5-2023

A Framework for Identifying Malware Threat Distribution on the Dark Web

Shelby Caldwell

A FRAMEWORK FOR IDENTIFYING MALWARE THREAT DISTRIBUTION ON THE DARK WEB

A Thesis

Submitted to the Graduate Faculty of the
University of South Alabama
in partial fulfillment of the
requirements for the degree of

Master of Science

in

Computer and Information Sciences

by
Shelby Caldwell
B.S., University of South Alabama, 2021
May 2023

ACKNOWLEDGEMENTS

In December of 2020, I was inducted into the CyberCorps: Scholarship for Service (SFS) program at the University of South Alabama. This research was supported in part by the National Science Foundation under grant DGE-1564518. I want to thank my mentor Dr. Todd McDonald for his continued support throughout the research process. I also want to thank my parents and my family for all their encouragement and support over the years.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

LIST OF ABBREVIATIONS

ARPANET	Advanced Research Projects Agency Network
CIA	Central Intelligence Agency
CISA	Cybersecurity and Information Security Agency
CSV	Comma Separated Value
DNS	Domain Name System
DoD	Department of Defense
FBI	Federal Bureau of Investigation
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ML	Machine Learning
NSA	National Security Agency
OSINT	Open-Source Intelligence
PGP	Pretty Good Privacy
RAAS	Ransomware as a Service
RAT	Random Access Trojan
SOCKS	Socket Secure
SVM	Support Vector Machine
TAILS	The Amnesic Incognito Live System

TF-IDF	Text Frequency-Inverse Document Frequency
TOR	The Onion Router
URL	Universal Resource Locator
USB	Universal Serial Bus
USD	United States Dollar
VPN	Virtual Private Network

ABSTRACT

Caldwell, Shelby, M.S., University of South Alabama, May 2023. A Framework for Identifying Malware Threat Distribution on the Dark Web. Chair of Committee: Dr. J. Todd McDonald, Ph.D.

The Dark Web is an ever-growing phenomenon that has not been deeply explored. It is no secret that in recent years, malware has become a powerful threat to technology users. The Dark Web is known for supporting anonymity and secure connections for private interactions. Over the years, it has become a rich environment for displaying trends, details, and indicators of emerging malware threats. Through the application of data science and open-source intelligence techniques, trends in malware distribution can be studied. In this research, we create a framework for helping identify malware threat distribution patterns. We examine this type of Dark Web activity by utilizing an automated and manual approach for collecting data on malware exchanges. Furthermore, a comparative analysis is conducted to determine which approach is more effective and efficient. Our framework for identifying current or future malware threats that are distributed on the Dark Web is refined by examining the weaknesses and strengths of each gathering approach.

CHAPTER I

INTRODUCTION

The World Wide Web (WWW) is comprised of three main categories: surface web, deep web, and dark web [1]. Regular technology users mostly visit the surface web and deep web [1]. Surface web refers to the category of the Internet that is easily accessible by the click of a button [2]. This includes popular websites such as Google or YouTube. The deep web signifies information that is not widely accessible to the public [2]. This information is often hidden behind firewalls, passwords, or other security protocols and is not always discoverable via search engines. Examples of data found on the deep web are legal documents such as medical records or government reports, however, it is not limited to just that [1]. Moving forward, we reach a barrier known as the Dark Web. Information provided on the Dark Web is often illegal and anonymous. The Dark Web cannot simply be accessed by any internet browser [1]. Special software is required to access the Dark Web hence it being a protected platform. In Bradley's (2019) dissertation, he stated that the Dark Web involves a series of networks inside of the Internet that often has rigorous technical prerequisites before a connection is made [3]. In this paper, we will be focusing on The Onion Router (Tor) web browser as a tool to navigate the Dark Web. Tor is open-source software that can be used to access. onion domains within the Dark Web [4]. Tor will be utilized in this research to access the Dark

Web and gain firsthand insight into the exchanges that occur. The lack of knowledge surrounding the Dark Web presents an opportunity to discover the specific mechanisms cybercriminals use to distribute information and resources.

1.1 Motivation

Several top-tier United States government agencies including the Cybersecurity Infrastructure and Security Agency (CISA) have identified malware as one of the top cyber threats facing the nation today [5]. When a new malware attack is identified, government agencies will often send an alert or warning to the public to raise awareness. For example, when a business reports a malware attack, the Federal Bureau of Investigation (FBI) has a list of information they collect and disseminate to better understand the situation. This includes items such as the ransom note, amount of the demand, currency type, and a copy of the incident report [6]. In addition, the FBI provides a list of recommended mitigations and resources to help businesses defend against these attacks and keep systems online [6]. By investigating the spread of these malware attacks on the Dark Web, we are essentially aiding law enforcement in the pursuit of limiting and preventing these attacks from occurring.

We are studying to see what the limits of one approach are versus the other in terms of tracking down sources for malware distribution. Furthermore, we will examine and map out exactly how malware is purchased. Data will be collected via web scraping technology and pre-processing techniques to create hypertext markup language (HTML) formatted data. Furthermore, an analysis will be conducted to determine the weaknesses and strengths of both approaches. We want to determine if alias names are being used,

who is selling the malware, the methods of communication used, types of currency, and what steps are needed to obtain the malware. These specific details will give researchers, businesses, law enforcement, and other interested parties insight into the dispersal of malware on the Dark Web. With this information, they can determine what steps should be taken to prevent this software from spreading so quickly. Firsthand knowledge of the popularity and trends of malware spreading on the Dark Web will give an advantage to businesses and law enforcement who are trying to prevent these attacks. If businesses study what they are up against, they will be better prepared for an attack and could potentially create an offensive strategy

1.2 Research Goals

There are two research goals that we are striving to accomplish with this study. The first goal is to develop a framework for identifying current or future malware threats that are distributed on the Dark Web. This study will examine specific advertisements, announcements, and other means of communication on the Dark Web regarding malware distribution. We aim to develop and evaluate a manual process for documenting the distribution sources of prominent malware threats. Subsequently, we will develop and evaluate an automated machine learning approach for documenting the distribution sources of prominent malware threats. The goal is to have a successful implementation of both a manual and automated approach to examining malware distribution on the Dark Web. The second research goal is to compare the effectiveness of manual versus automated methods. By accomplishing these goals, we will have successfully studied

trends in malware distribution and created a solid foundation for future research in this area of study.

1.3 Research Questions

A combination of techniques will be used to examine a variety of onion domains from a series of collection methods. Emerging malware threats such as rootkits, ransomware, and spyware will be examined to see how they are advertised, distributed, and purchased on the Dark Web. We will develop and evaluate a manual process for documenting the distribution sources of prominent malware threats. Additionally, we will utilize an automated approach with machine learning techniques for documenting the distribution sources of prominent malware threats. Ultimately, there will be a comparison of the effectiveness and functionality of manual versus automated methods. The first research question we aim to address is: how is malware advertised and purchased on the Dark Web? The next question we will examine is: what are the strengths and weaknesses of an automated versus manual approach for identifying malware distribution trends? Finally, we want to know: can we effectively use data science techniques to classify or link parts of the distribution process?

CHAPTER II

BACKGROUND AND RELATED WORK

The Dark Web is notoriously known for illicit affairs including illegal activity, cybercrime, and keeping information both anonymous and hidden [7]. The Dark Web was created along with the Internet in the 1960s through a project by the United States Department of Defense (DoD) titled the Advanced Research Projects Agency Network (Arpanet) [7]. It was originally designed to link the DoD's computer network systems but as time evolves this platform has been used for entirely different motives. Takaaki and Atsuo (2019) state that a key difference between the Dark Web and the surface web is the orientation of domains and URLs [8]. On the surface web, URLs have a unique structure and usually contain a string of words or numbers. Surface web links are easily searchable and accessible through search engines and advertisements [7]. There are both registered and unregistered domains on the Dark Web. These domains are much longer on average and contain a random set of words or numbers. These are not easily accessible and due to the nature of the content, the source code behind the domains moves frequently. It is difficult for search engines on the Dark Web to keep up with these changes. On a Dark Web link, onion refers to the special-case domain used which is comparable to .com and .org that are used on the surface web. While the surface web uses several different top-level domains, Dark Web links will always end in an onion.

2.1 Dark Web and Dark Net

Although the terms are often used interchangeably, they are two different concepts. The Dark Net is the "dark" version of the Internet where illegal activity often occurs. The Dark Web is a collection of websites on the Dark Net. The following illustration in Figure 1 depicts how the Dark Net encompasses the Dark Web. This setup is comparable to the Internet and the World Wide Web (WWW). The World Wide Web is a collection of sites on the Internet [1].

