

ATHABASCA UNIVERSITY

A FEDERATED APPROACH TO IDENTIFYING AND REMOVING ADVANCED
PERSISTENT SECURITY THREATS ON ENTERPRISE COMPUTER NETWORKS

BY

WADE W. WESOLOWSKY

A INTERIM PROJECT REPORT submitted in partial fulfillment

Of the requirements for the degree of

MASTER OF SCIENCE in INFORMATION SYSTEMS

Athabasca, Alberta

September, 2018

© Wade W. Wesolowsky, 2018

ABSTRACT

Computer Security is an intense flash point of concern in the modern computer landscape. Many threats to computer systems both known and unknown prey on the mind of computer professionals. Advanced Persistent Security Threats (APST) are a species of clandestine attack which infiltrate computer systems to exfiltrate data and struggle to maintain an everlasting foothold inside the target network. They are of particular concern thanks to their use and sponsorship by state actors in proxy battles and covert operations. In this landscape we chart the rise of the threat and search for free and open source software which can be used on an enterprise network to decrease the threat. Uncovering the threat will be a collaboration between the security tools cooperating together as components within a Federated Security Module (FSM) built using the Elasticsearch, Logstash, and Kibana (ELK) Stack. A federation between pfSense, SNORT, Sweet Security (Bro), Wazuh (OSSEC), and Honeytrap will offer a centralized data source from which their combined knowledge can be accessed. The efficacy of the federation will be evaluated using a test suite on a simulated enterprise network to provide us with opportunities for improvement and lessons learned for future research.

TABLE OF CONTENTS

1	INTERIM PROJECT REPORT	1
1.1	The Problem/Opportunity	1
1.2	The Goal	2
1.3	Impact or Significance of the Problem/Opportunity	3
1.4	Potential Causes of the Problem/Opportunity	4
1.4.1	Network Medium	5
1.4.2	Vulnerabilities Everywhere	5
1.4.3	The Value of Data	6
1.4.4	Cyberwarfare	7
1.4.5	System Complexity	8
1.4.6	Countermeasures	8
1.4.7	Aging Infrastructure	8
1.5	Literature Review	9
1.6	Mind Map	12
1.7	Potential Solutions to the Problem/Opportunity	14
1.8	Tool Selection	15
1.8.1	ELK Stack / Elastic Stack	16
1.8.2	Perimeter Firewall	17
1.8.3	Snort vs. Suricata	18
1.8.4	Network Monitoring	18
1.8.5	Host Monitoring	19
1.8.6	Honeypot	20
1.8.7	Anti-virus Software	20
1.8.8	Tool Summary	21
1.9	Penetration Testing Lab Network Setup	21
1.10	Security Software Configuration	26
1.11	Security Software Setup Within the Network	28
1.12	Difficulties and Future Recommendations on Setup	28
1.13	Tests	31
1.13.1	Baseline	31
1.13.2	EICAR Test File Download	33
1.13.3	APST Testing Suites	35
1.14	Assessment of Test Findings	35
1.14.1	Results and Future Work	37
	REFERENCES	38
	Appendix A Raw JSON Log Files	54

LIST OF TABLES

1.1	Attack techniques and countermeasures in each stage of an APT attack taken from [1, Table. 3]	16
1.2	Tool Summary	22
1.3	Log Files Gathered by the ELK Stack	30

LIST OF FIGURES

1.1	APST Mind Map	13
1.2	APST Lifecycle Model With Selected Tools for Identifying and Removing Threats derived from [2, Fig. 1]	23
1.3	Penetration Testing Lab Configuration	24
1.4	Security Software Setup Within the Network	29
1.5	Example Snort Alerts	32
1.6	Generic Device Information in Sweet Security	33
1.7	Expanded Device Information in Sweet Security	34
1.8	New DNS/IP Entries in Sweet Security	34

CHAPTER 1

INTERIM PROJECT REPORT

1.1 The Problem/Opportunity

Curiosity surrounding ideal ways of protecting Information Technology (IT) assets from security threats is a growing area of study. The interconnected nature of computer systems in the modern world makes this problem even more difficult due to the myriad of ways devices can communicate and exchange data on a network. All the components in a computer network, including the human operators, introduce attack vectors for a determined adversary to exploit. The system administrator and other computer professionals tasked with protecting the network will often take baseline precautions to mitigate these threats. These precautions might take the form of frequent patches, password policies, user education, running anti-virus software, or other protection measures. Often baseline protections are adequate when trying to prevent damage from automated, routine security threats like spam and botnets. However, they often fall short when there is a human intelligence directing the attack - when the attacker takes the time to learn the network and tailor the attacks to the vulnerabilities therein. One exercise often undertaken is to hire a penetration tester to attempt a break-in of the network in order to expose and report on potential vulnerabilities [3]. Often, the attempt is successful at illustrating systemic problems that make the total compromise of the computer system possible. The custodians of the network often ask how this is possible when baseline security precautions were taken and studiously followed.

The inevitable conclusion must be that baseline network security is inadequate to the security challenges present in a modern computer network. Even when a minor system glitch is detected, it can be a warning indicator of a much larger and deeper issue [4]. Without the ability to police these security threats, there is little chance of the enterprise network remaining secure. While many threats exist, we would like to focus on one of the most salient threats which is the Advanced

Persistent Threat (APT). APT can be defined as 'a sophisticated attack composed of several steps and performed by experienced and highly skilled actors with a great determination to achieve their goal' [5, p. 1][6]. These threats are *advanced* due to their sophisticated nature and multi-staged methods used to attack the IT system. They are *persistent* because their foothold within the network is non-volatile and tends to be of a permanent nature. They are a *threat* due to their sneaky, often undetected, exploitation of vulnerabilities within the computer system. Within our terminology we have scoped the definition of APT as Advanced Persistent Security Threat (APST) to highlight our focus on computer security threats versus other threats which maybe be present on computer systems such as coding errors, bit rot, data loss, etc.

Since APST are so difficult to mitigate there is ample opportunity to explore available options for their detection and removal. This field is also exploding with the rise of malware and hacking tools created by state actors [7], and we see only expansion within this domain. Some writers even suggest the deployment of systems to monitor your employees to watch for insider threats [8]. There is no lack of innovation, and we hope that our Project will add further ideas to this endeavor.

1.2 The Goal

The research objective is the study of existing tools and methods which can be leveraged to detect and remove APST within enterprise computer networks. After a review of the literature, a federated security model (FSM) design will be proposed and implemented to integrate select tools and techniques together in a federated manner which will help to mitigate threats and vulnerabilities within an enterprise environment. The primary goal of the FSM will be detection of intrusions and vulnerabilities, while the subjugate goal will be removal or elimination of the identified threats or vulnerabilities.

Creating the software to perform the federation of various software security tools will be a key focus of our work. Due to the federation of components, there will be problems of overlapping, non-deterministic tool behaviour, and mismatch between the tools while still having the require-

ment to maintain interoperability between separate components [9].

Another goal of our research will be to prototype our FSM on a simulated enterprise network. This will require the setup of a lab environment running similar services and roles that might be present in an enterprise network environment. Then we would run the components of our FSM which will be used for the detection of APST. Since these threats are often directed by a human intelligence, the research will implant several types of malware within the enterprise environment to observe the efficacy of the FSM at finding the threats. An analysis of how the FSM performs during the simulation should suggest additional avenues for future research.

1.3 Impact or Significance of the Problem/Opportunity

There has been a consistent growth of security threats which target computer systems and networks over the years[10]. Each year many new and previously unknown threats are discovered by security researchers who studiously document them along with potential fixes for their exploited vulnerabilities. This happens with such a regular basis that anti-virus software must be updated daily and most software vendors have moved towards software patching models which allow for a consistent delivery of updates and upgrades to their software over the Internet[11].

Specifically in the realm of APST the sophistication of threats has slowly advanced as researchers find and analyze threats on a continual basis. When a new threat is discovered often it is reverse engineered by security researchers[12] in order to get a better idea of how it is constructed. This gives valuable insights into how the APST functions and sometimes even gives information about its source. However, it also allows researchers to learn new techniques of software exploitation which can be used to create new malware.

We would strongly believe that most corporate and government networks have already been compromised in some way and they are simply not aware of it. This is because one of the important features of APST is avoiding detection[13] in order to succeed at maximizing data exfiltration from the compromised network. Without tools and techniques to at least find this threat it will continue

to plague the network [14] and most likely allow long term loss of corporate secrets and other confidential data.

A quick browse of WikiLeaks is all that should be required to justify the present research. It is quite clear that software exploitation has become a central focus within the Central Intelligence Agency (CIA), especially in relation to the Vault 7 leaks which have been ongoing [15]. There is an important section regarding how to evade forensics and anti-virus software [16][17][18][19]. This information demonstrates that state actors are well versed in evading detection while exploiting target systems and exfiltrating data. There can be no doubt that such practices are ongoing today.

We have argued that the risks to network security are only growing and shown how one state actor (CIA) may be developing exploitation tools. Within this space there is an urgent need to for better defense techniques which will help to alert security professional to potential risks within their enterprise networks. Furthermore, the arguments presented tend to suggest that this defense should be able to handle unknown threats. The impact of the research could be great depending on the programming skill of researcher, how astute they are to federate tools together, and the novelty of their ideas!

1.4 Potential Causes of the Problem/Opportunity

In this section we explore some of the reasons why APST have become such a large nuisance in enterprise networks along with some of the reasons they are becoming an even more prevalent threat. The reasons explored include the expansion of network medium, the perpetual discovery of vulnerabilities, the ongoing value of data contained within the networks, the specter of cyberwarfare, complexity of enterprise networks, the evolution of countermeasures, and aging IT infrastructure. All these areas collectively illuminate reasons for the proliferation of APST and highlight some of the problems and opportunities present in that space. We do not claim these potential causes are exhaustive or complete.

1.4.1 Network Medium

Computer networks were historically limited by physical wires which connected the computer systems. However, these days the network media could be cellular or wireless signals. There is less reliance on wired connections and an increasing focus on non-wired devices. This means that any computer system that is able to send and receive is at risk from nefarious signals. This risk can even extend to computer microphones and speakers which can be used to transmit and receive in a local area [20]. In general, an air gap between systems is no longer a deterrent to information leaving your computer [21]. Additionally, these types of attacks are difficult for the user to know about because humans cannot perceive radio frequency signals or hear the ranges that the malware transmits at and thus there is no warning signs that malware is working against them.

1.4.2 Vulnerabilities Everywhere

Each day software vulnerabilities are discovered and exploited. All one has to do is glance at the Bugtraq Mailing List or the Full Disclosure Mailing List to get a sense of how frequently this occurs. The onslaught of such notices gives even the most stoic reader pause, and highlights just how frequently software contains bugs, undocumented features, or other quirks which allow it to be compromised in so many ways. The sheer number of vulnerabilities shows the dangers of using an unpatched system, and even if you are using a patched system it shows that it would be impossible to trust your software to be bug free. The simple fact is that no matter which system you are running someone will have an exploit for it.

As a software consumer, there is even the risk that the software you have purchased contains back doors or other side-channels which allow covert access and surveillance. One long running debate is that you should have access to the source code of products which you run on your computer and whether that will contribute to the security of the software [22]. Without auditing the source code, it is very difficult to trust that your computer is executing the code you would expect.

Even areas of the computer system which historically have been considered safe are now coming under attack. There are well known techniques for re-flashing the BIOS of a computer system and embedding malware inside it [23]. There are many places for malware to hide, "between Linux and the hardware are at least 2 1/2 kernels" [24]. A more recent scare has broken out regarding the use of the Minix operating system within Intel processors, "Thanks for putting a version of MINIX inside the ME-11 management engine chip used on almost all recent desktop and laptop computers in the world"[25].

