Document Title:	**Technical Evaluation and Legal Opinion of Warden: A Network Forensics Tool, Version 1.0**
Author(s):	**Rod Yazdan, Newton McCollum, Jennifer Ockerman, Ph.D.**
Document Number:	**252944**
Date Received:	**May 2019**
Award Number:	**2013-MU-CX-K111**

JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

11100 Johns Hopkins Road • Laurel, Maryland 20723-6099

AOS-18-1223

NIJ RT&E Center Project 15WA

October 2018

TECHNICAL EVALUATION AND LEGAL OPINION OF WARDEN: A NETWORK FORENSICS TOOL

Version 1.0

Rod Yazdan
Newton McCollum
Jennifer Ockerman, PhD

Prepared for:

NIJ | National Institute of Justice
STRENGTHEN SCIENCE. ADVANCE JUSTICE.

Prepared by:
The Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Rd.
Laurel, MD 20723-6099

Task No.:	FGSGJ
Contract No.:	2013-MU-CX-K111/115912

This project was supported by Award No. 2013-MU-CX-K111, awarded by the National Institute of Justice, Office of Justice Programs, U.S. Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this publication/program/exhibition are those of the author(s) and do not necessarily reflect those of the Department of Justice.

CONTENTS

EXECUTIVE SUMMARY

The National Criminal Justice Technology Research, Test, and Evaluation (RT&E) Center at the Johns Hopkins University Applied Physics Laboratory (JHU/APL) was tasked by the National Institute of Justice (NIJ) to evaluate the forensic electronic data collection tool, *Wide-scale, Agentless and Rapid collection of Digital Evidence from Networks (WARDEN)*, developed by Assured Information Security, Inc. The goal of the evaluation was to answer five questions:

1. How does WARDEN identify, acquire, and preserve data of an investigative value?
2. What are WARDEN's analysis and reporting capabilities with regards to investigative data?
3. Does the functionality of WARDEN operate as intended?
4. Is WARDEN forensically sound? If not, how can it be enhanced to be more forensically sound?
5. What are the pros and cons of other forensic solutions?

During an initial technical review to understand and document WARDEN capabilities, however, it was discovered that WARDEN is not forensically sound. For example, it doesn't encrypt stored information or create an audit trail to help with chain of custody control. In addition, WARDEN data analysis does not appear to be matched to the needs of law enforcement personnel since it provides highly summarized information and does not allow for organization of data by case. Given these initial findings, effort was directed toward explaining the shortcomings of WARDEN and possible improvements. In addition, a legal opinion was documented on the requirements for a forensically sound digital evidence collection tool.

The legal opinion provided is included as Appendix B and notes that WARDEN will likely provide data and information that is admissible in a court of law but as it does not allow for adequate chain of custody control, the resulting data and information is less valuable as persuasive evidence—perhaps much less valuable. Although WARDEN does not produce information of probative value, it may still be useful as an investigative tool to identify entities for formal search.

1. INTRODUCTION

In September of 2013, the Johns Hopkins University Applied Physics Laboratory (JHU/APL) was selected by the U.S. Department of Justice, National Institute of Justice (NIJ) to establish the National Criminal Justice Technology Research, Test, and Evaluation (RT&E) Center within the National Law Enforcement and Corrections Technology Center (NLECTC) System. The purpose of the RT&E Center is to provide in-depth technical reports and support for NIJ's research and development efforts.

For this project, NIJ tasked the RT&E Center to verify and validate the capabilities of the WARDEN software product developed by Assured Information Security, Inc., (AIS) to process large-scale computer networks for digital evidence in a forensically sound manner that preserves the probative value of the evidence that the computer network may contain.

1.1 Digital Forensics

In law enforcement, digital forensics techniques are used to collect information to prosecute crimes, both cyber and noncyber crimes, which occur on a network or digital platform. Digital forensics is "the science of identifying, preserving, recovering, analyzing and presenting facts about digital evidence found on computers or digital storage media devices.[1]

Identifying digital data involves determining where the data is stored. Data can be stored on many types of devices from computer hard drives and network devices to mobile platforms and flash drives. There are many places digital data may reside and all likely locations need to be identified before planning the data collection process.

Preserving digital data refers to capturing the data of interest in its original state without any alterations. It is extremely important that the data be preserved with integrity, so that it is admissible in and acceptable to a court of law. Often this is done by copying a digital source with high and verifiable precision and then working with the verified copy and not the original source.

Recovery is usually needed and potentially involves the actions of restoring deleted files (both normal operating system deletions and purposeful user deletions), accessing password-protected data, and capturing damaged and corrupted data.

Analysis is the gathering of all the digital data that is connected to the crime. The more that can be gathered to cross corroborate user activities the better.

Presenting the facts of the findings in a clear and concise matter is the extremely important final step of digital forensics. Additionally, the presentation should be understandable by non-technical personnel such as other law enforcement personnel, lawyers, judges, and jury members.

[1] Interworks, "What is Digital Forensics?" https://www.interworks.com/blog/bstephens/2016/02/05/what-digital-forensics, accessed: March 22, 2018

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

Depending on the crime being investigated, this may be all or a part of the evidence against a suspect. "As well as identifying direct evidence of a crime, digital forensics can be used to attribute evidence to specific suspects, confirm alibis or statements, determine intent, identify sources (for example, copyright cases), or authenticate documents."[2]

1.2 Admissibility and Value of Evidence

Admissibility of evidence is unrelated to its value as evidence. Evidence that is deemed authentic by digital forensic subject matter experts (SMEs) would be admissible by a judge to be used in a case, but it might not be valuable to the case if the jury believes the evidence is tainted or irrelevant to the case. The jury or other fact finder fact may draw such conclusions due to uncertainties introduced by opposing council. Admissibility is covered by Federal Rules of Evidence (FRE) 104(a) and (b).

FRE 104(a) – "The court must decide any preliminary question about whether a witness is qualified, a privilege exists, or evidence is admissible. In so deciding, the court is not bound by evidence rules, except those of privilege."[3]

FRE 104(b) – "When the relevance of evidence depends on whether a fact exists, proof must be introduced sufficient to support a finding that the fact does exist. The court may admit the proposed evidence on the condition that the proof be introduced later."

> The significance of these rules is that when an objection is lodged to proffered digital data that includes a bona fide dispute of fact as to its authenticity, the court's decision to admit the digital data will only be conditional, subject to a determination by the jury (or other fact finder) on the basis of admissible evidence. For example, consider a case in which a company e-mail written by a specific employee is proffered as an exhibit. The basis for its authenticity is the fact that it was found on the company's server, it purports to have been sent by an employee, it bears the employee's company e-mail address. If an objection is made on the basis that anyone could have written the e-mail using the employee's e-mail account, the judge's decision on admissibility will be final; no issue of fact has been raised. In other words, it is not enough to speculate about what facts may theoretically impact the authenticity of the e-mail. If on the other hand, the objection is that anyone could have written the e-mail, and the defense will produce 5 witnesses who will testify that they were with the employee at the time the e-mail was sent, and the employee did not send the e-mail, the judge's decision will be conditional, such an objection would raise an actual dispute of fact and the judge may find the e-mail has been authenticated, and admit it into evidence conditionally; the jury will decide the factual dispute based on the evidence actually produced at trial. The significance of this interplay between the

[2] Wikipedia, "Digital forensics," https://en.wikipedia.org/wiki/Digital_forensics, last edited on 18 March 2018, viewed on 22 March 2018.
[3] "Privilege" as in attorney-client privilege or spousal privilege.

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

> rules may greatly increase the quality of supporting evidence required to authenticate the e-mail.[4]

Until relatively recently, documentary evidence was of a physical nature. With the proliferation of digital devices into our lives, however, much of what used to be paper is now in a digital format. Overall, court rules of procedure and evidence are still dependent on rules and procedures developed for physical, paper documents. Furthermore, there is little judicial authority to date that addresses the issues of admitting digital evidence as data.

In spite of the lack of definitive legal guidance on the use of digital data, there has been significant discussion in legal and technical journals and some courts. The main admissibility issues that have been discussed are the difficulty of proving (or rebuffing) the authenticity and accuracy of digital data.

The admissibility of digital evidence is only part of the issue. Much evidence is legally admissible but may not have legal value. The value is the amount of weight given by the jury or other fact finder in a case. The value of evidence is often tied to the chain of custody, and the ability to show that the evidence has not been tampered with in any manner.

1.3 WARDEN

WARDEN, which stands for "Wide-scale, Agentless and Rapid collection of Digital Evidence from Networks," was proposed as a tool to allow digital forensics investigators to query large-scale computer networks for digital evidence in a forensically sound manner, and thus preserve the probative value of the evidence that the computer network may contain.

The multi-aspect goal of WARDEN is to identify, preserve, acquire, analyze, and report data of investigative value from large-scale computer systems and computer networks in a forensically sound manner; focusing on the collection of evidence regarding criminal intent and criminal activity (not necessarily affecting the network).

In addition, WARDEN attempts to streamline and enhance forensic data collection and analysis to provide investigators an ability to quickly and remotely extract evidentiary data from remote devices to reconstruct indicators of compromise or criminal intent and activity by searching for anomalies or potentially incriminating evidence in said data.

In documentation provided to the RT&E Center, the developer provided the following brief introduction to WARDEN's capabilities:

> The WARDEN architecture is an innovative and agentless incident response and network forensics framework. The framework supports data collection, normalization and analysis of collected data. It has a modular, flexible plug-in interface that is designed for rapid deployment of custom scripts all while utilizing native system interfaces. Warden can extract information from selected hosts without the use of an agent by gaining remote access to the computer using

[4] Wolf, M.E., "Admissibility of Digital Evidence Derived Using WARDEN," provided in full as Appendix B.

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

> one of WARDENs transports. Transports open a connection and deliver a payload of specified scripts, called collectors, to the selected computer which collect specific information which is then sent back on the transport's connection to WARDEN. When incoming data is received from a host, WARDEN performs automatic recognition, organization and storage of data in a local or remote MongoDB instance [database] where it is analyzed. Each collector has specific data that it will attempt to collect from a system. The data collected can range from a full system information analysis, to currently installed programs and drivers, or the information that is currently stored in memory. WARDEN ships with many collectors and transports able to gather data on an array of different systems. The WARDEN framework was developed in such a way that new collectors and transports can be continuously developed and added in the framework not only by the WARDEN Team but by operators as well. [5]

When using WARDEN, the investigator must determine what collectors, transports, and related tools are to be part of the investigative task or "job" that they specify for WARDEN. The investigator then schedules and starts the job. The job will run to completion or until the investigator deems enough time has passed for sufficient data to have been collected. The data is then returned, stored in the database for analysis, and put into a report for the investigator.

1.4 Project Goal

The RT&E Center was tasked by NIJ to investigate the following questions in regards to WARDEN:

1. How does WARDEN identify, acquire, and preserve data of an investigative value?
2. What are WARDEN's analysis and reporting capabilities with regards to investigative data?
3. Does the functionality of WARDEN operate as intended?
4. Is WARDEN forensically sound? If not, how can it be enhanced to be more forensically sound?
5. What are the pros and cons of other forensic solutions?

To answer these questions, the RT&E Center planned to conduct an Independent Verification and Validation (IV&V) of the WARDEN tool, however an initial technical review to understand and document WARDEN capabilities revealed that WARDEN is not forensically sound. That is, as stated in the WARDEN final technical report to NIJ, "if an active adversary knew that the computer was under investigation by WARDEN, the data that is collected has the potential to be subjected to integrity attacks before it is picked up."[6] In addition, WARDEN is not as focused on law enforcement data needs as might be desired. Therefore, effort was directed toward explaining the shortcomings of WARDEN and possible improvements. In addition, a legal

[5] Assured Information Security, Inc., "Wide-Scale, Agentless and Rapid collection of Digital Evidence from Networks (WARDEN)," Final Technical Report, p. 4, February 2017, provided in full as Appendix A.
[6] Ibid., 50.

opinion was documented on the requirements for a forensically sound digital evidence collection tool (the full legal opinion is Appendix B).

1.5 Organization of Report

Section 2 of this document provides background on forensic collection of digital data: why it is needed, the technical and legal issues, and what attributes forensic data collection software must have to satisfy all these needs and issues. Section 3 provides the details on the technical review of WARDEN. Section 4 provides information on the forensic software tool Encase® that has been used successfully to collect digital evidence that was admissible and valued in the court system.

2. FORENSIC DATA COLLECTION BACKGROUND

2.1 What is the need for forensic data collection?

Digital information and data are pervasive in our current society, and that will likely only increase. As a result, it is now an important source of information about criminal activities. Forensic data collection refers to the collection of digital evidence, which has been defined as "information and data of value to an investigation that is stored on, received, or transmitted by an electronic device."[7] Digital evidence is the same as other evidence in that it is used to implicate a particular person with criminal activities. However, digital evidence is somewhat ephemeral in nature, requiring different tools and training, and demanding more rigorous methods of collection.

Digital evidence can be found on numerous platforms. One of the original sources of digital evidence came from early message boards and chat rooms. These initial electronic communications have evolved into the current internet and social media sites. These sites are often real-time, encrypted communications, which can be hard to collect at a later date. It is also possible to communicate in an anonymous manner, making it very difficult to easily attribute data to a particular person.

As noted by the NIJ publication, High Priority Criminal Justice Technology Needs,[8] there is the need for "improved capability to use and process digital evidence, including:

- Tools to investigate the use of peer-to-peer technologies used to facilitate criminal activity, such as distribution of contraband, that address decentralized and unstructured peer-to-peer network protocols.
- Tools that can recover system files, operating system information, applications, deleted files and unallocated space from small-scale mobile devices, such as cell phones and personal digital assistants.

[7] National Institute of Justice, "Electronic Crime Scene Investigation: A Guide for First Responders, second edition," U.S. Department of Justice, Washington, DC," 2008.

[8] National Institute of Justice, "High-Priority Criminal Justice Technology Needs," U.S. Department of Justice, Washington, DC," 2010, pp. 24–25.

- Full data imaging solutions for networks and network-attached or -connected devices addressing:
  - Redundant Array of Independent Disks (RAID).
  - Wireless network devices, including routers, gateways, network interface cards, repeaters, switches, hubs and wirelessly connected external digital media.
  - Network data storage devices that are either directly connected or connected by computer to the network.".

WARDEN strives to address all or portions of the third bullet, full data imaging solution for networks and network-attached or -connected devices.

2.2 Why is forensic data collection done?

Forensic data collection and analysis is done by law enforcement to gather evidence of investigative value and probative value of criminal activity. Evidence of investigative value has less rigorous requirements of its authenticity and integrity and is used within an investigation to provide leads to criminal activity or to provide evidence of the need for a search warrant. Evidence of probative value has highly rigorous requirements as to its authenticity and integrity to be used profitably within a court of law. Forensic data collection can be useful to collect evidence of criminal activity which is computer-based or cyber in nature and criminal activity that is more traditional but has been supported by digital platforms or communications in some way. The data collected has the power to show that the suspect(s) are responsible for or knowledgeable of the criminal activity.

2.3 What are the technical issues of forensic data collection from networks?

Digital data collection that is forensically sound and can be used for investigative and probative purposes can be difficult and error-prone even from "dead' or static devices such as unplugged, stand-alone computer hard drives. But, as noted in Section 2.1, there is a need for data collection and analysis for networks and network-attached or -connected devices, which are "live" or not static by nature with data changing constantly as they are used. As a result most network-oriented forensics data collection focuses on communication packets.

> Packet capture and analysis is an essential capability for any digital investigation, but post hoc network analysis requires more than just the ability to see and interpret communication packets on the wire. For instance, the network traffic produced during a one-time insider threat attack cannot be recovered, and a more precise and diverse tool is required to query the running processes on live machines, pull registry key values, and examine random access memory (RAM), firewall logs, content accessible memory (CAM) tables in switches, and routing tables in routers.[9]

[9] Assured Information Security, Inc., "Wide-Scale, Agentless and Rapid collection of Digital Evidence from Networks (WARDEN)," Final Technical Report, pg. 1, February 2017, provided in full as Appendix A.

The collection of this volatile data from live machines without investigator disruption is very difficult. Simply accessing this information will then show the investigator as one of the people who has accessed the information. Therefore, the information needs to be recorded without introducing investigator activities into internal logs, such as registries in rapid access memory (RAM). Tools are needed that assist investigators in recording in a manner that preserves the integrity of these data and follows the order of volatility,[10] which collects the most volatile data first and allows the investigator to control their digital footprint.

2.4 What are the legal issues of forensic data collection?

There are two main legal issues that have been identified with forensic data collection, proving authenticity and showing proper chain of custody control (see Appendix B for a full discussion of authenticity and chain of custody). Authenticity is addressed by FRE 901(a): "To satisfy the requirement of authentication or identifying an item of evidence, the proponent must produce evidence sufficient to support a finding that the item is what the proponent says it is." Legally, this is a relatively easy standard to meet and is what determines the admissibility of evidence into a court proceeding. The judge simply needs to find that a jury is likely to find that the evidence is what it is said to be to admit it to the court.

Chain of custody control issues are not addressed by the FRE but it can be argued that it is the more important of the two considerations. Chain of custody control affects the value that a jury places on a piece of evidence during its deliberations. If the chain of custody control is not clear and provides doubt that the evidence was generated in the manner that is being claimed, then the jury may put a low value on the evidence or disregard it completely. Even though the FRE does not address the issue of chain of custody control, there is some guidance from *The Sedona Conference Commentary on ESI Evidence & Admissibility*. Specifically, it discusses the use of metadata for proof of data generation method (who, when, where), and the use of hash values[11] to show that the electronic data used for investigation matches the original data.

2.5 What attributes are required in forensic data collection software?

The answers to the above questions provide guidance for what is needed in a forensic software package.

> …the software: (1) preserves the target files without alteration, (2) obtains hash values of the target files, (3) obtains copies of the target files, (4) obtains hash values for each copy, and (5) maintains the integrity of each file and hash value until admitted into evidence, including adequately documenting the chain of custody. (See Appendix B, p. 10)

[10] SANS Digital Forensics and Incident Response Blog, "Best Practices in Digital Evidence Collection," http://digital-forensics.sans.org/blog/2009/09/12/best-practices-in-digital-evidence-collection/, accessed on July 24, 2018.

[11] "Hash values can be thought of as fingerprints for files. The contents of a file are processed through a cryptographic algorithm, and a unique numerical value – the hash value – is produced that identifies the contents of the file. If the contents are modified in any way, the value of the hash will also change significantly." https://www.trendmicro.com/vinfo/us/security/definition/hash-values, accessed on October 15, 2018.

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

These activities are much more easily accomplished when the investigation is static and the data and metadata are not continuing to change. When dealing with a live system, however, these requirements are much harder to meet, since the data and metadata could be changing, making the need for a superb digital chain of custody all the more important.

3. WARDEN TECHNICAL REVIEW

As noted earlier, WARDEN was a response to an NIJ request for a system to process large-scale computer networks for digital evidence in a forensically sound manner that preserves the probative value of the evidence that the computer network may contain. More specifically, a means for criminal justice agencies, in particular state and local agencies, to identify, preserve, acquire and/or analyze and report data of an investigative value from large-scale computer systems and computer networks in a forensically sound manner; focusing on the collection of evidence regarding criminal intent and criminal activity not necessarily affecting the network.[12]

To accomplish this, the RT&E Center conducted a technical review of WARDEN's capabilities. The technical review environment consisted of two Windows 2008 R2 machines, one running WARDEN and one serving as a client, and 12 active virtual machines. The technical review consisted of pulling text test files, ports and generic system information from this environment. The technical review took approximately eight hours.

3.1 Digital Evidence Collection Using WARDEN

When using WARDEN, an investigator must identify the types of data to be collected and select or develop the corresponding collector scripts, or plugins. Collector plug-ins provided by WARDEN support retrieving data such as: certificates, Dynamic Host Configuration Protocol (DHCP) logs, event logs, hash processes, open ports, and others. (See Appendix A for full list of data collector plugins provided by WARDEN). WARDEN does not provide support for these collection decisions but does process the collected data to provide the investigator with a concise summary view of the information that has been collected.

3.2 WARDEN Technical Review Results

Five capabilities were analyzed during the technical review. These included Transport and Collection Process, Data Preservation, Information Filtering, Data Analysis, and Reporting. The analysis of the first two capabilities addresses the first question [How does WARDEN identify, acquire, and preserve data of an investigative value?] and the fourth question [Is WARDEN forensically sound?]. If not, how can it be enhanced to be more forensically sound? The remaining capability analyses address the second question [What are WARDEN's analysis and reporting capabilities with regard to investigative data?] The fifth question [What are the pros and cons of other forensic solutions?] is addressed in the next section about Encase®. The third question [Does the functionality of WARDEN operate as intended?] was not addressed once the technical review determined that WARDEN was not forensically sound. In the following sub-sections, each capability analysis is described and possible improvements are suggested.

[12] NIJ Solicitation SL001136, "Collecting Digital Evidence from Large-Scale Computer Systems and Networks," December 22, 2014.

1. How does WARDEN identify, acquire, and preserve data of an investigative value?
2. What are WARDEN's analysis and reporting capabilities with regards to investigative data?
3. Does the functionality of WARDEN operate as intended?
4. Is WARDEN forensically sound? If not, how can it be enhanced to be more forensically sound?
5. What are the pros and cons of other forensic solutions?

3.2.1 Capability Analysis 1 –Transport and Collection Process

WARDEN uses transport and collector plugins to collect data of investigative value, as specified by the investigator. The transport plugin transfers the collector plugin to the remote machine and runs it. The information is then returned to WARDEN via the selected transport plugin. Some of the transport plugins that WARDEN supports are WMI, SMB, SNMP, and SSH (Linux). WARDEN supports several ways of transporting and collecting information from a remote host and that is one of the biggest benefits of its ability to collect investigative data.

On the negative side, if the username or password of the remote machine are unknown, then WARDEN is unable to collect much critical information, such as process data and logs showing the activity history of the machine. In addition, there are several other configurations of the remote machine that could hinder WARDEN's collection performance such as blocked communication ports, disabled execution of unsigned scripts in the operating system, and disabled remote access. Another concern is that the only transport that is available for gaining access to Linux, Mac, and other physical appliances (e.g., firewall routers) is Secure Socket Shell (SSH), so if this transport option isn't supported there is no other available method to communicate with the machine.

Possible Improvements to Transport and Collection Process – There are at least two ways WARDEN could be more forensically sound in the transport and collection process. One way would be to implement encryption on all transport plugins so that information is protected while moving over the network. A second way would be having more transports for non-Windows systems which would allow for more information to be collected across a wider range of devices. Finally, for law enforcement use, having the ability to collect information from devices without knowing the username and password would make the tool much more useful.