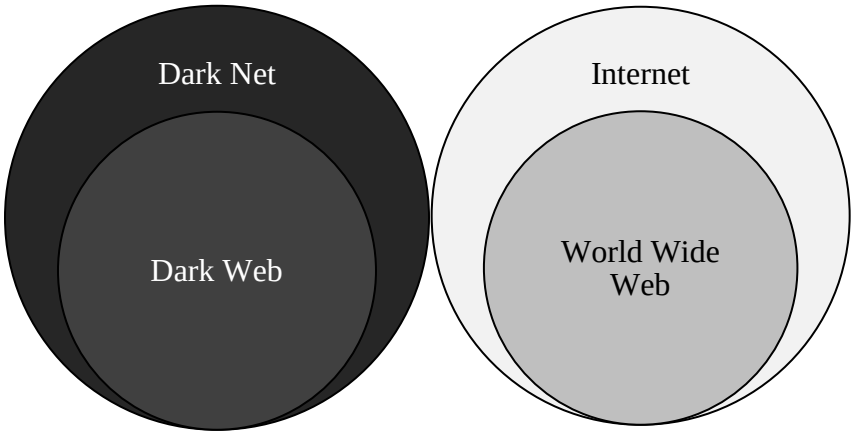

Figure 1. Dark net is often confused with dark web. Dark net encompasses the dark web similar to how the internet encompasses the world wide web.

2.2 OSINT on the Dark Web

To access the Dark Web, we will utilize The Amnesic Incognito Live System (Tails). Tails is a free and open-source operating system that runs Debian and encrypts files on the system [9]. Tails is quick and easy to install and for this study, it will be run using version 4.18 on a USB drive.

Tor is a web browser that is used primarily to access the Dark Web. From Chertoff, we learn that Tor was developed by the United States federal government [7]. It was created for military personnel in undisclosed locations to allow anonymous access to the Internet [7]. Tor is a free and open-source software that provides encrypted and anonymous communication. Open-source software (OSS) entails that the code or software is available for anyone to use, and they are allowed to contribute or make changes to fit their needs [10]. By adding several layers of encryption, Tor provides an anonymous experience while implementing privacy and security measures [4]. Tor guarantees end-to-end encryption between the user and an onion server [7]. Tor has a very straightforward name: an onion can be peeled to reveal many layers, and the Tor network has several layers of encryption in place to protect users [4]. Tor is used so that the traffic will be routed via the Tor network for guaranteed anonymity [11]. Any user trying to visit an onion domain without using Tor will not be successful due to the proxy servers set in place before the site is reached [11].

2.3 Machine Learning Techniques

Machine learning (ML) is a quick method for analyzing and aggregating data. It is defined as "emulating human intelligence by learning from the surrounding environment [12]." We chose to implement machine learning techniques to analyze our data for several reasons. The datasets we gather are text-based and require categorization. Machine Learning is an approach that will allow for classification and categorization [13]. Machine learning provides a way to look at patterns between HTML data and determine statistical information regarding the data. Machine Learning algorithms are

also a fast way to analyze data and can be used to display data in a variety of different ways [12].

2.4 Malware

Malware is considered an umbrella term that encompasses many different attack techniques. The term malware can refer to several different types of software including rootkits, adware, spyware, ransomware, viruses, worms, and target-specific code. Simply put, malware is malicious computer software. Broadhurst et al. describe malware as "computer code that is capable of compromising computer systems [13]." This can be a small-scale or large-scale attack depending on how it is executed. Malware can cause mass chaos and panic due to a shortage of resources such as gas or oil. Malware distribution often occurs through markets on the Dark Web. Examples of Dark Web markets include hacking services, weapons, technology, jewelry, cryptocurrency, fake IDs, credit cards, and much more. Users typically turn towards the Dark Web to fulfill any services they cannot access on the surface web. Hackers construct and design specific malware instances such as attacking a school or hospital and sell it for profit on the Dark Web. There is a plethora of known malware-oriented organizations on the Dark Web. One of the most notorious is Revil which is a Ransomware-as-a-service (RAAS) enterprise [14]. By looking at Figure 2, we see that Revil makes up 13.1% of ransomware threats detected over five months [14]. This is a very high percentage compared to other malware organizations on the Dark Web.

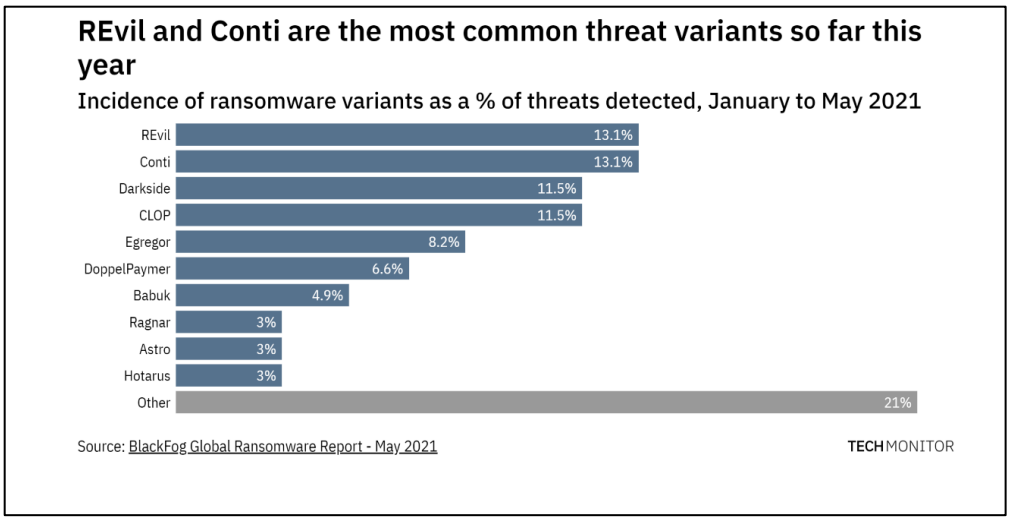

Figure 2. A list of several different ransomware groups that were prevalent from January to May 2021. Adapted from a BlackFog global ransomware report [14].

2.5 Threat Analysis

Other research has been completed in this area of study however, it is slightly different in nature. Cerys Bradley conducted a dissertation solely on law enforcement intervention regarding the Dark Net market ecosystem [3]. His contribution was to analyze Dark Net markets and how law enforcement can legally intervene. He also touched on the psychology and social aspect behind Dark Net markets. In contrast, this research focuses solely on the characteristics of Dark Net markets, specifically those that sell malware. The framework provides specific mechanisms to collect and analyze data on the Dark Web. In another published research paper, a group of students at Thompson Rivers University carried out a temporal analysis of radical Dark Web forum users [15]. Their study is similar in that they analyzed forum posts on the Dark Web, however, their main goal was to target radical and extremist groups and discover trends in their behavior

through data mining. This study examines cybercriminals and hackers selling malware on the Dark Web, and we are interested in how these transactions are carried out and what steps are needed to purchase malware. The journey from point A to point B is more crucial than the motive behind the screen.

CHAPTER III

METHODOLOGY

In this study, we explore the Dark Web and examine onion domain addresses via an automated and manual approach. The problem being addressed is an influx of malware and other malicious code being distributed on the Dark Web. To address this problem, we provide a framework for pinpointing current or future malware threats that are being spread on the Dark Web. We are observing malware transactions that occur on the Dark Web and comparing two approaches to gathering data.

The methodology we conducted in this research is based on work completed by Thorat et al., 2020 [16]. Their research is focused on classifying and visualizing all illegal activities on the Dark Web [16]. They trained classifiers using Indian laws and regulations related to each illegal activity they were searching for. This included a wide range of categories such as illegal drugs, gambling, weapons, child pornography, and counterfeit credit cards. Our research is based entirely on malware interactions rather than just illegal activity. However, we will be conducting text pre-processing, text frequency-inverse document frequency and testing machine learning algorithms during our automated approach. Figure 3 shows an overview of the research methodology including browsing sources, creating the keyword list, executing the approaches, reviewing the results, and conducting a comparative analysis.

