Overview of the Network Forensics

Chang Yoon Lee, Ji Won Koo

Dankook University, Korea Republic
32183641@gmail.com

Abstract. This paper contains an overview of the methods used in network forensics and their comparison. We will look over the methods of gathering evidence to analyze them and their future in network forensics.

Keywords: Network Forensics, Network Investigation, Forensics Framework, Traffic Analysis, Machine Learning on Forensics.

1 Introduction

Nowadays, computing has become network centered as more people rely on the internet and other network resources. It is no longer adequate to think about computers in isolation because many of them are connected using various network technologies. Therefore, the importance of network forensics, which finds the related digital evidence on the public internet, private networks, and other commercial systems, is increased. Network forensics is a sub-branch of digital forensics related to the monitoring and computer network traffic for information gathering, legal evidence, or malicious action detection (*Buchanan, 2010*). In this paper, we will discuss the methods in digital forensics that contain the related topics to network communication, which is network forensics. We will explain the current technologies to gather and analyze the network evidence, and the future works for the following field.

2 Gathering Evidence

In this section, we will discuss the methodologies to capture the live network forensics data. Also, we will discuss searching for artifacts of network activity wherever they may exist throughout the network.

2.1 DHCP Logs

DHCP, which stands for Dynamic Host Configuration Protocol, is a network management protocol used to dynamically assign an Internet Protocol (IP) address to any device on a network so they can communicate using an IP. DHCP automates and centrally manages these configurations rather than requiring network administrators to manually assign IP addresses to all network devices (*Gillis, 2019*).

If the network that we are performing digital forensics uses DHCP, it is vitally important that the organization records and preserves the DHCP logs for the duration of an examination. DHCP logs provide us with the log of the IP address in the security event log and the firewall log of the suspected computer or the computer of interest. It also provides a way to physically locate the computer within the network. These logs explain which device issued the IP address to a computer with a specific MAC address.

One of the vulnerabilities of DHCP has been the use of man-in-the-middle attacks, which intercept and relay messages between two parties who believe they are communicating directly with each other. To prevent such an example, using DHCP logs is important in network forensics.

2.2 TCP Dump and Win Dump

The TCP dump is the granddaddy of all open-source packet sniffers. It is a command line tool designed to operate under most versions of Unix including Linux, Solaris, etc. Win Dump is a port of TCP dump for use in Windows systems (Lillard, 2010). These days, most open-source sniffers are wrappers for the LIBPCAP. The LIBPCAP contains a set of system-independent functions for packet capture and network analysis. The TCP dump provides the user interface (UI) to communicatee with the LIBPCAP, which talks with the network device driver.

All applications of the TCP dump should be done with the root privileges. The Advanced Packing Tool apt-get utility can be used to retrieve and install the TCP dump in most Unix implementations. For Win dump, we can download the Win dump binaries for the Win PCAP and Win dump from www.winpcap.org.

Even though TCP dump can be used for any general packet-monitoring methods in promiscuous mode, it has limitations. First, TCP dump is command-line utility. The user is required to know all the options for screening the specific packets. Second, Packets can be blocked by a gateway firewall, router, or switch that might not be seen. Third, to replay the recorded traffic or perform additional analysis, usage of tools such as TCP replay, or TCP opera is required.

2.3 Wireshark

Wireshark is the world's most popular network protocol analyzer. It has a rich and powerful feature set and runs on most computing platforms including Windows, OS X, Linux, and Unix. It is freely available as an open source. Wireshark can be used to troubleshoot network problems, examine security problems, debug protocol implementations, learn network protocol internals, and more.

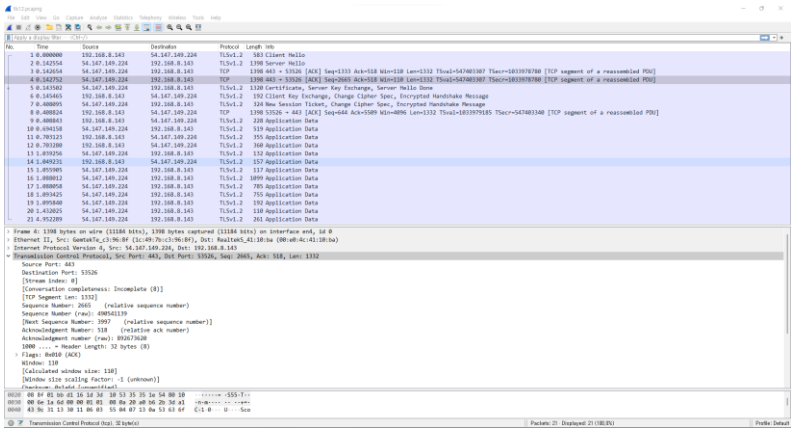

Fig. 1. Graphical User Interface (GUI) for Wireshark

Figure 1 shows the Graphical User Interface (GUI) of the Wireshark displays the network traffic of the tls12.pcapng (Ryšavý, 2022). According to the following figure, we can check that Wireshark offers information such as TCP stream, ethernet connections, internet protocol (IP), transmission control protocol, transport layer security, etc. Also, if we use Wireshark libraries, such utilities as TShark, RawShark, Dumpcap, MergeCap, EditCap, Text2pcap, etc. we can do more operations to analyze the network traffic.

