations with ECATV. First, the tool relies on Windows Event Logs to create its power events abstraction nodes. If logging was not enabled, ECATV ultimately will be unable to show when the computer in question was started. This brings us to the second limitation being the operating system. The tools used during the ingest process were specific to Windows images. ECATV would not work on Linux and Mac operating systems. Lastly, the tool was unable to locate key pieces of digital evidence if they were hidden in the slack space. For these cases, the examiner would have to manually search through the space to find the evidence.

6.3 Future Work

Future work for the development of ECATV would involve streamlining the process into a single module for Autopsy. Currently the ingest process combines the use of various tools and different stages. It can be a labor intensive process that requires knowledge of multiple different languages, databases, and schemas. The tools written for the first phase of the ingest process was also split up into multiple different docker containers. Streamlining the ingest process into a single pipeline would allow a larger population of forensic examiners to take advantage of ECATV's benefits. Developing a single module for Autopsy would also enable the ability for adhoc keyword searches. ECATV currently takes the results from Autopsy's keyword search and combines its results into ECATV's Neo4j database. But if it were developed as a single module for Autopsy, it would be able to communicate with Autopsy through the Blackboard and automatically add the results into its Neo4j database. This would allow for evolving search terms and examiners to add search keywords on-the-fly.

The other piece of future work would involve conducting a user study on ECATV. This research evaluated the objective benefits of ECATV but was unable to test the subjective ones. A user study would test whether the tool works as intended and reduces the human analysis time spent during a digital forensics examination. It would also determine the ease of the tool's usability. If the tool was difficult to utilize for the average user, it would defeat the purpose of attempting to reduce the human analysis portion.

Appendix A. Windows 7 Lab 3

Cyber Forensics - CSCE 527 Media Imaging and Analysis

Purpose: Image and examine a HD and unlock its many secrets.

Scenario: Millionaire and noted philanthropist Mr. Boddy has been murdered. Since Mr. Boddys wife was out of the country and is the primary interest owner of the company, the investigators have ruled out romantic and inheritance as motives. The investigators are certain that whoever murdered Mr. Boddy must work in his AF contracting company Boddy, Inc. Boddy, Inc. is a small business and consists of Mr. Boddy, Col Mustard, Mr. Green, Prof Plum, Miss Scarlet, Mrs. Peacock, and Mrs. White. Col Mustard is the vice president, and is in charge of schmoozing clients and garnering contracts. Mr. Green is in charge of Information Technology support. Prof Plum is director in charge of research and development. Miss Scarlet is the office manager. Mrs. Peacock is the companys accountant. Mrs. White works with Prof Plum as the lead researcher.

Assignment: Perform an analysis of the imaged drive and media. The hard drive image is available on the LISSARD share drive. You will have imaged all of the media, except for the hard drive during scene processing. The goal is a full reconstruction of the crime(s). Find as much evidence as you can. Determine who has committed the crime, why they committed the crime, and the timeline associated with the crime. Build a narrative of the case for the lead investigator. Document where and how you find it. The search terms cleared by the warrant for this search include all the employees names, and aspects concerning Mr. Boddys demise. If you evolve search terms, justify why you evolved them. Prepare a written report indicating the teams findings and the techniques used. Be careful to maintain a correct trail of evidence, including tagging, checking in and out of evidence, etc. Also, discuss in length the tools that you used.

Appendix B. Windows 10 Lab 3

Cyber Forensics - CSCE 527 Media Imaging and Analysis

Purpose: Examine a HD image and unlock its secrets.

Scenario: A hard drive from a computer from the Bredon Library has been seized. It is requested that you examine the image, it is suspected of containing evidence related to illegal narcotics. The hard drive is part of an ongoing investigation in which police responded to a report of an altercation between Mr. Golden Mustard and a Miss Ruby Scarlet arguing over the computer. During the course of the response, narcotics were found in Mr. Mustards possession. There are also a pile of USB drives that were seized as well. The lead investigator has submitted a request to the email providers for the local users. You should have the cloud email available for analysis in a week.

Assignment: Perform an analysis of the imaged drive and media. The hard drive image is available on the LISSARD share drive. You will have imaged all of the media, except for the hard drive during scene processing. The goal is a full reconstruction of the crime(s). Find as much evidence as you can. Determine who has committed the crime, why they committed the crime, and the timeline associated with the crime. Build a narrative of the case for the lead investigator. Document where and how you find it. As presented in the scenario, the search terms associated with the warrant are for information on narcotics. If you evolve search terms, justify why you evolved them. Prepare a written report indicating the teams findings and the techniques used. Be careful to maintain a correct trail of evidence, including tagging, checking in and out of evidence, etc. Also, discuss in length the tools that you used.