1.4.3 The Value of Data

All organizations contain valuable data. This data often has technological and commercial value, which means that other people would like to gain access to this information. APST would be a great way for an actor to gain and maintain access to this data. The type of data which might be stolen can be almost anything from passwords to behavioural profiles.

There is an argument made that malware may be used in a stealing-reality attack [26]. This means that malware may be present in a system specifically to slowly spread and steal data for behavioural analysis. This type of attack is deadly because it can be used to build profiles on specific human targets on patterns of their behaviour that may not be easily changed.

Malware can even be used to gain political [27] advantage over opponents. We can imagine the computer accounts of a powerful people being compromised by an attack in order to track or blackmail these people. It is apparent from the Edward Snowden leaks [28] that this type of surveillance has already been wildly implemented by the National Security Agency (NSA). APST definitely plays a role in these types of attacks, as often computer systems must be compromised over the long term in order for reliable intelligence to be extracted over a long time horizon. Furthermore, often the adversary has an IT environment that is configured in such a way that generic attacks would be unsuccessful, so custom advanced attacks are necessary.

Another recent article [29] points out that cracking computer networks to monitor news reports in order to find information which could affect stock prices, and then using that information to place

stock trade requests is a viable way of using APST to make money. When criminal enterprises are able to gain access to information exchanges they have the ability to use that information to gain monetary advantage.

1.4.4 Cyberwarfare

With nation states taking notice of the importance inherent in Information Technology infrastructures and their ability to subvert these infrastructure for various ends, there has been some dialogue about cyberwarfare. The digital frontier can be thought of as trench warfare in many respects, as it is still new and untested. The best defenses are multi-layered [30], however the visual of many trenches might be more apt. An aerial bombardment which bypasses the trenches, shows how inadequate this method of defense might be. We definitely see that superpowers [30] often blame each other for network intrusions and lack of cooperation in these issues.

While the specter of cyberwarfare is often maligned, the threat of other types of operations, like PsyOps [31] show that gaining control of a system might allow you to plant disinformation. So, while stealing information might be important - planting information might be just as effective. Therefore, securing systems often means that you need to make sure the information in them is not modified or altered in ways which benefit your adversary.

Cyberwarfare may not be limited to state groups, and many of the current threats originate from elsewhere [32]. The conflict between hacker groups (which happen to be different national groups) are in some cases international conflicts of ideology. This paper accepts that using patriotic groups within the country to advance the national agenda is possible.

Irregardless of the actors in question, it is clear that cyberwarfare is a discussed topic. While at this time there is scant evidence of a true cyber-war happening between nation states. There is evidence for skirmishes between long time rivals, like the USA and China. Any large company tied to a nation state would be wise to consider the possibility they may be targeted in the future and plan accordingly.

1.4.5 System Complexity

The enterprise network environment is always trying to push the boundaries to offer new services to its users [33]. We can see the shift in the last few years to bring-your-own devices and allowing workers to work from home using company provided computer systems. Within the network itself, systematic design of VLANs and other interconnectedness is often poorly understood [34]. This means that often the corporate network environment is in a constant state of redesign engineering flux where new threat avenues may be created from many different angles. Furthermore, the sheer complexity of the environment often means that no single individual has a full picture of how all the services work together to create a unified whole. Often there is many specialists which oversee their own small parts of the enterprise network. This is exacerbated with the often inadequate communication that exists within corporate structures. Should a security threat find its way into an enterprise network removing it will be a difficult challenge and it may be impossible to verify that it has been removed completely.

1.4.6 Countermeasures

APST have undergone many different evolutions over the years, and their writers have come up with more and more advanced countermeasures to prevent their detection and removal[35]. This has been briefly studied using such systems as BEAGLE[36] which show how malware authors refine their malware over time. It is important for us to realize during our research that new techniques will be created which may make our detection tools ineffective or obsolete and thus the current research cannot arrive at a perfect solution. Rather a balance will need to be supposed which allows adequate protection on the enterprise network.

1.4.7 Aging Infrastructure

No analysis would be complete without the mention of aging legacy IT systems present within any computing environment. From rumors about a forgotten server hiding in a back room or the

necessity of keeping an aging computer system operational to ensure business continuity, often the enterprise depends on legacy IT systems. These systems present a unique challenge and their modernization is a concern to computing professionals[37]. However, these systems present a unique target for malware as they are often poorly understood and often sparsely maintained. Furthermore, skill sets within industry for their operation are often sparse or difficult to find [38] so they may be maintained by system administrations who do not often understand their full complexities.

1.5 Literature Review

Our short literature review used online databases to search for topics related to "Advanced Persistent Threats" and "APT". The five main databases queried were: ScienceDirect, IEEE Xplore Digital Library, ACM Digital Library, Google Scholar, and SpringerLink. We found the the Google Scholar "related article" feature particularly helpful in tracking related articles which were indirectly related to our topic of interest.

APST have been documented to have several distinct phases in their life cycle [2][39][40][41]. While these stages may differ between published papers, it is widely accepted that humans are exploited in the early stages of attack[42]. The stages also provide differentiation for the strategies that can be deployed for detection as summarized in Table 1.1. Therefore, we will briefly look at the various stages as identified in the papers for completeness of the literature review and to make sure the reader is familiar with them. Also, depending at which stage an infection is detected suggests potential areas for removal or containment of the malware within the enterprise network. For a real-world analysis of APST stages of attack and some of the related actions taken by the malware at each stage please see [43].

The APT attack stage model adopted during our investigation will be the one contained in [2]. This model is selected because it is compact with four stages, each stage is clearly laid out in the paper presented. It is not as in depth as the phases identified in [41], but nevertheless we believe it is sufficient for our purposes and you can refer back to the comparison between the models for

comparisons. This model has four main phases: prepare stage, access stage, resident stage, and harvest stage. Within each of these main stages there are several sub-stages which the threat actors will take [44].

One of the high level methods of mitigating APST is to model the network and the various vulnerabilities that exist on it. One method which could be used by a security analyst is the Optimized Attribute Attack Graph which models the network and potential attack vectors and methods [45]. This graph can show the potential attack avenues for multiple hosts in the network, however the authors do not present an automated method of generating these graphs so usefulness would be limited for us. The idea of presenting a holistic view of the network lead us to another paper which presents an ontology based approach where various data providers can contribute to behaviour detection on the network. This approach known as TAON gives, "an OWL-based ontology offering a holistic view on actors, assets, and threat details, which are mapped to individual abstracted events and anomalies" [46]. This method will aggregate data from various data providers which are software components which provide information. This is a breed of behavioral detection systems which are posited as the next generation of APST detection methods because they model the stages of attack and try to use those as methods of detection. Unfortunately, this approach did not come with a corresponding software implementation so while the idea is novel we cannot implement it during our project. However, it does highlight the use of semantic web capabilities for the detection of threats. Also, it provides a assistant with data providers which could be used on a network for detection.

Most detection methods usually rely on network traffic analysis and inspection[47]. One paper which stood out was [48] which compared the network monitor implementations which were broken into packet capture, deep packet inspection, and flow-based observation. The paper brought to our attention Bro which is flexible and allows the creation of various detection methods. Another method of APST network detection is to detect traffic flows for Remote Access Tools running within internal enterprise networks [49]. The inspection of traffic by a host-based detection mod-

ule on each IP enabled device was an effective way of finding over 90% of the threats of infected systems. It is important to monitor internal hosts because often defensive strategies use an eggshell approach where the perimeter firewall is the only place where attacks might be stopped. The collaborative monitoring of hosts within the network is important to stop the lateral movement of APST within the network [50]. A final paper presents various real world recommendations for detection of threats on the network [51] however it is only a proposal and does not present real-world results for analysis.

Another proactive defense strategy is the use of honeypots which sit on the network and mimic real systems to lure APST to them[52]. The use of KFSensor was cited as the selection solution by this author, for our uses we will need to select more open source tools for our investigation. However, the main idea is the use of proactive notifications which are sent to the network administrator when the honeypot receives incoming network traffic. A recent source of information regarding honeypots can be found in [53], [54] which provides many examples of existing honeypots to choose from.

The removal of APST was not an easy find in our investigation. When a network is compromised it would be prudent to simply take all infected hosts off line and rebuild them from known good backups. The detection components can be used to show when a host is compromised,”Across the first organisation’s estate of 5,000 plus machines, there were four that were infected with the malware” [55], and then a backup solution can be used to restore the host after forensic analysis. Removal becomes even more difficult because often the best malware authors build on the successes and mistakes of other malware authors when creating their malware designs [35].

During the search for similar tools we came across Open Source Security Information Management (OSSIM) which is a collaboration of open source tools to perform security and asset monitoring. While not specifically tailored towards APST they may use some tools which are similar to our proposed solution. Also, their solution is more broadly based to take into account a

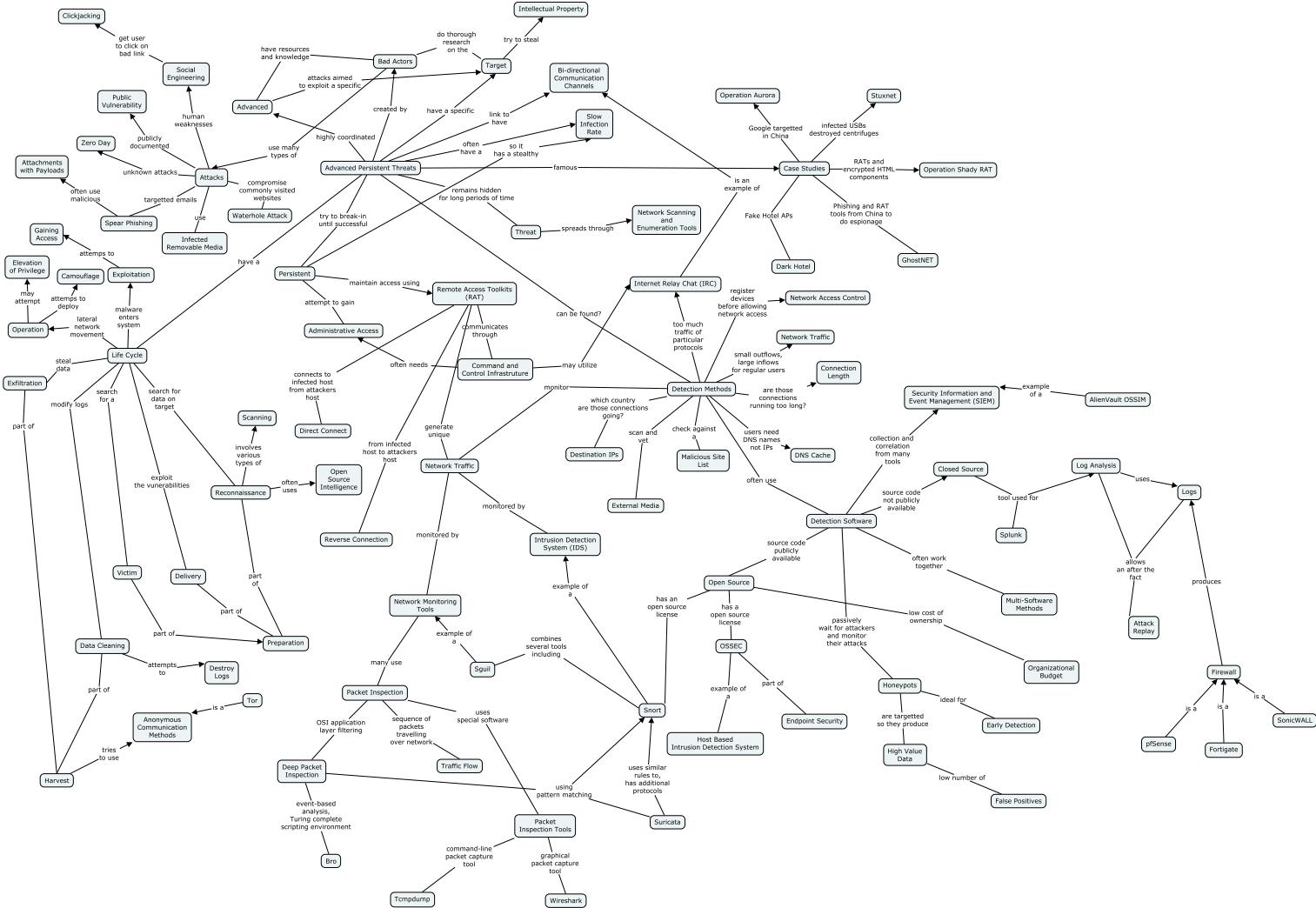
wider variety of threats.