3.2.2 Capability Analysis 2 – Data Preservation

WARDEN's ability to preserve data of an investigative value is severely limited. It does not provide investigators with the tools they need for preserving chain of custody of the potential evidence collected. One of the known limitations with WARDEN is that "if an active adversary [knows] that the computer [is] under investigation by WARDEN, the data that is collected has potential to be subjected to integrity attack before it is picked up."[13]

[13] Assured Information Security, Inc., "Wide-Scale, Agentless and Rapid collection of Digital Evidence from Networks (WARDEN) Final Technical Report," p. 50, February 1017, provided in full as Appendix A.

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

Even once the data is retrieved, there are issues. The storage database used by WARDEN is a version of MongoDB that is a "NoSQL" database. As a result, jobs cannot be correlated to one another and are comingled with jobs from different investigations. There is no capability for WARDEN to group jobs by case. There are technical benefits to WARDEN's choice of database. It is fast, efficient with space, and the formatting is easily readable and importable from a programmatic standpoint. Therefore, tools can be developed in the future to connect and perform analytics on the data; but from a policy perspective, there is an insurmountable drawback—the lack of an available audit trail for tracking changes to the database including the inability to track the investigator making the changes. Without an audit trail, it is impossible to prove the integrity of the data, making it less likely to be acceptable and useful in court.

Other drawbacks are not attributable to the use of MongoDB in general, but rather to the default security features WARDEN implements in their instance of MongoDB. The WARDEN default implementation of MongoDB is not secure and must be configured after installation for SSL communications and for adding read-only users. Finally, the version shipped with WARDEN, MongoDB 3.0, does not support at-rest encryption, which is required to ensure that the data is not readable by outside entities. To implement at-rest encryption would require the installation and configuration of MongoDB 3.2+ in place of the default MongoDB 3.0.

Possible Improvements to Data Preservation Capability – There are several ways that WARDEN may be made more forensically sound in its data preservation capability. Some recommendations would be to implement an audit trail, to have a separate database for redundancy, to upgrade to the latest Mongo database for encryption at rest, and to enable two factor authentications so that access to the machine is more secure.

3.2.3 Capability Analysis 3 – Information Filtering

Some of the benefits of WARDEN's design are that an investigator receives the information that they are expecting, they won't be overloaded by useless information, and they can make specific queries of a remote machine. On the negative side, much of the information retrieved is filtered for displaying, down to percentages and numbers for the report. The underlying data cannot be viewed in the browser-based user interface but only in a custom WARDEN power shell command interface that would be very challenging for law enforcement investigators to master. The summarized data is not always useful for law enforcement forensics. For example, the *netstat* collector results will display the number of ports open but not which ports are open or what their respective processes are.

Possible Improvements to Information Filtering Capability – WARDEN data displays would be improved by showing the investigator the raw collected data in addition to the results of summarizing, filtering, manipulating, or correlating the raw data. For example, if a *netstat* collector is chosen, the information returned and displayed should include the open port and what process is running on it. Additionally, WARDEN might also recommend further data collection jobs to the investigator based on the returned information. To continue with the example, using the information about the process running on a port, WARDEN could recommend that the *hashprocess* collector be selected by the investigator to retrieve the hash of

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

the running process followed by analysis of the process and display the determination of whether it is a known malicious hash.

3.2.4 Capability Analysis 4 – Data Analysis

WARDEN does not supply native analysis capabilities for collected data but it does supply plugins to analysis toolsets that can perform analyses as configured by an investigator. The two analysis toolsets that WARDEN supports are Elastic Stack and Splunk.[14]

The benefit of using analysis toolsets is that they are specifically built to do relevant data analysis. These particular tools are also able to spot commonalities across multiple jobs and investigations such as a file hash or IP address. The downside is that these analysis toolsets were designed for cyber security data such as event logs which doesn't meet law enforcement needs for non-cyber-security criminal investigations. Another disadvantage of relying on these analysis toolsets is their relatively steep learning curve. Finally, the transfer of information would need to be secured between WARDEN and the analysis applications, requiring the data to be stored in two locations instead of one location, adding to storage space requirements and possibly introducing data integrity issues.

Possible Improvements to Data Analysis Capability – The data analysis capability of WARDEN could be improved by providing an automated method to configure the stand-alone data analysis applications to provide analytic capabilities specific to forensics with appropriate dashboards for visualizing the results and correlations. This process would allow for a more controlled installation and integration between WARDEN and the stand-alone analysis tools as it would be done via the automated installation process and not by an end user who may or may not be familiar with the stand-alone analysis tool. These applications could also be configured to use the existing Mongo database, thereby reducing required storage space requirements and increasing security by eliminating the need to send potentially sensitive information over a network again [e.g., personally identifiable information (PII)].

3.2.5 Capability Analysis 5 – Reporting

WARDEN provides some reporting capabilities about the data that has been collected by specific jobs, but the displayed data is aggregated and filtered before being shown to the analyst. The reports page provides an investigator with quantitative data about the job, and the Impact Report shows the commands that were run on the remote machine, the machine state, and various information about the job such as deployment start and end time.

Another example of WARDEN's limited reporting is the Certificate Collector. The report pages show how many certificates are on the remote computer and how many are unique, but it does not supply the names of the certificates or who signed them. This reporting loses value to the investigator due to its level of abstraction, such as resolving information down to a summary quantitative measure instead of reporting specific information collected which is often needed in forensic investigations.

[14] Note that Elastic Stack and Splunk Community Edition are free. The enterprise edition of Splunk is not.

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

Possible Improvements to Reporting Capability – WARDEN could improve the reporting functionality by including the specific underlying data so that an investigator or an analysis tool could run analysis on more granular data to produce more actionable results.

3.3 Summary

This technology review of WARDEN's capabilities shows that it is not forensically sound. WARDEN lacks key security features due to outdated MongoDB software versions and lack of scripted configuration procedures. Shortcomings in preserving the forensic integrity of collected data include the lack of encryption for data at rest, use of default SSL support, and the lack of database redundancy. Crucially, WARDEN does not support an audit trail to ensure that all data modifications are properly attributed to the people and procedures responsible. WARDEN would require significant updates to become a viable tool for computer forensics.

4. ENCASE – A SUCCESS STORY

Encase is currently one of the most popular software packages used, and accepted in the court room, for digital forensic applications as discussed in this paper.

From legal opinion in Appendix B:

> EnCase is a suite of forensic software products produced by OpenText. Originally created in 1998, it has become the standard for obtaining digital evidence for use in court.[15] Evidence acquired through the use of EnCase forensic software has been accepted by numerous courts over the years. One attribute cited by courts in describing the veracity of such evidence is the fact that EnCase ensures the accuracy of forensic copies through the use of matching hash values, that is, the unique hash value of the original target disk drive or file (whatever the case may be) matches the hash value for the copy being offered into evidence.
>
> To ensure the integrity of the matching hash values, OpenText employs a specific file format for EnCase products known as EnCase Evidence File Format ("EEFF"). The EEFF is broken into three sections, the header, data blocks, and footer. The header contains case information such as the date and time of acquisition, the examiner's name, notes on the acquisition, etc. The data blocks contain the actual acquired data, but at the time of acquisition, the copy is split

[15] Although not the only widely-used forensic software (FTK Forensic Toolkit is also cited frequently in caselaw), it has been tested and evidence derived from the use of EnCase has been accepted by many courts over a relatively long time. See, Williford v. State, 127 S.W.3d 309, 312-13 (Tex. App. 2004); Sanders v. State, 191 S.W.3d 272, 278 (Tex. App. 2006)(two early decisions relying on EnCase); see also, Criminal Cases State v. Pratt, 200 Vt. 64, 77, 128 A.3d 883, 891 (2015); United States v. Romm, 455 F.3d 990, 995 (9th Cir. 2006); United States v. Ganias, 824 F.3d 199, 204 (2d Cir. 2016); United States v. Gaynor, 2008 WL 113653, at *1 (D. Conn. Jan. 4, 2008); United States v. McCoy, 2015 WL 7770181, at *3 (D. Minn. Oct. 1, 2015); and Civil Cases In re Hitachi Television Optical Block Cases, 2011 WL 3563781, at *2 (S.D. Cal. Aug. 12, 2011); Malibu Media, LLC v. Doe, 82 F. Supp. 3d 650, 656 (E.D. Pa. 2015); Xpel Techs. Corp. v. Am. Filter Film Distributors, 2008 WL 744837, at *1 (W.D. Tex. Mar. 17, 2008).

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

into 32KB sections, with a "Cyclic Redundancy Check" between each section. The CRC includes a hash value calculated for the section. If the data is ever accessed, the CRC hash can be recalculated and compared to the original to determine if any change has been made. The footer contains an MD5 hash of the entire image. Essentially, the identifying information regarding the target file, the copied file, and the process of acquisition, are locked in a "vault" that itself is uniquely identified by a hash value.

One of the software products in the EnCase suite is EnCase Endpoint Investigator, OpenText describes as follows:[16]

> EnCase Endpoint Investigator is designed for corporations and government agencies to perform remote, discreet, and secure internal investigations without disrupting an employee's productivity or impacting day-to-day operations of the business.

Endpoint Investigator appears to be similar to WARDEN in purpose and function.[17] Although we have been unable to find a published decision discussing Endpoint Investigator specifically, this particular program shares an important attribute with EnCase Forensics (which has been discussed and accepted by many courts; see n. 5). One of the points stressed in OpenText's marketing materials is that, "Evidence collected from remote machines is stored in the EnCase Evidence File Format, which has been accepted and proven in courts worldwide as forensically sound."[18]

The significance of this method of storage is that it provides an extremely robust chain of custody. As explained in Section 2.2, a jury may consider any challenge to the chain of custody in deciding the weight to give to a piece of evidence. By embedding the file hash values, system metadata, and job information in a file, and recording the hash value of that file, Endpoint Investigator establishes a formidable digital chain of custody for the files obtained.

[16] EnCase Endpoint Investigator Product Overview located at: EnCase Endpoint Investigator for Internal Forensic Investigations, https://www.guidancesoftware.com/docs/default-source/document-library/product-brief/encase-endpoint-investigator-product-overview.pdf?sfvrsn=f4a08dad_84 (last visited 6/8/2018).

[17] With the exception that Endpoint Investigator can actually obtain copies of files as well as metadata, and verify the copies by hash value. Although the description in marketing materials describes Endpoint Investigator as being used for internal control projects, it appears to be suitable for criminal investigations and prosecutions as well. In one case study reported by OpenText, an internal investigation uncovered a scheme among high-level financial employees who had destroyed documents in order to improve the corporation's position with the Securities and Exchange Commission. Based on the evidence uncovered by Endpoint Investigator, the employees were prosecuted. Endpoint Investigator Case Studies located at:
EnCase Endpoint Investigator in Action, https://www.guidancesoftware.com/document/product-brief/guidance-software-encase-endpoint-investigator-in-action (last visited on June 8, 2018)

[18] See n. 6, supra; see also cases collected in n. 5, supra.

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

5. CONCLUSION

The RT&E Center at JHU/APL was asked by NIJ to evaluate the forensic electronic data collection tool WARDEN developed by AIS. The goal was to answer five questions:

1. How does WARDEN identify, acquire, and preserve data of an investigative value?
2. What are WARDEN's analysis and reporting capabilities with regards to investigative data?
3. Does the functionality of WARDEN operate as intended?
4. Is WARDEN forensically sound? If not, how can it be enhanced to be more forensically sound?
5. What are the pros and cons of other forensic solutions?

During an initial technical review to understand and document WARDEN capabilities, however, it was discovered that WARDEN is not forensically sound. For example, it doesn't encrypt stored information or create an investigation trail to help with chain of custody control. In addition, WARDEN data analysis doesn't appear to be matched to the needs of law enforcement personnel since it provides highly summarized information and doesn't allow for organization of data by investigation. Given these initial findings, effort was directed toward explaining the shortcomings of WARDEN and possible improvements. In addition, a legal opinion was documented on the requirements for a forensically sound digital evidence collection tool.

The legal opinion provided is included as Appendix B. It notes that WARDEN will likely provide data and information that is admissible in a court of law but due to WARDEN's inadequate chain of custody control, the resulting data and information is less valuable as persuasive evidence—perhaps much less valuable. Although, WARDEN does not produce information of probative value, it may still be useful as an investigative tool to identify entities for formal search.

APPENDIX A. FINAL TECHNICAL REPORT ON WARDEN

Wide-Scale, Agentless and Rapid collection of Digital Evidence from Networks (WARDEN)

Final Technical Report

Grant Number: 2015-IJ-CX-K003

Prepared for: National Institute of Justice

February 2017

Team Members:

Adam Meily
Richard Gloo
Jason Nashold
Sidney Borne
Sean LaPlante
Keara Hill
Stephen Pape
Austin Benincasa
Richard Cook
Salvatore Paladino

Assured Information Security, Inc.
153 Brooks Road
Rome, NY 13441
(315) 336-3306
www.ainfosec.com

Assured Information Security. Inc.

Table of Contents

List of Figures

List of Tables

Assured Information Security, Inc.

1 SUMMARY

1.1 Objective

Most network forensics tools concentrate on the capture and analysis of communication packets; however, we argue that networks contain a wide range of evidentiary data across their workstations, servers, and other miscellaneous network devices. We are offering WARDEN not only as a capability that not only integrates with existing intrusion detection system (IDS) and packet capturing tools [1] [2], but also provides a single platform to allow an investigator to canvass network activity in a much deeper and comprehensive level. WARDEN does not minimize the importance of communication traffic analysis; rather it extends the investigators reach and allows them to build a more focused and coherent picture with data residing on a network. In general, the Internet Protocol (IP) suite defines the responsibilities of devices at specific layers of the network stack. Under the WARDEN effort, the IP suite will serve as the basis for logically grouping data that is collected from endpoint systems on an enterprise network as part of a large-scale investigation.

1.2 Background

Digital forensics evidence on a computer network often remains uncollected or ignored. This is largely in part to the fact that there are no available tools where an outside investigator can be introduced to a network of any size, mine the data, and process the highly-dimensional data set for a clear and concise reconstruction of the events.

During an investigation, preserving integrity of gathered evidence is cornerstone for delivering justice beyond reasonable doubt. Unlike a physical crime scene, digital forensic evidence exists on a more abstract level and in many ways, is more delicate. Investigators must extract relevant digital information from a crime scene, prove that it is unaltered, and create a coherent reconstruction of the scene. Although tools are available to do this for a single computer [3] [4], they do not scale well nor have the features required for network forensic evidence. This obstacle can be attributed to the ephemeral nature of network communication data.

Network forensics tools are available but mostly focus on the capture and analysis of network traffic [5] [6] [7]. Packet capture and analysis is an essential capability for any digital investigation, but post hoc network analysis requires more than just the ability to see and interpret communication packets on the wire. For instance, the network traffic produced during a one-time insider threat attack cannot be recovered, and a more precise and diverse tool is required to query the running processes on live machines, pull registry key values, and examine random access memory (RAM), firewall logs, content accessible memory (CAM) tables in switches, and routing tables in routers [8]. Similarly, a malware attack may not necessarily still be actively running when investigators arrive to canvass the network. This scenario too cannot benefit from traffic-monitoring network forensic tools.

Network forensic tools need to be scalable for today's intricate attacks and must be able to collect and analyze heterogeneous data from multiple network resources. With the cooperation of network administrators there are scores of network communication artifacts that may be collected. Network administrative privileges, however, come with the added responsibility of preserving forensic data in its natural state without any disruption. Using the Windows® operating system (OS) as an example, if an administrator remotely accesses a suspect or victim's account, navigates to a folder on their machine and opens a file, there are time-stamped values stored in the registry that are updated to reflect this action [9]. This ultimately spoils any useful facts that are gleaned, because the data will show the administrator was the last person to access the file. These volatile registry values exist in RAM and must be recorded before an investigation may continue. This example can be generalized to say that knowing the order in which network information is extracted is specialized knowledge fundamental to computer network forensics. Currently, there are no tools that are available to guide investigators through the appropriate steps needed to gradually increase and control their digital footprint in an enterprise network environment. The challenge to provide such a tool to law enforcement is complicated by improper network configurations, poor firewall settings, inconsistent security policies, diverse OSs, and the absence of data logs. Law enforcement agents need generic network forensics tools that follow the order of volatility principles consistently despite the divers set of obstacles that may impede their investigation [10].

The challenge associated with network forensics has been described here as a problem involving transient data that is difficult to extract without disturbing other sensitive information. These complications are compounded further by the audit reduction problem, where massive volumes of noisy network data obscure relevant information [11]. Not only are the data volumes massive, but also the relevant data from an attack may extend to multiple hosts. This makes the network analysis stage of an investigation appear intractable, but current research has shown promising results using correlation and graph-based approaches [12]. Recent advancements in retracing network events typically involves one of two approaches: the interpretation of data from multiple machines at one chosen layer of the OSI model [12] or from one machine at multiple layers of the OSI model [13]. The next step in this line of research is to build models that support the replay of multiple and concurrent network events over multiple machines in which evidence from multiple OSI layers has been collected.

1.3 Scope

The scope of this effort was to develop a robust and innovative product to enhance the capabilities of those challenged with incident response and enterprise forensics.

1.4 Principal Results and Conclusions

This effort resulted in the successful development of technologies that satisfy operational requirements. These technologies include WARDEN, ANTIGEN, WARDEN Training Environment (WTE), and the WARDEN Web User Interface, which are summarized below and are described in detail in Section 3 of this report.

1.4.1 WARDEN

WARDEN was rapidly prototyped based on end-user requirements, matured via rapid development based on user feedback from continuous releases, and has become an integral component in the arsenal employed by investigators. WARDEN currently serves as an enterprise network forensics framework which enables incident responders and investigators to rapidly conduct incident response and network forensics operations.

1.4.2 WARDEN Training Environment

The WTE was developed to train operators in the use of WARDEN. WTE is an interactive, web-based training application which focuses on providing investigators with the skills necessary to use the WARDEN command line interface (CLI). WTE contains eight lessons which guide the operator through fundamental operations such as invoking the **help** command, to navigation of complex tasks such as those involving the macro system.

1.4.3 WARDEN Web User Interface (WebUI)

The WARDEN WebUI was developed to enable WARDEN operation without a command line interface. The WebUI makes the capability more accessible to the investigator. The WARDEN WebUI enables all aspects of the normal WARDEN workflow, including the ability to create, deploy and view basic job data, to utilize all available transports and collectors, and to access helper functionality built into the framework.

1.4.4 ANTIGEN

ANTIGEN is a software script that performs a live memory analysis of a system to discover evidence of malicious artifacts. ANTIGEN can inspect artifacts to provide host-based incident response and data collection to support the analysis of a system compromise.

2 INTRODUCTION

Law enforcement investigators are responsible for investigating a large, heterogeneous, and dynamic landscape. Adding to that difficulty, investigators must continually adapt to changing tactics, techniques, and procedures. For these reasons, it is expected that investigators' tools slowly become dated and, in some cases, obsolete, over time.

The Agile Cyber Solutions team at AIS worked in very close conjunction with operators throughout this effort to identify requirements put forth by the Department of Justice and rapidly developed and prototyped capabilities which help enhance the capabilities of law enforcement and other professionals performing incident response and enterprise forensics.

Each capability developed during this effort was in direct response to a specific requirement, and included socializing the approach and technical requirements of the proposed solution with end-users.

3 METHODS, ASSUMPTIONS, AND PROCEDURES

This section identifies the methods used in the research and development of technologies, assumptions made, and the procedures used during the process.

3.1 WARDEN

3.1.1 Problem

Current incident response and network forensics tools depend on inflexible, agent based architectures that collect specific data based on preconceived expectations of incident response requirements. Additionally, agent based tools require a specialized software installation, and are prone to compatibility problems. Ultimately, existing incident response and network forensics tools are difficult to tailor to new and unexpected events.

3.1.2 Approach

The WARDEN architecture is an innovative and agentless incident response and network forensics framework. The framework supports data collection, normalization and analysis of collected data. It has a modular, flexible plug-in interface that is designed for rapid deployment of custom scripts all while utilizing native system interfaces. Warden can extract information from selected hosts without the use of an agent by gaining remote access to the computer using one of WARDENs transports. Transports open a connection and deliver a payload of specified scripts, called collectors, to the selected computer which collect specific information which is then sent back on the transport's connection to WARDEN. When incoming data is received from a host, WARDEN performs automatic recognition, organization and storage of data in a local or remote MongoDB instance where it is analyzed. Each collector has specific data that it will attempt to collect from a system. The data collected can range from a full system information analysis, to currently installed programs and drivers, or the information that is currently stored in memory. WARDEN ships with many collectors and transports able to gather data on an array of different systems. The WARDEN framework was developed in such a way that new collectors and transports can be continuously developed and added in the framework not only by the WARDEN Team but by operators as well. The adaptability and flexibility of the framework allows WARDEN and its operators to stay at the bleeding edge of incident detection/response and network forensics.

The WARDEN process can be outlined in five steps:

1) The operator installs scripts from the file system called Collectors. These Collectors can be imported as PowerShell scripts, Python scripts, Windows® Management Instrumentation (WMI) scripts, native executables, .Net binaries, or Batch scripts. (The necessary dependencies must exist on the remote host to allow for successful execution of collectors)

2) The operator combines Collectors into a job, which is scheduled and executed against a list of remote hosts on a network.

3) After job completion, the collected data is picked up.

4) The information is placed in a data store for analysis.

5) The operator then performs post-process analysis of the data. WARDEN allows operators to plug in tools and custom scripts for immediate use. Existing incident response and defensive collection capabilities can be ported directly into WARDEN with minimal modification.

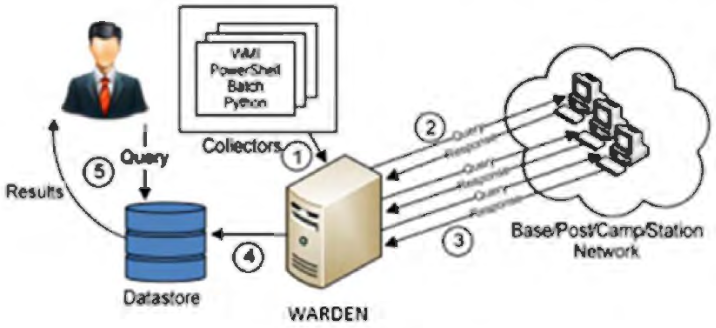

Figure 1: WARDEN Workflow

The WARDEN installation contains all the required dependencies and several documents that provide a complete overview of the WARDEN installation, administration, configuration, architecture, and extension. The table below provides the recommended reading sections for operators:

Table 1: WARDEN Manuals

Document	Targeted Personnel	Description
WARDEN User Manual	Everyone	Basic installation, architecture overview, and CLI guide
WARDEN Administration Guide	Administrators	Advance administration and configuration
WARDEN PowerShell Interface	Analysts Plugin Developers	WARDEN PowerShell Provider usage and .Net bindings
WARDEN Extension Guide	Plugin Developers	Extending WARDEN with Plugins

3.1.2.1 WARDEN Plugin Types

Collectors gather data from a remote system, whereas Transports provide the means for the collectors to gather data from Remote Hosts; these two plugin types are independent. Remediators

perform actions on a remote system while Analyzers visualize and sum up data collected from a WARDEN Job.