This powerful network traffic analysis tool also has limitations. First, most of the users run Wireshark as an administrator. This way is convenient, but if someone causes either a buffer overflow (BOF) or any other exploit, then the application may fail and leave the attacker as the administrator. Second, all the network-monitoring systems that rely on the libpcap have the same limitations, which also corresponds to the Wireshark. These limitations are a result of the open-source development method. Libpcap has little or no capability with the interfaces until someone develops drivers for physical and virtual interfaces.

2.4 SPAN Ports or TAPS

In the case of TCP dump / Win dump and Wireshark, we have discussed the frameworks that use a host to access the traffic that they can see. Nonetheless, we can also gather network traffic from some useful locations in the network for network forensics.

In an ideal case, we want to access traffic at some point in the network, but do not want to interfere with traffic or be detected.

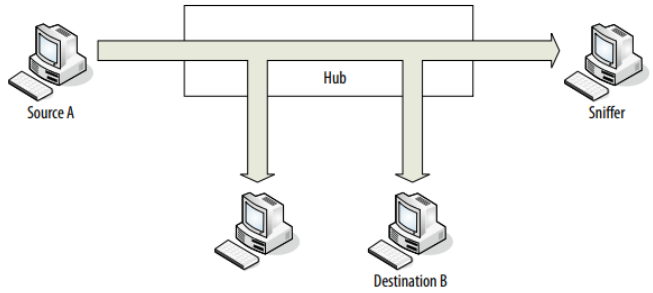

Fig. 2. Hubs and Monitoring (*Lillard, 2010*).

However, if the hub fails, all connections through the hub will be broken. Cheap hubs may not be able to support the throughput of the network and create a bottleneck for our network traffic. According to Figure 2, in the network that consists of hubs and routes, a sniffer to a hub could see all the traffic that passes through the hub.

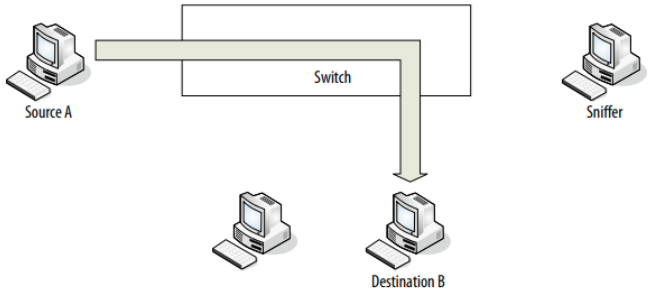

Fig. 3. Switches and Monitoring with no SPAN Port (*Lillard, 2010*).

In contrast, according to Figure 3, switched network connections are point-to-point (P2P). A sniffer attached to a switch in a switched network will only see broadcast traffic and traffic addressed to itself.

We can operate a sniffer reliably on a switched network without a SPAN port or a network tap only if the sniffer is located on the host of interest, all traffic of interest involves the host, and the host is not compromised. Therefore, we need to use the SPAN port or a Network TAP to see the traffic of other devices and to ensure that the compromised host will not interfere with the collection of traffic.

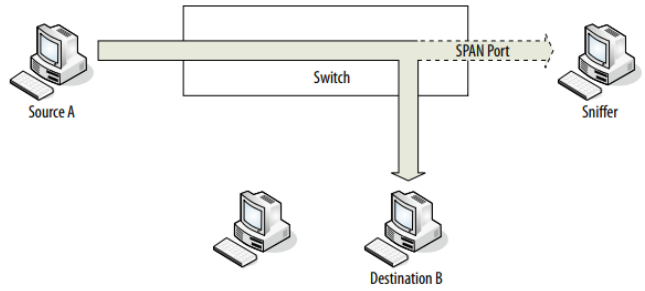

Fig. 4. Switching and Monitoring with SPAN Port (*Lillard, 2010*)**.**

SPAN, which stands for Switch Port for Analysis or Switched Port Analysis, is the port mirroring that sends a copy of all network packets from one port to another port, by allowing the packets to be analyzed. According to Figure 4, if the sniffer uses the SPAN port, it can see the traffic in the switched network.

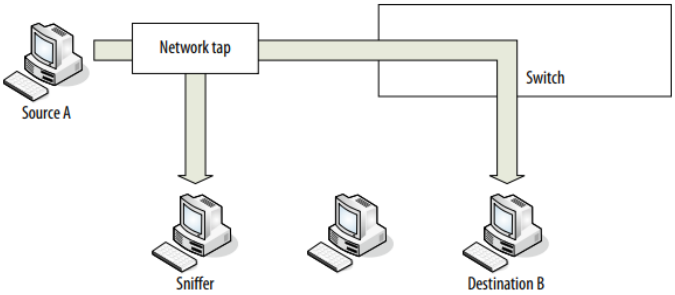

Fig. 5. Switching and Monitoring with SPAN Port (*Lillard, 2010*).

TAP, which stands for Network Test Access Point, is a hardware tool that allows us to access and monitor the network. TAP simultaneously transmits or receives data streams through separate dedicated channels and delivers all data to monitor or secure the devices in real-time.