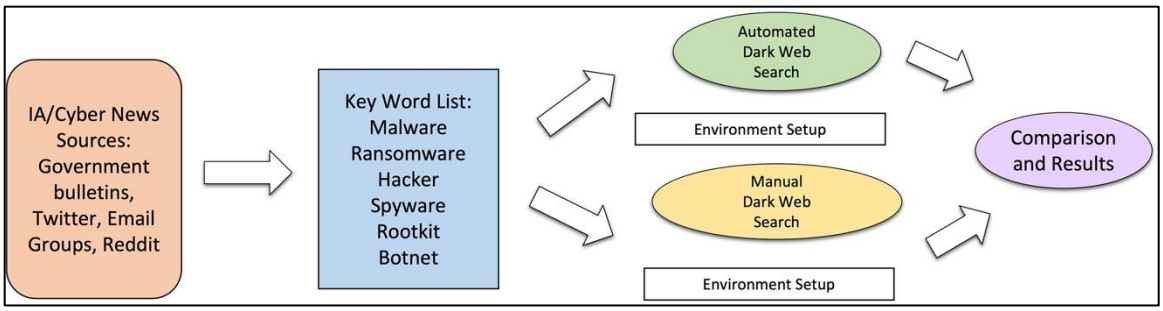

Figure 3. A visualization of the methodology used for this research.

The figure depicts the linear process for our proposed research methodology. The first step in our research is monitoring information assurance and cyber news sources. This includes resources such as Government bulletins (FBI, NSA, CIA, CISA, etc.), Twitter threads, email newsletters, and Reddit forums regarding malware. Subsequently, a keyword list is created in response to those sources. From there the host environment is configured to deploy the automated and manual approach. Ultimately, there will be a comparison of the two approaches and an overview of the results we expect to see.

3.1 Information Assurance and Cyber News Sources

We used open-source news, blogs, and social media channels to identify a list of current malware threat trends. Sources we heavily monitored include darknetdaily.com, Reddit, Dread (Dark Web version of Reddit), and popular social media personalities such as Vx-underground on Twitter. Darknetdaily.com is a website that regularly provides active onion domains from the Dark Web. Reddit is a forum-based website on the Internet that provides up-to-date and even real-time information regarding malware transactions on the Dark Web. While some users speak secretively or in code, it is still a great source to gather information and starting points for navigating the Dark Web. Dread

provides similar insight; however, it is located directly on the Dark Web rather than the surface web. Social media personalities are a useful source because they are often directly involved in malware distribution on the Dark Web. Vx-underground highlights popular malware groups on the Dark Web and updates on recent malware attacks occurring worldwide.

3.2 Search Engines and Key Word List

After monitoring several cyber news sources, we decided on four search engines to scrape and compare results. The four search engines are Ahmia.fi, OnionLand Search, Haystak, and DevilSearch. The intent is to have a wide variety of results, so each search engine is slightly different in nature. For instance, DevilSearch is a relatively new search engine on the Dark Web that has no censorship. Their website states that it has been live since 2020. Ahmia.fi, however, is open-source and has been in existence since 2014. It is available on the surface web and the Dark Web. It is censored and the site states that they strive to filter child pornography and other abusive services [17]. We originally aimed to search for malware organizations as well as malware keywords, but ultimately it is difficult to trace these organizations by their names and we decided that the information found was not sufficient. The keywords chosen to search for are malware, ransomware, hacker, spyware, rootkit, and botnet.

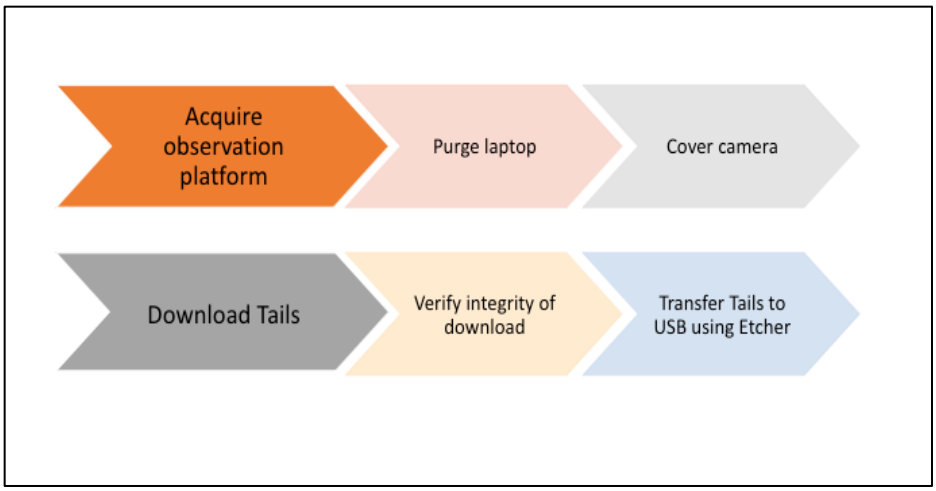

Figure 4. Process of environment setup for exploring the dark web.

3.3 Environment Setup

For this experiment, we set up an observation platform and a host laptop. The purpose of the observation platform is to explore the Dark Web and to gather live onion domains. The host laptop is more suited for scripting purposes and is used for Python coding and bash scripting. For setup protocols and establish a deeper understanding of the Dark Web, we utilized the Dark Net Market Buyer's Bible as a guide for setting up the observation platform. The Dark Net Bible is a document written specifically to help new users navigate the Dark Web. This portion of the environment setup applies to both the automated and manual approaches. To begin, a host laptop was set up to access the Dark Web safely and securely. For this research, we are using a Lenovo Ideapad 310 Touch. This laptop is completely devoid of any personally identifiable information (PII) and is used solely for research. For further protection, the web camera and microphone port are always covered. A ProtonMail email account was set up to transfer information from the observation platform to the host laptop. ProtonMail provides end-to-end encrypted email

services [18]. In addition to encrypted email, we use ProtonVPN any time emails are exchanged. Each time the observation laptop boots, a temporary account is used, and no data is saved after it is shut down. Instead of booting directly from the computer, a USB drive with Tails 4.18 downloaded is used to boot the operating system. The Tails image software was downloaded directly from the Tails website. The Dark Net Bible recommends using Tails via USB for optimal protection; however, it can be used on any system with a VPN for extra protection [19]. In Figures 5-9, there are images that display the process of downloading Tails, verifying the integrity of the image, and moving that image onto a USB-drive.

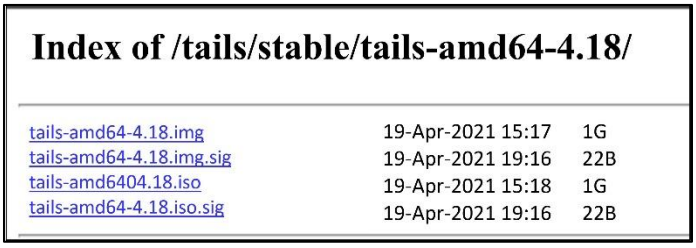

Figure 5. A screenshot of the image used for downloading tails.

Using GTKHash, a SHA-256 hash verification was completed to ensure integrity of the file. Tails requires users to authenticate their signing key via the OpenPGP Web of Trust, so Kleopatra is used to store and view the PGP certificate for Tails. This allowed Tails to be verified and downloaded. Etcher is an open-source utility that is used to transport Tails onto the USB-drive. Tor is a built-in software on the Tails operating system. Once Tails is booted and a Wi-Fi connection is established, Tor can be executed.

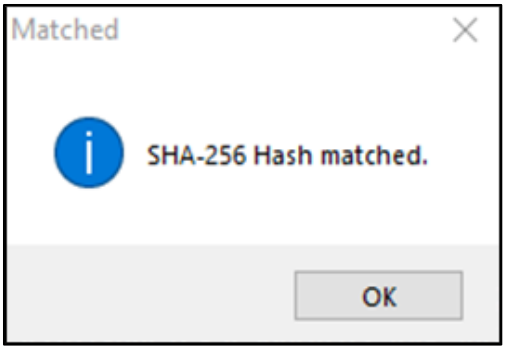

Figure 6. Verifying the SHA-256 hash of tails to ensure the integrity of the download

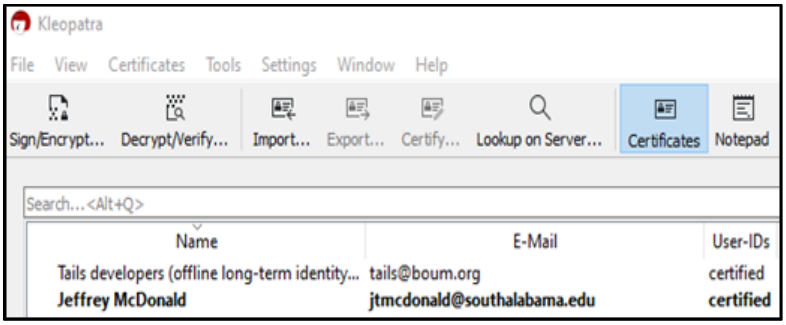

Figure 7. Saving the PGP certificate for tails to keep track of the public key for encrypted messaging.