Collectors – Plugins that collect data from a remote system.

Table 2: WARDEN Collector Description

Collector	Collector Description
Collectors/autoruns	The publicly available Autoruns Collector is a Windows® native executable that retrieves information about executables registered to run automatically at boot and other specific times. Internally, the Autoruns collector uses the Autoruns for Windows® utility from Sysinternals.
Collectors/certificates	The Certificates Collector is a Windows® .NET 2.0 executable that enumerates the installed security certificates on a computer.
Collectors/eventlogs	Collects all Windows® event logs from the remote machine and returns them to the database.
Collectors/hashfs	The HashFileSystem Collector is a Windows® .NET 2.0 executable that recursively walks the filesystem, hashing files that match a specified set of file extensions. All drives can be walked and hashed, including network mounted drives. HashFileSystem is intended to increase the number of baseline hashes in MekaDB, and hashes from a known clean system can be used to populate the database whitelist.
Collectors/hashproc	Hash Process is a .Net application that computes hashes for all of the modules loaded in memory within the current system. For each loaded module, three hashes are computed: MD5, SHA-1, and SHA-256. Hash Process gathers the list of loaded modules by first listing the currently running processes and then, for each process, listing the loaded modules. For each module in the loaded module list, the on-disk origin is found and the physical file is hashed.
Collectors/logins	The Logins Collector is a Windows® native executable that retrieves authentication events from the Windows® Event Log and writes them to a standard .zip file (DEFLATE-compressed) that contains a comma separated values (CSV) file. The application is configurable through command line arguments which can be used to set verbose record information, filter records based on username, and specify the output file path to write the collected data.
Collectors/netstat	The netstat command is a commonly operating system used utility for enumerating a host's network connections and their origin. This is useful for discovering suspicious network activity on a host. The Netstat collector consists of two scripts (PowerShell for Windows® and bash for Unix-based operating systems (Oss)) that wrap the netstat command. It is important to note that any reasonably sophisticated adversary will likely takes precautions to hide their activity from the netstat command.
Collectors/nullpipes	The NullPipes Collector is a Windows® native executable that searches for pipes that have been registered within the OS and whether it is Null or not. Malware typically uses null pipes for unauthenticated command and control channels.
Collectors/packages	Many popular Linux distros have package managers used to install and remove programs from the system. The Packages collector is a bash script which wraps two such package managers, yum and apt, to enumerate the installed packages on a CentOS or Debian-based system, respectively.

Collectors/pii	The PII Collector is a Windows® .NET 2.0 executable that scans a directory tree for files containing information that is personally identifying and could be used for identity theft and fraud. The PII Collector specifically searches for Social Security Numbers and credit card numbers from a variety of vendors. The PII collector provides the necessary foundations for extending to other pattern based disk searches.
Collectors/procdump	The ProcDump Collector is a Microsoft Sysinternals tool that collects a running process's memory and saves its contents to disk as a binary memory dump file. Along with providing a process memory dump, procdump can also provide a full RAM acquisition.
Collectors/antigen	The ANTIGEN application is a native Windows® executable that performs memory forensics to detect and alert on generic indicators of compromise. Rather than identifying specific malware samples or detecting certain hash signatures, ANTIGEN enumerates the memory of each process and loaded module to perform deep inspection for the identification of generic indicators of: • Hidden and injected modules • Hidden processes • Code modifications • Function hooks • Weak or modified memory region protection These five indicators are generic to all malware samples. For example, there are many methods available for malware to inject itself into another process. Rather than search for signatures related to this functionality, ANTIGEN will detect the following indicators in memory, regardless of how a malware sample injected itself. • Memory regions with Read/Write/Execute permissions • A hidden module • Code modifications • Function hooks Each process in a Windows® system has a Virtual Address Descriptor (VAD) tree, which defines all allocated and reserved memory within the process. The VAD tree contains critical information such as loaded modules, reserved memory regions, and memory allocation permissions. ANTIGEN primarily uses each process's VAD tree to enumerate allocated memory.
Collectors/sessions	The Sessions Collector is a Windows® .NET 2.0 executable that lists all active sessions and their related user information.
Collectors/splunk	Allows for bi-directional data transfer between WARDEN and a Splunk datastore. This collector can be configured to send all data to the Splunk datastore, or it can send Splunk query data to Mongo DB. This collector requires a Splunk database to be set up prior to use.
Collectors/survey	Performs a basic system survey of a remote machine using WMI queries. Information is gathered from the remote system by querying several WMI classes.
Collector/sysinfo	The SysInfo Collector is a Windows® .NET executable that is able to gather general information about the Windows® system. This collector is compatible with all versions of Windows® that have the .NET 2.0 framework or greater installed. The application writes the gathered information as JSON to the standard output stream or can optionally write the JSON output to a file specified as a command line argument.

Collectors/unhidenet	Unhidenet is a public open source tool that queries Windows® Application Programming Interface (API) for the TcpTable and UdpTable objects which hold data for all the ports that are currently in use. UnHideNet will attempt to use all available ports on the computer, if the collector cannot connect to the port it will look up the port in the loaded tables. If the port is not registered in the tables, it is assumed that the ports has been hidden for malicious reasons.
Collectors/usbdrives	Discovers USB devices that are currently connected or have been recently connected to a specific host computer. Collecting this information can help determine the origins of certain files as well as the disappearance of files from a host computer. USB devices can pose a potential threat to networks by allowing a passageway for malicious files to enter a secure network or by allowing a similar passageway for sensitive files to be exfiltrated.
Collectors/verifyproc	The VerifyProc collector utilizes an XML formatted ruleset file to determine what types of anomalies it will look for and alert on. WARDEN contains a default ruleset XML file that will find the most common anomalies that might indicate the presence of a malicious process or actor on a system. However, this ruleset file is modifiable to meet the environment's specific constraints.

Transports – Plugins that provide Communication methods to the remote systems.

Table 3: WARDEN Transport Description

MS Transport/capability	Transport Description
Transports/deploy/wpt	Installs WPT on remote host(s)
Transports/generic/launch	Launches a list of local and/or remote processes
Transports/generic/push	Configurable Transport to push files to a remote system
Transports/wpt	The WPT provides WARDEN with the capability to push, pull, and execute files on remote machines via a secure communication channel over a Named Pipe. WPT is a 32-bit native executable that installs as a service on remote systems and uses an X.509 certificate to establish a trust relationship with client system. Once an initial handshake is completed and the trust relationship is verified, WPT will receive encrypted control messages over a configurable Named Pipe to perform any of the following tasks: 1. Write a file to disk 2. Read a file from disk 3. Create a directory 4. Create a directory in the system's temporary files folder 5. Execute a file with a specified priority 6. List a directory 7. Delete a file 8. Recursively delete a directory 9. Report the architecture and version of the system 10. Uninstall the WPT service and optionally delete the WPT executable. WPT is integrated into the WARDEN framework as the transports/wpt plugin. The WPT Transport is designed to be used as an alternative to the transports/smb, transports/mswmi, and transports/task plugins based on mission requirements. Existing Jobs and Plugins will continue to work with WPT without modification.

Assured Information Security, Inc.

Transports/mswmi	The WMI Transport enables Plugins that require remote execution access to a system. The WMI Transport cannot modify nor interact with the remote system's file system; however, the WMI Transport provides a method of creating new processes and detecting the remote system's configuration. The WMI Transport is solely used for launching new processes remotely. In a typical WARDEN Job, a Collector will push its payload to the remote system using a Transport capable of pushing files, such as the SMB Transport, and then use the WMI Transport to launch the payload. Launching processes is accomplished using the WMI class Win32_Process and calling its Create method. Processes created with the WMI Transport are executed using the current user's credentials. This means that all processes spawned by WMI will be executing with the current user's privileges and no SYSTEM-level privileges. For SYSTEM-level privileges, use either the transports/task or transports/wpt Plugins.
Transports/smb	The SMB Transport uses the Windows® SMB subsystem and protocol to manipulate files on remote systems. The SMB Transport provides the push, pull, and execution capabilities, which gives it the ability to perform Job deployment, host architecture detection and file pickup tasks.
Transports/ssh	The SSH Transport uses the Paramiko1 SSH 2.0 library to manipulate files and execute commands on remote systems. The SSH Transport provides push, pull, and execute capabilities, which means that the SSH Transport performs Job deployment, collector execution, host architecture detection and file pickup.
Transport/task	The Robo application attempts to replace both the at.exe and schtasks.exe applications to provide a single tool for scheduling Windows® tasks regardless of a remote system's configuration. Robo is able to seamlessly schedule tasks on Windows® 2000 through Windows® 10 systems, and both workstation and server installations (32 and 64-bit). Robo is designed to be used both within a WARDEN Job as an execution Transport and outside of WARDEN to schedule remote tasks from the command line. The Robo application is designed to alleviate several compatibility issues with the existing task scheduler applications, including performance and interoperability. When used within WARDEN, Robo provides the core of an updated Remote Scheduled Task Transport (collectors/task), which previously used the schtasks.exe application as the underlying Transport. Therefore, previous Jobs that have used the Remote Scheduled Task Transport can use the new Robo-based transport without any modifications or configuration changes. Robo can login to a remote system with a specific username and password combination, and tasks can be executed as either the local SYSTEM account, or a specified authenticated user.

Remediators – Plugins that perform an action on a remote system.

Table 4: WARDEN Remediators Description

MS Remediators/capability	Remediators Description
Remediators/basic	The Basic Remediator is a Windows® .NET 2.0 executable that can delete files, stop or uninstall services, and unload or uninstall drivers.

	The Basic Remediator will output a JSON file that describes the actions it performed and their success rate. For example, when deleting a file, the Basic Remediator will first check if the file exists and, if it does, the file will be deleted. Both the files existence and the result of deleting the file will be saved as a single JSON document. The Basic Remediator can be used as a standalone command line application or within a WARDEN Job by loading the remediators/basic Remediator.	
Remediators/firewall	The Firewall Remediator is a Windows® MSVC executable that can create, enable, disable, and remove Windows® Firewall rules. The program can be run either with a single operation specified, or with a manifest file describing a set of operations. The success or failure of each operation is printed to standard output in CSV format. To perform Windows® Firewall manipulation, the Remediator uses the INetFwProfile1 interface of the Windows® API. The endpoint for this functionality is Hnetcfg.dll on Windows® XP SP2, and FirewallAPI.dll elsewhere.	
Remediators/kick	The UserKick Remediator utilizes the remote desktop services API to logoff and disconnect sessions. The API functions associated with these operations are listed below.	
	Function	**Use**
	WTSLogoffSession()	Used to logoff a specific user session
	WTSGetActiveConsoleSessionID()	Used to retrieve the unique session identifier of the currently active console, the result of this is used to logoff or disconnect a user if no session identifier is provided to the UserKick Remediator.
	WTSDisconnectSession()	Used to disconnect a specific user session

Analyzers – Plugins that help visualize and graph job data.

Table 5: WARDEN Analyzers Description

WARDEN analyzers/capability	Analyzers Description
analyzers/elastic	Exports job data to a ElasticSearch Engine where it is then imported and visualized in a Kibana Server.

WARDEN generates a vast amount of data for the operator after a job has successfully completed. This data is stored in a MongoDB instance and can be accessed using the WARDEN PowerShell Provider. Each piece of job data is tagged with a JobID, organized in such a way that enables the data to be rapidly, and accurately presented to the operator for analysis. For each completed Job a report is automatically generated and stored in the database. The report consists of statistical information on the job and includes but is not limited to these fields: Elapsed Time, Alive Hosts, Dead Host, and Network Bandwidth. The report also includes settings information about the job that was run. Examples of this information are, hosts selected, collectors used and set jobs options.

The WARDEN PowerShell Provider is a CLI built upon Windows® PowerShell that includes standard PowerShell functionality along with custom WARDEN cmdlets that aid the operator in accessing job data in a very intuitive and functional way. The main cmdlets are:

Get-Artifact

The Get-Artifact cmdlet enables the operator to retrieve a malware sample from the MekaDB collection of the MongoDB. The operator can specify options for retrieving an artifact. These options include whether it is whitelisted, blacklisted or graylisted.

Add-Artifact

The Add-Artifact cmdlet enables the operator to add new Artifacts to MekaDB from files that exist at a physical path on disk. The source file's hashes will be calculated and a new MekaDB Artifact will be created, saved to the data store, and written to the pipeline.

Get-ChildItem

The Get-ChildItem cmdlet will, with no parameters specified, list all objects within a Collection. For large scale Jobs with thousands of hosts, listing all objects within a Collection is not feasible because of limited time and resources. Therefore, the WARDEN Provider adds several additional parameters to the Get-ChildItem cmdlet to aid in filtering results retrieved from the Datastore.

3.1.3 Usage

The first time WARDEN CLI is launched, the operator will he prompted ahout beginning a training session. The beginner is encouraged to launch the training and go through the eight lessons included in the WTE, which will introduce them to its CLI and provide instructions on how to build and execute WARDEN jobs.

There are two options for launching the WARDEN CLI.

- Double-click the WARDEN desktop icon
- **Start Menu > All Programs > WARDEN > WARDEN**

Assured Information Security, Inc.

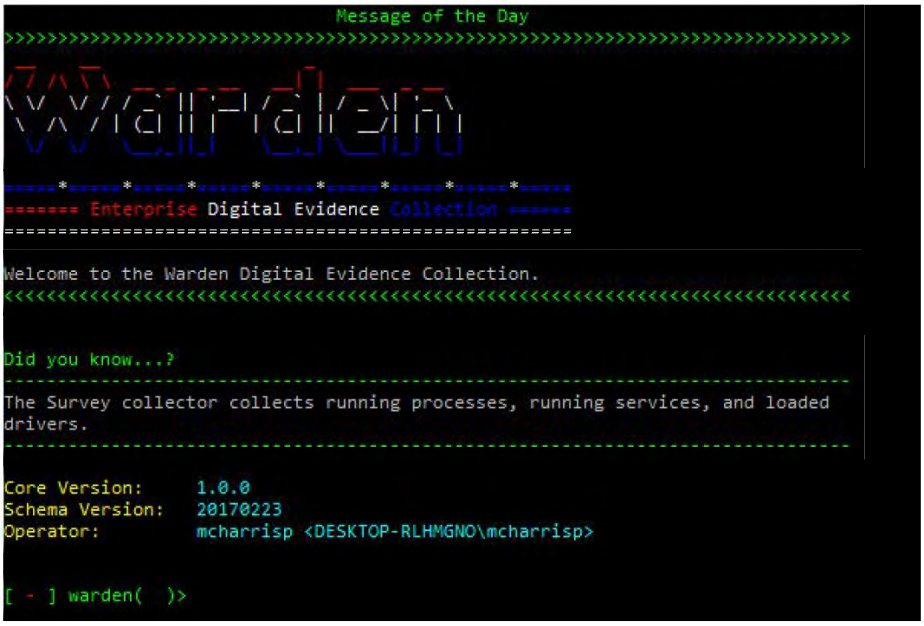

Figure 2: WARDEN CLI

The **help** command displays all the commands available within WARDEN with a brief description. Each command has help text that is displayed by adding -h or –help flags to the command. Additionally, the **help** command also recognizes Plugins and Job Options and will display help for both accordingly. For example, there is a **collectors/survey** topic that can be queried with the following command:

```
[ - ] warden(  )> help collectors/survey

Survey (collectors/survey)
==========================

The Survey collector gathers details about processes, services, and drivers
from the Remote Host.

Survey uses remote Windows Management Interface calls to query system
attributes.

Output
""""""

The Survey Collector will save objects to three collections, depending on the
survey.processes, survey.drivers, and survey.services options. Each record in
the collections corresponds to a single instance on the Remote System. For
example, each object in the processes collection will be a single process
running on a single remote system.

Analysis
""""""""

The Survey Collector will perform analsyis on collected processes, services,
and drivers after the Job has executed if the job.analysis option is set to
True.

Plugin Options
""""""""""""""

  - survey.drivers - Retrieve list of loaded drivers
  - survey.processes - Retrieve list of running processes
  - survey.services - Retrieve list of running services

Use "help --full collectors/survey" for more information

#
# Job name is not defined:
# Use "job rename <name>" to modify job name
#
# No Plugins are currently loaded:
# Use "show plugins" to list available Plugins
# Use "use <plugin_path>" to include a Plugin
#
[ - ] warden(  )>
```

Figure 3: Example Help Topic

Loading Plugins – To view available Collectors, use the **show collectors** command:

```
[ - ] warden(  )> show collectors
Path                    Description
=======================================================================
collectors/antigen      Antigen Native Client
collectors/autoruns     Detect the autoruns currently installed using the
                        SysInternals autoruns tool
collectors/certificates Collects all locally installed security certificates
collectors/cisco        Collect basic Cisco network device information
collectors/dhcp         Collect active DHCP leases from remote DHCP servers
collectors/eventlogs    Gathers all event logs from a remote system
collectors/hashfs       Recursively generate hashes for all executable files
                        within a path
collectors/hashproc     Generates hashes for all running processes and their
                        loaded modules
collectors/logins       Login attempt collector
collectors/macresponse  MacResponse
collectors/netstat      Collect active UDP and TCP sessions
collectors/nullpipes    Detect null session Named Pipes typically used for
                        backdoor command and control channels
collectors/packages     Linux installed packages collector
collectors/pii          Searches for personal information on a computer
collectors/procdump     Collect the memory of a running process
collectors/sessions     Collect list of active user sessions
collectors/snmp         Queries target via SNMP
collectors/survey       Basic system survey
collectors/sysinfo      Collects information about the remote system's hardware
collectors/unhidenet    Detects TCP and UDP ports and connections on the system
                        that are hidden
collectors/usbdrives    Gathers the USB drive history from a remote computer
collectors/verifyproc   Verifies the integrity of core system processes

# Use "use <plugin_path>" to include plugin
[ - ] warden(  )>
```

Figure 4: Available Collectors Displayed

Configuring a Job

Operator loads Plugins by using the **use collectors/<Collector Name>** or **use remediators/<Remediator Name>** command.

```
[ - ] warden(  )> use collectors/survey
Loaded Survey plugin

# Use "use -d <plugin_path>" to unload plugin from job
[ - ] warden(  )>
```

Figure 5: Plugins Loaded

Operator executes **check** command to view options needed to get a job in the "ready" state.

Figure 6: Check Command Used to View Job State

After executing the **check** command, the operator can now see there are unresolved dependencies as well as job options/job name that needs to be set. To view the unresolved dependencies, the operator runs the **check deps** command.

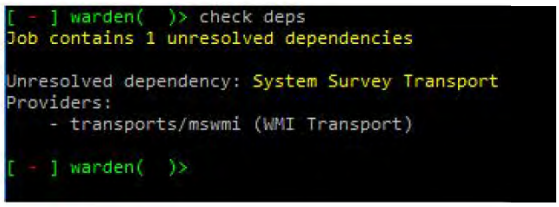

Figure 7: Unresolved Dependencies Displayed

The operator loads Transports based on the unresolved dependencies found by using the **check deps** command. Transports are then loaded by using the **transports/<Transport Name>** command. After the dependent Transports are loaded the **check deps** command is executed and returns "All dependencies are resolved".

```
[ - ] warden(  )> use transports/mswmi
Loaded WMI Transport plugin

# Use "use -d <plugin_path>" to unload plugin from job
[ - ] warden(  )> check deps
All dependencies are resolved
[ - ] warden(  )>
```

Figure 8: Dependencies Resolved by Loading of Transports

The remote host/s being investigated need to be set. The operator uses the **set job.rhost <IP address or Hostname>** command to update this job option

```
[ - ] warden(  )> set job.rhost 192.168.1.1
job.rhost option has been set/updated

# Use "set -c <option>" to clear option
# Use "set -r <option>" to reset option to default value
[ - ] warden(  )>
```

Figure 9: job.rhost Set

The operator now needs to rename the job. The **job rename <Unique Job Name>** command is used.

```
[ - ] warden(  )> job rename WARDEN_TEST_1
Job renamed to WARDEN_TEST_1
[ + ] warden( WARDEN_TEST_1 )>
```

Figure 10: Job Renamed

To ensure the job is ready to execute the operator runs the **check** command.

```
[ + ] warden( WARDEN_TEST_1 )> check
Job is ready to execute

[+] Job name specified
[+] Plugins loaded
[+] Dependencies resolved
[+] Plugins validated
[+] Options validated

Uncertain job runtime: no plugin statistics have been collected

# Use the 'schedule' command to start the job
[ + ] warden( WARDEN_TEST_1 )>
```

Figure 11: Job Ready to Execute

The operator now launches the job by using the **schedule now** command.

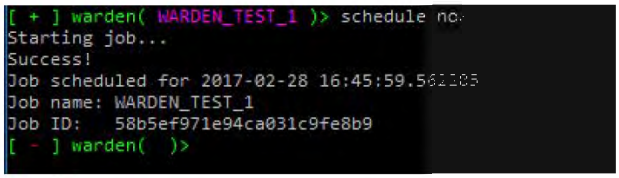

Figure 12: Job Executed

3.2 WARDEN Training Environment

3.2.1 Problem

The WARDEN CLI is a complex and highly specialized operational environment that requires a degree of familiarity and proficiency to use effectively and to its full abilities. Many of the more powerful features of the WARDEN CLI are not immediately discoverable without some level of training.

3.2.2 Approach

The WARDEN team considered several approaches to a training scenario for operators. A classroom-oriented lesson-based course was designed to iteratively introduce operators to the features of the WARDEN CLI. WTE was designed with an emphasis on allowing operators to interactively use features of the CLI as they would in an operational environment. Common tasks and workflows are emphasized and repeated as new features are taught to improve familiarity and efficiency of the operator. A leaderboard is used to track operators' progress through the course.

The WTE is an interactive, web-based training application designed to provide operators with the skills necessary to use the WARDEN CLI. It contains eight lessons. The course begins with a lesson introducing the help system and progressively moves through how to schedule jobs, troubleshoot common issues when creating jobs, importing Plugins, and how to use the Macro system.