The life cycle of the APST is important when considering different detection and removal methods for them. There is much research done into detection, but much less research was discovered for removal.

1.6 Mind Map

The Mind Map is a visual representation of interconnected concepts to assist in finding non-obvious connections. We have used it here to summarize some of our initial investigation into the properties of APST. The mind map in Figure 1.1 was created by using CmapTools. In order to fit it on the page, we have had to scale the image down to make it fit.

Figure 1.1: APST Mind Map



1.7 Potential Solutions to the Problem/Opportunity

There are many claimed solutions to the APST scourge, but many of them are difficult and costly to implement. We will attempt to secure our simulated enterprise network using free and open source tool. Therefore, while some proprietary security products perform well in this space we discount them from our analysis [56]. The main reason for this is due to funding constraints and to make the research more accessible to a wider audience.

In this section we will explore some solutions suggested by the literature review which might help to assist us. One of the most cited defense strategies is simply *user education about the dangers of cyberspace*[1]. However, we are going to focus on specific technologies we can implement within the enterprise to attempt to secure it better.

Some initial suggestions are good, and a first look presents some novel areas to consider. A PhD thesis gives voluminous suggestions without recommending any specific pieces of software [57]. A high level analysis provides many examples of signature based, anomaly based, and specification based detection methods [58]. An Intel presentation gives us some future considerations about how Intel Chips may be protected against these threats [59]. We also encounter general advice that all security appliances and software should be kept up-to-date [60], furthermore we should limit physical access to all facilities. These are excellent starting suggestions, but do not provide the concrete recommendations we need for our purposes.

Force the DNS servers on the local network to make use of fast-flux DNS domain [61]. Also, by limiting access that malware has to Command and Control servers prevents it from updating itself. The [61] paper gives us a list of several countermeasures which are highly recommended, which include: patch management, network segregation, white listing common Internet traffic, controlling access to software and hardware within the environment.

In [62] additional suggestions are given at the end of the paper. They include controlling all external media, making sure the endpoint security is kept up-to-date, and implementing network access control. Some of the tools recommended in this paper for network monitoring are OSSECC,

Snort, Sguil, and Splunk. However, some papers [63] only give general advice on how to monitor the network but do not provide us with a software tool recommendation for the endeavor. An additional paper expands on these tool recommendations and offers more granular suggestions [64].

Other papers [14] offer some suggestions about using the Microsoft Enhanced Mitigation Experience Toolkit to stop attacks, but this toolkit is currently end of life, so is not useful for us. This paper does suggest, "we propose to implement a detection approach that monitors if a process requests a handle to the LSASS process and performs suspicious function calls with the help of this handle.", however no source code is offered for this task. The final recommendation for Windows log analysis is indeed good, and we think having a tool which will be able to look at all the logs in a reasonable period of time could be useful.

The search for APST[1] must take into account event anomaly detection, data loss prevention. This paper presents an excellent summary of the defense strategies that can be employed at each stage of the APST life cycle, we have reproduced it in Table 1.1. This discussion illustrates the ongoing observation that different countermeasures should be used to stop APST depending on the attack stage it is currently in.

1.8 Tool Selection

The selection of tools for our project was important for the successful detection of APST. We wanted to find tools which met several criteria which included: open source driven, active development, supportive user base, well known and proven technologies. Moreover it was imperative to make sure that we selected tools which were supported by our literature review to ensure they had the backing of our peers when dealing with APST. Please note that the four stage lifecycle model [2] was our chosen model for APST attacks. We will try to link each tool to the stages it will be most effective for detection of APST.

Table 1.1: Attack techniques and countermeasures in each stage of an APT attack taken from [1, Table. 3]

Stages	Attack techniques/tools	Countermeasures
Reconnaissance and Weaponization	OSINT, Social engineering Preparing malware	Security awareness training, Patch management, Firewall
Delivery	Spear phishing, Watering hole attack	Content filtering software, NIDS, Anti-virus software
Initial Intrusion	Zero-day exploits, Remote code execution	Patch management, HIDS, Advanced malware detection
Command and Control	Exploiting legitimate services, RAT, Encryption	NIDS, SIEM, Event Anomaly detection
Lateral Movement	Privilege Escalation, Collecting data	Access control, HIDS, NIDS, Event Anomaly detection
Data Exfiltration	Compression, Encryption, Intermediary Staging	Data Loss Prevention

1.8.1 ELK Stack / Elastic Stack

The federation of software tools was at the heart of our project. It was imperative to have a method of getting the tools to work together to solve the problems faced when hunting for threats within our enterprise network. In order for the federation to be successful, we needed a straightforward way for all our components to work together without having additional overhead or the need for extensive programming or modification. Originally, we considered the use of Splunk [65], as the marketing material told us, "Splunk is uniquely suited to collect, index, correlate and analyze all data, and to monitor patterns of activity over the very long periods of time required to see a potential attack." However, Splunk often has a high cost associated with it [66] (unless using the limited free edition), so the search for alternatives lead us to the Elasticsearch, Logstash, and Kibana stack commonly referred to as ELK. The combination of these technologies offered the open source alternative to Splunk which satisfied our need for a system which has the capabilities, but not the cost. A logging system is important [67] as it offers the ability to figure out what has happened on a system, which provides transparency and a record of past events which can be audited in the future. The ELK stack was a perfect example of the technology we were looking for

to provide the heart of our FSM.

ELK has three main components:

- Elasticsearch [68] is a "real-time distributed search and analytics engine".
- Logstash [69] "provides an integrated framework for for log collection, centralization, parsing, storage, and search."
- Kibana offers a framework so you can create graphs, charts, and visualizations using the data collected in Elasticsearch.

The three tools within ELK provide a complete framework for gathering logs from many different sources, storing and parsing them so they can be searched in an efficient manner, and finally a way of creating visualizations and dashboards to view the logs in a user acceptable manner.

All selected tools must be able to log to the ELK stack for further processing and completeness. The method of logging may operate at slightly different layers within the stack using either Logstash or Elasticsearch.

1.8.2 Perimeter Firewall

Our literature review suggested that it was important to have a perimeter firewall when defending against attacks [6]. Finding a firewall which is both open source and has a solid reputation was difficult, but eventually we decided on the use of pfSense. pfSense is an open source firewall based on FreeBSD which claims to be the leading open source firewall [70]. The firewall is able to run on a standalone computer system which suits our purposes, as we acquired a computer with two networking ports for a reasonable price. Furthermore, pfSense comes with an extensive list of add-ons which extend its existing functionality which is important when adding an IDS to our system. A comparison of various open source firewalls concluded that pfSense [71] was, "...found to be a financially effective solution because of its easy upgradability, simple web configurator, and its wide range of extension and features." The perimeter firewall will be effective in protecting

against attacks in the preparation and access stages. The attacker will need to penetrate the firewall protections over the external network to access the internal corporate network.

1.8.3 Snort vs. Suricata

The two competing technologies which can be used within pfSense as an Intrusion Detection System (IDS) were Snort and Suricata[48]. Snort is the mature technology which has been around for many years, while Suricata is a fairly new multi-thread technology. Due to the limitations of our environment, a single threaded application is more appropriate. However, it does appear that Suricata would be able to handle a much larger workload [72]. Our decision focused on which technology was more proven in the industry, although several sources felt that Suricata outperformed Snort in terms of dropped packets [73] and better performed on Linux [74]. Since we were using a FreeBSD firewall with low throughput it made more sense to use Snort even with its known limitations.

Snort will be important in defending in the preparation, access, and harvest stages. In the preparation stage, Snort will assist in blocking websites and various types of port scans. The access stage will be important to use Snort as an NIDS system, to alert about possible attacks. Finally, in the harvest stage Snort can be used to identify personally identifiable information leaving the network [75].

1.8.4 Network Monitoring

Our search for a network monitoring system was originally focused on Bro [48] a packet capturing technology created in 1995 for passively monitoring a network line. This seemed to be the perfect, proven technology for monitoring our network [76]. This technology has been in development for over twenty years and is well-proven. Of course, it is also open source which met another one of our selection criteria. Some of the other reasons for choosing Bro include [77]: grounding in research, Bro community, and powerful design.

Another difficulty within our setup was how to implement Bro within our network. We at-

tempted to find a low cost way of deployment, and found an open source project known as Sweet Security [78]. This project provided all the infrastructure to run Bro on a Raspberry Pi and log what was detected to the ELK stack. This fit well within our overall project plan so we immediately decided to use this as part of our project. We will discuss in later sections some of the difficulties this tool decision created during setup.

Sweet Security (Bro) will be useful in the access, resident, and harvest stages on the internal network. Bro has a large number of prebuilt protocol filters which allow traffic to be captured and interpreted in real-time. This deep packet inspection will allow us to detect suspicious activities on the network. This usefulness will continue into the resident and harvest stages because we will be able to filter for network traffic that suggest lateral movement and data exfiltration from the network.

1.8.5 Host Monitoring

Open Source HIDS Security (OSSEC) has been around for a while and is used as a host-based intrusion detection system (HIDS) [79]. What this means is that it monitors the individual computers on the network through agents running on them, instead of monitoring the network traffic itself. This project was chosen because [80] it is open source, has been in development for a long period of time, has the capability of monitoring many different host operating systems, and is still under active development. Additionally, we found there is an parallel project which incorporates OSSEC known as Wazuh. Wazuh takes OSSEC and combines it with the ELK stack [81] which is exactly our goal for this project. Wazuh also has an extensive developer community which is constantly updating their tool to work with the latest versions of the ELK stack.

Wazuh will be most beneficial in the access and resident stages as it monitors the hosts to see whether and intrusion might have taken place. Wazuh has many capabilities, but with the default ruleset certain indicators of compromise can be found in altered registry values and common files. Additional rulesets may need to be expanded for more targeted threats.

Since our enterprise environment focuses so closely on Windows operating systems, we will in-

clude Sysmon[82] to add some additional Windows process, network connection, and file creation monitoring. Sysmon will dump the additional information in the Windows logs and then we will use Wazuh to export the log files from the host machines [83]. Sysmon requires a configuration file, so we use one available from GitHub [84].

1.8.6 Honeypot

We spent a large amount of time trying to find a honeypot which was under active development. Most projects are only active for a few years and then are abandoned as new techniques become available. Our main source [53] had an excellent table on pages 17-18 which summarized various historical honeypot projects, their protocols, and areas of focus. We needed to located a honeypot which would we usable within an internal network to similar some of the services which might be available therein, we also had a strong preference for log files to be made available in the ELK stack we had setup. Another reference we located was [54] which had a brief section discussing honeypots as they specifically apply to APST. Only serendipity lead us to the current Honeytrap project, but once we located it, it fulfilled our requirements. It is under active development, has several output channels which includes the ELK stack, is able to pretend to be various services, has a fairly straightforward setup and configuration, and is open source. All of these requirements lead us to choose Honeytrap for use within our project.