According to Figure 5, the tap is on the ingress connection side of the switch so that the tap will be able to duplicate all inbound and outbound internet traffic. Network tap duplicates of all traffic including corrupted packets and packets that are below the minimum size. As such, they are ideal for forensics and troubleshooting layer 1 and 2 network errors. If they fail, most taps are designed not to open so that throughput is not affected even though the device is online. In addition, capturing data from a tap eliminates political issues around granting administrators access to switches to security.

2.5 Firewalls

Firewall logs are the primary source of many forensic investigations. Firewall logs do not contain content. Instead, they gather information about the communications. Some firewalls provide additional information such as source and destination OP address, source and destination port, interface, time and date, and the rule that caused the event to record.

Analyzing firewalls can be tricky because firewalls only record the traffic that they are told to record. Therefore, an investigator must know what firewall and network address translation (NAT) rules are in effect when they attempt to interpret the traffic they see in the logs. Frequently, investigators draw a conclusion based on the absence of certain traffic, only to find that the traffic was not being recorded.

3 Analyzing Evidence with Open-Source Software

In this section, we will discuss the operations of the open-source software for network forensics, which uses the evidence mentioned previously.

3.1 Deciphering a TCP Header

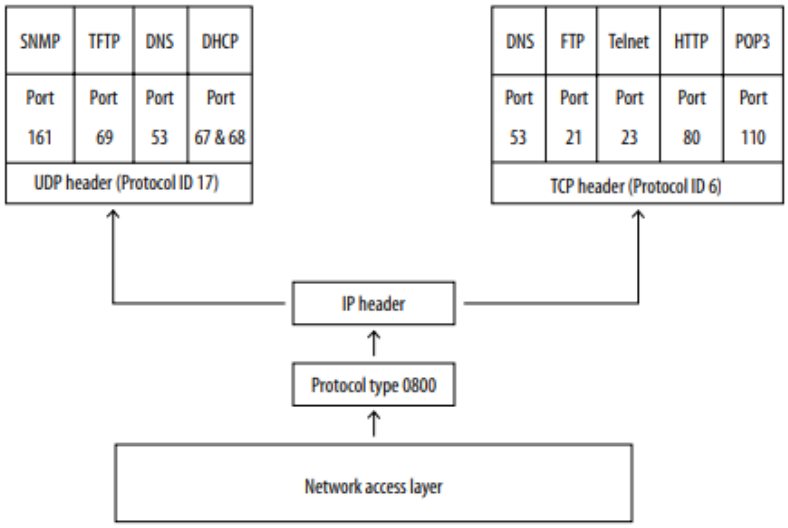

Fig. 6. TCP / IP Model Structure (*Lillard, 2010*).

Transmission Control Protocol (TCP) is a key protocol for the successful implementation of network computing. The TCP / Internet Protocol (IP) model works as an industry framework for end-to-end communications between a source and destination device. Figure 6 presents the structure of the TCP protocol. In this paper, we will not discuss the details and features of the TCP protocol. However, only methodologies that use TCP protocol for network forensics are discussed.

```
0020  08 8f 01 bb d1 16 1d 3d  0b 1f 35 35 1e 54 80 10   ·····= ··55·T··
0030  00 6e 51 e2 00 00 01 01  08 0a 20 a0 b6 2b 3d a1   ·nQ····· ·· ··+=·
0040  43 9c 27 39 60 63 83 ca  22 35 e1 d5 65 0c af 92   C·'9`c·· "5··e···
```

Fig. 6. Sample TCP Packet Capture TCP Header Format

Figure 6 shows the network binary capture of one Ethernet frame using the Wireshark tool. The IP datagram and the TCP segment are encapsulated within the Ethernet frame. For the analysis of network binary captures, the network examiner may be required to decipher the entire Ethernet frame or a subset of the frame. Based on the TCP segment decipherment table, we can analyze the given hexadecimal TCP segment and check which information is in the TCP segment.

The signature analysis of TCP-based packets will allow the network forensics examiner to determine whether the analyzed traffic packets between the source and destination device are normal or suspicious. The following analysis is processed during an investigation to determine the authorized flow of legitimate network traffic or the unauthorized flow of illegitimate network traffic. In the case of normal TCP-based packets, it does not contain any malicious payload data and adheres to the proper use of TCP-based flags which coordinate with RFC 793. However, abnormal TCP-based traffic can contain the use of Malicious Payload Data attacks, the creation of Malformed TCP Header Information attacks, the injection of Single Packet attacks, and the injection of Multiple Packet attacks.

The first is Malicious Payload Data attacks. This occurs when the data is inserted into the TCP segment in the application layer of the TCP / IP model. In the following attacks, Instruction Detection Systems (IDSes) can be used to match binary, or text string sets of characters located within the data payload.

The second is the creation of the Malformed TCP Header Information. This occurs when the attacker uses specifically crafted software tools to alter or generate malformed TCP header fields.

The third is a Single Packet attack. This occurs when the attacker sends a single TCP packet from the source device to the destination device. The following attack is typically used for crushing the TCP protocol stack of the destination device or perform port-scanning techniques to determine the presence of an operating system.

The last is Multiple Packet attacks. This occurs when one device sends multiple network packets to a different device to establish session connectivity. The following attacks are more difficult to detect compared to the methods above.