WTE forwards all commands entered by the operator to a live WARDEN CLI session. WTE performs as little emulation as possible so that operators are always using the latest version of the WARDEN CLI. WTE will emulate some commands, such as **schedule**, to ensure that no Jobs created in the training environment are executed on the network. Additionally, the core WARDEN data collections are separated from the data generated from WTE. This is to ensure that Jobs created in the WTE do not appear in the WARDEN CLI and Jobs created in the WADEN CLI do not appear in the WTE.

3.2.3 Lessons

Eight lessons are included in the WTE:

Lesson 1: **Interacting with the shell** – Introduction to the WARDEN CLI. Upon completion, the operator will be familiar with the basic usage of the WARDEN CLI.

Lesson 2: **Using Variables in WARDEN** – Helps operator become familiar with the use of variables within WARDEN.

Lesson 3: **Working with WARDEN Plugins** – Demonstrates how to use and remove Plugins from a job as well as searching for Plugins and displaying more information about them.

Lesson 4: **Working with Multiple Jobs Concurrently** – The basics of building more than a single job concurrently.

Lesson 5: **Scheduling a ANTIGEN Job** – Takes operator through building and scheduling a ANTIGEN job to run.

Lesson 6: **Interacting with Running Jobs** – Teaches interactive job commands, **job abort, job pickup,** and other commands used to interact with a running job.

Lesson 7: **Creating a Basic Macro in WARDEN** – Demonstrates how to create a macro that will automatically run the survey collector on remote hosts that are specified in a variable.

Lesson 8: **Importing a Custom Collector** – Teaches operator how to use the **plugin import** command to import a custom Plugin into WARDEN.

3.2.4 Usage

To test the WTE, the lessons themselves can be followed as seen in the following screenshots:

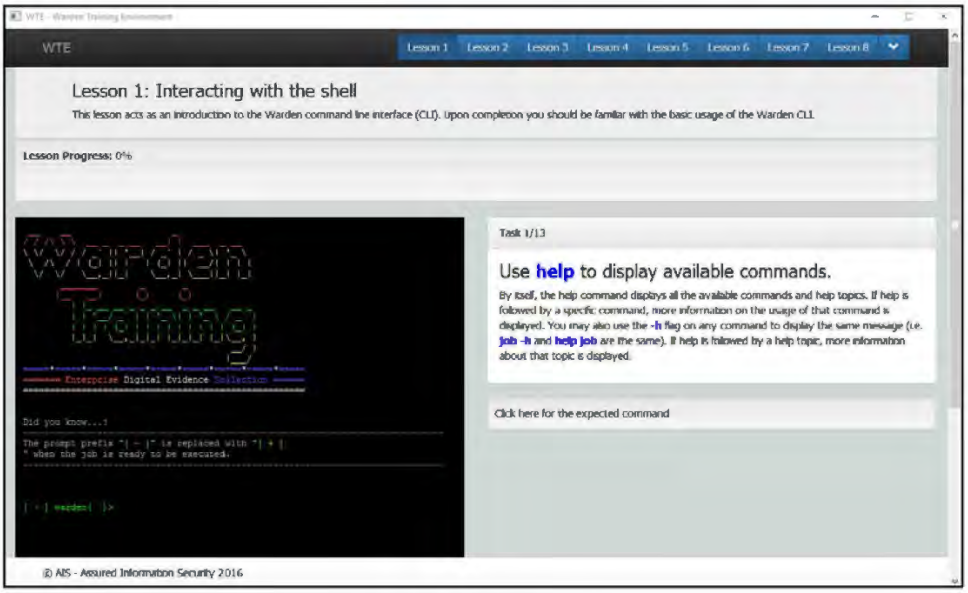

Figure 13: WTE - Lesson 1

Figure 14: WTE - Lesson 4 Completed

3.3 WARDEN Web User Interface

3.3.1 Problem

The CLI interface to WARDEN is an obstacle to operators without previous experience or familiarity with a CLI. The WebUI aims to help these operators become proficient in the use of WARDEN.

3.3.2 Approach

The approach for the WARDEN WebUI required the utilization of well-established web development frameworks for quickly creating user interfaces and data models. Additionally, the team could provide a unified interface to the WARDEN framework to seamlessly present job data

to the operator regardless of the interface whether it be CLI or WebUI (i.e. – jobs created and managed in the CLI are available in the WebUI and vice versa). Much of the WebUI back-end was written using the Python Flask [2] web library and visual elements were developed using the Bootstrap library [3].

The WARDEN WebUI functions very similar to how the WTE functions. The WARDEN WebUI displays input fields that accept operator input that is then reflected to a running session of WARDEN. The WebUI dynamically adapts and tailors the fields that are displayed to the different transports and collectors that are selected when creating a job.

3.3.3 Usage

The WARDEN WebUI can be launched via the shortcut on the system's desktop or through the Start Menu. Launching the WebUI will automatically start a compatible web browser that is included within the release.

The Job Listing page can be accessed through the navigation bar at the top of default page by clicking the **Job List** item.

Figure 15: Job Listing Page

The Job Details page can be accessed by clicking a Job name within the Job List page and the details page has additional information about a Job including its current state, deployment statistics, and pickup statistics.

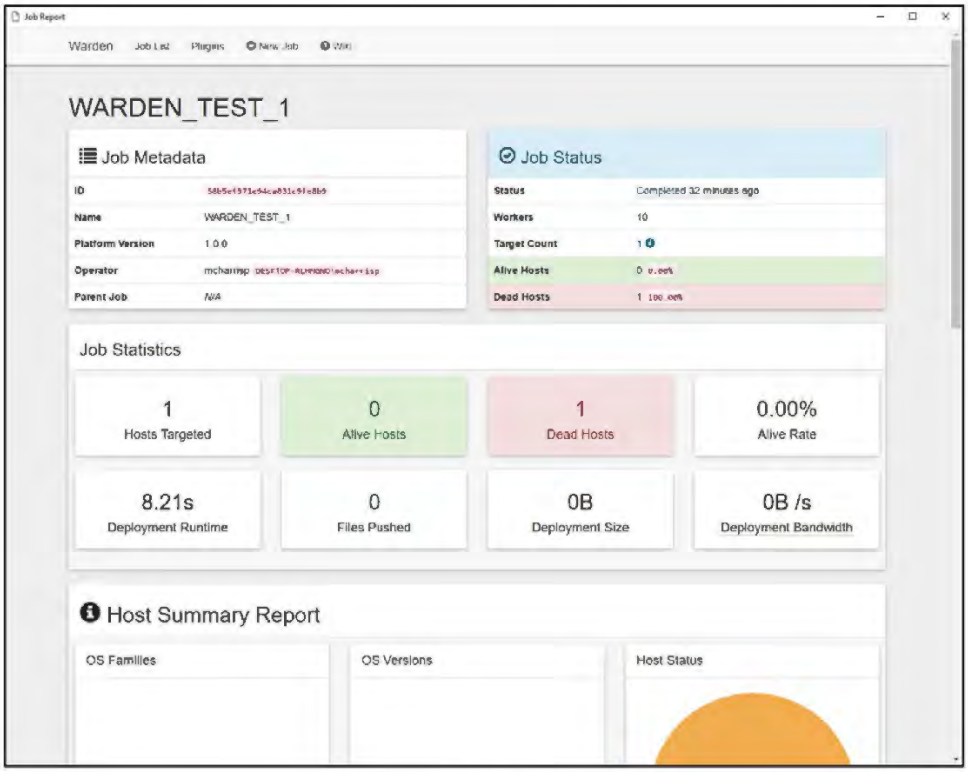

Figure 16: Job Details Page

WARDEN will automatically generate a report after a Job has completed execution if **job.report** is set to **True**. The report is accessible from the WARDEN WebUI interface by navigating to the Job Details page and then clicking the **Report** button.

Assured Information Security, Inc.

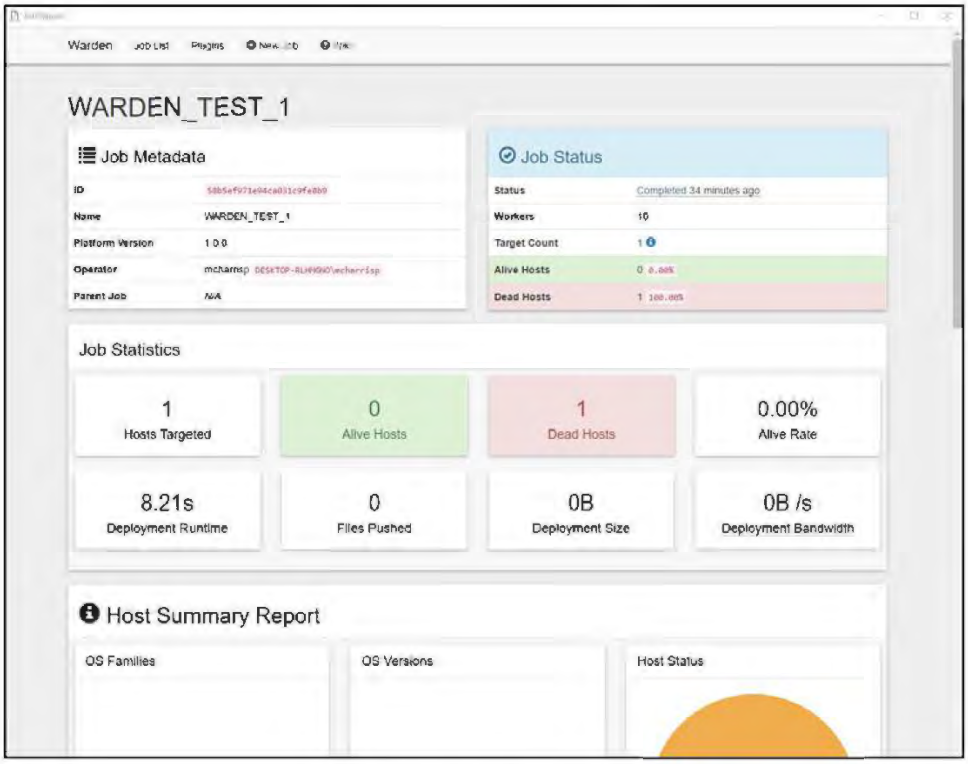

Figure 17: Job Report Page

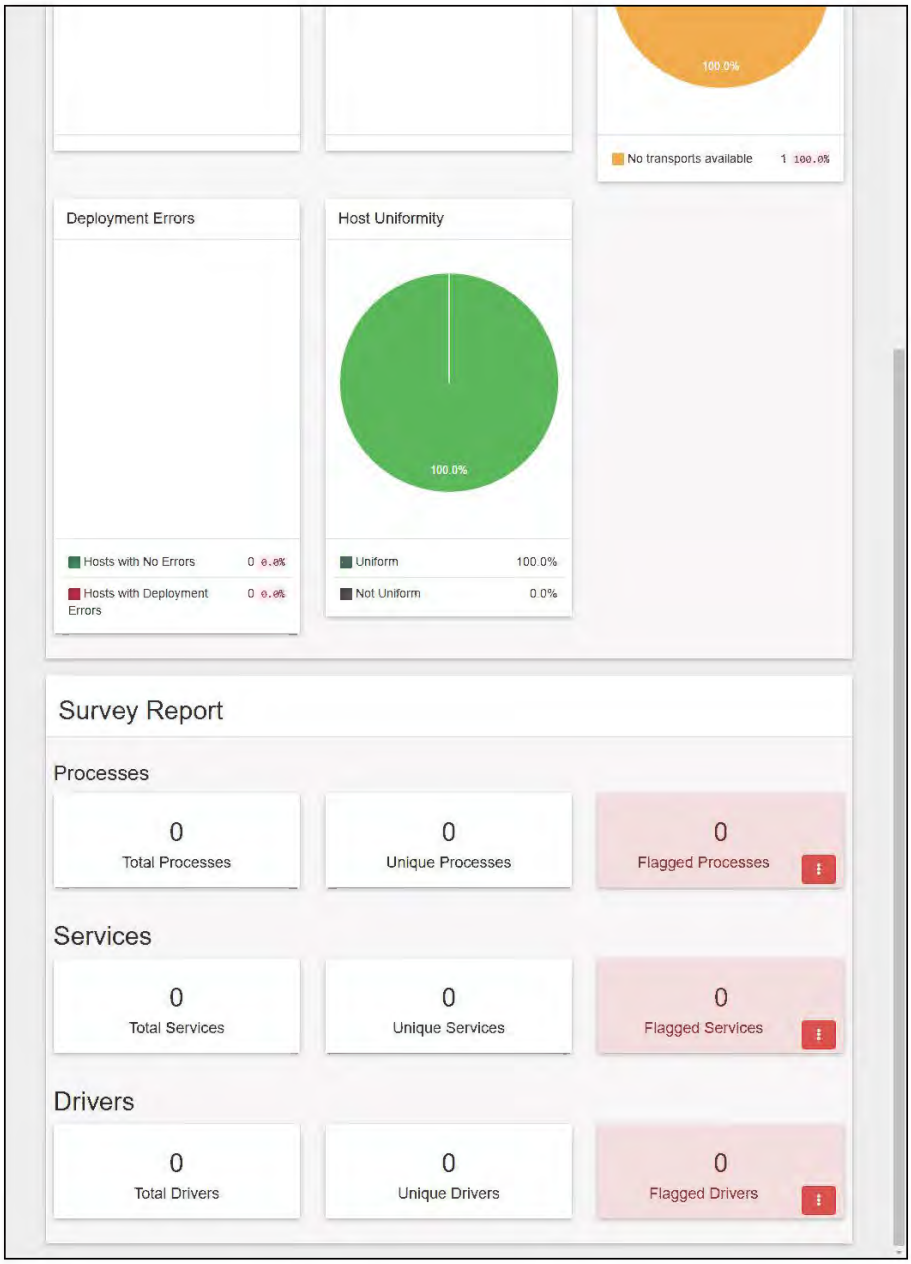

Figure 18: Job Report Page Continued

The generated report will contain additional Job statistics and any plugin-specific analysis that was performed after data was collected. Currently WARDEN will execute the following analysis:

Table 6: Included Collector Analyzers

Plugin	Analysis Description
collectors/autoruns	Identify the number of autorun entries that were found on a remote system. The data consists of total autorun entries, total autorun entries enabled, and the d as well as total number of signed autorun entries. Additionally, each autorun entry will be compared against MekaDB and a report will be generated that contains the number of unique, known, unknown, and blacklisted item hashes.
collectors/certificates	Identify digitally signed certificates installed on a remote system. Data consist of total number of installed certificates as well as total number of unique certificates installed on the system.
collectors/hashfs	Identify unique, known, and unknown files collected by comparing results to the MekaDB. Refer to the **Hash Filesystsem Collector** plugin document for more information. Additionally, a signing report will be generate if the **hashfs.sigcheck** Job option is set to **True** which will show an overview of trusted and untrusted executables.
collectors/hashproc	Identify unique, known, and unknown files collected by comparing results to the MekaDB. Refer to the **Hash Process Collector** plugin document for more information. Additionally, a signing report will be generate if the **hashproc.sigcheck** Job option is set to **True** which will show an overview of trusted and untrusted executables.
collectors/logins	Identify number of logins on a system. Data consists of total attempts which occur locally and remotely along with the number of failed for each. Also includes the number of unique users and IP Address that accessed the host performed remote authentication.
collectors/netstat	Identify total sockets in use on a system. This consists of number of TCP, UDP and other sockets in use. Listening, established and other connections are also listed. Aggregate network connection information across the selected range based on protocol.
collectors/nullpipes	Identify total number null r anonymous (null) pipes on a remote host. Data includes total number of named pipes found on the remote system along with and the total number of null pipes and authorized pipes.
collectors/pii	Identify number of hosts with personally identifiable information (PII) on the filesystem. This data consists of total documents containing PII, and average file age of documents with PII. Also, identifies total number of Social Security numbers and cCredit cCards numbers found along with the number documents and hosts where it was found.
collectors/antigen	Identify the total number of processes that have been maliciously injected into. The data will include the total number of injections discovered, total number

	of hosts with injections and total number of processes with injections. Refer to the ANTIGEN **Collector** documentation for more information about process injections.
collectors/sessions	Identify the total number of active sessions on a remote system. Data will include the total number of active sessions on a remote system along with the number of sessions that are active and the number of user sessions.
collectors/survey	Identify unique process, services, and drivers and attempt to assign a suspicion score to unique items. Refer to the **Survey Collector** plugin document for more information.
collectors/unhidenet	Identify total number of hidden ports on a remote system. Data will include the total number of hidden ports discovered on a remote system. Additionally, network aggregation on connection protocol will be performed if **unhidenet.all** Job option is set to **True**.
collectors/usbdrives	Identify the total number of USB drives currently attached or that have been attached to a remote system. Data consists of total number of currently attached usbUSB drives as well as total number of previously attached number of USB drives.
collectors/verifyproc	Verifies all running processes on a remote system. Data consists of the total number of processes on a remote system as well as the "Top Flagged" executable and "Top Reason" for why an executable was flagged.

The WARDEN web interface provides access to the centralized help system that is available within the WARDEN CLI's **help** command through a Wikipedia-style component. The WARDEN Wiki can be accessed by clicking the Wiki item in the navigation bar at the top of the page.

Assured Information Security, Inc.

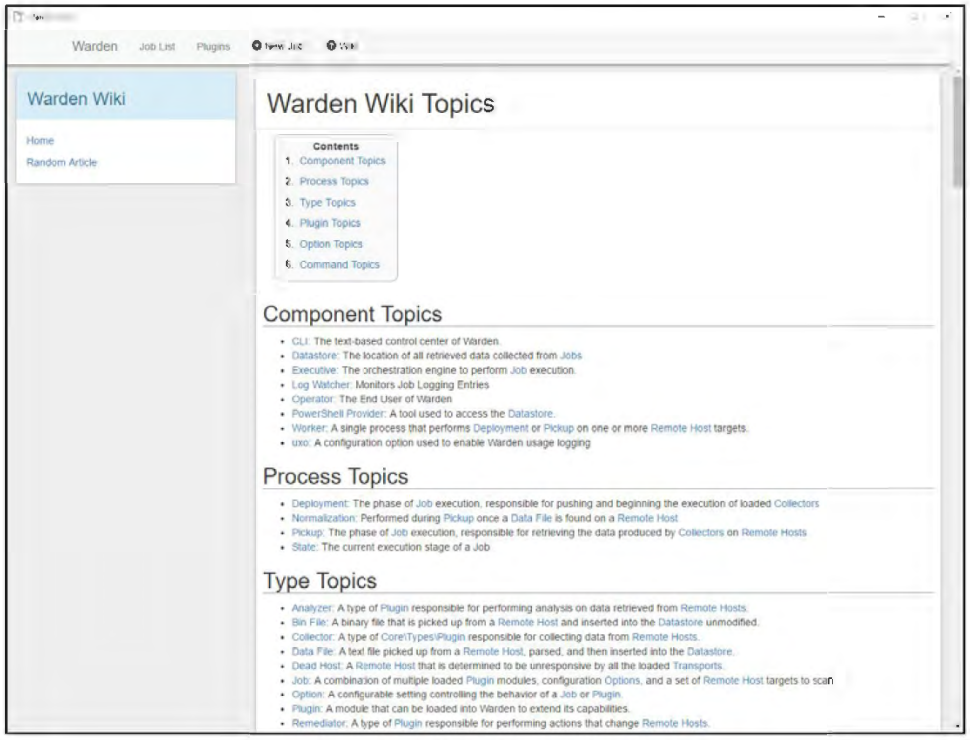

Figure 19: WARDEN Wiki Page

Job setup or building a new job is done by selecting the New Job item in the navigation bar at the top of the UI.

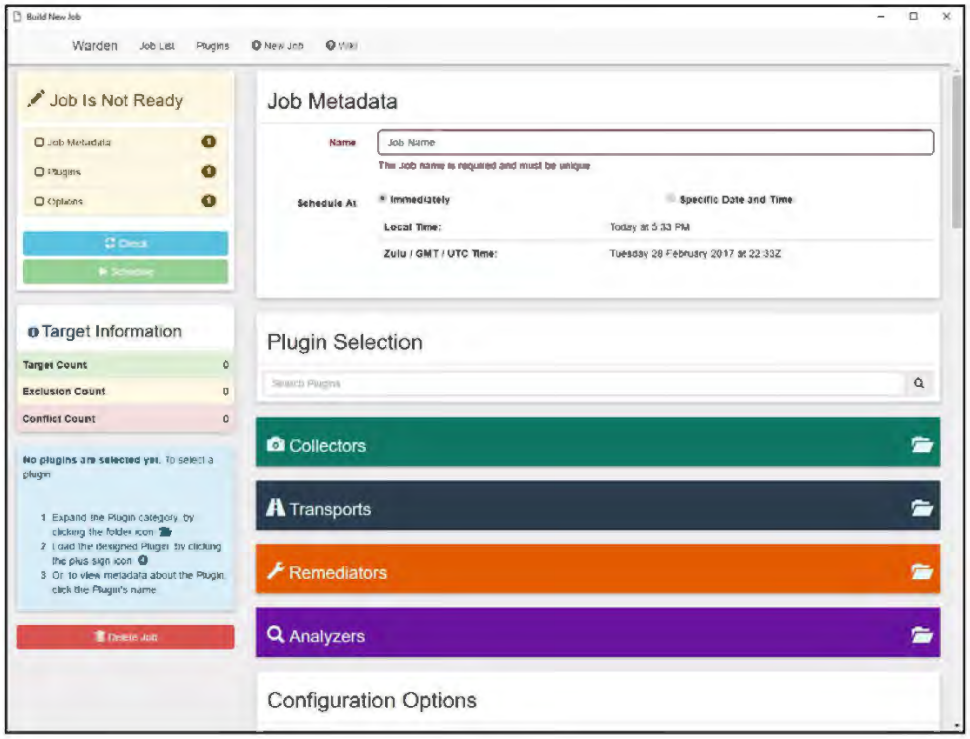

Figure 20: Job Setup

3.4 ANTIGEN

3.4.1 Problem

Incident response is generally a lengthy, error-prone process. This can lead to inaccuracy and a delayed response. Traditionally, a RAM image of a selected host is acquired and analyzed to determine the nature of the attack. Human error is a common problem when using this method because memory contains important artifacts that can be miscorrelated or overlooked entirely when not automated.

The ANTIGEN application is a native Windows® executable that performs memory forensics to detect and alert on generic indicators of compromise. Rather than identifying specific malware samples or detecting certain hash signatures, ANTIGEN enumerates the memory of each process and loaded module and performs deep inspection to detect generic indicators of:

- Hidden and injected modules
- Code modifications
- Hidden processes
- Function hooks

- Weak or modified memory region protection

These five indicators are generic to all malware samples. For example, there are many methods available for malware to inject itself into another process. Rather than search for signatures related to this functionality, ANTIGEN will detect the following indicators in memory, regardless of how a malware sample injected itself.