Bibliography

1. Bardia Safaei, Amir Mahdi Hosseini Monazzah, Milad Barzegar Bafroei, and Alireza Ejlali. Reliability side-effects in internet of things application layer protocols. In *2017 2nd International Conference on System Reliability and Safety (ICSRS)*, pages 207–212. IEEE, 2017.

2. Office of Juvenile Justice U.S. Department of Justice, Office of Justice Programs and Delinquency Prevention. When your child is missing: a family survival guide. `http://www.ojjdp.gov/pubs/childismissing/`.

3. Mark Scanlon. Battling the digital forensic backlog through data deduplication. In *2016 Sixth International Conference on Innovative Computing Technology (INTECH)*, pages 10–14. IEEE, 2016.

4. George Mohay. Technical challenges and directions for digital forensics. In *First International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'05)*, pages 155–161. IEEE, 2005.

5. Yunus Yusoff, Roslan Ismail, and Zainuddin Hassan. Common phases of computer forensics investigation models. *International Journal of Computer Science & Information Technology*, 3(3):17–31, 2011.

6. Rodney McKemmish and Australian Institute of Criminology. *What is forensic computing?* Australian Institute of Criminology, 1999.

7. Emmanouil Vlastos and Ahmed Patel. An open source forensic tool to visualize digital evidence. *Computer Standards & Interfaces*, 29(6):614–625, 2007.

8. Salvatore Amato Catanese and Giacomo Fiumara. A visual tool for forensic analysis of mobile phone traffic. In *Proceedings of the 2nd ACM workshop on Multimedia in forensics, security and intelligence*, pages 71–76. ACM, 2010.

9. James J. Thomas and Kristin A. Cook. A visual analytics agenda. *IEEE Computer Graphics and Applications*, (1):10–13, 2006.

10. Kristin A. Cook and James J. Thomas. Illuminating the path: The research and development agenda for visual analytics. Technical Report No. PNNL-SA-45230, Pacific Northwest National Lab.(PNNL), Richland, WA (United States), 2005.

11. Sushilkumar Chavhan and S.M. Nirkhi. Visualization techniques for digital forensics: A survey. *International Journal of Advanced Computer Research*, 2(4):74, 2012.

12. Melanie Tory and Torsten Moller. Rethinking visualization: A high-level taxonomy. In *IEEE Symposium on Information Visualization*, pages 151–158. IEEE, 2004.

13. Sheldon Teelink and Robert F. Erbacher. Improving the computer forensic analysis process through visualization. *Communications of the ACM*, 49(2):71–75, 2006.

14. Rodney McKemmish. When is digital evidence forensically sound? In *IFIP international conference on digital forensics*, pages 3–15. Springer, 2008.

15. Albert Antwi-Boasiako and Hein Venter. A model for digital evidence admissibility assessment. In *IFIP International Conference on Digital Forensics*, pages 23–38. Springer, 2017.

16. Christopher Tassone, Ben Martini, and Kim Kwang Raymond Choo. Forensic Visualization: Survey and Future Research Directions. In *Contemporary Digital*

Forensic Investigations of Cloud and Mobile Applications, pages 163–184. Elsevier Inc., oct 2016.

17. Brian Carrier. Open source digital forensics tools: The legal argument. Technical report, stake, 2002.

18. John Patzakis and Victor Limongelli. *EnCase Legal Journal*. Guidence Software: Pasadena, California, 2002.

19. Brian Carrier. Defining digital forensic examination and analysis tools. In *Digital Research Workshop II*, 2002.

20. Brian Carrier. Defining digital forensic examination and analysis tools using abstraction layers. *International Journal of digital evidence*, 1(4):1–12, 2003.

21. AccessData. Forensic ToolKit. `https://accessdata.com/products-services/forensic-toolkit-ftk`, 2019.

22. LinuxConfig.org. Learning Linux commands: dd. `https://linuxconfig.org/learning-linux-commands-dd`, 2016.

23. ILook-Forensics. ILook IXImager: A forensic data imaging system. `http://www.ilook-forensics.org/iximager.html`, 2006.

24. SIFT. SIFT Workstation. `https://digital-forensics.sans.org/community/downloads`, 2008-2019.

25. Mohd Sohail. DEFT Linux A Linux Distribution For Computer Forensics. `http://www.linuxandubuntu.com/home/deft-linux-a-linux-distribution-for-computer-forensics`, 2019.

26. Brian Carrier. The Sleuth Kit. `http://www.sleuthkit.org/sleuthkit/`, 2003-2019.

27. Brian Carrier. The Sleuth Kit Framework. https://www.sleuthkit.org/sleuthkit/docs/framework-docs/index.html/, 2011-2013.