By understanding the following types of attacks based on TCP-based packets and processing the TCP signature analysis, we can avoid the following types of attacks.

3.2 Using Snort for Network-Based Forensics

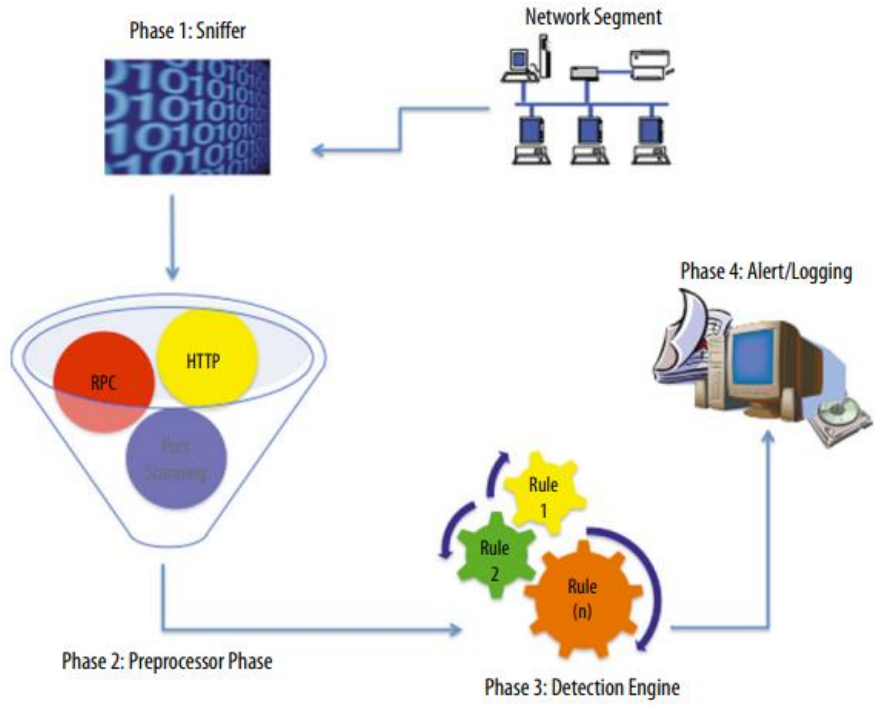

Fig. 7. Snort Architecture (*Lillard, 2010*).

Snort is a free open-source multiplatform product that can be configured to run in four modes: sniffer, packet logger, network instruction detection system (NIDS), and inline. The first is the sniffer. In this mode, it functions as a packet sniffer that reads the packets of the network. The captured packets can be displayed in a continuous stream on a monitor. The second is a packet logger, which can be configured to log the packets to the disk. The third is NIDS, which allows Snort to analyze decoded network traffic against predefined preprocessors and rules and performs several different actions if a match is found. The last is inline. This mode allows Snort to obtain the packets from IP tables and drop or pass those packets based on Snort inline-specific rule types. Figure 7 presents the architecture of the Snort which is based on the four motions that we previously mentioned.

A network forensics investigation that entails the use of Snort involves three forms of data that the network forensics examiner must address within the court of law. The first form is capturing binary network sniffer data. At this stage, the network forensics examiner must prove that the captured data was obtained using business record procedures. The second form occurs in the preprocessor and detection rule criteria used to identify the security intrusion or security violations. The last form is the IDS alerts generated and saved as a log file or in a database.

If we use Snort, we can collect multiple types of generated evidence for network forensics. Therefore, by using the Snort, the network forensics investigator must plan for and address this issue as soon as possible before the collection of any must network-based evidence within the organizations.

4 Commercial Network Forensics Applications

In this section, we will discuss the operations of the commercial network forensics tools, which use the evidence mentioned previously and are dependent on devices and services.

4.1 Commercial NetFlow Application

NetFlow is a technology developed by Cisco that collects and categorizes Internet Protocol (IP) traffic as it passes through the supported network devices. NetFlow runs on many Cisco Internetwork Operating System devices and a handful of third-party solutions from Linux, Jupiter, and FreeBSD. It is used for network traffic accounting, usage-based billing, network planning, and security and network monitoring.

NetFlow is built into supported devices and records all IP traffic passing through specific device interfaces. NetFlow does not collect or export the entire payload of network packets. It creates a cache on the router for each network flow. The NetFlow determines whether each packet is related to individual flow or not by scanning the following seven fields: source IP address, destination IP address, source port number, destination port number, IP, type-of-service byte, and input logical interface. If the seven fields are matched to an existing flow, the byte count for the flow entry is incremented within the device cache. If the fields are different, then the packet is considered part of the new network flow.

NetFlow focuses on the flow rather than the full packet capture. This allows it to keep up with the increasing speeds and utilization of business networks. Packet sniffers capture full data content, including packet payloads, but they lose effectiveness as the network gets busier due to the sheer volume of the duplication effort. However, if we step back from the packets and analyze the flow, it can be greatly reducing the amount of data needed to be analyzed and make it simpler to identify any suspicious traffic for future investigation.