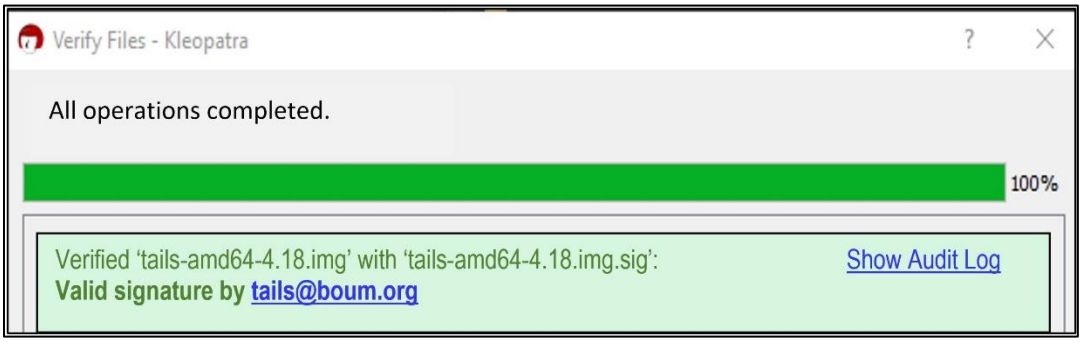

Figure 8. Notification that tails has been successfully downloaded.

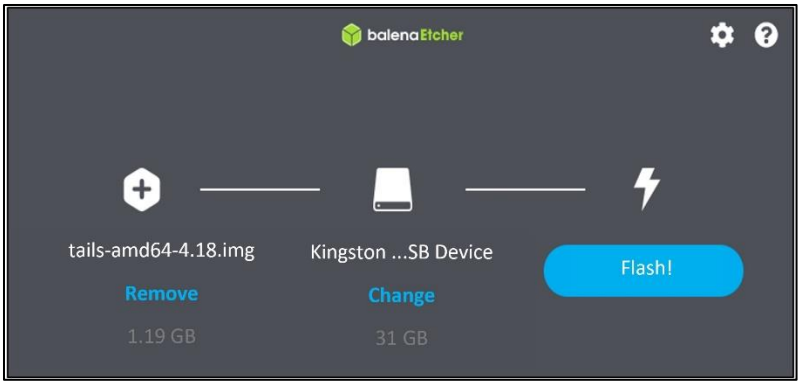

Figure 9. Process of transferring tails to a USB via etcher.

3.3.2 Automated Environment Setup

The automated approach is conducted on the host laptop using a virtual machine. The host laptop is a Dell Inspiron 7506 2n1 using Windows 11. The virtualization platform is VirtualBox 6.1.26 and the operating system is Linux Ubuntu 64-bit. For further protection, ProtonVPN is used for all research investigations.

CHAPTER IV

RESULTS

4.1 Automated Approach

The automated approach encompasses four major steps for completion. The four steps are web scraping, pre-processing, word vectorization, and machine learning. The operational flow for the automated approach is described in Figure 10.

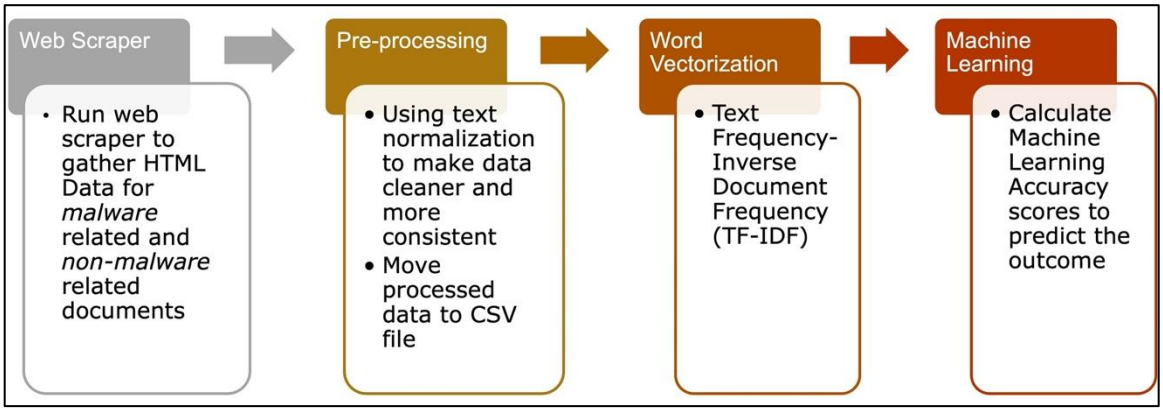

Figure 10. The process for the automated approach used to gather data from the dark web.

4.1.1 Web Scraper

This method is considered automated because it is done primarily with the use of computers and automation. For the automated approach, a Dark Web scraper was adapted

to extract onion links using Tor and Python. The source code was created by Stephan Fleig and published on LinkedIn and Github in 2020 [20]. The source code for the Dark Web scraper is provided in Appendix B. The code imports several necessary Python modules including socks, socket, requests, and BeautifulSoup. The socks and socket modules allow traffic through SOCKS and HTTP Proxy servers [21]. This includes client and server programs. The requests module is simply a Python HTTP library [22]. BeautifulSoup is a Python library that is used to scrape all information from a particular web page [23]. The web scraping script configures socks to use Tor and defines the localhost as port 9050. Using the defined modules, the script uses Tor for DNS resolution of the onion websites. Afterward, the script executes the scraping tools to extract content from the specified site. An HTML parser is used to clean up the data in a readable format. We executed this script using different onion sites as a target to build the corpus. Figure 11 shows the process for executing the automated web scraper using Python.

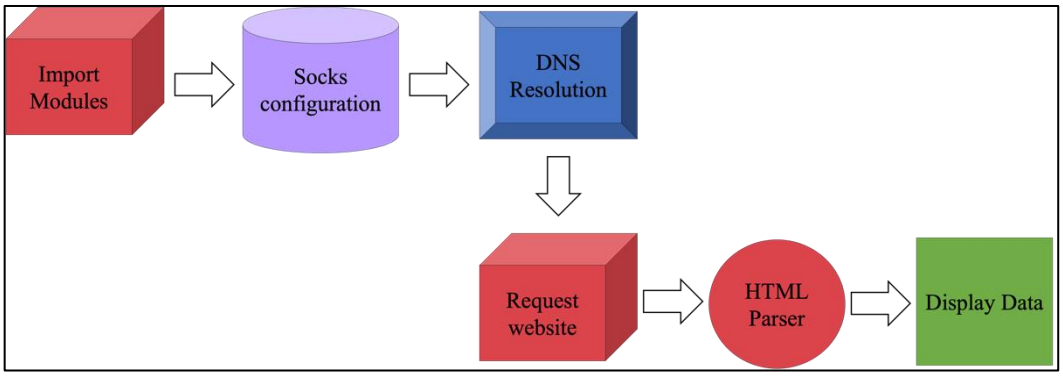

Figure 11. The process for the automated web scraper used to grab information from dark web onion sites.

We began building the corpus by searching for a specific keyword on a search engine. For data variation, we scraped malware-related and non-malware-related onion domains. The non-malware-related domains included sites about categories completely disassociated from computing such as food and drinks. Those sites were gathered from the search engines mentioned previously and were completely devoid of any computing related topics. We collected the HTML text from the onion sites and extracted data containing the specified keyword for malware and any currency related terms.

4.1.2 Pre-Processing the Data

Pre-processing the data is important to clean up the data and remove all HTML formatting. The process began with removing any empty rows and modifying the text to lowercase so that case sensitivity does not affect the results of the data. The next action was to perform word tokenization by breaking the group of text into words. Next, we removed stop words which are just common words in the English language that are not useful for the experiment [24]. This was completed using the Python module NLTK. Then we removed non-alpha text meaning any character that is not a letter or number [25]. Examples of non-alpha text include commas, brackets, spaces, asterisks, and so on [25]. Word lemmatization is the exercise of removing inflectional endings from a word [26]. The next step is condensing the data. The purpose of this task is to limit the amount of information we are looking at. HTML data from a website will contain a lot of unnecessary information that can be hidden underneath the website. We want to consolidate the data into information related to malware and the selling of malware-related products. For this reason, we used a Python script to search for lines containing the specified keyword and anything related to currency such as the abbreviation or full

currency name. Afterward, we formatted the data in a comma-separated value (CSV) file. Then, we constructed the text analysis script. The text analysis script was adapted from Gunjt Bedi on the technology website Medium [26]. A full-text copy of the script is provided in Appendix C. The script requires several modules imported from different Python libraries. After implementing the modules, we specified the corpus which is the CSV file created from earlier. The corpus contains 224 documents total including 137 malware-related and 87 non-malware related.