The honeypot will be useful in the resident stage in detecting the lateral movement of the APST as it attempts to explore and map the network. By pretending to be a high value target on the internal network attempts to access specific ports like SSH, FTP, etc. will be logged and will provide an indication that certain hosts within the network have been compromised.

1.8.7 Anti-virus Software

Open source anti-virus software is available through the ClamAV project [85]. ClamAV is the only open source anti-virus product available at this time [86]. While ClamAV is useful in detection of viruses and various forms of malware, modern Windows operating systems come installed

with Windows Defender. One paper on spyware concluded, "Windows Defender is competitive enough to compete with the existing Anti-Spyware products" [87]. We decided to leave Windows Defender as the default anti-virus on our Windows hosts as it was free to download and install on these operating systems. Furthermore, it regularly updates and has a better detection rate than ClamAV due to commercial backing by Microsoft. Windows Defender will form part of our APST protection suite, but will not be considered as one of the open source tools making up our ELK stack.

While an effective open source anti-virus was not forthcoming, we felt it was important to include an anti-virus within the overall toolkit. Anti-virus software is most important during the resident phase, as it can sometimes detect suspicious files or exploits that are being brought onto compromised hosts by the attackers.

1.8.8 Tool Summary

The tools selected for this project are summarized in Table 1.2. All tools contain open source software which we believe will be useful in the detection of APST and meets our commitment to build our solution using open source tools. Furthermore, all tools can be setup and configured to use the ELK stack.

Figure 1.2 summarizes how the various tools will fit into detection and removal of APST throughout the threat lifecycle. The four stage lifecycle model [2] was our chosen model for APST attacks.

1.9 Penetration Testing Lab Network Setup

In order to test our tool we setup a penetration testing lab as shown in Figure 1.3. This network has been setup to simulate a small-scale corporate network. The setup include a domain containing two domain controllers and one client operating system joined to the domain.

We have drawn from many sources to get a sense of the best way the lab should be built and

Table 1.2: Tool Summary

Tool	License	Website
Bro	BSD license	https://www.bro.org/
ELK Stack (Elastic Stack)	Apache License Version 2.0	https://www.elastic.co/
OSSEC	GNU GPL v2	https://www.ossec.net/
HoneyTrap	GNU Affero General Public License version 3	https://github.com/honeytrap/honeytrap
pfSense	Apache License 2.0	https://www.pfsense.org/
Snort	GNU GPL 2 and Commercial	https://snort.org/
Sysmon	Freeware	https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon
Sweet Security (Bro and ElasticStack)	Apache License 2.0	https://github.com/TravisFSmith/SweetSecurity
Wazuh (OSSEC and Elastic Stack)	GNU GPL 2	https://wazuh.com/
Windows Defender	Commercial	https://www.microsoft.com/en-ca/windows/comprehensive-security

Figure 1.2: APST Lifecycle Model With Selected Tools for Identifying and Removing Threats derived from [2, Fig. 1]

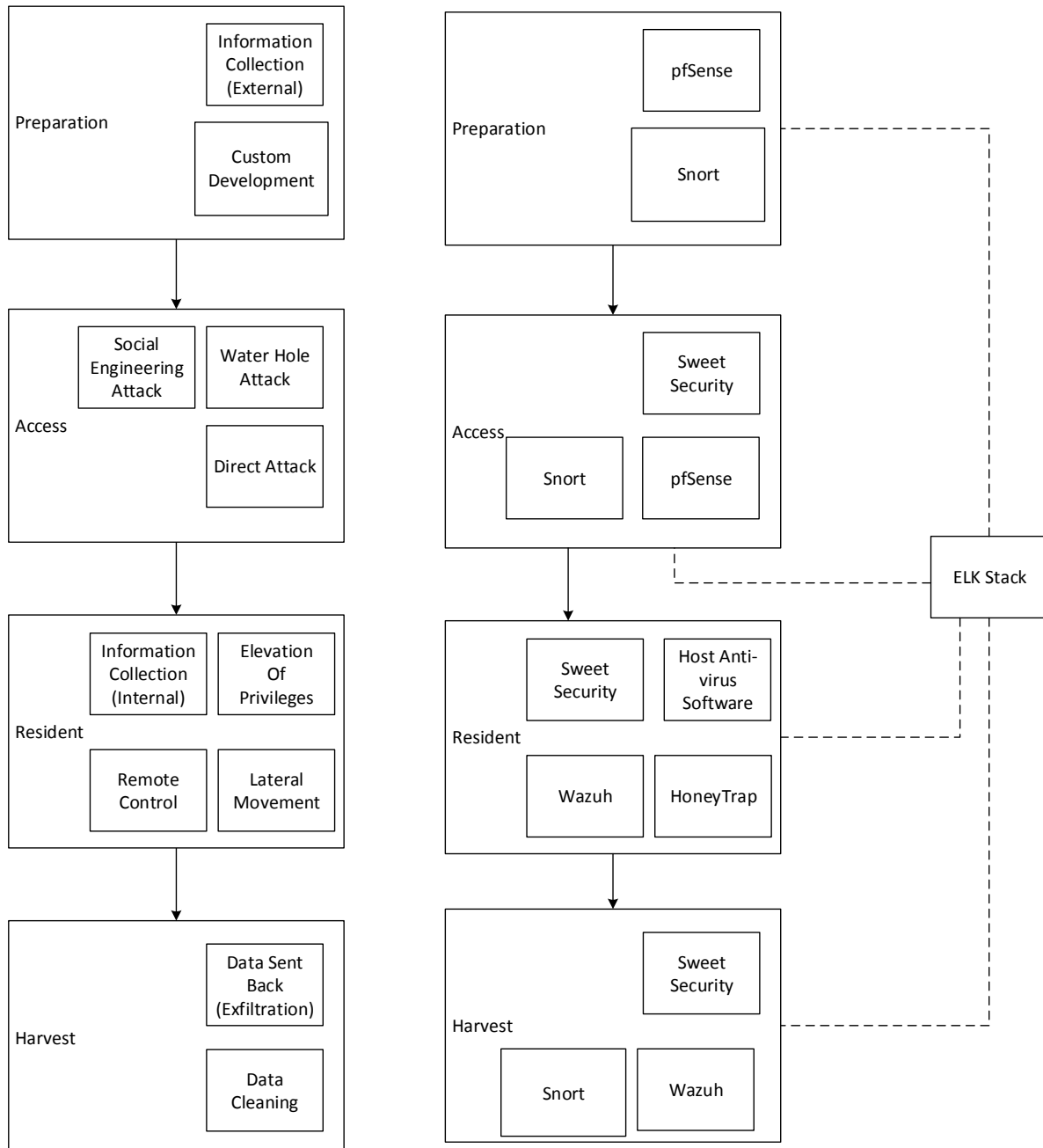
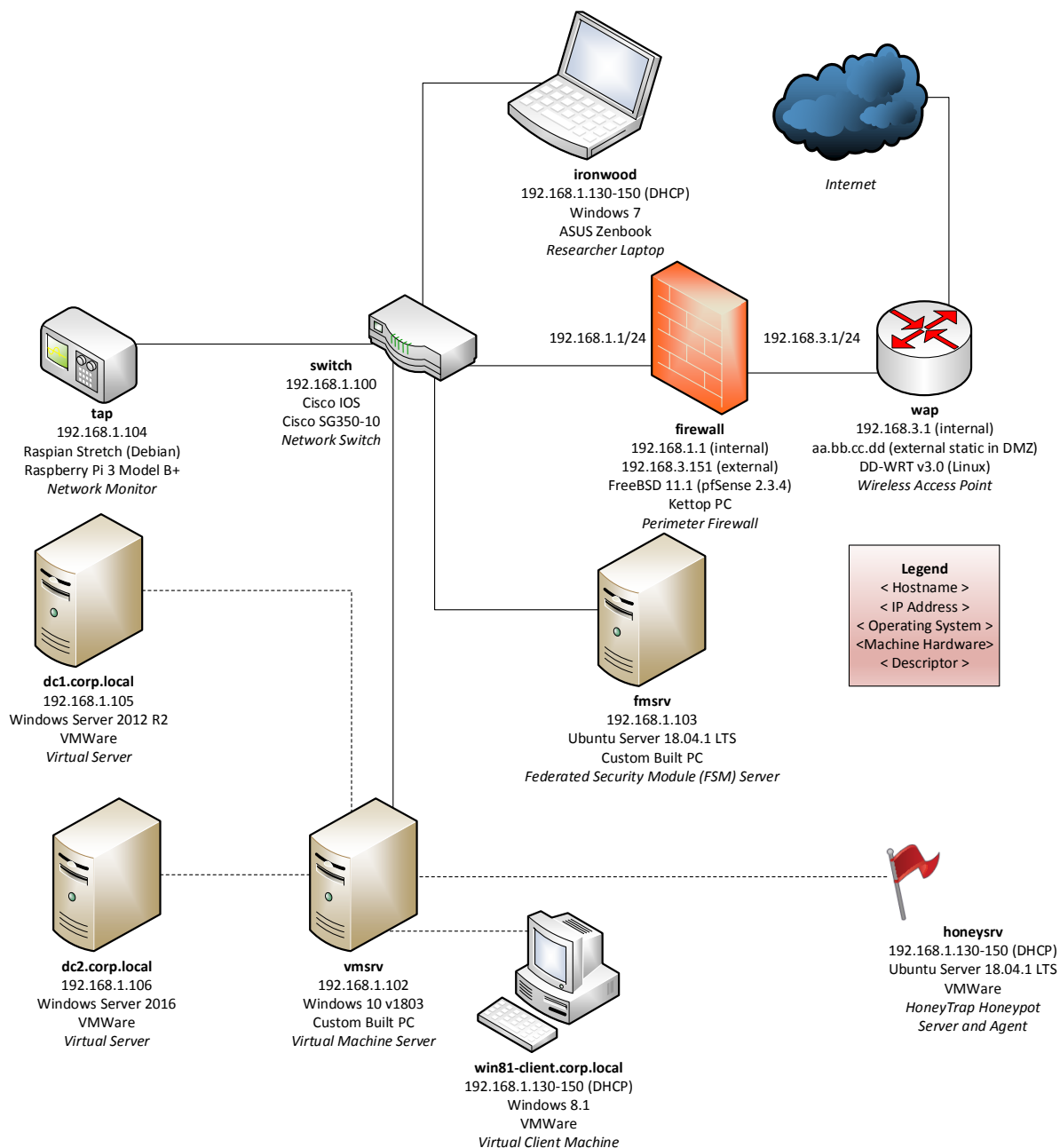


Figure 1.3: Penetration Testing Lab Configuration



configured, with a distinct emphasis on easy setup, configuration, and restoration of machines to a pristine state. One source of information would be chapter 4 and 5 in [88] which talks about setting up a penetration testing lab. Another source identified on setup of the lab environment for malware analysis is chapter 2 in [89]. Both these references give some guidance for the setup of a lab environment. Our specific environment uses virtual machines created in VMware Workstation.