One of the NetFlow analyzers is the Scrutinizer, which provides an extremely granular view into the network-utilization information for resident devices and applications. Once it is configured, the interface point of NetFlow flows the data either directly to the scrutinizer or indirectly using a collector that will in turn forward it to the scrutinizer. Scrutinizer is the central aggregation point for network-wide flow utilization and historical traffic patterns within the environment.

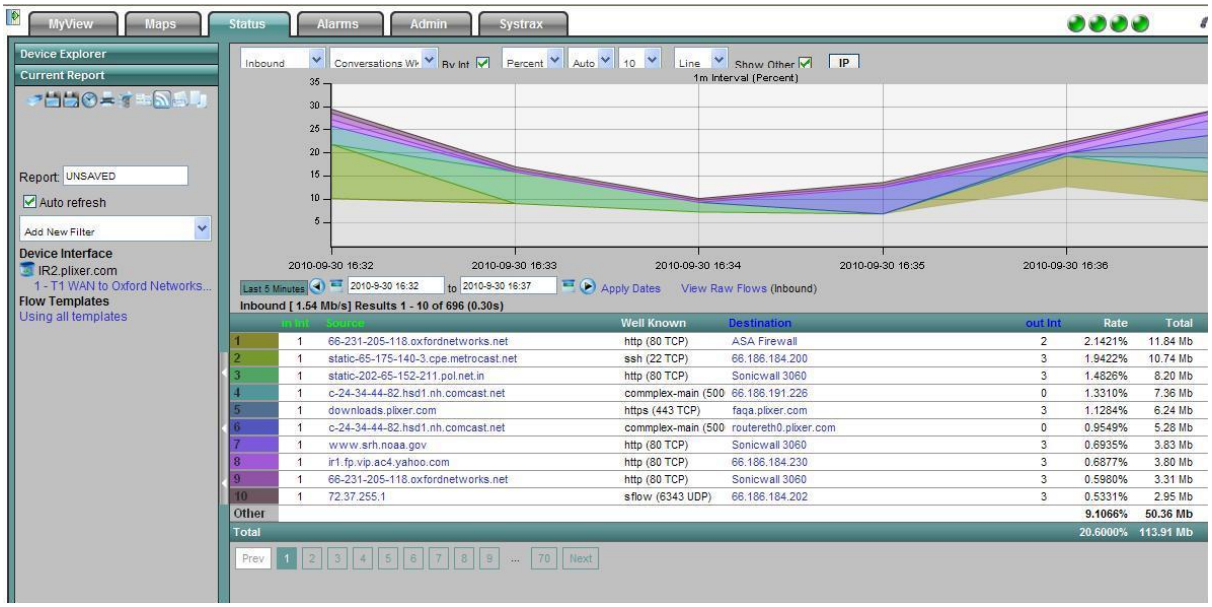

Fig. 8. Default Graphical Report of NetFlow Presented by Scrutinizer (*Scott, 2016*).

Figure 8 shows the graph report of the NetFlow by the scrutinizer. As we previously mentioned, the scrutinizer can be a useful tool for forensic investigation that supports the analysis of the NetFlow. One dimension of NetFlow analysis is to have NetFlow analyze the packets traversing the configured interfaces and forward the key data elements back to the central collector for the event. For the reason the list of fields exported by NetFlow is short, we can use these fields in flow analytics.

Flow Analytics contains dozens of default algorithms that look for odd behavior patterns by searching NetFlow data which is received by select routers and switches. The run time of each algorithm is tracked, as well as the violation count. Since the scrutinizer and flow analytics are very comprehensive tools; we can use the following tools for the network forensics investigation.

4.2 Net Witness Investigator

Fig. 9. Components of Net Witness Investigator (*Lillard, 2010*).

The Net Witness Investigator tool, which is a Microsoft Windows-based application, enables the network forensics examiner to audit and monitor the network traffic by analyzing the captured network traffic through unique investigative lenses. These lenses allow the network forensics examiner to conduct different network traffic analyses using various types of customized filters. There are six main components in the Net Witness Investigator tool which are presented in Figure 9: import / live captures, collections, parsers / feeds rules, navigation views, data analysis, and export captures.

The first component is import / live capture of the network traffic. Net Witness Investigator provides three possible ways to insert the network packet data into the network forensics application. The first option is the one that allows the downloading of captured network data from previously deployed Net Witness remote devices. The second option allows real-time capturing of network data using a local wired or wireless network interface. The network interface can be configured in stealth mode to make the device appear logically invisible. The third option is importing the previously captured network data, which allows the pre-captured network traffic to be read as file-based input.

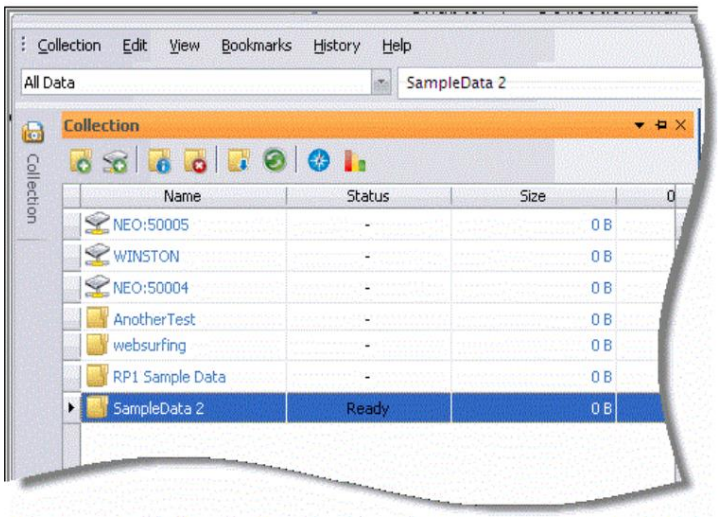

Fig. 10. Collection naming in New Witness (*Net Witness Investigator User Guide* 2020).