28. Brian Carrier. The Blackboard. https://www.sleuthkit.org/sleuthkit/docs/framework-docs/mod_bbpage.html, 2011-2013.

29. Brian Carrier. Autopsy Analysis Features. https://www.sleuthkit.org/autopsy/features.php, 2003-2019.

30. Brian Carrier. Autopsy Timeline Analysis. https://www.sleuthkit.org/autopsy/timeline.php, 2003-2019.

31. Brian Carrier. Autopsy Keyword Searching and Indexing. http://www.sleuthkit.org/autopsy/keyword.php, 2003-2019.

32. Apache. Solr Features. https://lucene.apache.org/solr/features.html, 2019.

33. Daniel J. Schelkoph, Gilbert L. Peterson, and James S. Okolica. Digital forensics event graph reconstruction. In *International Conference on Digital Forensics and Cyber Crime*, pages 185–203. Springer, 2018.

34. James S. Okolica. Temporal event abstraction and reconstruction. Journal Article No. AFIT-ENG-DS-17-D-004, Air Force Institute of Technology Wright-Patterson AFB OH, 2017.

35. Plaso. log2timeline/Plaso. https://github.com/log2timeline/plaso, 2016.

36. Nikolai A. Adderley. Graph-based temporal analysis in digital forensics. Master's thesis, Air Force Institute of Technology Wright-Patterson AFB OH, 2019.

37. GRANDstack. GRANDstack; Build full stack graph applications with ease. https://grandstack.io/.

38. GRANDstack. Getting Started With GRANDstack. `https://grandstack.io/docs/getting-started-neo4j-graphql.html`.

39. Neo4j. Indexes for full-text search. `https://neo4j.com/docs/cypher-manual/current/administration/indexes-for-full-text-search/`.

40. Docker. What is a Container? A standardized unit of software. `https://www.docker.com/resources/what-container`.

41. elastic. The Elastic Stack. `https://www.elastic.co/products/elastic-stack`.

42. Brian Carrier and Eugene H. Spafford. An event-based digital forensic investigation framework. In *Digital Forensic Research Workshop*, pages 11–13, 2004.

Acronyms

AFIT Air Force Institute of Technology. 57

API Application Program Interface. 30

CFIP Computer Forensic Investigative Process. 5

CSCE Computer Science and Computer Engineering. 57

DART Digital Advanced Response Toolkit. 20

DEFT Digital Evidence and Forensics Toolkit Zero. 20

DHS Department of Homeland Security. 9

ECATV Event Correlation and Abstraction Timeline Visualization. vii, viii, xi, 3, 4, 5, 7, 34, 35, 36, 37, 39, 46, 47, 48, 51, 52, 54, 57, 58, 59, 66, 67, 68, 83, 84, 85, 92, 96, 97, 98, 99

EnCase EnCase Analytics. 17

FTK Forensic Toolkit FTK AccessData. 17

GCFIM Generic Computer Forensic Investigation Model. 6, 7, 8

i2 i2 Analysts Notebook by IBM. 17

IEF Internet Evidence Finder IEF Magnet Forensics. 17

Intella Intalla Vound. 17

IT Information Technology. 58

Katana Katana Forensics Lantern. 17

NIST National Institute of Standards and Technology. 20

Nuix Nuix: Visual Analytics. 17

NVAC National Visualization and Analytics Center. 9

Oxygen Oxygen Forensic Suite 2015 Analyst. 17, 20

PGER Property Graph Event Reconstruction. 3, 4, 28, 29, 30, 33, 34, 39

SIFT SANS Investigative Forensic Toolkit. 20

Susteen Susteen Secure View 3. 17

TAIMA Temporal Analysis Integration Management Application. 3, 4, 28, 30, 33, 34, 51, 52

TEAR Temporal Event Abstraction and Reconstruction. 28, 29, 36

TSK The Sleuth Kit. 21, 22, 23, 25, 36, 48

UFED Cellbrite's UFED Link Analysis. 17

UI user interface. 30, 34, 36, 37, 39, 52

XAMN Micro Systemation: XAMN/XRY. 17

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 26-03-2020	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From - To)* May 2018 - Mar 2020

4. TITLE AND SUBTITLE Digital Forensics Tools Integration	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Kim, Alexander D.H., Capt	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way Wright-Patterson AFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT-ENG-MS-20-M-031
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Intentionally Left Blank	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution Statement A. Approved for Public Release; Distribution Unlimited.

13. SUPPLEMENTARY NOTES
This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT
As technology has become pervasive in our lives we record our daily activities both intentionally and unintentionally. Because of this, the amount of potential evidence found on digital media is staggering. Investigators have had to adapt and change their methods of conducting investigations to address the data volume. Digital forensics examiners current process consists of performing string searches to identify potential evidentiary items. Items of interest must then go through association, target comparison, and event reconstruction processes. These are manual and time consuming tasks for an examiner. This thesis presents a user interface that combines both the string searching capabilities that begin...

15. SUBJECT TERMS
digital forensics, autopsy, timeline visualization, forensic examination

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. Gilbert L. Peterson, AFIT/ENG
U	U	U	UU	109	**19b. TELEPHONE NUMBER** *(Include area code)*