4.1.3 TF-IDF

Term frequency-inverse document frequency (TF-IDF) will be computed with the data retrieved from the Dark Web. Term frequency refers to how often a particular word appears within a document [27]. Inverse document frequency is essentially the opposite. Inverse document frequency diminishes the value of words that are frequent across documents [27]. Together, the term frequency-inverse document frequency emphasizes the more significant words within a given document [27]. TF-IDF can be calculated using the following formula:

$$Tfidf(t,d,D) = tf(t,d) * idf(t,D)$$

In this equation, tf(t,d) refers to the raw count of a term within a document otherwise called term frequency [27]. Term frequency is multiplied by idf(t,D) which is inverse document frequency. Inverse document frequency is the total number of documents in the corpus divided by the number of documents where the keyword appears [27]. The closer the TF-IDF score is to 1, the more relevant that term is. Table 1 and Figure 12 each display the results of our term frequency inverse-document frequency calculation.

Table 1. TF-IDF scores in table format from the automated approach.

Botnet	Hacker	Malware	Ransomware	Rootkit	Spyware
0.20609	0.269502	0.317061	0.095118	0.015853	0.063412

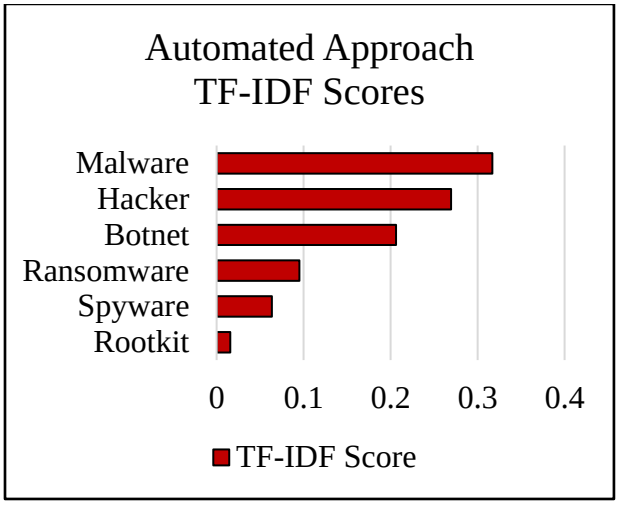

Figure 12. TF-IDF scores displayed in bar graph format from the automated approach.

The results from the TF-IDF calculation show that malware had the highest score, and rootkit had the lowest. Botnet and hacker were close in relevance for the documents that were analyzed.

4.1.4 Machine Learning Techniques

The data is split into two random train and test subsets allowing for 70% training data and 30% test data. Naïve Bayes Accuracy is the first machine learning algorithm tested on the data. It is based on Bayes' Rule which is a principled way of

calculating conditional probability [28]. It is widely used because of its simplicity in the training and classifying stage [28]. A simple form of calculating Bayes' Rule is:

$$P(y|x) = p(x, y)/ p(x)$$

where P(y | x) is the posterior probability; the probability that we are interested in calculating [25]. P(y) is the prior; or probability of the event [28]. The Naïve Bayes accuracy score depicts how well the dataset can be classified using the keyword classifiers [28]. The Naïve Bayes accuracy score we computed for the corpus is 85.29.

We learn from Hodo et al., that "Support Vector Machines (SVM) is a machine learning algorithm that learns to classify data using points labeled training examples falling into one or two classes." It is typically used for pattern recognition and will be compared against Naïve Bayes [29]. Decisions are based upon support vectors hence the name support vector machine [29]. Accuracy can be computed by comparing actual test set values against predicted values. The SVM accuracy score we computed for the corpus is 83.82. Figure 13 shows a comparison between the Naïve Bayes accuracy score and SVM accuracy score.

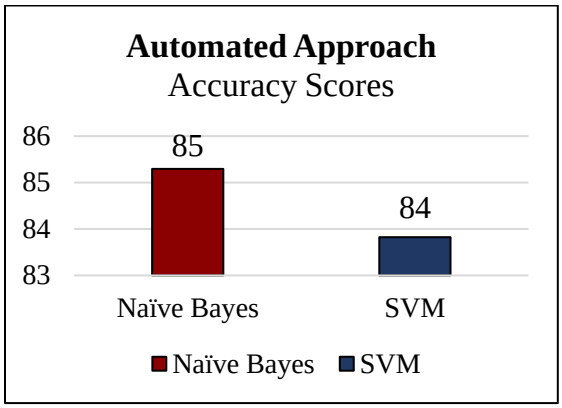

Figure 13. Comparison of Naïve Bayes and SVM accuracy scores for the automated approach.

4.2 Manual Approach

For the manual approach, we began with the six keywords we gathered from examining various sources and trends. The six keywords we are searching for are malware, ransomware, hacker, spyware, rootkit, and botnet. Each keyword is searched for using a search engine and from there we recorded the onion site link, search engine name, site name, number of clicks, result (success/failure), price of malware, and any notes about the site that are relevant. For this experiment, success refers to the presence of a malware-related listing. A failure determines that there is no presence of any malware-related listings. The protocol for data collection was to gather as much information as possible without interacting with the seller.

4.2.1 Malware

Starting with malware, we began searching for malware using the search engine Haystak. We made six attempts to locate malware using Haystak which all resulted in an "Invalid Onionsite Address" meaning the site cannot load for an unidentified reason. After those six failures, we switched to the search engine OnionLand Search. OnionLand search provided a working link on the first attempt and after three clicks we were able to identify a listing for malware. The site name was "Premium Malware", and they were offering malware for $100 USD. This price, however, would be converted to the Monero and Bitcoin cryptocurrency equivalent. Notes about the site include that it has a simple design utilizing black and green. The seller provided an email address for contact and described the product as a "quick solution for ending a system." He also stated "it writes over the whole drive, zeros, up, until the system completely fails. This will render a system useless."

4.2.2 Ransomware

The next keyword searched for was ransomware. The search began with the last search engine which was OnionLand Search. The first three sites we selected resulted in failures. The first failure was a site full of generic ads for items such as Bitcoins, credit card information, and other similar categories. The next failure included an article related to ransomware written in an unidentified foreign language. The last failure resulted in another "Onionsite Not Found" error. On the fourth attempt to find ransomware, we discovered a site titled "DarkHidden marketplace – guaranteed secure escrow system." It is interesting to note that the site developer misspelled the word "guaranteed" when designing the site. Often a misspelled word on a website is a red flag that the site may not be legitimate. The site was advertising custom-made ransomware for a discounted price of $350. The regular listing price was $450, and it is unclear why the price was discounted. The user provided an accurate description of ransomware and there were 26 reviews by verified owners of the product. It is also unclear whether these reviews were from actual people who have received a product from the seller. The reviews were very generic in nature and there is a possibility that the site owner posted these reviews as a ploy to convince buyers that this is a good product. Products on the site that are listed as related to ransomware are hacking services for operating systems, Facebook, Instagram, and networks.

4.2.3 Hacker

Searching for the keyword hacker had one failure due to the onion site being invalid. The next attempt resulted in success. The site name was "Hacker Forces" and after navigating four clicks we reached the section for user input. The site claimed to

provide ethical hacking services meaning that they only want to provide legal services. Users must provide a first and last name, email address, subject, and message to contact the hacker for a specific hacking service. The services this hacker provides include bitcoin recovery, computer hacking, compromising databases, protection against cyber stalkers, phone hacking, grade changing, social media hacking, special hacking, and network hacking. To determine a price point for any of these hacking services, one must message the seller and wait for further instructions.

4.2.4 Spyware

The quest for finding spyware on the Dark Web began with four failures. The second failure was a forum discussing different types of repositories. The third failure was a site offering web hosting, virtual servers, dedicated servers, blocking VPNs, and domain registration. The fifth attempt to acquire spyware was successful on a site titled "Spyware – Mr. Snow Fall's Hacking Products and Services." After 3 clicks, we were able to see listings for spyware ranging from $20-150 USD. The seller stated "here is a list of spyware that I currently have for sale" although, none of the spyware listed is explicitly labeled as spyware. They are labeled as RATs (Random Access Trojans) and keyloggers. Appendix A provides the variety of listings shown on this Dark Web site.