- Memory regions with Write/Execute permissions
- A hidden module
- Code modifications
- Function hooks

3.4.2 Approach

ANTIGEN has the following workflow:

1. Generate a list of running processes on a remote system
2. Begin running various memory tests
3. Information needed for analysis is copied into ANTIGEN's address space
4. Analyze information gathered and report discrepancies or suspicious evidence

First, ANTIGEN gathers the list of currently running processes. This information is retrieved using the Windows® Tool Help32 API. Because ANTIGEN is linked statically at compile time, these Windows® API calls are unaffected by any current hooks on the system and the results of these calls can be trusted. For each process running, ANTIGEN then gets the list of loaded modules (Dynamically Linked Library (DLL)) and contiguous memory regions using the Toolhelp32 API. With the list of running processes ANTIGEN then begins further manual analysis and parsing of each process only loading into memory the information that it needs.

While each process is being analyzed by ANTIGEN, the required information on the process is loaded in ANTIGEN's address space using the ReadProcessMemory Windows® API call. Once copied in, ANTIGEN begins manually parsing the memory locations of the process and its loaded modules. After the tests have been completed for the process the processes information that was loaded into memory is discarded and any findings on the process are reported and written to a JSON file for further analysis. This method for memory analysis greatly reduces the overhead and efficiency of ANTIGEN while running on a remote system. To reduce complexity and runtime of ANTIGEN any findings uncovered by ANTIGEN are not correlated to other running processes, therefore multiple processes in a report may show similar findings.

ANTIGEN generates a single JSON report that is, by default, printed to the standard output stream. The report can instead be written to a file with the **–o/--output** command line argument (see table 8 for additional arguments).

The output format is a single JSON object containing a list of processes that were scanned. Each process then contains metadata and any detected malicious indicators of compromise.

ANTIGEN operates on the process-level. By design, ANTIGEN will alert on duplicate malicious indicators if they exist across several processes. If, for example, a malware sample injects a hidden module into all the processes on the system, the injected hidden module will be reported in each of the process objects.

ANTIGEN does not de-duplicate indicators because process memory, in most circumstances, is copy-on-write. That is, when a change occurs to a loaded module's code, the change is not propagated to all the processes on the system that have that module loaded. Rather, the change is localized to the one process. Therefore, if a malware sample infects multiple processes by hooking a function, each infected process will contain the function hook.

3.4.2.1 Hidden Module Detection

It is very common for malware to hide any modules that it has loaded in order to hide functionality or persistence within a process. Hidden modules can be achieved through either reflective module injection or modifying the Windows® process loaded module lists. The detected hidden module will be processed by the other ANTIGEN analyzers regardless of the method used to hide the module.

3.4.2.2 PE Header Detection

Each process in a Windows® system has a virtual address descriptor (VAD) tree, which defines all allocated and reserved memory within the process. The VAD tree contains critical information such as loaded modules, reserved memory regions, and memory allocation permissions. ANTIGEN primarily uses each process's VAD tree to enumerate allocated memory. ANTIGEN will use PE header detection to find dynamically loaded modules within a process by scanning each VAD entry for a valid PE header. A VAD entry that begins with a valid PE header and does not belong to a known loaded module will be flagged as a hidden module.

Figure 21: ANTIGEN Hidden Modules - PE Header Detection

3.4.2.3 Inconsistent Loaded Module Lists

Each Windows® process contains a Process Environment Block (PEB) which stores three lists of loaded modules. Hidden modules that were loaded using the legitimate Windows® API will typically be removed from one of these lists. ANTIGEN can detect this by manually parsing the

PEB loaded module lists and then correlating the results. Modules that do not exist within all three lists will be flagged as hidden.

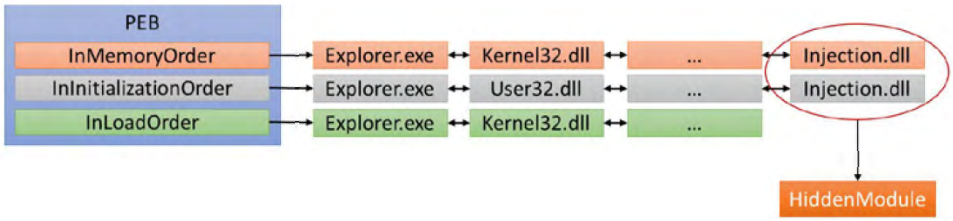

Figure 22: Hidden Modules - Inconsistent Loaded Module Lists

ANTIGEN will create and write a **HiddenModule** object whenever a hidden module has been detected (see section 4.5.9.1).

3.4.2.4 Hidden Process Detection

Windows® processes can be hidden by modifying the kernel list of running processes. A hidden process will still be scheduled to execute and receive processor time, however, the hidden process will not appear in applications that list currently running processes. For example, a hidden process will not appear in the Windows® Task Manager. A hidden process can only be detected if the scanning tool knows the unique process identifier (PID) for the hidden process. Hiding a process, like hiding a module, is commonly performed to obfuscate or conceal malware execution.

ANTIGEN detects hidden processes by gathering a list of running processes using two different methods and then comparing the results. First, ANTIGEN uses the Windows® process API to list the running processes, the list provided will not contain hidden processes. Then, ANTIGEN will perform a brute force scan of all running processes by checking if each possible PID is currently executed. The brute force scan will return processes that are currently hidden. Next, the list of running processes is retrieved using the Windows® process API. Finally, the three lists are compared and hidden processes are detected if they only exist in the brute force scan.

3.4.2.5 Code Modification Detection

Malware can hijack an existing and trustworthy process and co-opt its execution. Malware will do this through the method of process hollowing where:

1. A selected process is paused
2. The selected process executable memory is overwritten in place with the malware's code
3. The selected process is resumed, causing the malware's payload to execute

Process hollowing and code modifications in general are detected by ANTIGEN. ANTIGEN can correlate code in memory to its on-disk origin. Then, for each code region in memory, ANTIGEN compares the in-memory code against the code as it exists on disk. The comparison algorithm

takes into account legitimate code modifications that Windows® performs on all modules and processes, such as:

- Resolved imported functions
- Relocated code

If the two code regions differ drastically, more than 10%, the code in memory has been modified and ANTIGEN will create a **CodeModification** object and write it to the host scan report (see section 4.5.9.7).

Figure 23 shows how the code modification detection operates.

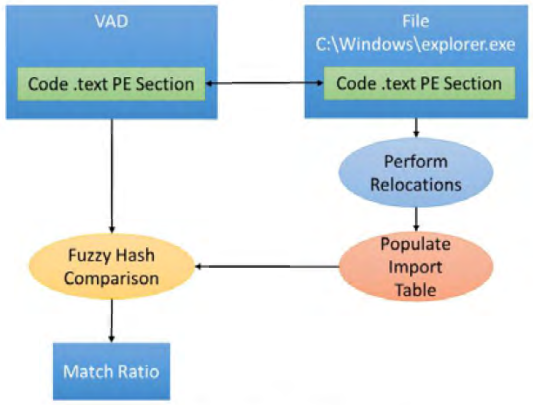

Figure 23: Code Modification Detection

1. The loaded module or executable is identified in memory.
2. The file in memory is traced to its on-disk origin.
3. All code sections are pulled out from both the loaded module and the on-disk PE header.
4. Relocations are performed and the import address table is populated for the file on disk. This step is performed to make sure that relocations and the import address table do not affect the match ratio result.
5. Both code sections are processed to produce a fuzzy hash.
6. The fuzzy hashes are compared against each other and a match ratio is calculated.

3.4.2.6 Function Hook Detection

Malware will commonly attempt to modify the behavior of some common Windows® functions with the intent of changing a function's input or output. For example, malware can hide registry keys by modifying the results of the **RegOpenKey** and **RegEnumKey** Windows® API functions, which are used to open a Registry key and list all child Registry keys, respectively. Malware will modify the inputs or outputs to functions by using function hooking.

Function hooking is the process of inserting custom code into a pre-existing function to modify its behavior or redirect control flow. The most common use of function hooking in malware is control flow redirection, which passes execution to a third-party, and usually malicious, code block, thereby bypassing the original function's body. This way, an application may try to open a Registry key, with **RegOpenKey**, but instead, the malware will gain control and run an arbitrary payload before returning control back to the application. There are several methods of function hooking through redirection, listed below:

- Interrupt-based hooks
- Unconditional jump hooks
- Push and return hooks

These three hooks are all performed at the processor instruction level. When a hook is detected, ANTIGEN will create a **FunctionHook** object and then write it to the report (see section 4.5.9.3).

Figure 24 shows the process used to detect exported functions that are hooked.

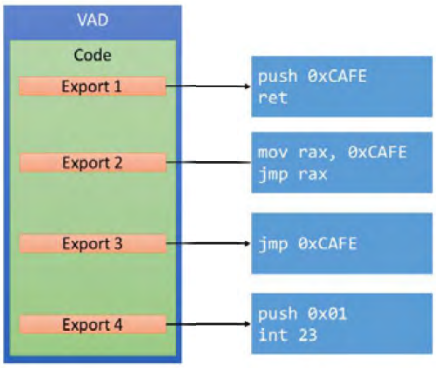

Figure 24: Function Hook Detection

The left side of the diagram shows that a module in memory exports four functions. The right side shows the disassembled code at each function's entry point address. Each of the four exported functions are hooked in this example. Between each there is an unconditional jump.

3.4.2.7 Weak or Modified Memory Protection Detection

Each memory region within a process has permissions that control how the memory can be accessed. The memory region permissions are:

- Read (R) - the memory region can be read
- Write (W) - the memory region can be modified
- Execute (X) - the memory region can be executed as code

Memory regions can have any combination of the above permissions. It is generally accepted within the host-based security community that executable code regions should not be modifiable, that is, regions that can be executed as code should not be able to be changed. These regions that are readable, writable, and executable are denoted as RWX memory regions. ANTIGEN detects the following memory region permission anomalies.

Table 7: ANTIGEN: Detected Memory Region Permission Anomalies

Anomaly	Description
Allocated with RWX	The region was allocated as having read, write, and execute permission
Is Currently RWX	The region currently has read, write, and execute permission
Is Currently Executable and Permissions have Changed	The region is executable and its permissions have changed since it was originally allocated.

Hooking a function requires a modification to the code region of a module, which is, by default, not writable. Therefore, a function hook in a region of memory that is reported as both writable and executable is very suspicious and ANTIGEN will alert on both indicators.

ANTIGEN detects these anomalies by walking each process VAD tree and querying for the region's initial allocated and current protection permissions. For each VAD entry that ANTIGEN finds at least one of the memory protection anomalies, ANTIGEN will create a **VadViolation** object and write it to the report (see section 4.5.9.9). Figure 25 shows how two VAD entries that had their protection permissions changed during process execution. The first entry shows that the permissions changed from **PAGE_EXECUTE_READ** (read and execute) to **PAGE_EXECUTE_READWRITE** (read, write, and execute), which will generate a VAD violation due to an executable page gain write permission. The second entry shows that the permissions changed from **PAGE_READ** (read) to **PAGE_READWRITE** (read and write). The second entry does not have execution permission so the entry is not flagged.

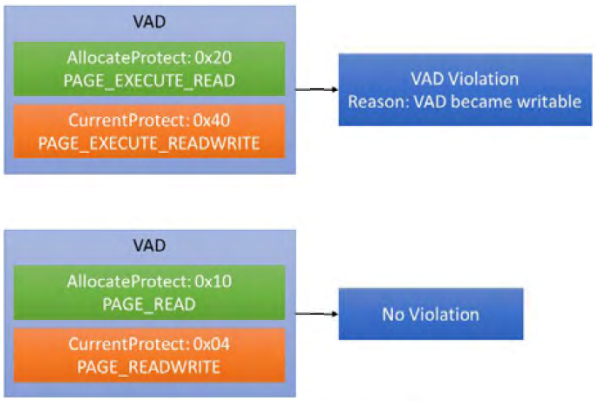

Figure 25: VAD Violation Algorithm

3.4.3 Analysis of ANTIGEN Output

The AIS team approached the analysis of the problem by first referencing materials that outline well-established malware attributes. Once identified, the team studied numerous types of malware to determine the validity of the attributes. The next step in the analysis was to develop software to identify these attributes and create a method to measure the effectiveness of the software. Because the identified attributes are potentially available in non-malicious processes, the team used a scaling threshold system in order to "hone-in" on the test levels to accurately identify malware while mitigating false positives. The solution was an iterative adjustment. This method was performed on images containing malware as well as non-malicious images until a minimum of false positives and false negatives were identified.

ANTIGEN is designed to execute on a compromised system, alongside malware that is trying to hide itself or modify the system in some way. Therefore, ANTIGEN does not perform any analysis on the compromised system itself. Rather, ANTIGEN will produce a dataset that should be analyzed offline on a partitioned and trusted system. As a result, ANTIGEN can produce a very large dataset for a given system.

There are two PowerShell cmdlets to analyze ANTIGEN output generated as part of a WARDEN Job: **Invoke-AntigenStdDevAnalyzer** and **Invoke-AntigenStatistics**.

3.4.4 Standard Deviation Calculation

Invoke-AntigenStdDevAnalyzer will perform standard deviation calculations to find the most suspicious hosts that are outliers in the amount of malicious artifacts that were detected. **Invoke-AntigenStdDevAnalyzer** is designed to run on many ANTIGEN host scans and is intended to quickly narrow down the most suspicious hosts.

3.4.5 Invoke-AntigenStatistics

Using the narrow list of remote systems that contain the most suspicious information, the **Invoke-AntigenStatistics** analyzer can be used to perform a more detailed and in-depth analysis of the remote systems. **Invoke-AntigenStatistics**, unlike **Invoke-AntigenStdDevAnalyzer**, should only be used on a limited number of remote systems.

Invoke-AntigenStatistics is able to correlate modules across processes and label data appropriately.

3.4.6 Usage

A process consists of three primary parts: (1) control structures, (2) executable code, and (3) raw data. The control structures govern how the process executes and the environment that is maintained during execution. Under normal operating situations, these control structures are typically only manipulated by the Windows® kernel and associated drivers. However, due to lack of security, user-land processes can manipulate these critical components. Modifications can be detected by checking for inconsistencies in other structures and correlating the discrepancies. Executable code is typically static and not significantly altered, if at all, during execution. Changes can be detected by comparing the code section in memory to the section it corresponds to on disk. In addition, checking if a code section can be written to and modified can be accomplished by verifying the permissions on the code regions. Raw data sections, however, will change during execution, varying on what is being stored in memory. Just as code sections are typically not writable, data sections are typically not executable. A write-execute region is uncommon and considered dangerous if present at all. In incident response, identification and classification of malware is necessary to properly respond to and quarantine a malicious process.

The resulting ANTIGEN capability runs on Windows® systems and accepts the following command line arguments:

Table 8: ANTIGEN Command Line Arguments

Argument	Description
-s/--silent	Do not write any status messages to the console
-v/--verbose	Print verbose status messages to the console
-o/--out PATH	Write the ANTIGEN JSON report to the file located at PATH
-p/--pretty	Write the output JSON file so it is more legible for human consumption
-V/--version	Print ANTIGEN version information and exit.

ANTIGEN will attempt to legitimately escalate its privileges to SYSTEM level so that it can scan processes that are running at a higher privilege level than the current user. Therefore, ANTIGEN should ideally be launched with SYSTEM privileges. Sufficient privileges can be granted to the ANTIGEN process by scheduling a Windows® task or running ANTIGEN through a tool like

PSExec[1]. If, however, it isn't possible to execute ANTIGEN as SYSTEM for a particular environment or setup, ANTIGEN must be run with Administrator privileges instead. Running ANTIGEN with Administrator privileges can be done by performing one of the following:

- Right-clicking the ANTIGEN executable and selecting **Run As Administrator**
- Starting a **cmd.exe** or **PowerShell** session as Administrator, by launching either with right-clicking and selected **Run As Administrator**, and then launching ANTIGEN from within the session.

The ANTIGEN capability, while able to run standalone, has been fully integrated into WARDEN. WARDEN contains the **collectors/antigen** Plugin that wraps the ANTIGEN application for use within a Job. It is recommended to execute ANTIGEN with the **transports/task** or **transports/wpt** Plugins, which will cause ANTIGEN to execute with the SYSTEM privilege. The **transports/mswmi** Plugin is also capable of launching ANTIGEN; however, ANTIGEN will not run as SYSTEM and, therefore, won't scan SYSTEM processes.

WARDEN will automatically detect whether each remote system is 32 or 64-bit. ANTIGEN will then transparently push the correct executable, antigen32-2.0.exe or antigen64-2.0.exe, depending on the remote system's architecture.

Collected ANTIGEN data will be saved to the antigen collection in the WARDEN Provider. When queried in the WARDEN Provider, the antigen collection will output Antigen**V2** objects (see the WARDEN PowerShell Interface document for more details).

ANTIGEN creates a JSON file containing a single JSON object, a **HostScan** object.

3.4.6.1 HostScan Object

Table 9: HostScan Members

Name	Type	Description	Mandatory	Can be null?
Processes	List<Process>	List of scanned processes	Yes	No

3.4.6.2 Process Object

Each **Process** object maps to a process that was executing on the system when ANTIGEN executed.

[1] https://technet.microsoft.com/en-us/sysinternals/bb897553.aspx

Assured Information Security, Inc.

Table 10: Process Members

Name	Type	Description	Mandatory	Can be null?
CodeModifications	List<CodeModification>	List of code modifications	No	No
CommandLine	string	The command line arguments used to launch the process	Yes	No
HiddenModules	List<HiddenModule>	List of hidden modules	No	No
FunctionHooks	List<FunctionHook>	List of function hooks	No	No
IsHidden	bool	Whether the process is hidden or not	Yes	No
IsWow64	bool	Whether the process is a WoW64 process or not	Yes	No
FileName	string	The process short image name	Yes	No
ParentPid	int	Parent Process Identifier	Yes	No
ParentProcessImageName	string	Parent Process name	Yes	No
FilePath	string	Path to the process on disk	No	No
Pid	int	UPI	Yes	No
UserName	string	The user that the process is running as	Yes	No
VadViolations	List<VadViolation>	List of VAD protection violations	No	No
VirtualAddress	int	The base virtual address of the process module	No	No
VirtualSize	int	The virtual size, in bytes, of the loaded process module	No	No
AntigenWarnings	List<string>	A list of warning that were encountered during processing, such as memory copy errors	No	No
FileInfo	FileInfo	The collected fie information of the process on disk	No	Yes

3.4.6.3 Sample Process JSON

```
{
    "Pid": 100,
    "ParentPid": 200,
    "ParentProcessImageName": "winlogon.exe",
    "UserName": "ComputerName\\Username",
    "FileName": "explorer.exe",
    "FilePath": "C:\\Windows\\System32\\explorer.exe",
    "VirtualAddress": 65536,
    "VirtualSize": 4096,
    "HiddenModules": [ ... ],
    "FunctionHooks": [ ... ],
    "CodeModifications": [ ... ],
    "VadViolations": [ ... ],
```

```
    "DynamicallyAllocatedCode": [ ... ],
    "PackedCode": [ ... ],
    "IsHidden": false,
    "IsWow64": false,
    "CommandLine": "C:\\Windows\\System32\\explorer.exe",
    "FileInfo": { ... }
}
```

3.4.6.4 Module Object

Module objects map to a loaded module in memory for a specific process.

Table 11: Module Members

Name	Type	Description	Mandatory	Can be null?
FileName	string	The module short image name	Yes	No
FilePath	string	Path to the module on disk	Yes	No
VirtualAddress	int	The base virtual address of the module	Yes	No
VirtualSize	int	The virtual size, in bytes, of the loaded module	Yes	No
FileInfo	FileInfo	The file information of the module on disk	No	Yes
OriginLists	list<string>	The source lists that the module was detected through	Yes	No

3.4.6.5 Sample Module JSON

```
{
    "FileName": "ws2_32.dll",
    "FilePath": "C:\\Windows\\System32\\ws2_32.dll",
    "VirtualAddress": 1087312,
    "VirtualSize": 1048576,
    "FileInfo": { ... },
    "OriginLists": ["Peb.InMemoryOrder"]
}
```

3.4.6.6 VAD Object

A VAD object is a contiguous region of memory that has uniform memory protection. Each contiguous memory region is stored inside of the process VAD tree.

Table 12: VAD Members

Name	Type	Description	Mandatory	Can be null?
VirtualAddress	int	The base virtual address of the VAD region	Yes	No
VirtualSize	int	The virtual size, in bytes, of the VAD region	Yes	No
CurrentProtection	VadProtection	The current protection permissions	Yes	No
AllocateProtection	VadProtection	The original allocated protection permissions	Yes	No

IsAnonymousMemory Region	Bool	Whether the VAD is backing a loaded **Module**	Yes	No
MemoryState	VadState	The current memory region state	Yes	No
MemoryType	VadType	The type of memory region	Yes	No
SectionName	string	The memory region's section name, which, if the region is a mapped file on disk, will be the file path.	Yes	No

The **CurrentProtection** and **AllocateProtection** members are bitmaps for Windows® memory protection constants[2]. The **state** member is an integer bitmap for Windows® memory state constants[3]. The **type** Member is an integer bitmap for Windows® memory type constants[4].

3.4.6.7 VadProtection Object

The WARDEN Provider ANTIGEN object wraps the VAD protection bitmap into an object with convenience accessor to remove the need to perform bitwise operations in the Provider.

Table 13: VadProtection Members

Name	Type	Description	Mandatory	Can be null?
CanExecute	Bool	Whether the VAD entry has executable permission	Yes	No
CanRead	bool	Whether the VAD entry has read permission	Yes	No
CanWrite	bool	Whether the VAD entry has write permission	Yes	No
IsCopyOnWrite	bool	Whether the VAD entry is copy-on-write	Yes	No
IsRWX	bool	Whether the VAD entry has read, write, and execute permission	Yes	No

3.4.6.8 VadState Object

The WARDEN Provider ANTIGEN object wraps the VAD state bitmap into an object with convenience accessor to remove the need to perform bitwise operations in the Provider.