Within our setup there are many different components:

- Desktop (Virtual Client Machine) - one virtual workstation running Windows 8.1 end user operating systems.
- FSM Server - the main analysis server where all data will be sent for processing as part of the FSM. This server is running Ubuntu Server 18.04.1 LTS on custom built hardware.
- Honeypot - a fake high value host running on the network. The honeypot is implementing using HoneyTrap running in a Ubuntu Server virtual machine.
- Laptop (Researcher Laptop) - one mobile workstations running Windows 7 end user operating systems.
- Network Monitor - a Raspberry Pi 3+ system which is setup to use port mirroring on the switch.
- Perimeter Firewall - a pfSense firewall appliance which will separate the internal network from the Internet.
- Router - The Wireless Access Point (WAP) functions as a router for our network and connects the network to the Internet.
- Switch - a Cisco switch which provides network connectivity for all the components in our internal network

- Servers - two Windows servers running in VMware Workstation on a single host computer.

In order to manage all these components remote access to all devices was setup either by web interface, SSH, RDP, or VNC, then a tool known as Remotix for Windows was used to remote into all systems for ease of access. A laptop wired on the internal network was used to connect with all services.

1.10 Security Software Configuration

The configuration and the setup of the software tools which were selected as part of this project involved a large amount of time and effort. We have detailed a high level overview of the steps required for the setup of the lab and respective software environment. You can refer to Figure 1.4 for the installation location of various pieces of security software. All reference to setup documentation in individual projects will be linked to, but for brevity only important steps are highlighted.

It is important to make sure that all systems within the research environment are synchronized to the correct time using a local network time protocol (NTP) server. This will make sure that all log files have correct date and time stamps.

1. Install pfSense on *firewall* [90].
2. Install SNORT on the *firewall* [91].
3. Install Sweet Security on *fmsrv*.
 - (a) Use the latest Ubuntu Server ISO to install the operating system [92].
 - (b) Follow the normal setup steps to install and run Linux.
 - (c) Make sure to configure the time server [93].
 - (d) Download Sweet Security and install it using the Web Server Only configuration [94].

4. Do the upgrade and install of latest version of ELK stack that Wazuh will support.
This install will use the single-host architecture [95].
 - (a) Install Wazuh server with DEB packages, but do not install FileBeats [96].
 - (b) Install Elastic Stack with Debian packages [97].
5. Upgrade Sweet Security information by re-indexing to the latest version of the ELK stack [98].
6. Install Wazuh latest version plugins.
7. Restore the default logstash template. (Sweet Security messes with this a bit, and when we was trying to get the logs from the pfSense sent to the ELK stack this was causing problems.)
8. Sweet Security Client install on the Raspberry Pi 3 B+ and get it to report back to the ELK server.
 - (a) Download NOOBS network install to the Pi SD card [99].
 - (b) Install latest Raspian OS on the Pi.
 - (c) Download Sweet Security from GitHub.
 - (d) Do additional install steps for Debian 9 [100].
 - (e) Patch the required files so the software can compile successfully, modify line 24, 26, 38, 40, 45, 47 in file Sweet Security/install/packages.py change libssl-dev to libssl1.0-dev [101].
 - (f) Install Sweet Security on the Raspberry Pi and give it the settings to send logs to the *fmsrv* ELK stack.

9. Install Wazuh agents on all Windows virtual machines [102]. You will need to register each agent [103].

(a) The Wazuh agent configuration was mainly left with the default. However, the capabilities of the agent are quite extensive and evolving [104]. Default settings are available for Windows agent clients. [105]. The main configuration file *ossec.conf* is modified on each host to add additional capabilities required for our research.

(b) Install Sysmon on all Windows hosts [82].

10. Setup pfSense to log to the ELK stack [106], [107].

11. Setup SNORT to log directly to the pfSense system log, then both the Snort and pfSense logs will come over at the same time.

12. Create a new Linux virtual machine and setup and Honeytrap Server and Agent [108].

13. Setup port mirroring on the Cisco switch to send information to *tap* [109].

Now you should have a working installation of the ELK stack with all logs going to it! Table 1.3 gives a summary of the main indexes in ElasticSearch and which tools they collect logs files from. When you see a * at the end of log file name that means that it is actually a series of log files denoted with a timestamp of -YYYY.MM.DD, for example *wazuh-alerts-3.x-2018.08.19*.

1.11 Security Software Setup Within the Network

1.12 Difficulties and Future Recommendations on Setup

Many concerns came to light during the setup, here are some of the high level difficulties encountered:

Figure 1.4: Security Software Setup Within the Network

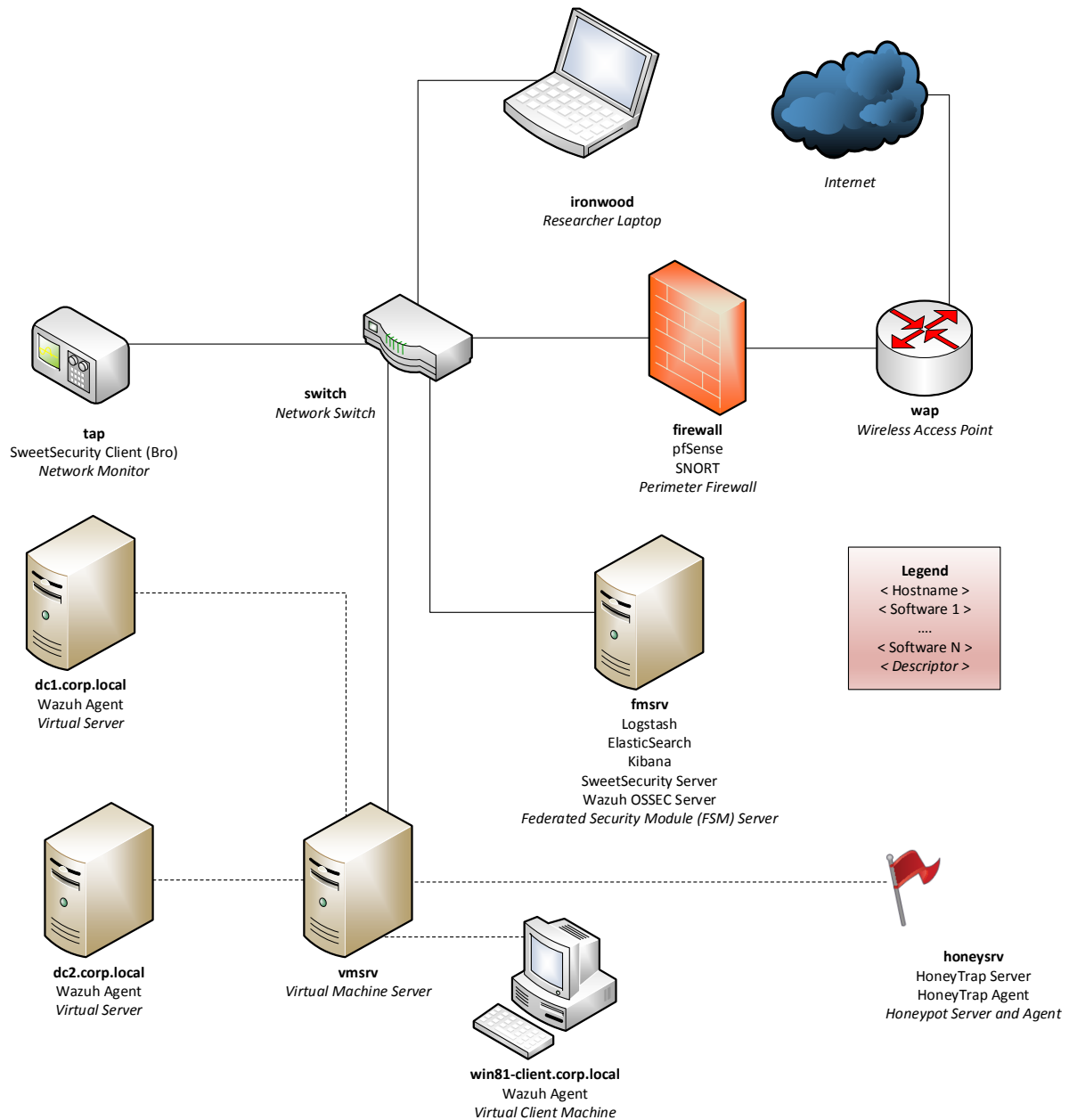


Table 1.3: Log Files Gathered by the ELK Stack

Elasticsearch Index	Source Tool	Type of Logs
honeytrap*	HoneyTrap	Status Messages, Honeypot Connection Attempts
logstash-*	Bro	Bro Network Traffic Monitoring
pfsense-*	pfSense Firewall, Snort	Firewall Status Messages, Snort Alerts
sweet_security	Sweet Security	Detected Device Information, Port Scans
sweet_security_alerts	Sweet Security	New(Unique) Sweet Security Log Events
tardis	Sweet Security	Historical Hosts, IP Addresses, and Websites
wazuh-alerts-3.x-*	Wazuh	Log Events Above the Alert Threshold
wazuh-monitoring-3.x-*	Wazuh	All Wazuh Monitoring Logs

- Sweet Security uses ELK 5.5, and there was a major change from ELK 5 to 6 under which many of the mappings are broken. This means that the indexes which are created in ELK 5.x will not work in ELK 6.x unless ELK stack is upgraded [110].
- Sweet Security does not appear to be under active development, and many problems reported on their official bug forum went unanswered by the developer during the months we worked on the project.
- ELK stack and Wazuh have a very quick development cycle and was releasing a new version every one or two months. This forced updating and upgrading these software applications on an ongoing basis which posed an issue with making sure project files were up-to-date.
- Honeytrap is under active development, so we had to request the developers do several software updates to their documentation and code before it was able to be used in this project.

- A good guide on getting pfSense to log to ELK stack does not exist, so we had to use several unofficial guides found on the Internet. We were able to get pfSense and Snort to log to the ELK stack, but a tutorial from the developers would have been more ideal.
- The Apache HTTPS wrapper Sweet Security uses around the ELK stack does cause some troubles with logging.
- Bro compilation time on the Raspberry Pi exceeds an hour, so rebuilding it from scratch can take a very long time.
- The Raspberry Pi sometimes gets overloaded and fails to respond, it must then be hard booted.
- The Raspberry Pi does not have sufficient RAM to have adequate performance when running Sweet Security full configuration.




1.13 Tests

1.13.1 Baseline

The baseline is an examination of normal traffic and reports generated by the various tools in which form part of the FSM. To generate the traffic, we allowed the network to run normally for a number of weeks. During this time our Researcher Laptop was connected to DHCP on the network, and we completed normal tasks we would do like checking e-mail, surfing the web, watching videos, and downloading programs.