4.2.5 Rootkit

The first two attempts to acquire rootkits were unsuccessful. The first failure was a site containing anti-malware and anti-rootkit software being sold on the Dark Web. The third attempt at finding rootkits was a success and took three clicks to reveal. The name of the site was Venom RAT, and the seller was providing a subscription service for

rootkits. The base price for the RATs was $350 per month. In Table 2, there is a breakdown of each subscription service.

Table 2. Rootkits for sale on the dark web.

1 Month	3 Months
• HVNC Hidden Desktop • Icon Hidden Browser Chrome/Firefox • Icon Easy Hidden Control • Icon Stealer: Passwords/ History/Autofill/bookmarks/ cookies • Icon Hidden Log-in via social media • Icon Friendly Support • Icon Support WebGL • Icon Auto Clone profile	• Icon System Information • Icon TCP Connection • Icon UAC Exploit • Icon Disable WD • Icon Execution Policy Editor • Icon Password Recovery • Icon Remote Fun • Icon Encrypted connection

4.2.6 Botnet

The only piece of malware that took one search attempt was botnets. After two clicks, a correct listing was found. The site name was Anon Market, and the botnet was $7 USD. The seller described the package as the ultimate guide to using and operating a personal botnet. It includes everything you need to know for owning and operating a botnet. The package details how to spread your botnet, the different types of botnets, and how to monetize them for maximum profit.

4.3 Comparative Analysis

The final step of the methodology is a comparative analysis of the manual and automated approaches. It is important to assess the weaknesses and strengths of both the automated and manual methods. By observing these metrics, we can determine ways to improve this process moving forward. During the manual approach, there were 6 successes and 16 failures. Figure 14 displays the percentage of failures by keyword and

shows that the keyword ‘malware’ had the highest percentage of failures. This could be attributed to the fact that during the manual approach when looking for malware we began by using the search engine Haystak but ultimately decided to switch to OnionLand Search. The keyword with the least number of failures was botnet which had zero failures.

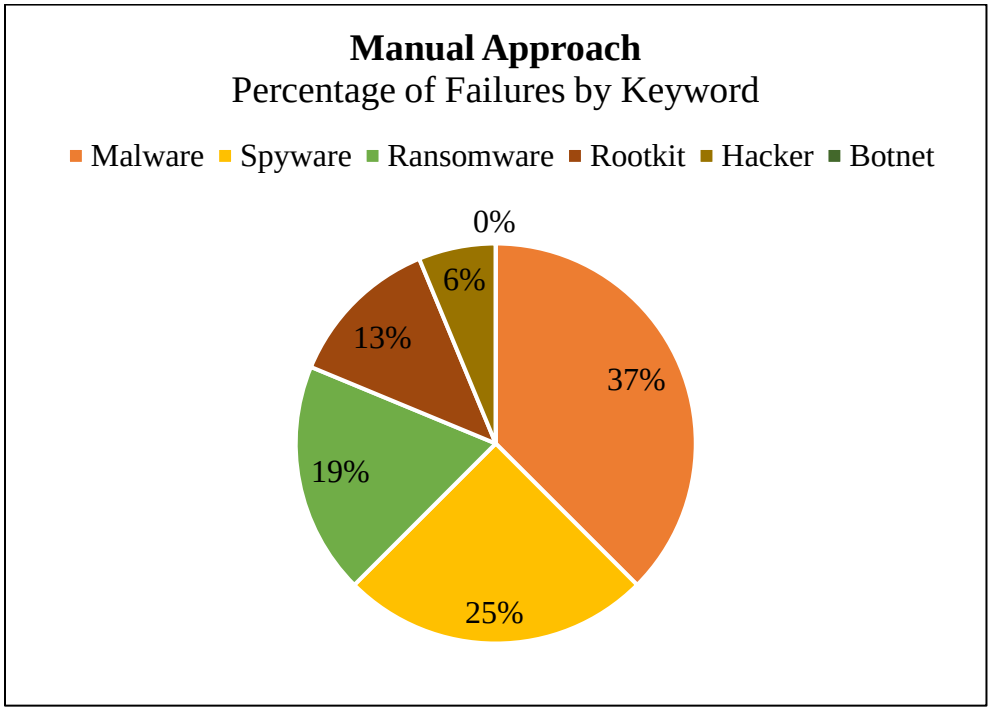

Figure 14. Percentage of failures by keyword for the manual approach.

The manual approach revealed a set of price points for malware threat distribution on the Dark Web. Rootkits and ransomware were the most expensive pieces of malware we found during our investigation priced at $350 USD. Botnets were the least expensive at only $7 USD. This range of prices proves that depending on the seller, a piece of malware can be very expensive or relatively inexpensive. Figure 15 shows a comparison of the malware prices found on the Dark Web using the manual approach.

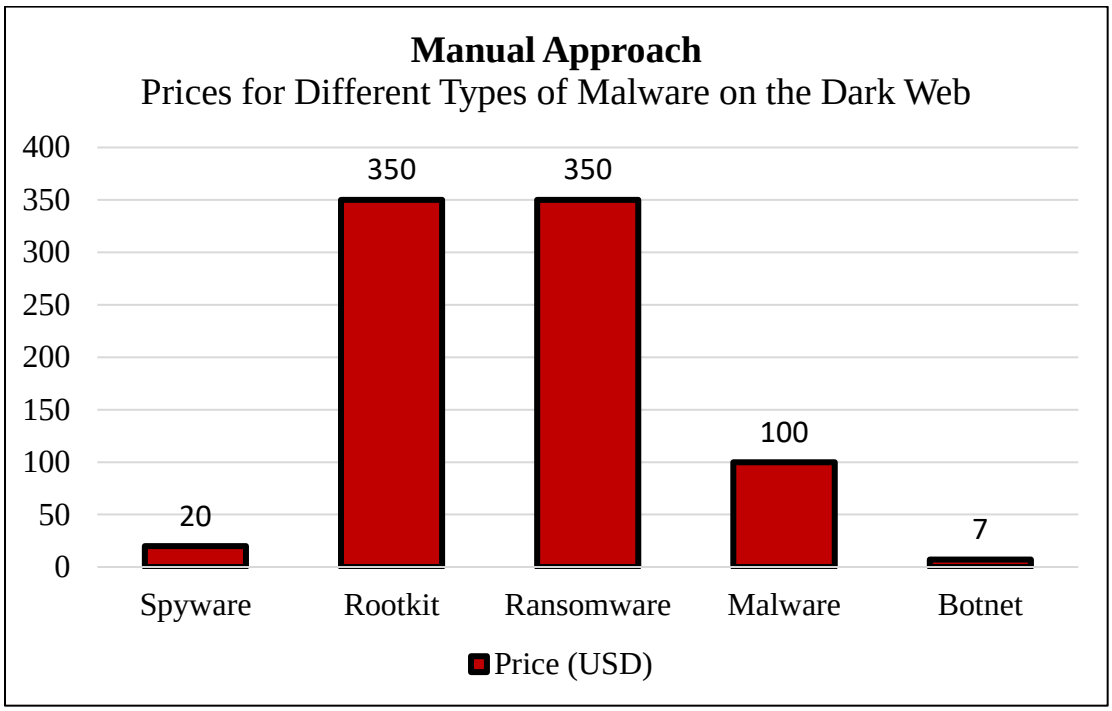

Figure 15. Price comparison for different malware types on the dark web.

Ultimately, the manual approach was successful at identifying malware threat distribution on the Dark Web. However, this approach had several strengths and weaknesses. The manual approach was technically faster than the automated approach due to there being fewer steps involved. The manual approach provides the ability to examine the graphical user interface (GUI) and draw conclusions about the products based on the design and layout of the website. The manual approach also has the option to interact with sites or sellers. We recorded the number of clicks involved to reach a particular type of malware because it is rare that a listing will be available after just one click. Navigating the website and documenting the steps required to access the malware can be crucial to understanding the protocols for buying malware. In contrast, there are

also several weaknesses to the automated approach. Depending on the type of site, this process can be very time-consuming. Some sites take longer to load, and it is crucial to document as much information as possible in case the onion site is shut down indefinitely. It is also easy to glance over or miss something. When investigating a website with the human eye rather than a computer, it is very probable for human error as in missing an important detail about the site. Most importantly, this process could be considered more dangerous than the automated approach. During the manual approach, we physically visited the website and exposed ourselves to potential threat actors. The automated approach, however, uses scraping tools to gather information, therefore, protecting the user even further.