Table 14: VadState Members

Name	Type	Description	Mandatory	Can be null?
IsCommitted	bool	Indicates committed pages for which physical storage has been allocated, either in memory or in the paging file on disk	Yes	No

[2] https://msdn.microsoft.com/en-us/library/windows/desktop/aa366786%28v=vs.85%29.aspx
[3] https://msdn.microsoft.com/en-us/library/windows/desktop/aa366775%28v=vs.85%29.aspx
[4] https://msdn.microsoft.com/en-us/library/windows/desktop/aa366775%28v=vs.85%29.aspx

IsFree	bool	Indicates free pages not accessible to the calling process and available to be allocated	Yes	No
IsReserved	bool	Indicates reserved pages where a range of the process's virtual address space is reserved without any physical storage being allocated	Yes	No

3.4.6.9 VadType Object

The WARDEN Provider ANTIGEN object wraps the VAD state bitmap into an object with convenience accessor to remove the need to perform bitwise operations in the Provider.

Table 15: VadType Members

Name	Type	Description	Mandatory	Can be null?
IsMapped	bool	Indicates that the memory pages within the region are mapped into the view of a section	Yes	No
IsMappedImage	bool	Indicates that the memory pages within the region are mapped into the view of an image section	Yes	No
IsPrivate	bool	Indicates that the memory pages within the region are private (that is, not shared by other processes)	Yes	No

3.4.6.10 Sample VAD Object JSON

The following sample includes the Windows® constant symbols for the bitmap members in the comments: **currentProtect**, **allocateProtect**, **state**, and **type**.

```
{
      "VirtualAddress": 198472,
      "VirtualSize": 928,
      "CurrentProtection": 64,            // PAGE_EXECUTE_READWRITE
      "AllocateProtection": 64,           // PAGE_EXECUTE_READWRITE
      "IsAnonymousMemoryRegion": false,
      "MemoryState": 4096,                // MEM_COMMIT
      "MemoryType": 16777216              // MEM_IMAGE
}
```

3.4.7 Module PE Header Types

Each **Module** loaded into memory, including the **Process** object, includes the Portable Executable (PE) header. ANTIGEN parses the PE header for each loaded **Module** and running **Process**. Unless noted, all addresses in the PE header are relative virtual addresses (RVA) which are virtual addresses that are relative to the base address of the **Module** (**Module.baseAddress** and **Process.baseAddress**). Using the PE header in memory, ANTIGEN can detect changes and modifications to the process and loaded modules.

3.4.7.1 PEImageHeader Object

The main PE header, **PEImageHeader**, contains top-level metadata about the module.

Table 16: PEImageHeader Members

Name	Type	Description	Mandatory	Can be null?
BaseAddress	int	The preferred base virtual address when loaded into memory	Yes	No
BaseOfCode	int	RVA to the primary code section	Yes	No
CRC32	string	The calculated CRC32 value of the entire PE header	Yes	No
PEExportCount	int	The number of exported symbols that the PE header contains (**PEImageExport** objects)	Yes	No
PEImportCount	int	The number of import modules that the PE header contains.	Yes	No
Is64Bit	bool	Whether the module is built for 64-Bit systems	Yes	No
IsManagedAssembly	bool	Whether the module is a managed (.Net) assembly	Yes	No
MajorOsVersion	int	The major operating system version targeted by the module	Yes	No
MinorOsVersion	int	The minor operating system version targeted by the module	Yes	No
Name	string	The module name, which is only valid for modules that contain at least one **PEImageExport** object	Yes	No
PeSectionCount	int	The number of sections that the PE header contains (**PEImageSection** objects)	Yes	No
SizeOfCode	int	Virtual size, in bytes, of the primary code section	Yes	No
SizeOfHeaders	int	The virtual size, in bytes, of the entire PE header	Yes	No

The **MajorOsVersion** and **MinorOsVersion** members correspond to the Windows® kernel version for which the module was built[5].

3.4.7.2 Sample PEImageHeader Object JSON


```
{
    "BaseOfCode": 4096,
    "SizeOfCode": 256,
    "SizeOfHeaders": 4096,
    "BaseAddress": 65536,
    "MajorOsVersion": 5,
    "MinorOsVersion": 2,
    "PESectionCount": 3,
```


[5] https://msdn.microsoft.com/en-us/library/windows/desktop/ms724832%28v=vs.85%29.aspx

```
    "PEImportCount": 0,
    "PEExportCount": 24,
    "Is64Bit": false,
    "IsManagedAssembly": false,
    "Name": "ntdll.dll",
    "CRC32": "3e7bace9"
}
```

3.4.7.3 PEImageSection

A PE section header, **PEImageSection** object, describes a contiguous memory region that will be mapped into memory when the module is loaded. A section can either contain data, executable code, or both.

Table 17: PEImageSection Members

Name	Type	Description	Mandatory	Can be null?
SectionCharacteristics	int	Section attributes and types bitmap	Yes	No
PESectionName	string	The section name	Yes	No
PhysicalAddress	int	Physical address of the section content on disk	Yes	No
PhysicalSize	int	Physical size, in bytes, of the section content on disk	Yes	No
VirtualAddress	int	RVA to the section content	Yes	No
VirtualSize	int	The virtual size, in bytes, of the section content	Yes	No

The **Characteristics** field is a bitmap of PE image section characteristic constants[6].

3.4.7.4 Sample PEImageSection JSON

```
{
    "VirtualAddress": 4096,
    "VirtualSize": 9820,
    "PhysicalAddress": 1024,
    "PhysicalSize": 9860,
    "PESectionName": ".text",
    "SectionCharacteristics": 1610612768 // IMAGE_SCN_CNT_CODE
                                         // IMAGE_SCN_MEM_EXECUTE
                                         // IMAGE_SCN_MEM_READ
}
```

[6] https://msdn.microsoft.com/en-us/library/windows/desktop/ms680341%28v=vs.85%29.aspx

3.4.7.5 PEImageExport

A **PEImageExport** object describes a function or symbol that is exported for public use. Typically, malware will hook or modify exports to gain access to control flow and manipulate a process execution and/or behavior. Exported symbols can be exported by name or by forwarding. Forwarded exports allow for another function to be called instead of the exported function. This is useful in situations where an API has changed but the author doesn't want to hinder backwards compatibility.

Table 18: PEImageExport Members

Name	Type	Description	Mandatory	Can be null?
EntryPoint	int	RVA to the symbol's entry point; only present if the symbol is exported by name	No	No
ExportForwarder	string	The name of the forwarded symbol, in the format of "module.symbol"; only present if the symbol is a forwarded export	No	No
SymbolName	string	The symbol name; only present if the symbol is exported by name	No	No
FunctionOrdinal	int	The unique symbol ordinal	Yes	No

3.4.7.6 Sample PEImageExport JSON

```
// Export by name
{
      "SymbolName": "LoadLibraryW",
      "EntryPoint": 2983,
      "FunctionOrdinal": 92,

}

// Export by ordinal
{
      "EntryPoint": 283,
      "FunctionOrdinal": 12
}

// Forwarded export
{
      "FunctionOrdinal": 23,
      "ExportForwarder": "kernel32.LoadLibraryW"
}
```

3.4.8 Malicious Indicator Objects

3.4.8.1 HiddenModule Object

ANTIGEN will create a **HiddenModule** object for each hidden module that is detected in memory. The **HiddenModule** object is specific to a single process and contains a parsed PE header and backing VAD entry.

Table 19: FunctionHook Members

Name	Type	Description	Mandatory	Can be null?
PEHeader	PEImageHeader	The hidden module's PE header information	Yes	No
Vad	VAD	The backing VAD object	Yes	No

The **Vad.SectionName** member may contain the path to the hidden module on disk.

3.4.8.2 Correlating Common Hidden Modules across Processes

The same module may be hidden in multiple processes. Further analysis can detect this by comparing the PE header's **crc32** field across all hidden modules in a **HostScan**.

3.4.8.3 FunctionHook Object

Detected function hooks in memory are written to the report as **FunctionHook** objects. Each **FunctionHook** describes where the hook is, the hook source, and where control flow transfers to the hook target.

Table 20: FunctionHook Members

Name	Type	Description	Mandatory	Can be null?
Disassembly	string	The disassembled assembly code of the hook	Yes	No
PEExport	PEImageExport	The exported function that is hooked	Yes	No
HookMethod	string	The hooking method detected	Yes	No
Module	Module	The source module that contains the hook	Yes	No
TargetHexFingerprint	string	A 40-character hex signature of the memory region that is the hook target (the memory region jumped to by the hook)	Yes	Yes
HookTarget	HookTarget	The target VAD and/or **Module** that receives control flow	Yes	No

The **Target** member stores the target **VAD** or **Module** object that receives control flow.

Table 21: HookTarget Members

Name	Type	Description	Mandatory	Can be null?
Module	Module	The Module that contains the hook target	Yes	Yes

Vad	VAD	The VAD entry that contains the hook target	Yes	Yes
VirtualAddress	int	The virtual address that is immediately jumped to	No	No

The **Method** member signifies how the function is being hooked. Based on the **Method** member value, additional members may be present in the **FunctionHook** object.

Table 22: FunctionHook: Method-specific Members

Method	Method Description	Member Name	Type	Member Description
int	Interrupt instruction hook	InterruptVector	int	The interrupt vector used
		HookCode	int	The possibly unique hook code used to determine which function was hooked
push-ret	push/ret instruction combination hook	N/A	N/A	N/A
jmp	An unconditional jump instruction hook	N/A	N/A	N/A

3.4.8.4 Identifying Common Hooks across Processes

The **FunctionHook** object's **signature** member can be used to correlate hooks that span multiple processes. By grouping all hook signatures that are exactly equal together, unique hooks can be found that exist in multiple processes.

3.4.8.5 Memory-resident Payloads

Memory-resident payloads are blocks of code that did not originate from disk. Rather, the executable memory was allocated at runtime and was populated from an origin other than disk, such as a network payload. Detecting memory-resident payloads can be done by checking if the following is true about a **FunctionHook** object.

- The **target.module** member is **null**- And -
- The **target.vad.sectionName** is an empty string

3.4.8.6 Sample JSON For Memory Resident Payloads

```
// interrupt instruction hook
{
      "HookMethod": "int",
      "Disassembly": "0x00ff8014 push 20\n0x00ff8019 int 48"
      "TargetHexFingerprint": "ff01ba93ee9f0e9e9e9e9e9e9e9e9",
      "Module": { ... },
      "PEExport": { ... },
      "HookTarget": {
            "Vad": { ... },
            "Module": null
      },
      "InterruptVector": 72,
      "HookCode": 32
```

```
}

// push/ret instruction combination hook
{
      "HookMethod": "push-ret",
      "Disassembly": "0x0062ba70 push e09df\n0x0062ba75 ret",
      "TargetHexFingerprint": "ff0191103b8a720bcc9a9bdeec8f",
      "Module": { ... },
      "PEExport": { ... },
      "HookTarget": {
            "Vad": { ... },
            "Module": { ... },
            "VirtualAddress": 920031
      },
}

// Unconditional jump instruction hook
{
      "HookMethod": "jmp",
      "Disassembly": "0xff019beda jmp 2caf5",
      "TargetHexFingerprint": "91ba830ed819c1b37ba0c8e01ba8",
      "Module": { ... },
      "PEExport": { ... },
      "HookTarget": {
            "vad": null,
            "module": null,
            "virtualAddress": 183029
      }
}
```

3.4.8.7 CodeModification Object

ANTIGEN will create a **CodeModification** object for each memory region that contains modified code. The **CodeModification** object contains the modified PE section, source **Module**, and a comparison match ratio.

Table 23: CodeModification Members

Name	Type	Description	Mandatory	Can be null?
PESection	PEImageSection	The code section that has been modified	Yes	No
Module	Module	The Module that the code section belongs to	Yes	No
BinarySimilarity	int	The calculated match between the on-disk and in-memory section, **100** being an exact match (no modifications) and **0** being no similarity.	Yes	No

3.4.8.8 Sample CodeModification Object JSON

```
{
      "PESection": { ... },
      "Module": { ... },
      "BinarySimilarity": 72          // 72% similarity, 28% difference
}
```

3.4.8.9 VadViolation Object

ANTIGEN creates a **VadViolation** object for each VAD region that is found that contains weak or modified protection permissions. Each **VadViolation** object contains the backing VAD entry, whether the region was allocated with RWE permissions or currently has them, and a VAD entry signature.

Table 24: VadViolation Members

Name	Type	Description	Mandatory	Can be null?
Vad	VAD	The VAD entry	Yes	No
IsRwx	bool	Whether the VAD entry is currently readable, writable, and executable	Yes	No
WasAllocatedRwx	bool	Whether the VAD entry was allocated readable, writable, and executable	Yes	No
HexFingerprint	string	A 40-character hex string of the beginning 20-bytes of the VAD entry	Yes	Yes

3.4.8.10 FileInfo Object

Table 25: FileInfo Members

Name	Type	Description	Mandatory	Can be null?
FileSize	int	The total file size in bytes	Yes	No
MD5	string	MD5 hash of file contents	Yes	No
SHA1	string	SHA-1 hash of the file contents	Yes	No
SHA256	string	SHA-256 hash of the file contents	Yes	No
CRC32	string	CRC32 hash of the file contents	Yes	No
SSDeep	string	SSDeep fuzzy hash of the file contents	Yes	No
VersionInfo	FileVersionInfo	File version information	Yes	Yes

3.4.8.11 FileVersionInfo Object

Table 26: FileVersionInfo Members

Name	Type	Description	Mandatory	Can be null?
Comments	string	File comments	Yes	No
InternalName	string	Internal file name	Yes	No
ProductName	string	Product name	Yes	No
CompanyName	string	Company name	Yes	No
ProductVersion	string	Product version	Yes	No

FileDescription	string	File description	Yes	No
FileVersion	string	File version	Yes	No
OriginalFileName	string	Original file name	Yes	No

3.4.8.12 Whitelisting Scans

The WARDEN Provider adds a method in the **HostScan** object to remove commonalities between two ANTIGEN executions. Removing commonalities in scans across hosts and WARDEN Jobs enables scan whitelisting and can drastically reduce the amount of data within a ANTIGEN scan.

To remove commonalities between two scans, perform the following steps:

1. Retrieve the whitelisted ANTIGEN scan from the datastore. Typically, this is done by executing a WARDEN Job, targeting a single known-good host. Then, in the Provider, retrieve the **HostScan** object for the whitelisted host and store it in a variable. Here, the Job name that contains the whitelisted scan is "AntigenWhitelist" and the variable that will store the whitelisted scan is "$Whitelist".

```
warden:\> cd AntigenWhitelist
warden:\AntigenWhitelist> $Whitelist = ls .\antigen
```

2. Navigate to the WARDEN Job that contains suspicious ANTIGEN scans. In this example, the Job name is "TestAntigen".

```
warden:\AntigenWhitelist> cd ..\TestAntigen
warden:\TestAntigen>
```

3. Using the **Where-Object** cmdlet, remove whitelisted artifacts from the suspicious scans and drop them from the pipeline if there are no suspicious artifacts remaining in the host scan. The **RemoveCommonalities()** method is used here to remove artifacts that exist in the whitelisted scan.

```
warden:\TestAntigen> ls .\antigen | Where-Object {
    $_.RemoveCommonalities($Whitelist)
    -not $_.IsEmpty
}
```

   The **HostScan** object contains an **IsEmpty** member that indicates if the **HostScan** contains any malicious artifacts or not. If **IsEmpty** is **True**, the **HostScan** is dropped from the pipeline.

3.4.9 Testing

The ANTIGEN capability has undergone internal development testing and testing via operational technical transition partners. ANTIGEN is used operationally by various stakeholders on a daily basis.

4 RESULTS AND DISCUSSION

This section details results and lessons learned from the design, development, and deployment of each capability developed in this effort. Limitations of the capabilities are also briefly discussed, which in some cases relate to future or recommended research.

4.1 WARDEN Incident Response and Network Forensics Framework

4.1.1 Analysis

With the ability to test WARDEN on many live networks, the team was able to receive meaningful operator feedback. This feedback enabled WARDEN to become more efficient as well as more operator-friendly. During previous versions of WARDEN, if the remote host was defined in Classless Inter-Domain Routing (CIDR) notation, the subnet would be expanded prior to a job running which would cause jobs that had thousands of hosts to allocate several gigabytes of memory. With feedback from operators using WARDEN for a large numbers of hosts, the operators found this method of remote host IP handling to be very inefficient. Using the operator feedback WARDEN rapidly adapted and now handles remote hosts defined in CIDR notation on demand. This means that when a job deploys, WARDEN generates the subnet list and the workers' address field is dynamically assigned right before they are spawned to complete their task. Handling the remote host on job deployment greatly reduced the memory overhead needed for jobs on a large network and greatly improved the overall efficiency of WARDEN.

A challenge of early WARDEN deployments discovered during testing was that transport methods (sometimes high overhead) were initiated prior to determining if a host was online or not. If many hosts are selected on a job where a large portion of hosts are offline, a large portion of network bandwidth and job execution was incurred without successfully connecting to a host. This inefficient use of resources was recognized in early functional and operational tests. This finding resulted in the implementation of an alive-check that determines if a host system is online before network bandwidth is used to deliver a job's plugins to that host. This "alive" check greatly reduced the network impact of WARDEN as well as improved the overall efficiency of WARDEN.

WARDEN administration also greatly improved with the implementation of the WARDEN installer. This installer guides an operator through the set-up of WARDEN to completely install and configure all of the necessary dependences. Early releases of WARDEN required a lengthy, tedious, and potentially error-prone installation process that was overly burdensome to operators. By automating this process, the team was able to more quickly obtain extensive testing and feedback from operators to attain a very stable and efficient operational framework and respective capabilities.

4.1.2 Known Limitations

A current known limitation is that once a job has been executed the framework no longer has insight into the progress of job execution. If job pickup is initiated before the collectors have

finished no data will be collected for that collector. This limitation is an inherent result of the "fire-and-forget" job deployment methodology. While an acceptable design decision for some operators, this approach to deployment can be a limitation for others, and the team is seeking methods of alleviating this issue in future releases. In future versions of WARDEN the team hopes to be able to actively maintain contact with hosts so that the operator knows when a job can be picked up without data loss.

Another known limitation of WARDEN is that collectors are not "stealthy" when they are being run on a system. Meaning that if a host is being actively monitored by an adversary then they will observe a running collector process and can take steps to cause its termination. Also, if an active adversary knew that the computer was under investigation by WARDEN, the data that is collected has potential to be subjected to integrity attacks before it is picked up. Future releases of WARDEN will take steps to mitigate these known limitations by running entirely out of memory and using methods less susceptible to adversary attack.

4.1.3 Summary

WARDEN allows operators to more efficiently and effectively identify and respond to compromises as well as collect data from compromises on machines residing on a network. WARDEN provides a single source for scheduling and executing jobs across the network and then analyzing the collected data in a flexible and customizable manner.

4.2 WARDEN Training Environment

4.2.1 Analysis

The team received positive feedback from operators. The graphical, step-by-step training model employed by WTE proved much more palatable to trainees, and was reported as much more effective to simply read and absorb the contents of the WARDEN user manual. In addition, operators were more likely to return to the WTE training when they came across a challenge in the use of WARDEN, rather than consult the user manual.

The WTE training model received such accolades from users that it has been as the basis of other training modules, for other users on different projects.

4.2.2 Known Limitations

There are several known limitations of the WTE:

- WTE does not execute any Jobs on the network
- WTE does not train operators in the use of the WARDEN PowerShell Provider
- WTE does not train operators to identify the Collector plugins necessary to respond to specific threats

4.2.3 Summary

The WTE is an effective tool for training operators in the use of the WARDEN CLI. Operators will become familiar with the workflow of creating, modifying, and executing a WARDEN job. With both the use of the WTE and included documentation, new operators are able to rapidly acquire the necessary knowledge to begin using WARDEN and its available features.

4.3 WARDEN Web User Interface

4.3.1 Results

Although the WebUI was released near the end of effort and only initial operator feedback has been received, the preliminary reviews have been promising. From extensive operator demonstrations, training, and WARDEN team pilot efforts, many improvements to the WebUI have been suggested and will likely be incorporated into the interface in the upcoming releases.

4.3.2 Known Limitations

- Jobs cannot be controlled once started
- Job log data cannot be viewed
- Custom plugins cannot be imported

4.3.3 Summary

With the creation of the WebUI the WARDEN team was able to add an interactive and adaptable user interface to the WARDEN framework. This interface allows WARDEN to be easily used in a wide variety of system environments thus adding to the overall effectiveness and flexibility of the framework.

4.4 ANTIGEN

4.4.1 Analysis

Based on feedback provided by operators, ANTIGEN has shown itself to be capable of identifying indicators of compromise that went undetected by best-of-breed security products.

4.4.2 Known Limitations

On all Windows® OSs the system process (PID: 4) cannot be analyzed due to kernel level security implemented by Windows®. On Win x64 systems, the following processes cannot be analyzed:

- Audiodg.exe
- Searchfilterhost.exe
- Searchprotocolhost.exe

4.4.2.1 Operating System Compatibility

ANTIGEN is designed and built to execute on systems running the Windows® 2000 operating system or newer versions of Windows®.

Table 27: ANTIGEN Windows® Operating System Compatibility

OS	32-Bit	64-Bit
Windows® Server 2000	✓	N/A
Windows® Server 2003	✓	✓
Windows® Server 2008	✓	✓
Windows® Server 2008R2	✓	✓
Windows® Server 2012	✓	✓
Windows® Server 2012R2	✓	✓
Windows® XP	✓	✓
Windows® Vista	✓	✓
Windows® 7	✓	✓
Windows® 8	✓	✓
Windows® 8.1	✓	✓
Windows® 10	✓	✓

There are two ANTIGEN executables: one for 32-bit Windows® systems and one for 64-bit Windows® systems, antigen32-2.2.exe and antigen64-2.2.exe, respectively.

Table 28: ANTIGEN Executables

Name	Target Operating System	Description
antigen32-2.2.exe	Windows® 2000 – Windows® 10, 32-bit	32-bit build of ANITGEN to execute on all 32-bit Windows® installations
antigen64-2.2.exe	Windows® XP – Windows® 10, 64-bit	64-bit build of ANTIGEN to execute on all 64-bit Windows® installation

32-bit Windows® installations need to execute the 32-bit executable, antigen32-2.0.exe. Likewise, 64-bit Windows® installations need to execute the 64-bit executable, antigen64-2.0.exe. Failing to execute the correct binary will result in an error message and ANTIGEN will not execute.

The **collectors/antigen WARDEN** Plugin will automatically execute the correct version of ANTIGEN based on each remote system's architecture and configuration.

4.4.3 Summary

ANTIGEN is a Windows® user-mode executable that detects suspicious executable code in running processes. A system is profiled by scanning active memory to identify characteristics indicative of malicious activity. ANTIGEN alleviates the currently time consuming manual process of incident response on an enterprise.

ANTIGEN solves current problems with traditional anti-malware techniques and software. Signature-based techniques take too long to be useful in near-term protection. The large footprint of host-based anti-malware is another problem that inhibits wide deployment. ANTIGEN maintains a minimal footprint, allowing it to be easily deployed enterprise-wide.