Our firewall had a basic ruleset, which used default SNORT rules to filter out malicious traffic. Figure 1.5 demonstrates some hosts which Snort blocked going through the firewall. These blocked sites were identified using a free Oinkmaster account to download updated rulesets for Snort. Reasons for why certain hosts were blocked are identified by the Alert Descriptions. Snort will

Figure 1.5: Example Snort Alerts

Last 500 Hosts Blocked by Snort		
#	IP	Alert Descriptions and Event Times
1	93.115.28.164 	ET SCAN Sipvicious User-Agent Detected (friendly-scanner) – 2018-08-27 01:54:45 ET SCAN Sipvicious Scan – 2018-08-27 01:54:45
2	5.188.10.241 	ET CINS Active Threat Intelligence Poor Reputation IP TCP group 6 – 2018-08-27 12:36:38 ET DROP Spamhaus DROP Listed Traffic Inbound group 1 – 2018-08-27 20:09:05 ET CINS Active Threat Intelligence Poor Reputation IP TCP group 5 – 2018-08-27 20:09:05
3	82.221.105.6 	ET CINS Active Threat Intelligence Poor Reputation IP TCP group 79 – 2018-08-22 15:06:41 ET CINS Active Threat Intelligence Poor Reputation IP UDP group 79 – 2018-08-22 12:25:01 ET CINS Active Threat Intelligence Poor Reputation IP TCP group 81 – 2018-08-23 17:45:28 ET CINS Active Threat Intelligence Poor Reputation IP TCP group 88 – 2018-08-27 12:41:20 ET CINS Active Threat Intelligence Poor Reputation IP UDP group 88 – 2018-08-26 17:49:32 ET CINS Active Threat Intelligence Poor Reputation IP TCP group 89 – 2018-08-27 22:31:49

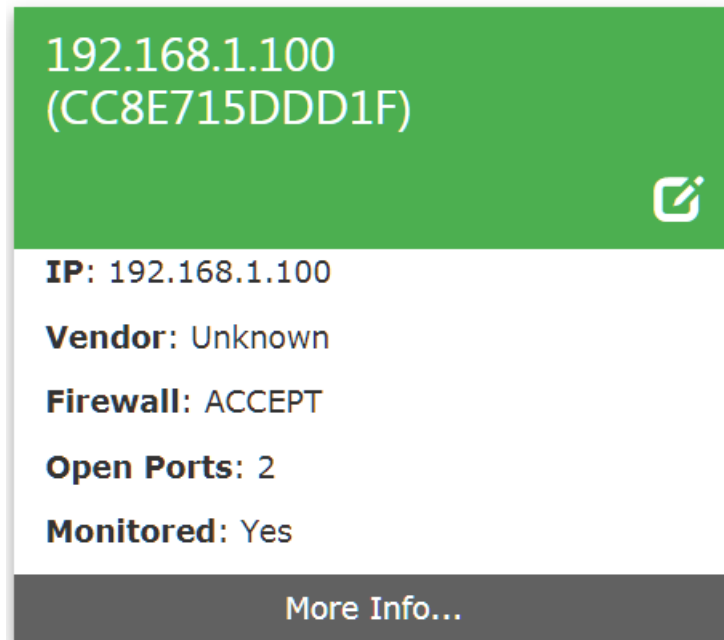
filter traffic traveling from the WAN to the LAN.

Sweet Security Server has a web interface which is hosted on *fmsrv* at <https://192.168.1.103>. This site has a simple password interface to limit access. Once you login, you are able to view devices which have been detected on the local network, an example is in Figure 1.6. When you click on a detected device, it will show you the expanded view similar to Figure 1.7. Finally, you can view new DNS / IP entries on the LAN as seen in Figure 1.8. These views show some of the information which is available in SweetSecurity including IP addresses, DNS queries, nmap scans of open ports, etc.

On the hosts, Wazuh and Windows Defender will provide endpoint protection and HIDS. Wazuh has been left with the default ruleset, so it will generate alerts in a standardized manner. Windows Defender has also been left with default settings. It is expected that Windows Defender will be running up-to-date signature definitions.

Within the internal network, HoneyTrap is active and presenting itself as a high value target. Besides the probes sent out by Sweet Security there should be no additional traffic attempting to connect to the HoneyTrap agent. Any other connection attempts from other hosts on the network mean they have been compromised, and should be investigated further.

Figure 1.6: Generic Device Information in Sweet Security



1.13.2 EICAR Test File Download

The EICAR test file [111] is a unique signature which is used to provide rudimentary baseline testing of anti-virus software. We are going to download an EICAR executable on to the *win81-client* to see whether some useful log files are generated. This is a basic test to make sure that our ELK stack is running properly and appears to be gathering information through Wazuh for the Windows Defender Alert message regarding this software.



Wazuh agent on the Windows host will need to be modified to correctly take the logs from Windows Defender. The local Wazuh configuration file `C:\Program Files (x86)\ossec-agent\ossec.conf` will be modified so collected logs will be sent to the ELK stack [112]. This will configure the Wazuh agent to monitor the specific Windows events for Windows Defender.

When the EICAR file was downloaded, an alert was displayed by Windows Defender that it is disinfecting the file and when we check the Windows logs it detects it as `Virus:DOS/EICAR_Test_File`. We verified that the Windows Event logs contain an alert for the EICAR test file [113].

Checking the Wazuh logs we found the results shown in Listing Appendix A.1 which clearly show that the Windows Defender scan result has been recorded by the ELK stack. This baseline

Figure 1.7: Expanded Device Information in Sweet Security

192.168.1.100 (CC8E715DDD1F)

Hostname	192.168.1.100 (CC8E715DDD1F)
Nickname	192.168.1.100 (CC8E715DDD1F) 
IP Address	192.168.1.100
MAC Address	CC8E715DDD1F
Vendor	Unknown
First Seen	2018-06-18 00:22:59
Last Seen	2018-08-27 23:48:49
Isolate Device	<input type="checkbox"/> No
Device Monitored	

Firewall Config (0)

Alerts (14)

Open Ports (2)

Baseline Info (14)

Delete Device

Figure 1.8: New DNS/IP Entries in Sweet Security

Baseliner	A new DNS query was added to the baseline: valve802.steamcontent.com	9CEBE8063EA1	2018-08-02 07:06:25
Baseliner	A new DNS query was added to the baseline: valve618.steamcontent.com	9CEBE8063EA1	2018-08-02 07:06:24
Baseliner	A new DNS query was added to the baseline: valve806.steamcontent.com	9CEBE8063EA1	2018-08-02 07:06:25

test shows how the anti-virus will alert us about possible malware on the Windows hosts.

1.13.3 APST Testing Suites

We accept that without any type of early warning system, the detection and removal of APST will be ineffective and virtually non-existent. We take this to be self-evident and do not proceed to prove it further within our experimental design.

One of the easiest ways to simulate APST on our network is to use open source projects which have been designed for this purpose. Two available test suites are APTSimulator [114] and FlightSim [115]. APTSimulator will simulate APST on our Windows hosts and FlightSim will simulate malicious traffic on our enterprise network. These tools will form the first level of testing we will do to evaluate our FSM solution. They will also give us information about what we should look for in the various log files to show malicious activity is happening within our network.

For each trial we run, we have gathered example log files which indicate that a compromise has occurred. These examples are gathered to provide documented proof of what our tools are reporting and allow analysis of the effectiveness of the alerts. After a qualitative examination of the alerts, we will provide an experimental framework to provide more rigorous testing of our results.

1.14 Assessment of Test Findings

There are many strengths to our current approach:

- Use of open source tools allows us to make available our full tool source code and capabilities, with the ability of other researchers to expand upon our work.
- Logstash allows easy log handling for a myriad of applications, which means we can easily expand the security tools which make up our FSM.
- Wazuh (OSSEC) allows monitoring of a large number of host operating systems

[116]. And the configuration file allows us to make many customizations depending on our requirements.

- Kibana allows the inclusion of user defined visualizations and dashboards to make sense of the threat intelligence that is being gathered.
- Our solution makes use of low-cost hardware like the Raspberry Pi 3 B+ and small PC as the firewall appliance.
- Most of the tools are under active development so they are improving and adding new features all the time.
- Our solution keeps all data within the internal enterprise network and does not make use of cloud services for the storing of any data. This means that privacy can be maintained in regards to proprietary data which the tools may collect from within the enterprise network, and there are no security or privacy concerns in regards to data collected by the tools.
- Our approach will scale to a large number of hosts and the tools we selected can be scaled to a large infrastructure.
- The ELK stack is very quick at making data available in real-time, so it can be searched and utilized when combating evolving threats.

There are some weaknesses in our current approach, which are limitations of the open source tools which were used:

- Encrypted traffic cannot be monitored effectively by Bro. The initial traffic showing that an encrypted connection is setup will be captured, but the encrypted traffic will not be plain text when captured [117]. This will severely limit our ability to intercept encrypted communications within our internal network. Decrypting

SSL/TLS traffic [118] is possible, but often requires corporate solutions which were not within the scope of our research.

- Critical Stack is not functional within Sweet Security, which will limit the ability to take note of certain types of traffic [119]. One example of traffic we may wish to flag within our network is Tor traffic as it indicates an attempt to send hidden traffic over our network to unknown destinations.
- Monitoring DNS requests for malicious sites is another important function which we did not select a tool for. While, Sweet Security will monitor DNS requests, it does not explicitly check for malicious DNS entries.
- A centralized logging solution within ELK stack would have been beneficial to have one unified dashboard for alerts. One example of this is to create a Kibana App which could contain these visualizations.
- The security of the FSM server is of course of paramount importance, and if our solution was deployed it would be prudent to spend additional time securing the server and associated logging devices.

1.14.1 Results and Future Work

Preliminary results are promising and some malware can be detected by our solution. The next investigation would be to perform a real-world test and monitor a production enterprise network to see whether threats are adequately flagged within the tool. Another opportunity for improvement is to create a dashboard which automatically highlights detected threats.

REFERENCES

- [1] P. Chen, L. Desmet, and C. Huygens, “A Study on Advanced Persistent Threats,” in *IFIP International Conference on Communications and Multimedia Security*. Springer, 2014, pp. 63–72. [Online]. Available: <https://lirias.kuleuven.be/bitstream/123456789/461050/1/2014-apt-study.pdf>
- [2] M. Li, W. Huang, Y. Wang, W. Fan, and J. Li, “The study of APT attack stage model,” in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, June 2016, pp. 1–5. [Online]. Available: <https://doi.org/10.1109/ICIS.2016.7550947>
- [3] Y. Wang and J. Yang, “Ethical Hacking and Network Defense: Choose Your Best Network Vulnerability Scanning Tool,” in *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, March 2017, pp. 110–113. [Online]. Available: <https://doi.org/10.1109/WAINA.2017.39>
- [4] C. Stoll, *The Cuckoo’s Egg: Tracking a Spy Through the Maze of Computer Espionage*. Gallery Books, 2005. [Online]. Available: <https://books.google.ca/books?id=9B1RfCAar2cC>
- [5] B. I. D. Messaoud, K. Guennoun, M. Wahbi, and M. Sadik, “Advanced Persistent Threat: New analysis driven by life cycle phases and their challenges,” in *2016 International Conference on Advanced Communication Systems and Information Security (ACOSIS)*, Oct 2016, pp. 1–6. [Online]. Available: <https://doi.org/10.1109/ACOSIS.2016.7843932>
- [6] C. Tankard, “Advanced Persistent threats and how to monitor and deter them,” *Network Security*, vol. 2011, no. 8, pp. 16 – 19, 2011. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1353485811700861>
- [7] M. Courtney, “States of cyber warfare,” *Engineering Technology*, vol. 12, no. 3, pp. 22–25, April 2017. [Online]. Available: <https://doi.org/10.1049/et.2017.0300>