The automated approach gave a different perspective to gathering data regarding malware threat distribution on the Dark Web. A strength of the automated approach is that there is less room for user error. There is guaranteed accuracy and validity to the data by allowing the computer to handle most of the work. There is also no need to visit the site which further protects the user's identity from the owner of the site. When scraping the site for HTML data, the user does not have to have to visit that onion site on the Dark Web. With the automated approach, there is more data to examine. By scraping several different onion sites, we could examine the differences between the layout and overall formatting each seller uses. Scraping the websites allowed access to the entire text population of the onion site from the click of a button. We were able to statistically analyze the TF-IDF scores, Naïve Bayes accuracy scores, and SVM accuracy scores related to the data. A weakness pertaining to the automated approach is that the scripts could be tweaked for efficiency. Several of the scripts were adapted from online sources

and fixed primarily to fit the scope of the research. If allotted more time, there could be changes made to the Python scripts for optimization and efficiency. Another weakness associated with this approach is that we were dealing with HTML data. HTML data can come in several formats and is often considered difficult to process. We utilized several pre-processing techniques to conform the data for analysis and consolidate the data. Lastly, scraping the Dark Web often leads to errors. As mentioned before, there are many layers of encryption involved with exploring the Dark Web. There is also a pattern of constant changes involved with onion domains whether they are taken down for a short time or indefinitely. For this reason, our web scraper was not able to scrape onions consistently. On occasion, the web scraper would time out or experience an interruption and we would have to attempt the scrape again.

CHAPTER V

CONCLUSIONS

The steps we have highlighted can be used as a framework to identify malware threat distribution on the Dark Web. Through this research, we were able to successfully identify malware by utilizing an automated and manual approach to gathering data. Examining different listings on the Dark Web allowed for the comparison of prices, selling tactics, and the features each seller is providing for the malware. After comparing the strengths and weaknesses of each approach, we can ultimately say that both approaches are successful and provide useful information for law enforcement, researchers, and other interested parties.

5.1 Limitations of Research

This experiment was successful in nature, however, there were many limitations of the research. It was evident early on that there are not many resources regarding the Dark Web in general. There were even fewer sources that focus directly on malware residing on the Dark Web. The sources we did find were often outdated or no longer relevant due to new revelations. The scripts we examined were also very outdated. As new updates and patches are released for software, it can be difficult or impossible to implement packages that are no longer in service. There were several instances during the testing phase where the script we wanted to adopt had resources that were phased out of

existence. Most importantly, the Dark Web is a dangerous subject matter. Visiting the Dark Web with no prior knowledge of the subject could result in possibly breaking the law. For that reason, we took several precautions to research the process and familiarize ourselves with as much literature as possible before exploring the Dark Web. Another limitation of the research is the fact that the Dark Web is not easily searchable. Sites on the surface are easy to remember seeing that they are comprised of actual words and letters. Sites on the Dark Web can rarely be converted to actual words. They are more like computational hashes and are difficult to remember or trace. The Dark Web is difficult to scrape or crawl since onion sites often change or shut down randomly. There are also several layers of encryption involved with accessing the Dark Web which also makes scraping the HTML data more problematic. Our experiment involved using an abundance of open-source software (OSS). There are always restrictions when dealing with open-source software including lack of security and compatibility issues. Dealing with these issues did cause delays in completing the research promptly. Another limitation of this research is that it is difficult to replicate due to constant changes on the Dark Web. The Dark Web is not a stagnant entity by any means. Onion links are always changing location and Dark Web marketplaces protect themselves by staying ahead of law enforcement and other entities that are trying to thwart their plans.

5.2 Future Work

With that in mind, there is certainly room for improvement in this research experiment. The scripts used can be upgraded and optimized for maximum efficiency. It would also be intriguing to interact with the seller to see what is received compared to

what was paid for and originally promised. A potential idea for future work is to compare the amount of time for the manual process versus the automated approach. It would also be interesting to pick a category of malware and buy multiple types on the Dark Web to see the differences in quality at different price points. There could be more than one seller selling the same product or the same product being sold at different price ranges. An additional topic of study is to examine the popularity of cryptocurrencies and determine how or why those specific currency types are trending. It would also be beneficial to look at more ways to implement machine learning techniques with Dark Web investigations. Concepts such as random tree or sentiment analysis would be interesting to see regarding malware threat distribution on the Dark Web. All in all, the possibilities are endless in this field of study.

REFERENCES

[1] M. Hatta, "Deep Web, Dark Web, Dark Net: A Taxonomy of 'Hidden' Internet," Annals of Business Administrative Science, vol. 19, no. 6, pp. 277–292, Sep. 2020, doi: https://doi.org/10.7880/abas.0200908a.

[2] M. K. Bergman, "White paper: The deep web: Surfacing hidden value," The Journal of Electronic Publishing, vol. 7, no. 1, Aug. 2021, doi: https://doi.org/10.3998/3336451.0007.104.

[3] C. Bradley, "On the Resilience of the Dark Net Market Ecosystem to Law Enforcement Intervention." UCL Discovery, Aug. 2019. [Online]. Available: https://discovery.ucl.ac.uk/id/eprint/10080409/8/Bradley_10080409_thesis.pdf.

[4] A. Biryukov, I. Pustogarov, and R. P. Weinmann, "Content and popularity analysis of Tor hidden services," 2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 188–193, Jun. 2014, doi: 10.1109/ICDCSW.2014.20.

[5] M. Wade, "Digital hostages: Leveraging ransomware attacks in cyberspace," Science Direct, vol. 64, no. 6, pp. 787–797, Nov. 2021, doi: https://doi.org/10.1016/j.bushor.2021.07.014.

[6] "RagnarLocker Ransomware Indicators of Compromise," Federal Bureau of Investigation, Mar. 2022. [Online]. Available: https://www.ic3.gov/Media/News/2022/220307.pdf.

[7] M. Chertoff, “A public policy perspective of the Dark Web,” Journal of Cyber Policy, vol. 2, no. 1, Mar. 2017, doi: https://doi.org/10.1080/23738871.2017.1298643.

[8] S. Takaaki and I. Atsuo, “Dark Web Content Analysis and Visualization,” International Workshop on Security and Privacy Analytics, pp. 53–59, Mar. 2019, doi: https://dl.acm.org/doi/10.1145/3309182.3309189.

[9] D. Patterson, “Getting started with Tails, the encrypted, leave no trace operating system,” TechRepublic, Aug. 2016, [Online]. Available: https://www.techrepublic.com/article/getting-started-with-tails-the-encrypted-leave-no-trace-operating-system/.

[10] A. Micard, “Building a fast modern web crawler for the dark web,” Dev Community, Sep. 2019. [Online]. Available: https://dev.to/creekorful/building-a-fast-modern-web-crawler-for-the-dark-web-l1.

[11] D. Markuson, “How to make a .onion site,” NordVPN, Apr. 2018. https://nordvpn.com/blog/how-to-register-onion-domain/#:~:text=When%20you%20create%20a%20.,to%20navigate%20to%20your%20server.https://nordvpn.com/blog/how-to-register-onion-domain/.

[12] I. Naqua and M. Murphy, “What is Machine Learning?,” Machine Learning in Radiation Oncology, 2015, doi: https://doi.org/10.1007/978-3-319-18305-3_1.

[13] R. Broadhurst et al., “Malware Trends on ‘Darknet’ Crypto-markets: Research Review.” Australian National University, Jul. 2018. [Online]. Available: https://www.researchgate.net/publication/326436666_Malware_Trends_on_'Darknet'_Crypto-markets_Research_Review.

[14] C. Glover, “Meet the ransomware gangs fuelling a global cybercrime spree,” TechMonitor, Jun. 2021. https://techmonitor.ai/technology/cybersecurity/top-ten-ransomware-gangs-fuelling-the-global-cybercrime-spree.

[15] A. Park, B. Beck, D. Fletche, P. Lam, and H. Tsang, “Temporal analysis of radical dark web forum users,” 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 880–883, 2016, doi: https://ieeexplore.ieee.org/document/7752341.

[16] H. Thorat, S. Thakur, and A. Yadav, “Categorization of Illegal Activities on Dark Web using Classification,” International Research Journal of Engineering and Technology, vol. 7, no. 5, May 2020.

[17] “About Ahmia,” Ahmia.fi. https://ahmia.fi/.

[18] “About Proton.” ProtonMail. [Online]. Available: https://proton.me/about.

[19] “DNM’S Buyer Bible,” Mar. 2018, [Online]. Available: https://darknetlive.com/post/dnm-bible-has-been-updated-shakybeats-aa20ede4.

[20] S. Fleig, “Scraping Onion Websites from the Darknet using Tor and Python,” LinkedIn, Jan. 2020. https://www.linkedin.com/pulse/scraping-onion-websites-from-darknet-using-tor-python-stephan-fleig/.