The second component is used after capturing or importing the network data, which uses the collections of the network data. Net Witness investigator stores the network traffic into a component known as collections which are presented in Figure 10. Collections are the logical grouping containers that store the unique sets of captured network packet data before processing the network traffic.

The third component is parsers, feeds, and rules, that are used to process the captured network traffic. Each of the three components provides predefined metadata values to conduct, organize, and present the captured traffic in an easy-to-review format for detailed analysis by the network forensics examiner. The parsers are used to process live or imported captured network data by decoding the network traffic by user-customizable or predefined metadata values. The feeds are process applications that use metadata values that are extracted from various external sources to create the metadata to process the captured network data. The rules are used to filter out network traffic that matches specific predefined patterns.

The fourth component is navigation views that enhance the analysis process for the network forensics examiner rather uniquely. The Net Witness Investigator allows users to display and arrange various views of the captured network traffic after parsers, feeds, and rules are applied to conduct a visualization analysis of the captured network traffic. This visualization of the network traffic allows the network forensics examiner to perform a detailed analysis of the captured network traffic more efficiently when they use the different data analysis techniques.

The fifth component is data analysis. This allows the network forensics examiner to conduct detailed ad-hoc and what-if analyses for specific network traffic patterns of normal suspicious or abnormal behavior to determine the occurrence of malicious activities. One of the unique terms in Net Witness Investigator is Breadcrumb, which enables the network forensics examiner to drill up and down throughout the capture network traffic, thus creating a data-analysis path. The data-analysis path represents the selection of different elements within the captured and processed collection traffic.

The last component is exporting the captured data. This component allows the network forensics examiner to extract the data of evidentiary value from a collection. The extracted data can be saved in pcap format.

Using the Net Witness Investigator tool, the network forensics examiner analyzes the captured network traffic. From the options that are supported by the Net Witness Investigator tool, such as import / live captures, collections, parsers / feeds rules, navigation views, data analysis, and export capture, the network forensics examiner can efficiently analyze the network traffic.

4.3 Silent Runner by Access Data

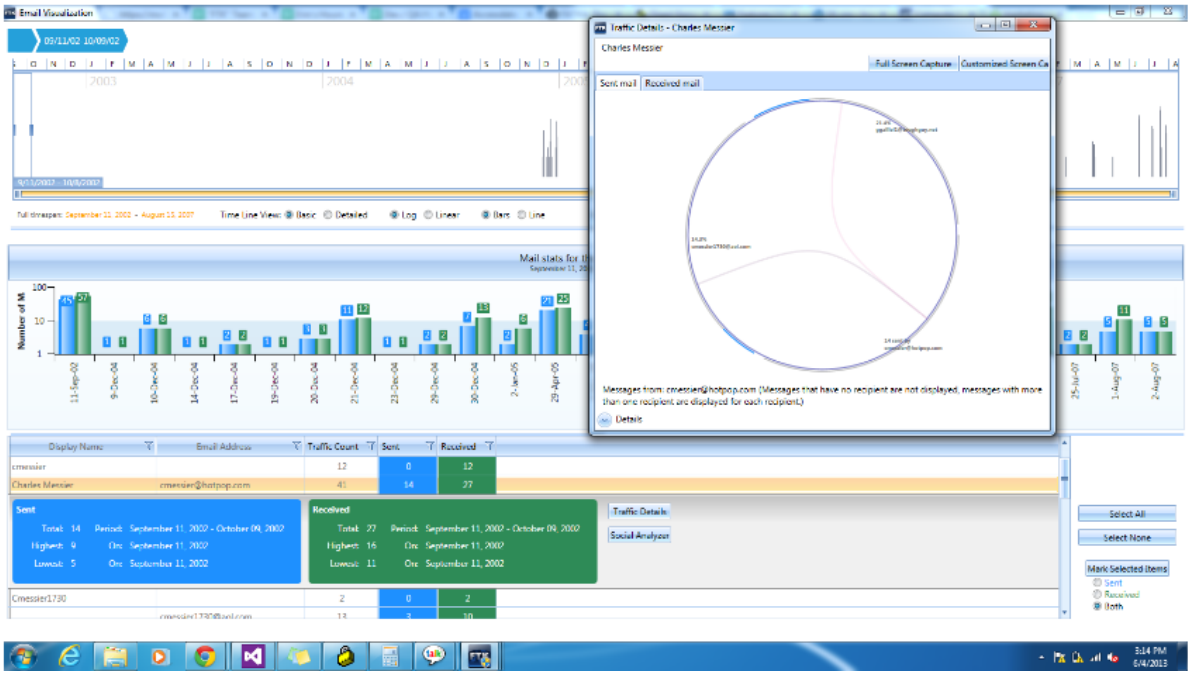

Fig. 10. Silent Runner (*Access data Silent Runner®*).