- [8] N. Shalev, I. Keidar, Y. Weinsberg, Y. Moatti, and E. Ben-Yehuda, “WatchIT: Who Watches Your IT Guy?” in *Proceedings of the 26th Symposium on Operating Systems Principles*, ser. SOSP '17. New York, NY, USA: ACM, 2017, pp. 515–530. [Online]. Available: <https://doi.org/10.1145/3132747.3132752>
- [9] J. Estublier, H. Verjus, and P.-Y. Cunin, “Building Software Federation,” in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA '2001, Las Vegas, USA, 2001*. [Online]. Available: <http://www-adele.imag.fr/Les.Publications/intConferences/PDPTA2001Est.pdf>
- [10] T. R. Jackson, J. G. Levine, J. B. Grizzard, and H. L. Owen, “An investigation of a compromised host on a honeynet being used to increase the security of a large enterprise network,” in *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, June 2004, pp. 9–14. [Online]. Available: <https://doi.org/10.1109/IAW.2004.1437791>
- [11] R. Souza, C. Chavez, and R. A. Bittencourt, “Rapid Releases and Patch Backouts: A Software Analytics Approach,” *IEEE Software*, vol. 32, no. 2, pp. 89–96, Mar 2015. [Online]. Available: <https://doi.org/10.1109/MS.2015.30>
- [12] S. Burji, K. J. Liszka, and C. C. Chan, “Malware analysis using reverse engineering and data mining tools,” in *2010 International Conference on System Science and Engineering*, July 2010, pp. 619–624. [Online]. Available: <https://doi.org/10.1109/ICSSE.2010.5551719>
- [13] P. OKane, S. Sezer, and K. McLaughlin, “Obfuscation: The Hidden Malware,” *IEEE Security Privacy*, vol. 9, no. 5, pp. 41–47, Sept 2011. [Online]. Available: <https://doi.org/10.1109/MSP.2011.98>
- [14] M. Ussath, D. Jaeger, F. Cheng, and C. Meinel, “Advanced persistent threats: Behind the scenes,” in *2016 Annual Conference on Information Science and Systems (CISS)*, March 2016, pp. 181–186. [Online]. Available: <https://doi.org/10.1109/CISS.2016.7460498>

- [15] (2017) Vault 7: CIA Hacking Tools Revealed. WikiLeaks. Accessed 2017-12-06. [Online]. Available: <https://wikileaks.org/ciav7p1/>
- [16] Network Operations Division Cryptographic Requirements. Central Intelligence Agency. Accessed 2017-12-06. [Online]. Available: <https://wikileaks.org/ciav7p1/cms/files/NOD%20Cryptographic%20Requirements%20v1.1%20TOP%20SECRET.pdf>
- [17] Network Operations Division CNE Operational Data Exchange Format (Codex) Specification. Central Intelligence Agency. Accessed 2017-12-06. [Online]. Available: <https://wikileaks.org/ciav7p1/cms/files/Codex-Spec-v1-SECRET.pdf>
- [18] Network Operations Division Persisted DLL Specification. Central Intelligence Agency. Accessed 2017-12-06. [Online]. Available: <https://wikileaks.org/ciav7p1/cms/files/ICE-Spec-v3-final-SECRET.pdf>
- [19] Network Operations Division In-memory Code Execution Specification. Central Intelligence Agency. Accessed 2017-12-06. [Online]. Available: <https://wikileaks.org/ciav7p1/cms/files/ICE-Spec-v3-final-SECRET.pdf>
- [20] M. Hanspach and M. Goetz, “On covert acoustical mesh networks in air,” *arXiv preprint arXiv:1406.1213*, 2014. [Online]. Available: <https://arxiv.org/pdf/1406.1213.pdf>
- [21] L. Shing, J. Astacio, A. Figueroa, and C. C. Shing, “Vulnerabilities of radio frequencies,” in *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, Aug 2015, pp. 2682–2686. [Online]. Available: <https://doi.org/10.1109/FSKD.2015.7382381>
- [22] G. Schryen and R. Kadura, “Open Source vs. Closed Source Software: Towards Measuring Security,” in *Proceedings of the 2009 ACM Symposium on Applied Computing*, ser. SAC ’09. New York, NY, USA: ACM, 2009, pp. 2016–2023. [Online]. Available: <https://doi.org/10.1145/1529282.1529731>

- [23] R. Wojtczuk and A. Tereshkin, “Attacking intel bios,” *BlackHat, Las Vegas, USA*, 2009. [Online]. Available: <http://www.blackhat.com/presentations/bh-usa-09/WOJTCZUK/BHUSA09-Wojtczuk-AtkIntelBios-SLIDES.pdf>
- [24] R. Minnich. Replace Your Exploit-Ridden Firmware with Linux. YouTube. Accessed 2017-12-06. [Online]. Available: <https://youtu.be/iffTJ1vPCSo>
- [25] A. Tanenbaum. An Open Letter to Intel. Accessed 2017-12-06. [Online]. Available: <http://www.cs.vu.nl/~ast/intel/>
- [26] Y. Altshuler, N. Aharony, A. Pentland, Y. Elovici, and M. Cebrian, “Stealing Reality: When Criminals Become Data Scientists (or Vice Versa),” *IEEE Intelligent Systems*, vol. 26, no. 6, pp. 22–30, Nov 2011. [Online]. Available: <https://doi.org/10.1109/MIS.2011.78>
- [27] F. Li, A. Lai, and D. Ddl, “Evidence of Advanced Persistent Threat: A case study of malware for political espionage,” in *2011 6th International Conference on Malicious and Unwanted Software*, Oct 2011, pp. 102–109. [Online]. Available: <https://doi.org/10.1109/MALWARE.2011.6112333>
- [28] A. A. Adams, “Report of a Debate on Snowden’s Actions by ACM Members,” *SIGCAS Comput. Soc.*, vol. 44, no. 3, pp. 5–7, Oct. 2014. [Online]. Available: <https://doi.org/10.1145/2684097.2684099>
- [29] I. Koshiw. (2018, Aug) How an International Hacker Network Turned Stolen Press Releases into \$100 Million. The Verge. Accessed 2018-08-23. [Online]. Available: <https://www.theverge.com/2018/8/22/17716622/sec-business-wire-hack-stolen-press-release-fraud-ukraine>
- [30] N. Kshetri, “Cyberwarfare: Western and Chinese Allegations,” *IT Professional*, vol. 16, no. 1, pp. 16–19, Jan 2014. [Online]. Available: <https://doi.org/10.1109/MITP.2014.4>
- [31] S. Bazan, “A New Way to Win the War,” *IEEE Internet Computing*, vol. 21, no. 4, pp. 92–97, 2017. [Online]. Available: <https://doi.org/10.1109/MIC.2017.2911419>

- [32] T. A. Berson and D. E. Denning, “Cyberwarfare,” *IEEE Security Privacy*, vol. 9, no. 5, pp. 13–15, Sept 2011. [Online]. Available: <https://doi.org/10.1109/MSP.2011.132>
- [33] J. L. G. Dietz and J. A. P. Hoogervorst, “Enterprise Ontology in Enterprise Engineering,” in *Proceedings of the 2008 ACM Symposium on Applied Computing*, ser. SAC '08. New York, NY, USA: ACM, 2008, pp. 572–579. [Online]. Available: <https://doi.org/10.1145/1363686.1363824>
- [34] Y.-W. E. Sung, X. Sun, S. G. Rao, G. G. Xie, and D. A. Maltz, “Towards Systematic Design of Enterprise Networks,” *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 695–708, Jun. 2011. [Online]. Available: <https://doi.org/10.1109/TNET.2010.2089640>
- [35] J. Moubarak, M. Chamoun, and E. Filiol, “Comparative study of recent MEA malware phylogeny,” in *2017 2nd International Conference on Computer and Communication Systems (ICCCS)*, July 2017, pp. 16–20. [Online]. Available: <https://doi.org/10.1109/CCOMS.2017.8075178>
- [36] M. Lindorfer, A. Di Federico, F. Maggi, P. M. Comparetti, and S. Zanero, “Lines of Malicious Code: Insights into the Malicious Software Industry,” in *Proceedings of the 28th Annual Computer Security Applications Conference*, ser. ACSAC '12. New York, NY, USA: ACM, 2012, pp. 349–358. [Online]. Available: <https://doi.org/10.1145/2420950.2421001>
- [37] R. Khadka, B. V. Batlajery, A. M. Saeidi, S. Jansen, and J. Hage, “How Do Professionals Perceive Legacy Systems and Software Modernization?” in *Proceedings of the 36th International Conference on Software Engineering*, ser. ICSE 2014. New York, NY, USA: ACM, 2014, pp. 36–47. [Online]. Available: <https://doi.org/10.1109/10.1145/2568225.2568318>
- [38] P. A. Sandborn and V. J. Prabhakar, “The Forecasting and Impact of the Loss of Critical Human Skills Necessary for Supporting Legacy Systems,” *IEEE Transactions on*

- Engineering Management*, vol. 62, no. 3, pp. 361–371, Aug 2015. [Online]. Available: <https://doi.org/10.1109/TEM.2015.2438820>
- [39] R. Brewer, “Advanced persistent threats: minimising the damage,” *Network Security*, vol. 2014, no. 4, pp. 5 – 9, 2014. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1353485814700406>
- [40] J. Chen, C. Su, K.-H. Yeh, and M. Yung, “Special issue on advanced persistent threat,” *Future Generation Computer Systems*, vol. 79, no. Part 1, pp. 243 – 246, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167739X17324913>
- [41] B. I. D. Messaoud, K. Guennoun, M. Wahbi, and M. Sadik, “Advanced Persistent Threat: New analysis driven by life cycle phases and their challenges,” in *2016 International Conference on Advanced Communication Systems and Information Security (ACOSIS)*, Oct 2016, pp. 1–6. [Online]. Available: <https://doi.org/10.1109/ACOSIS.2016.7843932>
- [42] M. Bere, F. Bhunu-Shava, A. Gamundani, and I. Nhamu, “How Advanced Persistent Threats Exploit Humans,” *International Journal of Computer Science Issues (IJCSI)*, vol. 12, no. 6, p. 170, 2015. [Online]. Available: https://www.researchgate.net/profile/Attlee_Gamundani/publication/301689293_How_Advanced_Persistent_Threats_Exploit_Humans/links/57223ec208ae586b21d3e6c6.pdf
- [43] (2017) Advanced Persistent Threat Activity Targeting Energy and Other Critical Infrastructure Sectors. United States Computer Emergency Readiness Team. Accessed 2017-12-09. [Online]. Available: <https://www.us-cert.gov/ncas/alerts/TA17-293A>
- [44] A. Lemay, J. Calvet, F. Menet, and J. M. Fernandez, “Survey of publicly available reports on advanced persistent threat actors,” *Computers and Security*, vol. 72, no. Supplement C, pp. 26 – 59, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167404817301608>
- [45] M. Li, W. Huang, Y. Wang, and W. Fan, “The optimized attribute attack graph

- based on APT attack stage model,” in *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, Oct 2016, pp. 2781–2785. [Online]. Available: <https://doi.org/10.1109/CompComm.2016.7925204>
- [46] R. Luh, S. Schrittwieser, and S. Marschalek, “TAON: An Ontology-based Approach to Mitigating Targeted Attacks,” in *Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services*, ser. iiWAS ’16. New York, NY, USA: ACM, 2016, pp. 303–312. [Online]. Available: <https://doi.org/10.1145/3011141.3011157>
- [47] M. Marchetti, F. Pierazzi, M. Colajanni, and A. Guido, “Analysis of high volumes of network traffic for Advanced Persistent Threat detection,” *Computer Networks*, vol. 109, no. Part 2, pp. 127 – 141, 2016, traffic and Performance in the Big Data Era. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1389128616301633>
- [48] I. Ghafir, V. Prenosil, J. Svoboda, and M. Hammoudeh, “A Survey on Network Security Monitoring Systems,” in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, Aug 2016, pp. 77–82. [Online]. Available: <https://doi.org/10.1109/W-FiCloud.2016.30>
- [49] M. Yamada, M. Morinaga, Y. Unno, S. Torii, and M. Takenaka, “RAT-based malicious activities detection on enterprise internal networks,” in *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, Dec 2015, pp. 321–325. [Online]. Available: <https://doi.org/10.1109/ICITST.2015.7412113>
- [50] A. Greco, A. Caponi, and G. Bianchi, “Facing lateral movements using widespread behavioral probes,” in *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, Dec 2016, pp. 159–160. [Online]. Available: <https://doi.org/10.1109/ICITST.2016.7856688>
- [51] I. Ghafir and V. Prenosil, *Proposed Approach for Targeted Attacks Detection*.