[21] Anorov, “Pysocks,” Sep. 2019. https://pypi.org/project/PySocks/.

[22] K. Reitz, “Requests 2.28.2,” Jan. 2023. https://pypi.org/project/requests/.

[23] L. Richardson, “BeautifulSoup4 4.11.2,” Jan. 2023. https://pypi.org/project/beautifulsoup4/.

[24] “Removing stop words with NLTK in Python,” Geeks for Geeks, Jan. 2023. https://www.geeksforgeeks.org/removing-stop-words-nltk-python/.

[25] “How to remove all non-alphanumeric characters from a string in Java,” Geeks for Geeks, Sep. 2022. https://www.geeksforgeeks.org/how-to-remove-all-non-alphanumeric-characters-from-a-string-in-java/.

[26] G. Bedi, “A guide to Text Classification (NLP) using SVM and Naive Bayes with Python,” Medium, Nov. 2018.

[27] B. Stecanella, “Understanding TF-ID: A Simple Introduction,” MonkeyLearn, May 2019. https://monkeylearn.com/blog/what-is-tf-idf/.

[28] K. Murphy, “Naive Bayes classifiers,” University of British Columbia, vol. 18, no. 60, pp. 1–8, Oct. 2006.

[29] E. Hodo, A. Hamilton, X. Bellekens, C. Tachtatzis, E. Iorkyase, and R. Atkinson, “Machine Learning Approach for Detection of nonTor Traffic,” Association for Computing Machinery, Aug. 2017, doi: http://dx.doi.org/10.1145/3098954.3106068.

APPENDICES

Appendix A: Spyware options found on the Dark Web

Malware	Description	Price
Simple Keylogger v1.0	Basic minimalistic keylogger	$20 USD
Simple Keylogger v2.0	Random file name that is unpredictable Cleaner special char output Recompiled for optimized size & speed Fully silent on run, no quickly flashing console window on startup	$30 USD
Nano Core	Remote Desktop Remote Webcam Built in Keylogger Password Recovery Chat Feature Remote File Manager Download and Execute Remote Terminal Process Manager Microphone Listening Registry Modification Reverse Proxy Advanced Setup	$30 USD
Advanced Keylogger v1.0	Shows what window is being typed to The log file is the current date and time Well defined and clear formatting with right click and left click Fully silent on run, no quickly flashing console window on startup	$50 USD
NjRAT	Remote Desktop Remote Webcam Built in Keylogger Password Recovery Remote File Manager Download and Execute Remote Terminal Process Manager Microphone Listening	$60 USD

Advanced-Keylogger v2.0	Built in installer to quickly install itself or retrieve logs to and from a USB Drive Shows what window is being typed to The log file is the current date and time Well defined and clear formatting with right click and left click Fully Silent On run, no quickly flashing console window on startup	$65 USD
Babylon Rat	Remote Desktop Remote Webcam 2 Built in Keyloggers Password Recovery Chat Feature Remote File Manager Download and Execute Remote Terminal Process Manager Reverse Proxy Auto Installer	$90 USD
Dark Comet	Remote Desktop Remote Webcam Built in Keylogger Password Recovery Remote File Manager Download and Execute Remote Terminal Process Manager Registry Modification Clipboard Access Advanced Setup Auto Installer	$150 USD

Appendix B: Source Code for web scraping script

```
#!/usr/bin/env python
# Importing modules
import socks
import socket
import requests
from bs4 import BeautifulSoup

# Configuring Socks to use Tor
from urllib.request import urlopen
socks.set_default_proxy(socks.SOCKS5, "localhost", 9050)
socket.socket = socks.socksocket

# It is necessary to use Tor for DNS resolution of Onion websites
def getaddrinfo(*args):
        return [(socket.AF_INET, socket.SOCK_STREAM, 6, '', (args[0], args[1]))]
socket.getaddrinfo = getaddrinfo

# Using requests package to read in the Hidden Wiki Onion Website on the Darknet
res =
requests.get("http://ytdfpiqzumzdejym5z7537spmn4geq4xqles6brxfkaipa7e6jpyofid.onio
n/2017/02/12/the-pizza-blog/")

# Using beautifulsoup to get the website content into a nice format
soup = BeautifulSoup(res.content, 'html.parser')

# Having a look at the Website content
print(soup.prettify())
```

Appendix C: Source code for TF-IDF, Naïve Bayes, and SVM

```
#!/usr/bin/env python

# Add the required libraries
import pandas as pd
import numpy as np
import nltk
from nltk.tokenize import word_tokenize
from nltk import pos_tag
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.preprocessing import LabelEncoder
from collections import defaultdict
from nltk.corpus import wordnet as wn
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import model_selection, naive_bayes, svm
from sklearn.metrics import accuracy_score

# Set random seed
np.random.seed(500)

# Add the corpus as a pandas Dataframe
Corpus = pd.read_csv(r"/home/usasoc/Downloads/myCorpus6.csv",encoding='latin-1')

# Data pre-processing: tokenization, word stemming/lemmatization
# Step - a : Remove blank rows if any.
Corpus['text'].dropna(inplace=True)
# Step - b : Change all the text to lower case. This is required as python interprets 'dog'
and 'DOG' differently
Corpus['text'] = [entry.lower() for entry in Corpus['text']]
# Step - c : Tokenization : In this each entry in the corpus will be broken into set of words
Corpus['text']= [word_tokenize(entry) for entry in Corpus['text']]
# Step - d : Remove Stop words, Non-Numeric and perfom Word
Stemming/Lemmenting.
# WordNetLemmatizer requires position tags to understand if the word is noun or verb or
adjective etc. By default it is set to Noun
tag_map = defaultdict(lambda : wn.NOUN)
tag_map['J'] = wn.ADJ
tag_map['V'] = wn.VERB
tag_map['R'] = wn.ADV
for index,entry in enumerate(Corpus['text']):
        # Declaring Empty List to store the words that follow the rules for this step
        Final_words = []
```

```
        # Initializing WordNetLemmatizer()
        word_Lemmatized = WordNetLemmatizer()
        # pos_tag function below will provide the 'tag' i.e if the word is Noun(N) or
Verb(V) or something else.
        for word, tag in pos_tag(entry):
        # Below condition is to check for Stop words and consider only alphabets
        if word not in stopwords.words('english') and word.isalpha():
        word_Final = word_Lemmatized.lemmatize(word,tag_map[tag[0]])
        Final_words.append(word_Final)
        # The final processed set of words for each iteration will be stored in 'text_final'
        Corpus.loc[index,'text_final'] = str(Final_words)

# Prepare train and test data sets
# Train_X -> Training Data Predictors
# Train_Y -> Training Data Target
# Test_X -> Test Data Predictors
# Test_Y -> Test Data Target

Train_X, Test_X, Train_Y, Test_Y =
model_selection.train_test_split(Corpus['text_final'],Corpus['label'],test_size=0.3)

# Encoding
Encoder = LabelEncoder()
Train_Y = Encoder.fit_transform(Train_Y)
Test_Y = Encoder.fit_transform(Test_Y)

# Word Vectorization
Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(Corpus['text_final'])
Train_X_Tfidf = Tfidf_vect.transform(Train_X)
Test_X_Tfidf = Tfidf_vect.transform(Test_X)


# Use the ML Algorithms to Predict the outcome
# fit the training dataset on the NB classifier
Naive = naive_bayes.MultinomialNB()
Naive.fit(Train_X_Tfidf,Train_Y)

# predict the labels on validation dataset
predictions_NB = Naive.predict(Test_X_Tfidf)

# Use accuracy_score function to get the accuracy
print("Naive Bayes Accuracy Score -> ",accuracy_score(predictions_NB, Test_Y)*100)

# Classifier - Algorithm - SVM
# fit the training dataset on the classifier
```

```
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)
# predict the labels on validation dataset
predictions_SVM = SVM.predict(Test_X_Tfidf)
# Use accuracy_score function to get the accuracy
print("SVM Accuracy Score -> ",accuracy_score(predictions_SVM, Test_Y)*100
```

BIOGRAPHICAL SKETCH

Name of Author: Shelby Caldwell

Graduate and Undergraduate Schools Attended:

University of South Alabama, Mobile, Alabama

Degrees Awarded:

Master of Science in Computer and Information Sciences, 2023, Mobile, Alabama
Bachelor of Science in Information Technology, 2021, Mobile, Alabama

Awards and Honors:

Women in Cybersecurity Student Scholar and Featured Poster Presenter, 2023
NICE/CAE Conference Featured Poster Presenter, 2022
Cybercorps: Scholarship for Service, 2021