Silent Runner is a network forensics tool built by Access Data. It is an application that is designed to work together, offering data capture, analysis, and visualization of the data. This tool includes the loading of the data into a relational database to provide complex query and correlation abilities. The main parts of the Silent Runner system are the Collectors, Loaders, Database, and Analysis Workstation. Collectors capture network traffic through their available network adapters. Loader facilitates the transfer of data from Collectors into the Database. The Analysis Workstations either perform queries against the database or import logs files and recreate the visualizations and reconstructions of the data.

Silent Runner is one of the venerable network forensics tools. Having been commercially available for around a decade, it has been deployed in many environments and has seen continuous development. The following tool provides the ability to visualize the network and other data and perform real-time traffic analysis from a centralized database.

5 The Future of Network Forensics

In this section, we will discuss the limitations and futures of network forensics and the application of ML technologies in network forensics, particularly on network traffic analysis.

5.1 The Future of Network Forensics

To effectively prosecute those who commit crimes with the use of networking technologies, the criminal justice system mandates the effective identification, preservation, analysis, and presentation of evidence to the courts. It is pervasive network technology and the enormous number of crimes committed within the network that is driving the future of network forensics.

Even though the hardware-based and software-based solutions, which are already mentioned in the previous sections, provided various aspects of security, still the network forensics that requires court-admissible evidence is very limited or not provided at all. The following information is today's challenges for the network forensics examiner to solve the current limitations.

The first is an analysis of encrypted traffic. End-to-end and link encryption technology prevents captured network traffic from being analyzed, such as pattern matching for malware, for security violations.

The second is a visual analysis of network traffic and log data. Many computers, networks, and security devices do not allow visual analysis to be performed on the network traffic and log data unless it is commercial software that is dependent on the devices. Therefore, if the raw log data can be visually represented in some sort of data, the examiner can easily analyze the network traffic.

The third is hashing captured network evidence. Most computers, networks, and security tools do not produce hash values for captured data or utilize the same hash algorithms. Without the collection of hash values, the integrity of the data can be called into question.

Still, there are more challenges in the field of network forensics. The goal of the new tool is to provide the basis for the network forensics examiner to address the criminal justice systems' mandate, the driving demands of organizations seeking effective network forensics tools, and the on-site and off-site investigative tools of the examiner based on the crimes committed with the use of the networking technologies.

5.2 Application of Machine Learning to Network Forensics

In the previous sections, we have been through the importance of traffic classification in network forensics, evidence of network traffic, and the tools that help network forensics examiners classify the network traffic from the evidence. Nowadays, due to the enhancement of machine learning techniques, researchers started to apply machine learning technologies to network forensics in the aspect of network traffic classification. Mainly, there are three board categories of significant works published in ML-based IP traffic classification: clustering, supervised, and hybrid (Nguyen, 2008).

The first is the clustering approach. It is related to the works whose main approach centers on works published with unsupervised learning techniques. Flow Clustering using Expectation Maximization (Nguyen, 2008) published one of the earliest works that applied ML in IP traffic classification using the Expectation Maximization algorithm. The approach clusters traffic similar observable properties into different application types. Automated application identification using Auto Class (Nguyen, 2008) is the work that uses Auto Class, which is an unsupervised Bayesian classifier, using the EM algorithm to determine the best clusters set from the training

data. EM is guaranteed to converge to a local maximum. To find the global maximum, the auto class repeats EM searches starting from pseudo-random points in the parameter space. The model with the highest probability is considered the best. TCP-based application identification using Simple K-Means proposed a technique using an unsupervised ML algorithm that classified different types of TCP-based applications using the first few packets of the traffic flow. Identifying web and P2P traffic in the network core (Nguyen, 2008) is the work that addressed the challenge of the traffic classification at the core of the network, where the available information about the flows and their contributors might be limited. The work proposed to classify a flow using only single directional flow information.

The second is supervised learning approaches. It is the work whose main approach centers around supervised learning techniques. The statistical signature-based approach using NN, LDA, and QDA algorithms (Nguyen, 2008) proposed the use of the nearest neighbors (NN), linear discriminate analysis (LDA), and Quadratic Discriminant Analysis (QDA) ML algorithms to map the different network applications to predetermined QoS traffic classes. Classification using Bayesian analysis techniques (Nguyen, 2008) proposed to apply the supervised ML Naïve Bayes technique to categorize Internet traffic by application. Traffic flow in the dataset used is manually classified, allowing accurate evaluation. Real-time traffic classification using Multiple sub-flow features (Nguyen, 2008) noted a method to address the issue by proposing classification based on only the most recent N packets of a flow, which is called a classification sliding window. The use of a small number of packets for classification ensures the timeliness of classification and reduces the buffer space required to store packets' information for the classification process. Real-time traffic classification using Multiple Synthetic Sub-Flow pairs (Nguyen, 2008) extended their work to various environments. The purpose of this paper is to train the ML classifier using statistical features calculated over multiple short sub-flows extracted from full-flow generated by the target application and their mirror-imaged replicas as the flow is in the reverse direction.