5 CONCLUSION

To conclude, the successful completion of the effort resulted in capabilities ready to be transitioned into the hands of operators.

5.1 WARDEN

The WARDEN framework has proven to be a successful and powerful tool. WARDEN's robust and expandable framework is constantly being updated. The power and agility of WARDEN has enabled operators to stay on the bleeding edge of large-scale incident response and network forensics.

5.2 WARDEN Training Environment

The WTE effectively takes an operator through the various steps of Job building. WTE successfully does this in a classroom like setting through eight lessons in which the operator learns everything from scheduling a Job to importing custom Collectors. Additionally, WTE allows operators to train in a real shell in order to make the transition of practice to actual job execution seamless.

5.3 WARDEN Web User Interface

With the creation of the WARDEN WebUI, the WARDEN framework now incorporates a multiplatform user interface that allows operators to access the full functionality of WARDEN via a web-based framework. The WebUI can support multiple operators accessing the framework from remote-locations and can be used to centralize data collection and reporting. This user interface significantly lowers the on-ramping time from new operators that are more comfortable with graphical user interfaces as opposed to command line interfaces.

5.4 ANTIGEN

ANTIGEN identifies suspicious objects in the memory of running Windows® systems that are symptomatic of the presence of malicious artifacts. ANTIGEN looks for entries in memory that are common in infection. These suspicious objects include:

- Executable code out of an image
- Modified images
- DLLs in unusual processes and unusual locations
- Hooked functions
- Packed code

The results of these tests are presented to the operator in either standard output, or saved to a file or database.

6 FUTURE WORK

Future work would extend WARDEN's current capabilities, provide new features and additional innovative functionality to accelerate the processing of large-scale computer systems and computer networks. Research and development would focus on extending the network data acquisition

capabilities to the network edge for a more complete view of the evidentiary data and evidence residing within large-scale networks. The expansion of network endpoints would include networked devices such as printers, fax machines, overhead projectors, network storage and drives, voice over IP phones, smart TVs, smart appliances, and scanners. Relevant data residing on these devices includes contraband files in the forms of text, voice, imagery or video as well as relevant event timestamps, connections to other devices, and user activities or interactions that occurred on the device or across the network. The collectors developed under a future effort would address issues typically encountered when acquiring data within large-scale networks, where access data is obscured by fragmented files, non-standard interfaces, as well as proprietary file formats or encodings.

7 ACRONYMS

API	Application Programming Interface
ATA	AT Attachment
ATAPI	AT Attachment Packet Interface
CIDR	Classless Inter-Domain Routing
CLI	Command Line Interface
CSV	Comma Separated Values
DLL	Dynamically Linked Library
EPT	Extended Page Tables
OS	Operating System
PE	Portable Executable
PID	process identifier
RAM	Random-Access Memory
SOW	Statement of Work
SWAT	Secure Workstation Attestation
VAD	Virtual Address Descriptor
VM	Virtual Machine
VMI	Virtual Machine Introspection
WebUI	Web Users Interface
WMI	Windows® Management Instrumentation

8 REFERENCES

[1] Cisco. (2015). Snort.org. Retrieved March 17, 2015, from https://www.snort.org

[2] MartinGarcia, L. (2015). TCPDUMP/LIBPCAP public repository. Retrieved March 17, 2015, from http://www.tcpdump.org/

[3] Guidance Software. (n.d.). Computer Forensic Software - Encase Forensic. Retrieved March 9, 2015, from https://www.guidancesoftware.com/products/Pages/encase-forensic/overview.aspx

[4] SANS Institute. (n.d.). SANS SIFT Kit/Workstation: Investigative Forensic Toolkit. Retrieved March 9, 2015, from http://digital-forensics.sans.org/community/downloads

[5] Niksun. (n.d.). NIKSUN NetDetector Alpine. Retrieved March 9, 2015, from https://www.niksun.com/product.php?id=4

[6] Access Data. (n.d.). SilentRunner Sentinel. Retrieved March 9, 2015, from http://www.jesc.co.za/wp-content/uploads/2014/03/2-SilentRunner-Brochure.pdf

[7] Shanmugasundaram, K., Memon, N., Savant, A., & Bronnimann, H. (2003). ForNet: A Distributed Forensics Network. Computer Network Security Lecture Notes in Computer Science, 2776, 1-16.

[8] Nurse, J., Buckley, O., Legg, P., Goldsmith, M., Creese, S., Wright, G., & Whitty, M. (2014). Understanding Insider Threat: A Framework for Characterising Attacks. 2014 IEEE Security and Privacy Workshops. San Jose, California

[9] Carvey, H. (2011). Windows Registry Forensics. Burlington, MA: Syngress.

[10] SANS Digital Forensics and Incident Response. (2009, September 12). SANS Digital Forensics and Incident Response Blog. Retrieved March 9, 2015, from http://digital-forensics.sans.org/blog/2009/09/12/best-practices-in-digital-evidence-collection/

[11] Corey, V., Peterman, C., Shearin, S., Greenberg, M. S., & Bokkelen, J. V. (2002). Network Forensic Analysis. IEEE Internet Computing, 6(6), 60-66.

[12] Wang, W., & Daniels, T. E. (2008). A Graph Based Approach Toward Network Forensics Analysis. ACM Transactions on Information and Systems Security, 12(1), 1-33.

[13] Neasbitt, C., Perdisci, R., Li, K., & Nelms, T. (2014). ClickMiner: Towards Forensic Reconstruction of User-Browser Interactions from Network Traces. 21st ACM Conference on Computer and Communications Security. Scotsdale, AZ.

Assured Information Security, Inc.

APPENDIX B. LEGAL OPINION

ADMISSIBILITY OF DIGITAL EVIDENCE
DERIVED USING WARDEN™

Martin E. Wolf, Esq.

Section 1 Introduction

The proliferation of computers into everyday life has transformed the nature of society's "documents" from primarily paper and ink, into "digital data."[1] An ever-increasing number of documents generated today are digital data,[2] including not only those that are analogous to traditional "documents" (such as word processing documents, and electronic spreadsheets), but also conversations and meetings carried-on through digital means that produce digital records: text messages, Skype, calls made over VoIP, video conferencing, and others.[3] Consider that in 2014, the number of cell phones in the world exceeded the number of people between the ages of fifteen and sixty-four.[4] The number of digital photos generated from the smart phones alone are overwhelming. Moreover, unlike its paper cousin, digital data bears unique characteristics: it is stored longer, by more people, and it is more difficult to discard.[5]

The legal system, in civil and criminal cases, has been forced to adapt to the changes in both the volume and nature of documents sought to be introduced into evidence. Given

[1] The terms "digital data," "computer-generated information," and "electronically stored information" or "ESI" are used interchangeably in this memorandum.

[2] Maj. Scott A. McDonald, *Authenticating Digital Evidence from the Cloud*, 2014 ARMY LAW 40 (generally describing historical growth of digital documents in online services such as webmail services, online data storage, cloud-based word processing application, social media and online storage sites), *citing* Pew Research Center, *Use of Cloud Computing Applications and Services*, 1 (2008); Cindy Pham, Article, *E-Discovery in the Cloud Era: What's a Litigant to Do?*, 5 HASTINGS SCI. & TECH. L.J. 139 (2013).

[3] 2 Raymond T. Nimmer & Holly K. Towle, *E-Mails and Evidence in E-Commerce Contexts*, § 13.09, pt. C (2d ed. 2018) (*"Commercial Trans."*).

[4] Serge Jorgensen, *Convergence of Forensics, Ediscovery, Security & Law*, 12 AVE MARIA L. REV. 291, 292 n.6 (2014).

[5] *Commercial Trans.* ("As those and group communications are forwarded from person to person or accessed or altered by multiple persons, storage places multiply as do repetitive documents, such as an e-message string or blog where the base messages are repeated each time a new person adds a comment. More so than in a paper world, exact copies and/or different versions of a particular document may be found in multiple locations: PC hard drives (home and office), network servers, cell phones (personal or company owned), wearables (e.g. smart watches etc.), videos, CD-ROMs, laptops, e-mail attachments, text messages, servers of third parties such as outsourcers and cloud or other service providers storing corporate or consumer records, pictures and so on. Voice-mail messages are converted by universal messaging systems into audible e-mails that can exist in all those places as well. The creation of backup tapes and archiving electronic documents create additional copy sets, and the ability to resurrect deleted files from electronic storage media increases the volume of potentially relevant and responsive material.").

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

the sheer volume of digital data, investigators rely more and more on technology, including forensic software, to keep pace with the number of "documents" to be identified, analyzed and produced. This forensic software is constantly evolving to address the expanding needs for document investigation.

We have been asked to evaluate a particular software product known as WARDEN™ ("Wide-Scale, Agentless and Rapid collection of Digital Evidence from Networks"). Specifically, we will render an opinion as to the admissibility of evidence derived through the use of WARDEN™, and compare the attributes of WARDEN™ to EnCase™ forensic software from OpenText (formerly Guidance Software).

This report will focus on digital evidence used in criminal investigations and prosecutions. We will examine the rules of evidence and their application as they are applied to digital data, and, specifically, when such data is acquired, analyzed and stored using forensic software (Section 2.1). We then will identify the attributes of forensic software that have been relied upon by courts in determining the admissibility of evidence derived through such software (Section 3.1). In Sections 4.1 and 4.2, we will review the attributes of WARDEN™, and compare them to the attributes of the EnCase™ suite of forensic software. Finally, in Section 5.1, we will offer an opinion as to the admissibility and relative forensic value of digital data produced by these software packages.[6]

Section 2 Evidentiary Rules Governing Digital Data

Admissibility of digital data presents a conundrum. On the one hand, courts have recognized that the issues surrounding admissibility of digital data raise the same issues as the admissibility of paper documents (*e.g.*, manipulated e-mail v. a forged letter, accuracy of fax banners v. stolen/duplicated letterhead, etc.). On the other hand, it can be more difficult to both prove or rebut allegations regarding the authenticity or accuracy of digital data because of the ways in which it is created, stored and produced (as opposed to paper documents, for example, a signed letter in a bank safe deposit box).

Admissibility of evidence in federal court is generally governed by the Federal Rules of Evidence ("FRE"). [7] The roadmap for determining the admissibility of digital data was

[6] The Oxford Dictionaries define "forensic" as: (1) "Relating to or denoting the application of scientific methods and techniques to the investigation of crime"; and (2) "Relating to courts of law." *See Forensic Definition*, oxforddictionaries.com, http://en.oxforddictionaries.com/definition/forensic (last visited June 15, 2018). Given the purpose of this report, the terms "forensic" or "forensically sound" (and similar uses), when used in relation to digital data acquired using software and searches of computers and other storage media, refer to the data's value as evidence in a criminal investigation or trial. In other words, whether the digital data is likely to be (1) admitted into evidence, and (2) given sufficient weight by the trier of fact to impact the fact-finder's decision.

[7] Computerized searches may invoke the protections against search and seizure of the Fourth Amendment. Such issues under the Fourth Amendment are beyond the scope of this project, which is limited to: (1) admissibility of digital data as evidence under the FRE; and (2) the use of digital data to establish probable cause in securing a search warrant.

set forth in *Lorraine v. Markel Am. Ins. Co.*, 241 F.R.D. 534 (D. Md. 2007). As used below "ESI" stands for Electronically Stored Information. The court set out a 5-step hierarchical process:

> Whenever ESI is offered as evidence, either at trial or in summary judgment, the following evidence rules must be considered: **(1)** is the ESI relevant as determined by Rule 401 (does it have any tendency to make some fact that is of consequence to the litigation more or less probable than it otherwise would be); **(2)** if relevant under 401, is it authentic as required by Rule 901(a) (can the proponent show that the ESI is what it purports to be); **(3)** if the ESI is offered for its substantive truth, is it hearsay as defined by Rule 801, and if so, is it covered by an applicable exception (Rules 803, 804 and 807); **(4)** is the form of the ESI that is being offered as evidence an original . . . under the original writing rule, or if not, is there admissible secondary evidence to prove the content of the ESI (Rules 1001-1008); and **(5)** is the probative value of the ESI substantially outweighed by the danger of unfair prejudice or one of the other factors identified by Rule 403, such that it should be excluded despite its relevance.

Id. at 538 (emphasis supplied)(all references to "rules" are to the FRE). Steps 2 and 4 involve issues that are germane to forensic software in general, and WARDEN™ in particular, they being issues of authenticity and accuracy (i.e., the "original writing rule").

Before turning to the authenticity rules themselves, it is important to understand that all such decisions are subject to the provisions of FRE 104(a) and (b).[8] The significance of these rules is that when an objection is lodged to proffered digital data that includes a bona fide dispute of fact as to its authenticity, the court's decision to admit the digital data will only be conditional, subject to a determination by the jury (or other fact finder) on the basis of admissible evidence. For example,[9] consider a case in which a

---

[8] FED. R. EVID. 104(a) - "The court must decide any preliminary question about whether a witness is qualified, a privilege exists, or evidence is admissible. In so deciding, the court is not bound by evidence rules, except those of privilege."

FED. R. EVID. 104(b) - "When the relevance of evidence depends on whether a fact exists, proof must be introduced sufficient to support a finding that the fact does exist. The court may admit the proposed evidence on the condition that the proof be introduced later."

[9] This example is taken from an article written, in part, by the Honorable Paul W. Grimm, the judge in *Lorraine v. Markel*, and a noted authority on digital evidence. *See*, Hon. Paul W. Grimm, Daniel J. Capra & Gregory P. Joseph, Esq., *Authenticating Digital Evidence*, 69 BAYLOR L. REV. 1, 6-10 (2017). Judge Grimm was appointed by the Chief Justice of the United States to the Advisory Committee for the Federal Rules of Civil Procedure when he served as Chief Magistrate Judge for the District of Maryland. Judge Grimm's official court biography can be viewed on the Court's website at Paul W. Grimm – Biography, U.S. District Court, District of Maryland, http://www.mdd.uscourts.gov/paul-w-grimm-district-judge (last visited June 20, 2018)

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

company e-mail written by a specific employee is proffered as an exhibit. The basis for its authenticity is the fact that it was found on the company's server, it purports to have been sent by an employee, it bears the employee's company e-mail address. If an objection is made on the basis that anyone could have written the e-mail using the employee's e-mail account, the judge's decision on admissibility will be final; no issue of fact has been raised. In other words, it is not enough to speculate about what facts may theoretically impact the authenticity of the e-mail. If on the other hand, the objection is that anyone could have written the e-mail, and the defense will produce 5 witnesses who will testify that they were with the employee at the time the e-mail was sent, and the employee did not send the e-mail, the judge's decision will be conditional, such an objection would raise an actual dispute of fact and the judge may find the e-mail has been authenticated, and admit it into evidence conditionally; the jury will decide the factual dispute based on the evidence actually produced at trial. The significance of this interplay between the rules may greatly increase the quality of supporting evidence required to authenticate the e-mail.

There is one further complicating factor. Whether the authenticity of digital data is determined by judge or fact finder, the jury (or other fact finder) is free to give it whatever weight it determines is appropriate.[10] As explained more fully below, the weight given to physical evidence (such as digital data) is often impacted by the chain of custody of the evidence.[11] Thus, even if digital data is admitted into evidence, different software may have different capabilities that could impact what effect, if any, the digital evidence will have on the jury's decision.

Section 2.1 Authenticity

Authenticity is generally governed by FRE 901 and 902. The standard for authenticity is set forth in FRE 901(a): "To satisfy the requirement of authentication or identifying an item of evidence, the proponent must produce evidence sufficient to support a finding that the item is what the proponent claims it is." Under this standard, authenticity is simply a special aspect of relevancy because "evidence cannot have a tendency to make the existence of a disputed fact more or less likely [i.e. "relevant"] if the evidence is not

---

[10] *See* MODEL CIV. JURY INSTR., 3rd Cir., 1.5 (2015) ("Consider it in light of your everyday experience with people and events, and give it whatever weight you believe it deserves."); FED. CRIM. JURY INSTR., 7th Cir., 2.02 (2012) ("Give the evidence whatever weight you decide it deserves."); Pattern Instruction No. 2.02 ("It is up to you to decide how much weight to give to any evidence, whether direct or circumstantial."); MODEL CIV. JURY INSTR., 9th Cir., 1.12 (2017) ("It is for you to decide how much weight to give to any evidence."); *United States v. Cardenas*, 864 F.2d 1528, 2531 (10th Cir. 1989)("deficiencies in the chain of custody go to the weight of the evidence, not its admissibility; once admitted, the jury evaluates the defects and, based on its evaluation may accept or disregard the evidence."); *United States v. Vidacak*, 553, F.3d 344, 350 (4th Cir. 2009)(same), *quoting Cardenas*; *United States v. Pantic*, 308 Fed.Appx. 731, 733 (4th Cir. 2009)(same), *quoting Cardenas*; *Flores v. City of Westminster*, 873 F.3d 739, 758 (9th Cir. 2017) *cert. denied sub nom, Hall v. Flores*, 138 S.Ct. 1551 (2018), *quoting Torta v. Las Vegas Metro. Police Dept.*, 556 F.2d 678, 681 (9th Cir. 1985).

[11] *Vidacak*, 553 F.3d at 350 ("deficiencies in the chain of custody go to the weight of the evidence, not its admissibility ...."); *Pantic*, 308 Fed.Appx. at 733 (same); *see also, United States v. Howard-Arias*, 679 F.2d 363, 366 (4th Cir. 1982)(noting that as a practical matter, chain of custody is a variation of the authenticity requirement); *United States v. Blank*, 2015 WL 4041408, at *8 (D. Md. June 30, 2015)(same).

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

what the proponent claims." [12] And the standard is not difficult to satisfy; the proponent need only make a prima facie showing that the document is what he claims it to be:

> [t]he question for the court under Rule 901 is whether the proponent of the evidence has 'offered a foundation from which a jury could reasonably find that the evidence is what the proponent says it is . . . .' the Court need not find that the evidence is necessarily what the proponent claims, but only that there is sufficient evidence that the jury might ultimately do so.[13]

FRE 901(b) sets forth a list of nine examples of ways to satisfy the standard. The list is not exclusive; there may be other ways to meet the standard. Of the nine methods, two are particularly relevant to authenticating digital data through forensic software: FRE 901(b)(4) Distinctive Characteristics and the Like; and FRE 901(b)(9) Evidence About a Process or System.[14]

In *Authenticating*, 69 Baylor L. Rev. at 11-34, the authors list by type of digital data various examples of ways to authenticate digital data under FRE 901(b). Many of these examples include the product of forensic software. For example, one of the first examples of ways to authenticate the authorship of an e-mail is through forensic information including, an e-mail's hash values, and testimony from a forensic witness that recovered metadata showed an e-mail was issued from a particular device at a particular time.[15] The way in

---

[12] *Lorraine*, 241 F.R.D. at 539, *quoting United States v. Branch*, 970 F.2d 1368, 1370 (4th Cir. 1992), *citing United States v. Sliker*, 751 F.2d 477, 497-99 (2d Cir. 1984).

[13] *Lorraine*, at 542, *quoting United States v. Safavian*, 435 F. Supp. 2d 36, 38 (D.D.C. 2006).

[14] FED. R. EVID. 901(b) states in pertinent part:

> (b) Examples. The following are examples only - not a complete list - of evidence that satisfies the requirement:
>
> * * *
>
> (4) Distinctive Characteristics and the Like. The appearance, content, substance, internal patterns, or other distinctive characteristics of the item, taken together with all the circumstances.
>
> * * *
>
> (9) Evidence About a Process or System. Evidence describing a process or system and showing that it produces an accurate result.

[15] *See Authenticating Digital Evidence, supra* note 9, at 17 n.47, (*quoting* Barbara J. Rothstein, Ronald J. Hedges & Elizabeth C. Wiggins, *Managing Discovery of Electronic Information: A Pocket Guide for Judges* 38 (2d. ed. 2007)):

which software identifies, collects and stores these two areas of forensics - hash values and metadata - are most critical to determining the value of the particular software for evidence collection.

FRE 902 sets forth 14 categories of evidence that are "self-authenticating." Of particular importance are two categories that were added to the list as of December 1, 2017: (13) Certified Records Generated by an Electronic Process or System; and (14) Certified Data Copies from an Electronic Device, Storage Medium, or File.[16] Both FRE 902 (13) and (14) are intended to provide a procedure to determine early in the process if there will be a real challenge to the authenticity of proffered digital data.[17] This process is borrowed by reference to FRE 902(11) and (12), which establish the process for authenticating both domestic and foreign business records. Part of that process is advance notice to the opposing party of the intention to authenticate the exhibit by affidavit of the custodian of the record or other qualified person that the exhibit is what it purports to be (i.e., a business record).[18] Only if the opponent intends to make a real challenge to

---

> A hash value is [a] unique numerical identifier that can be assigned to a file, a group of files, or a portion of a file, based on a standard mathematical algorithm applied to the characteristics of a dataset. The most commonly used algorithms, known as MD5 and SHA, will generate numerical values so distinctive that the chance that any two data sets will have the same hash value, no matter how similar they appear, is less than one in one billion. "Hashing" is used to guarantee the authenticity of an original data set and can be used as a digital equivalent of the Bates stamp used in paper document production.

[16] FED. R. EVID. 902 states in pertinent part:

> FRE 902. Evidence That Is Self-Authenticating
>
> The following items of evidence are self-authenticating; they require no extrinsic evidence of authenticity in order to be admitted:
>
> * * *
>
> (13) Certified Records Generated by an Electronic Process or System. A record generated by an electronic process or system that produces an accurate result, as shown by a certification of a qualified person that complies with the certification requirements of Rule 902(11) or (12). The proponent must also meet the notice requirements of Rule 902(11).
>
> (14) Certified Data Copied from an Electronic Device, Storage Medium, or File. Data copied from an electronic device, storage medium, or file, if authenticated by a process of digital identification, as shown by a certification of a qualified person that complies with the certification requirements of Rule 902(11) or (12). The proponent also must meet the notice requirements of Rule 902(11).

[17] *See*, FED. R. EVID. 902 advisory committee's notes.

[18] FED. R. EVID. 902(11) and (12):

authenticity will the party be required to provide a witness to lay the proper foundation. By reference to (11) and (12), FRE 902(13) and (14) essentially extend the business records exceptions embodied in FRE 902(11) and (12) to digital data.