The third is hybrid approaches. These are the works whose approach combines supervised and unsupervised learning techniques (Nguyen, 2008). Semi-supervised traffic classification approaches combine unsupervised and supervised methods. Motivations to the proposal are due to two main reasons. First, the labeled examples are scarce and difficult to obtain, while supervised learning methods do not generate well when being trained with the few examples in the dataset. Second, the new application may appear over time, and not all of them are well known as a priory, traditional supervised methods map unseen flow instances into one of the known classes, without the ability to detect new types of flows.

Also, some works compared the algorithms which were presented previously. Comparison of different clustering algorithms (Nguyen, 2008) compared three unsupervised clustering algorithms, which are K-Means, DBSACN, and Auto Class. Their results show that the Auto Class algorithm produces the best accuracy. In the case of Auckland and Calgary datasets, the Auto Class showed 92.4% and 88.7% of accuracy. K-Means, showing 79% and 84% of accuracy when the K is around 100. DBSCAN algorithm produced 75.6 and 72% of accuracy. In Comparison of clustering vs. supervised techniques (Nguyen, 2008), evaluated the effectiveness of supervised Naïve Bayes and clustering Auto Class algorithm. They are evaluated by the three-accuracy metrics which are recall, precision, and overall accuracy. The conclusion was that the Auto Class has an average overall accuracy of 91.2%, while the Naïve Bayes classifier has an overall accuracy of 82.5%. Auto Class also performed better in terms of precision and recall for individual traffic classes. However, in terms of the time taken to build the classification model, Auto Class took a much longer time than the Naïve Bayes algorithm, which is 2070 seconds vs. 0.06 seconds. Comparison of different supervised ML algorithms

There are many more ML algorithms for network traffic analysis and comparisons for that method. The various techniques and their comparisons are presented in Figure 11.

Work	Real-time Classification	Feature Computation Overhead	Classify Flows In Progress	Directional neutrality
McGregor et al. [48]	No	Average	Not addressed	No
Zander et al. [46]	No	Average	Not addressed	No
Roughan et al. [18]	No	Average	Not addressed	N/A
Moore and Zuev [14]	No	High	Not addressed	No
Barnaille et al. [53]	Yes	Low	Not addressed	No
Park et al. [44]	No	Average	Not addressed	Not clear
Nguyen and Armitage [56]	Yes	Average	Yes	Yes
Nguyen and Armitage [54]	Yes	Average	Yes	Yes
Erman et al. [47]	No	Average	Not addressed	No
Crotti et al. [61]	Yes	Average	Not addressed	No
Haffner et al. [57]	Yes	Average	Not addressed	N/A
Ma et al. [66]	No	Average	Not addressed	No
Auld et al. [55]	No	High	Not addressed	No
Williams et al. [65]	N/A	Average	N/A	N/A
Erman et al. [45]	N/A	Average	N/A	N/A
Erman et al. [64]	N/A	Average	N/A	N/A
Bonfiglio et al. [67]	Yes	Average	Not addressed	Not clear

Fig. 11. Considerations for Operational Traffic Classifications of ML algorithms (*Nguyen, 2008*).

6 Conclusion

Modern computers are network centralized. Also, as the importance of cyber security in the network increases, the importance of network forensics has increased. Therefore, in this paper, we discussed the flow of network forensics. In the first section, we presented the evidence used in the network forensics: DHCP logs, TCP dump and Win dump, Wireshark, Span ports or TAPS, and Firewall. In the second section, we explained how the open-source tools that help the network forensics examiner to analyze the network traffic by using this evidence. The third section explains the commercial network traffic analysis tools, such as Net Flow, Net Witness, and Silent Runner, which are dependent on the devices or the services. At the end of this paper, we discussed the future of network forensics and the application of Machine Learning technologies to network forensics. By understanding this paper, we can get an overview of the field of network forensics.

Citations

[1] Buchanan, W. J. (2010). Introduction to network forensics. Computer Security Journal (Vol. 21). http://doi.org/10.1007/978-1-61779-968-6

[2] Gillis, A. S. (2019, December 3). *What is DHCP (dynamic host configuration protocol)?* Search Networking. Retrieved July 17, 2022, from https://www.techtarget.com/searchnetworking/definition/DHCP

[3] Lillard, T. V. (2010). Digital Forensics for Network, Internet, and Cloud Computing. . Terrence V. Lillard.

[4] Ryšavý, O. (2022). Lab-Encrypted Traffic Analysis. Brno; Faculty of Information Technology, Brno University of Technology.

[5] Scott. (2016, December 28). Scrutinizer users: Which function do you use more, reporting or analytics? Plixer. Retrieved July 20, 2022, from https://www.plixer.com/blog/scrutinizer-users-which-function-do-you-use-more-reporting-or-analytics/

[6] RSA NETWITNESS PLATFORM. (2020). Net Witness Investigator User Guide.

[7] Access data Silent Runner®. Complex. (n.d.). Retrieved July 21, 2022, from https://www.complexbiz.com/2013/07/26/access-data-silent-runner/

[8] Nguyen, T. T. (2008). A Survey of Techniques for Internet Traffic Classification using Machine Learning. IEEE.