While this can be valuable, it is a limited procedure. In the e-mail example described above, for instance, the procedure will uncover if the opponent falls into category 1, in which case the court will determine authenticity, or category 2 where the court's decision will only be provisional subject to the final decision of the fact finder. In all cases, however, the procedure only applies to authenticity; the opponent is free to challenge the evidence on other grounds (e.g., hearsay, relevance, undue prejudice, right of confrontation, etc.).[19]

> (11) Certified Domestic Records of a Regularly Conducted Activity. The original or a copy of a domestic record that meets the requirements of Rule 803(6)(A)-(C), as shown by a certification of the custodian or another qualified person that complies with a federal statute or a rule prescribed by the Supreme Court. Before the trial or hearing, the proponent must give an adverse party reasonable written notice of the intent to offer the record--and must make the record and certification available for inspection--so that the party has a fair opportunity to challenge them.
>
> (12) Certified Foreign Records of a Regularly Conducted Activity. In a civil case, the original or a copy of a foreign record that meets the requirements of Rule 902(11), modified as follows: the certification, rather than complying with a federal statute or Supreme Court rule, must be signed in a manner that, if falsely made, would subject the maker to a criminal penalty in the country where the certification is signed. The proponent must also meet the notice requirements of Rule 902(11).

[19] FED. R. EVID. 902 advisory committee's notes ¶ 13:

> For example, assume that a plaintiff in a defamation case offers what purports to be a printout of a webpage on which a defamatory statement was made. Plaintiff offers a certification under this Rule in which a qualified person describes the process by which the webpage was retrieved. Even if that certification sufficiently establishes that the webpage is authentic, defendant remains free to object that the statement on the webpage was not placed there by defendant. Similarly, a certification authenticating a computer output, such as a spreadsheet, does not preclude an objection that the information produced is unreliable--the authentication establishes only that the output came from the computer.

Id. at ¶ 14:

> For example, in a criminal case in which data copied from a hard drive is proffered, the defendant can still challenge hearsay found in the hard drive, and can still challenge whether the information on the hard drive was placed there by the defendant.

For further examples of how these two paragraphs can be applied, *see* P. Grimm et al., *Authenticating Digital Evidence*, at 42-50.

One final note regarding FRE 902(14): this revision has highlighted the importance of "hash values" in addressing the admissibility of digital data beyond just authentication. Specifically, the Revisor's Notes to Paragraph 14 express a certainty that matching hash values (and future similar technologies) are a virtual guarantee of exactness:[20]

> Today, data copied from electronic devices, storage media, and electronic files are ordinarily authenticated by "hash value." A hash value is a number that is often represented as a sequence of characters and is produced by an algorithm based upon the digital contents of a drive, medium, or file. If the hash values for the original and copy are different, then the copy is not identical to the original. If the hash values for the original and copy are the same, it is highly improbable that the original and copy are not identical. Thus, identical hash values for the original and copy reliably attest to the fact that they are exact duplicates. This amendment allows self-authentication by a certification of a qualified person that she checked the hash value of the proffered item and that it was identical to the original. The rule is flexible enough to allow certifications through processes other than comparison of hash value, including by other reliable means of identification provided by future technology.

This note accords significant importance to hash values in authenticating digital data, and rightly so. But matching file hash values only assure that two files are identical; they say nothing about the files' provenance, including the chain of custody (see Section 2.2 below, n. 24).

Section 2.2 Chain of Custody

Once digital data has been authenticated and admitted into evidence (assuming all other requirements for admission have been satisfied as well), it will be up to the jury to attribute some evidentiary value to the data. In other words, does the evidence tend to prove a fact at issue in the case? In the example from Section 2.1, assume that through testimony regarding the acquisition of the e-mail including metadata or matching hash values, the company e-mail was admitted into evidence by the judge, and considered by the jury as evidence in the case. It is still possible that ultimately, the contents of the e-mail may be given no weight on the issue in the case for which it was offered.[21]

---

[20] *See*, FED. R. EVID. 902, advisory committee's notes ¶14.

[21] In a dissenting opinion, Judge Battaglia of the Maryland Court of Appeals described this interplay between authenticity and weight of evidence regarding proffered printouts of social media pages:

> I am not unmindful of the Majority Opinion's analysis relating to the concern that someone other than Ms. Barber could access or create the account and post the threatening message. * * * The technological heebie-

One of the important factors that impacts the weight of such evidence in the example is the chain of custody. For example, a Commentary of the Sedona Conference,[22] in discussing the use of technology to retrieve, analyze and store for trial, digital data, described the need for a well-documented chain of custody when dealing with system metadata:

> Metadata can be another useful checkpoint for determining authenticity. For example, email messages generally contain a substantial amount of metadata information, including a unique message ID as well as information on the unique Internet locations (IP addresses) where the message originated and was handled along the way to its destination.
>
> Similarly, operating system metadata can be a useful tool. Most operating systems maintain information about individual files – the dates that a file was created, last modified and last accessed. For example, in a case where an individual claims that it did not create a document until July 1, but the system metadata shows that the document was created on May 1, this data may be helpful.
>
> However, metadata can be unreliable and is usually subject to manipulation and non-obvious deletion. A moderately sophisticated user may be able to manipulate system dates, and although traces of this manipulation may be left behind, detecting such traces can be extremely difficult and expensive,

---

> jeebies discussed in the Majority Opinion go, in my opinion, however, not to the admissibility of the print-outs under [Maryland's Authentication Rule of Evidence], but rather to the weight to be given the evidence by the trier of fact. * * * Like many filters that are unable to remove completely all impurities, [Maryland's Authentication Rule of Evidence] does not act to disallow any and all evidence that may have "impurities" (i.e., in this case, evidence that could have come, conceivably, from a source other than the purported source). * * * The potentialities that are of concern to the Majority Opinion are fit subjects for cross-examination or rebuttal testimony and go properly to the weight the fact-finder may give the print-outs.

Griffin v. State, 419 Md. 343, 367-68 (2011).

[22] "The Sedona Conference (TSC) is a nonprofit, 501(c)(3) research and educational institute dedicated to the advanced study of law and policy in the areas of antitrust law, complex litigation, and intellectual property rights. The mission of TSC is to drive the reasoned and just advancement of law and policy by stimulating ongoing dialogue amongst leaders of the bench and bar to achieve consensus on tipping point issues. TSC brings together the brightest minds in a dialogue based, think-tank setting with the goal of creating practical solutions and recommendations of immediate benefit to the bench and bar." Taken from the website: The Sedona Conference, https://thesedonaconference.org/ (last visited June 6, 2018).

> or simply impossible. Worse, use of files after the fact, such as an investigator opening a file for review, can modify metadata and make it useless or misleading for authenticity purposes. Accordingly, careful attention should be paid to the methods used to authenticate metadata.

The Sedona Conference Commentary on ESI Evidence & Admissibility, at 13 (2008).[23] Similarly, in discussing hash values, the Commentary notes that, while matching hash values can determine that the contents of two files are identical, the reliability of the file produced as evidence is based on a trustworthy reference, that is, the subject file or its hash value, and the copy or its hash value.[24]

Accordingly, for the purpose of evaluating forensic software, it is important to determine that the software: (1) preserves the target files without alteration, (2) obtains hash values of the target files, (3) obtains copies of the target files, (4) obtains hash values for each copy, and (5) maintains the integrity of each file and hash value until admitted into evidence, including adequately documenting the chain of custody.

Section 2.3 The Original Writing Rule

---

[23] The commentary is a product of the Sedona Conference Working Group Series, Working Group on Electronic Document Retention & Production (WG 1). As described more fully on its website The Sedona Conference Working Group Series, https://thesedonaconference.org/wgs (last visited June 20, 2018):

> The Sedona Conference® Working Group Series℠ was established to pursue in-depth study of tipping point issues in the areas of antitrust law, complex litigation, and intellectual property rights, with the goal of producing high-quality, non-partisan commentary and guidance of immediate, practical benefit to the bench and bar.
>
> * * *
>
> The mission of Working Group 1 is to develop principles, guidance and best practice recommendations for information governance and electronic discovery in the context of litigation, dispute resolution and investigations. The group released the first public comment draft of The Sedona Principles in March 2003, and the impact was immediate and substantial. Within a few weeks, The Sedona Principles was cited by the Civil Rules Advisory Committee Discovery Subcommittee as one of the reasons to focus on possible amendments to the Federal Rules of Civil Procedure, and it was cited in the seminal *Zubulake* case in the Southern District of New York. Since then, WG1 has published updated editions of The Sedona Principles and several companion works, including guidelines for electronic document management, an authoritative glossary of e-discovery and electronic records management terms, several commentaries on e-discovery related topics, and cooperation guidance for trial lawyers, in-house counsel, and the judiciary.

[24] *Id.* at 12-13.

The Original Writing Rule[25] is embodied in FRE 1001-1008. The essence of the rule is that the contents of a writing, recording or photograph (in today's world most likely digital data) must be proved through an original or copy of the writing, recording or photograph unless secondary evidence is deemed acceptable. The key is to determine when the contents of the digital data are to be proved, as opposed to when an event that just happens to have been recorded is being proved.[26] For example, an eyewitness can testify that John and Mary are married because the eyewitness was at the wedding. The marriage license may also be used to prove the fact, but it is not necessary.

For purposes of this analysis, FRE 1003 is the rule on which we shall focus, because this rule solves one of the most insidious problems when dealing with digital evidence: what is the original?[27] As once noted by one commentator:[28]

> Because servers always carry the risk of catastrophic failure, "[a] cloud computing system must make a copy of all its clients' information and store it on other devices. Thus, a user of Dropbox may upload one copy of a photo they took on vacation and never realize that the photo has been duplicated and potentially stored on any one of many servers located throughout the world. As a result, cloud content gathered pursuant to a law enforcement investigation may not be the original content stored by the user.

At least for purposes of the original writing rule, copies of files are considered originals. The copied file with a matching hash value satisfies the purpose of the rule.

Section 3 Use of Forensic Software in Court

---

[25] Originally known in the common law as the "best evidence rule," that moniker has been found to be misleading. The rule never required the "best" evidence to prove the content of a writing, only the original. Thus, the FRE has used the more accurate term "original writing rule." *See, Lorraine*, at 156 n.54.

[26] *Lorraine*, at 156.

[27] FED. R. EVID. 1003:

> A duplicate is admissible to the same extent as the original unless a genuine question is raised about the original's authenticity or the circumstances make it unfair to admit the duplicate.

[28] *See Authenticating Digital Evidence from the Cloud, supra* note 2, at 41 n.26; *see also,* Orin S. Kerr, *Searches and Seizures in a Digital World,* 119 Harv. L. Rev. 531, 564 (2005)("From a technical perspective, it usually makes no sense to speak of having an "original" set of data. Given this, it would be troublesome and artificial to treat copies as different from originals.").

Decisions addressing digital evidence acquired, analyzed and stored using forensic software generally involve one of two processes: (1) static environment; and (2) live environment. In a static environment, a mirror image copy is made of a storage device, such as a computer hard drive. The accuracy of the copy is verified by matching the hash values of the target hard drive (and each file on the drive) with the hash values of the copy (and each file). Then, forensic software is used to acquire evidence from the copy. In a live environment, the forensic software acquires digital data while the system continues to operate. One of the key distinctions is that, while the system is operating, data and metadata will continue to change.

Both processes have produced admissible evidence in court,[29] however evidence acquired from a live system can be vulnerable to challenge if there is a break in the chain of custody. [30] WARDEN acquires digital data from a live system, which will place greater emphasis on the chain of custody.

Section 4 Software Attributes and Capabilities

Section 4.1 WARDEN[31]

WARDEN is a software product developed by Assured Information Security, Inc. as a tool to allow forensic investigators to search large-scale computer networks for digital evidence in a manner that preserves the forensic integrity of the data, including evidence of criminal activity and criminal intent. The goal is to: (1) identify, preserve, acquire, analyze and report data with investigative value; and (2) streamline and enhance forensic data collection to quickly and remotely extract evidence from remote devices by searching for anomalies or potentially incriminating evidence.

The WARDEN process begins with the operator creating a job, which entails selecting one or more scripts called "collectors" to collect data from the target device. Each collector has specific data that it will try to collect from the target. Once the job is started, WARDEN connects to the target machine through transports that open a communication link, creates a directory on the target machine, and delivers the selected collectors to the target. WARDEN then executes the collectors and they each identify the data they are designed to collect, such as certificates, DHCP, event logs, hash process, and open ports, and stores the data in the WARDEN directory on the target until pickup. WARDEN copies the

---

[29] *See, e.g., Sanders v. State*, 191 S.W.3d 272, 278 (Tex. App. 2006)(static); *United States v. Tippens*, 2016 U.S. Dist. LEXIS 184174, at *2 (W.D. Wash. November 30, 2016)(live)("*Tippens I*").

[30] *See, Tippens I*, at *10, *supra* note 29; *United States v. Cruz-Fajardo*, 2017 U.S. Dist. LEXIS 136030, at *3 (N.D. Ga. June 8, 2017).

[31] The descriptions of performance and attributes of WARDEN™ are taken from analysis conducted by the National Institute of Justice Criminal Justice Technology Research, Test, and Evaluation Center (the "RT&E Center"), including the report, *NIJ RT&E Center WARDEN Technical Review Regarding Forensic Integrity of Collected Data*, January 16, 2018.

data from the target into its own database, and deletes the WARDEN directory from the target.

It should be noted that WARDEN does not appear to provide a collector to acquire a copy of a target file. For example, if a collector identified a hash value for a specific file as one listed in the inventory of known child pornography files, the WARDEN collector would gather metadata about the file, but not the actual file itself. Similarly, WARDEN may collect system information, such as the fact that a USB drive was connected to a certain device at a certain time, and an unusual volume of file activity (e.g., downloads) occurred during the time it was connected. While the data collected may become evidence at trial, the immediate value is most likely to be the basis for obtaining a search warrant to obtain a forensic copy of the hard drive (or other storage medium) where this activity occurred.

The RT&E Center was tasked with investigating a number of questions regarding the performance of WARDEN, including WARDEN's ability to collect and preserve data in a forensically sound manner. The RT&E Center found several limitations in the collection of data, one of which is important for purposes of the present analysis: The only transport that can be used to access Linux, Mac, and other physical appliances (firewall routers) is unencrypted. The RT&E Center also found at least four deficiencies pertinent to the present analysis regarding the preservation of data: (1) lack of encryption for data at rest; (2) lack of two factor authentication; (3) lack of database redundancy; and (4) lack of an audit trail. Whether described as a "limitation" (e.g., only one unencrypted transport for certain types of devices), or "deficiency," these findings all point to weaknesses that would impact WARDEN's ability to document a substantial chain of custody. Ultimately, a defendant in a case where such evidence is introduced could have opportunity to challenge the admissibility and value of such evidence presented at trial. The significance of these deficiencies for purposes of admissibility and evidentiary value of evidence is set forth more fully in Section 5.1.

Section 4.2 EnCase

EnCase is a suite of forensic software products produced by OpenText. Originally created in 1998, it has become the standard for obtaining digital evidence for use in court.[32] Evidence acquired through the use of EnCase forensic software has been accepted by numerous courts over the years. One attribute cited by courts in describing the veracity

[32] Although not the only widely-used forensic software (FTK Forensic Toolkit is also cited frequently in caselaw), it has been tested and evidence derived from the use of EnCase has been accepted by many courts over a relatively long time. *See, Williford v. State*, 127 S.W.3d 309, 312-13 (Tex. App. 2004); *Sanders v. State*, 191 S.W.3d 272, 278 (Tex. App. 2006)(two early decisions relying on EnCase); *see also,* **Criminal Cases** *State v. Pratt*, 200 Vt. 64, 77, 128 A.3d 883, 891 (2015); *United States v. Romm*, 455 F.3d 990, 995 (9th Cir. 2006); *United States v. Ganias*, 824 F.3d 199, 204 (2d Cir. 2016); *United States v. Gaynor*, 2008 WL 113653, at *1 (D. Conn. Jan. 4, 2008); *United States v. McCoy*, 2015 WL 7770181, at *3 (D. Minn. Oct. 1, 2015); and **Civil Cases** *In re Hitachi Television Optical Block Cases*, 2011 WL 3563781, at *2 (S.D. Cal. Aug. 12, 2011); *Malibu Media, LLC v. Doe*, 82 F. Supp. 3d 650, 656 (E.D. Pa. 2015); *Xpel Techs. Corp. v. Am. Filter Film Distributors*, 2008 WL 744837, at *1 (W.D. Tex. Mar. 17, 2008).

of such evidence is the fact that EnCase ensures the accuracy of forensic copies through the use of matching hash values, that is, the unique hash value of the original target disk drive or file (whatever the case may be) matches the hash value for the copy being offered into evidence.

To ensure the integrity of the matching hash values, OpenText employs a specific file format for EnCase products known as EnCase Evidence File Format ("EEFF"). The EEFF is broken into three sections, the header, data blocks, and footer. The header contains case information such as the date and time of acquisition, the examiner's name, notes on the acquisition, etc. The data blocks contain the actual acquired data, but at the time of acquisition, the copy is split into 32KB sections, with a "Cyclic Redundancy Check" between each section. The CRC includes a hash value calculated for the section. If the data is ever accessed, the CRC hash can be recalculated and compared to the original to determine if any change has been made. The footer contains an MD5 hash of the entire image. Essentially, the identifying information regarding the target file, the copied file, and the process of acquisition, are locked in a "vault" that itself is uniquely identified by a hash value.

One of the software products in the EnCase suite is EnCase Endpoint Investigator, OpenText describes as follows:[33]

> EnCase Endpoint Investigator is designed for corporations and government agencies to perform remote, discreet, and secure internal investigations without disrupting an employee's productivity or impacting day-to-day operations of the business.

Endpoint Investigator appears to be similar to WARDEN in purpose and function.[34] Although we have been unable to find a published decision discussing Endpoint Investigator specifically, this particular program shares an important attribute with EnCase Forensics (which has been discussed and accepted by many courts; see n. 28). One of the points stressed in OpenText's marketing materials is that, "Evidence collected from remote

---

[33] EnCase Endpoint Investigator Product Overview located at: EnCase Endpoint Investigator For Internal Forensic Investigations, https://www.guidancesoftware.com/docs/default-source/document-library/product-brief/encase-endpoint-investigator-product-overview.pdf?sfvrsn=f4a08dad_84 (last visited 6/8/2018)

[34] With the exception that Endpoint Investigator can actually obtain copies of files as well as metadata, and verify the copies by hash value. Although the description in marketing materials describes Endpoint Investigator as being used for internal control projects, it appears to be suitable for criminal investigations and prosecutions as well. In one case study reported by OpenText, an internal investigation uncovered a scheme among high-level financial employees who had destroyed documents in order to improve the corporation's position with the Securities and Exchange Commission. Based on the evidence uncovered by Endpoint Investigator, the employees were prosecuted. Endpoint Investigator Case Studies located at: EnCase Endpoint Investigator in Action, https://www.guidancesoftware.com/document/product-brief/guidance-software-encase-endpoint-investigator-in-action (last visited on June 8, 2018)

machines is stored in the EnCase Evidence File Format, which has been accepted and proven in courts worldwide as forensically sound."[35]

The significance of this method of storage is that it provides an extremely robust chain of custody. As explained in Section 2.2, a jury may consider any challenge to the chain of custody in deciding the weight to give to a piece of evidence. By embedding the file hash values, system metadata, and job information in a file, and recording the hash value of that file, Endpoint Investigator establishes a formidable digital chain of custody for the files obtained.

Section 5 Opinion

The deficiencies in WARDEN (as set forth in Section 4.2) all bear upon the chain of custody, which goes to the weight of the evidence, not admissibility. Given the low threshold for authenticating evidence (i.e., sufficient evidence that a jury could find that the digital data is what the proponent claims it to be), and FRE 1003 (copy is deemed an original), it is my opinion, therefore that digital data acquired through WARDEN would be found to be authentic and admissible into evidence, subject to any other sustained objection (i.e., Hearsay-FRE 801 et seq., Undue Prejudice-FRE 403, etc.).

The deficiencies do, however, create a risk that a jury (or other fact finder) could give little or no weight to the evidence. Several relatively recent cases illustrate the potential challenge to the chain of custody in cases involving forensic software that appears to acquire digital data by a method similar to WARDEN. For example, *United States v. Tippens*, 2017 U.S. Dist. LEXIS 219162 (W.D. Wash. March 16, 2017)("*Tippens II*") provides an instance in which a chain of custody challenge resulted in dismissal of two counts in a child pornography case.[36] There, pursuant to a warrant, the FBI took control of a website that trafficked in child pornography. When members logged onto the website using the Tor network (which protects the anonymity of users), the FBI deployed a network investigative technique (NIT) that inserts an "exploit," a piece of software, onto the user's computer. Due to a weakness in the Tor network, the NIT negates the Tor anonymity protections in order to obtain the computer's IP address and other identifying information. The defendant challenged the warrant, and sought discovery of the source code for the NIT in order to challenge the chain of custody of the digital data acquired through the NIT.[37] The discovery request was denied because the challenges to the chain of custody were speculative, and had no factual support, and therefore failed to satisfy the standard for

[35] *See* n. 33, *supra*; *see also* cases collected in n. 30, *supra*.

[36] In this opinion, the Court dismissed two counts in the indictment after the Government rested its case at trial.

[37] The possible problems in the chain of custody mirror the weaknesses found in WARDEN (*e.g.*, lack of encryption, ability to tamper without detection, lack of audit capabilities, etc.).

This resource was prepared by the author(s) using Federal funds provided by the U.S. Department of Justice. Opinions or points of view expressed are those of the author(s) and do not necessarily reflect the official position or policies of the U.S. Department of Justice.

discovery under Fed.R.Crim.P. 16.[38] After the close of the Government's case, the defendant proffered exhibits that would have bolstered the chain of custody argument. The exhibits, however, were excluded because the Government asserted they were classified (the documents, although classified, were available through Wikileaks). The Court then dismissed Counts I and III, noting the proffered exhibits demonstrated that the Government should have been provided the requested discovery, but chose to go to trial relying on the belief the classified documents would not be discovered and used.[39]

It is unclear if the software used in *Tippens I* and *II* actually modified or planted evidence, allowed evidence tampering by a third party, or properly gathered evidence. In any event, *Tippens II* demonstrates the inherent risk in a weak digital chain of custody. There, the weakness resulted in dismissals, but it could just as well provide a defendant the opportunity to argue that the jury should give no weight to the evidence.

It is, therefore, my opinion that the deficiencies identified in WARDEN pose a risk that the digital data acquired through the software may lack an adequate digital chain of custody, and could jeopardize prosecutions relying on such evidence.

[38] *See, Tippens I, supra* note 29, at 10-11; *Cruz-Fajardo, supra* note 30, at 4-5; *United States v. Matish*, 193 F. Supp. 3d 585, 598-600 (E.D. Va. 2016).

[39] *See Tippens II*, at 4-7.