- Cham: Springer International Publishing, 2016, pp. 73–80. [Online]. Available: https://doi.org/10.1007/978-3-319-24584-3_7
- [52] Z. Saud and M. H. Islam, “Towards Proactive Detection of Advanced Persistent Threat (APT) Attacks Using Honeypots,” in *Proceedings of the 8th International Conference on Security of Information and Networks*, ser. SIN ’15. New York, NY, USA: ACM, 2015, pp. 154–157. [Online]. Available: <https://doi.org/10.1145/2799979.2800042>
- [53] M. Nawrocki, M. Wählisch, T. C. Schmidt, C. Keil, and J. Schönfelder, “A Survey on Honeypot Software and Data Analysis,” *CoRR*, vol. abs/1608.06249, 2016. [Online]. Available: <http://arxiv.org/abs/1608.06249>
- [54] C. NG, L. Pan, and Y. Xiang, *Honeypot Frameworks and Their Applications: A New Framework*, ser. SpringerBriefs on Cyber Security Systems and Networks. Springer Singapore, 2018. [Online]. Available: <https://books.google.ca/books?id=YydaDwAAQBAJ>
- [55] M. Auty, “Anatomy of an advanced persistent threat,” *Network Security*, vol. 2015, no. 4, pp. 13 – 16, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1353485815300283>
- [56] G. G. Granadillo, J. Garcia-Alfaro, H. Debar, C. Ponchel, and L. R. Martin, “Considering technical and financial impact in the selection of security countermeasures against Advanced Persistent Threats (APTs),” in *2015 7th International Conference on New Technologies, Mobility and Security (NTMS)*, July 2015, pp. 1–6. [Online]. Available: <https://doi.org/10.1109/NTMS.2015.7266480>
- [57] R. Mehresh, “Schemes for surviving advanced persistent threats,” Ph.D. dissertation, PhD thesis, Faculty of the Graduate School of the University at Buffalo, State University of New York, 2013. [Online]. Available: <https://www.cse.buffalo.edu/caeiae/documents/pdf/ruchika-mehresh-2013-dissertation.pdf>
- [58] H. M. Deylami, R. C. Muniyandi, I. T. Ardekani, and A. Sarrafzadeh, “Taxonomy of

- malware detection techniques: A systematic literature review,” in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, Dec 2016, pp. 629–636. [Online]. Available: <https://doi.org/10.1109/PST.2016.7906998>
- [59] D. Durham, “Mitigating exploits, rootkits and advanced persistent threats,” in *2014 IEEE Hot Chips 26 Symposium (HCS)*, Aug 2014, pp. 1–39. [Online]. Available: <https://doi.org/10.1109/HOTCHIPS.2014.7478798>
- [60] A. K. Sood and R. J. Enbody, “Targeted Cyberattacks: A Superset of Advanced Persistent Threats,” *IEEE Security Privacy*, vol. 11, no. 1, pp. 54–61, Jan 2013. [Online]. Available: <https://doi.org/10.1109/MSP.2012.90>
- [61] N. Virvilis, D. Gritzalis, and T. Apostolopoulos, “Trusted Computing vs. Advanced Persistent Threats: Can a Defender Win This Game?” in *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*, Dec 2013, pp. 396–403. [Online]. Available: <https://doi.org/10.1109/UIC-ATC.2013.80>
- [62] J. Vukalović and D. Delija, “Advanced Persistent Threats - detection and defense,” in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2015, pp. 1324–1330. [Online]. Available: <https://doi.org/10.1109/MIPRO.2015.7160480>
- [63] S. Torii, M. Morinaga, T. Yoshioka, T. Terada, and Y. Unno, “Multi-layered defense against advanced persistent threats (apt),” *Fujitsu Sci. Tech. J.*, vol. 50, no. 1, pp. 52–59, 2014. [Online]. Available: <http://www.fujitsu.com/global/documents/about/resources/publications/fstj/archives/vol50-1/paper09.pdf>
- [64] B. Binde, R. McRee, and T. J. O’Connor, “Assessing outbound traffic to uncover advanced persistent threat,” *SANS Institute. Whitepaper*, 2011. [Online]. Available: <https://www.sans.edu/student-files/projects/JWP-Binde-McRee-OConnor.pdf>

- [65] (2014) Advanced Threat Detection and Response: Using Splunk Software to Defend Against Advanced Threats. Splunk Inc. Accessed 2018-08-20. [Online]. Available: https://www.splunk.com/web_assets/pdfs/secure/Splunk_for_APT_Tech_Brief.pdf
- [66] C.-j. LU, Z. Heng, J.-y. LIU, R. ZHANG, Y.-k. CHEN, and Y.-g. YAO, “Network Security Log Analysis System Based on ELK,” *DEStech Transactions on Computer Science and Engineering*, no. cece, 2017. [Online]. Available: <http://www.dpi-proceedings.com/index.php/dtcse/article/download/14597/14114>
- [67] I. Y. M. Al-Mahbashi, M. B. Potdar, and P. Chauhan, “Network security enhancement through effective log analysis using ELK,” in *2017 International Conference on Computing Methodologies and Communication (ICCMC)*, July 2017, pp. 566–570.
- [68] C. Gormley and Z. Tong, *Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine*. O’Reilly Media, 2015. [Online]. Available: <https://books.google.ca/books?id=d19aBgAAQBAJ>
- [69] J. Turnbull, *The Logstash Book*. James Turnbull, 2014. [Online]. Available: <https://books.google.ca/books?id=5dIEBwAAQBAJ>
- [70] M. Arunwan, T. Laong, and K. Atthayuwat, “Defensive performance comparison of firewall systems,” in *2016 Management and Innovation Technology International Conference (MITicon)*, Oct 2016, pp. MIT-221–MIT-224.
- [71] A. Thakur. (2015) Open source firewall implementation: replacing traditional firewall with open source. [Online]. Available: <https://www.theseus.fi/handle/10024/96447>
- [72] W. Park and S. Ahn, “Performance Comparison and Detection Analysis in Snort and Suricata Environment,” *Wireless Personal Communications*, vol. 94, no. 2, pp. 241–252, May 2017. [Online]. Available: <https://doi.org/10.1007/s11277-016-3209-9>
- [73] B. Brumen and J. Legvart, “Performance analysis of two open source intrusion detection

- systems,” in *2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2016, pp. 1387–1392.
- [74] A. Alhomoud, R. Munir, J. P. Disso, I. Awan, and A. Al-Dhelaan, “Performance Evaluation Study of Intrusion Detection Systems,” *Procedia Computer Science*, vol. 5, pp. 173 – 180, 2011, the 2nd International Conference on Ambient Systems, Networks and Technologies (ANT-2011) / The 8th International Conference on Mobile Web Information Systems (MobiWIS 2011). [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1877050911003498>
- [75] (2018) Snort FAQ - README.sensitive_data. Accessed 2018-09-03. [Online]. Available: https://www.snort.org/faq/readme-sensitive_data
- [76] V. Paxson, “Bro: a system for detecting network intruders in real-time,” *Computer Networks*, vol. 31, no. 23, pp. 2435 – 2463, 1999. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1389128699001127>
- [77] (2014, 11) Why Choose Bro? Bro Network Security Monitor. Accessed 2018-08-22. [Online]. Available: https://www.bro.org/why_choose_bro.pdf
- [78] T. Smith. (2016, 01) Sweet Security Part 2 – Creating a Defensible Raspberry Pi. The State of Security Blog. Accessed 2018-08-22. [Online]. Available: <https://www.tripwire.com/state-of-security/security-data-protection/sweet-security-part-2-creating-a-defensible-raspberry-pi/>
- [79] R. Bray, D. Cid, and A. Hay, *OSSEC Host-Based Intrusion Detection Guide*. Elsevier Science, 2008. [Online]. Available: <https://books.google.ca/books?id=h37q2q3wvcUC>
- [80] J. Timofte, “Intrusion Detection using Open Source Tools,” *Informatica Economica*, vol. 0, no. 2, pp. 75–79, 2008. [Online]. Available: <https://ideas.repec.org/a/aes/infoec/vxiiy2008i2p75-79.html>

- [81] (2018) User Manual. Wazuh Inc. Accessed 2018-08-23. [Online]. Available: <https://documentation.wazuh.com/current/user-manual/index.html>
- [82] T. Mark Russinovich. (2018, July) Sysmon v8.0. Accessed 2018-09-29. [Online]. Available: <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>
- [83] Alberto Rodriguez. (2017, May) Using Wazuh to monitor Sysmon events. Accessed 2018-09-29. [Online]. Available: <https://blog.wazuh.com/using-wazuh-to-monitor-sysmon-events/>
- [84] @SwiftOnSecurity. (2018, Jan) sysmon-config — A Sysmon configuration file for everybody to fork. Accessed 2018-09-29. [Online]. Available: <https://github.com/SwiftOnSecurity/sysmon-config#sysmon-config--a-sysmon-configuration-file-for-everybody-to-fork>
- [85] L. E. S. Jaramillo, “Detecting malware capabilities with FOSS: Lessons learned through a real-life incident,” in *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*, June 2018, pp. 1–6.
- [86] M. I. Al-Saleh, F. M. AbuHjeela, and Z. A. Al-Sharif, “Investigating the detection capabilities of antiviruses under concurrent attacks,” *International Journal of Information Security*, vol. 14, no. 4, pp. 387–396, Aug 2015. [Online]. Available: <https://doi.org/10.1007/s10207-014-0261-x>
- [87] R. Shams, M. Farhan, S. A. Khan, and F. Hashmi, “Comparing Anti-Spyware products — A different approach,” in *2011 6th IEEE Joint International Information Technology and Artificial Intelligence Conference*, vol. 1, Aug 2011, pp. 75–80.
- [88] T. Wilhelm, “Professional Penetration Testing,” in *Professional Penetration Testing*. Boston: Syngress, 2010. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/B9781597494250000014>

- [89] M. Sikorski and A. Honig, *Practical Malware Analysis: A Hands-On Guide to Dissecting Malicious Software*. No Starch Press, 2012. [Online]. Available: <https://books.google.ca/books?id=FQC8EPYy834C>
- [90] (2018) Installing pfSense. Rubicon Communications LLC. Accessed 2018-08-23. [Online]. Available: <https://www.netgate.com/docs/pfsense/install/installing-pfsense.html>
- [91] (2018) Configuring the Snort Package. Rubicon Communications LLC. Accessed 2018-08-23. [Online]. Available: <https://www.netgate.com/docs/pfsense/install/installing-pfsense.html>
- [92] (2018) Download Ubuntu Server. Canonical Ltd. Accessed 2018-08-23. [Online]. Available: <https://www.ubuntu.com/download/server>
- [93] (2018) Time Synchronization. Canonical Ltd. Accessed 2018-08-23. [Online]. Available: <https://help.ubuntu.com/lts/serverguide/NTP.htm>
- [94] T. Smith. (2018) Getting Sweet Security. Accessed 2018-08-23. [Online]. Available: <https://github.com/TravisFSmith/SweetSecurity/wiki>
- [95] (2018) Installation Guide. Wazuh, Inc. Accessed 2018-08-23. [Online]. Available: <https://documentation.wazuh.com/current/installation-guide/index.html>
- [96] (2018) Installing Wazuh Server. Wazuh, Inc. Accessed 2018-08-23. [Online]. Available: https://documentation.wazuh.com/current/installation-guide/installing-wazuh-server/wazuh_server_deb.html#wazuh-server-deb
- [97] (2018) Install Elastic Stack with Debian packages. Wazuh, Inc. Accessed 2018-08-23. [Online]. Available: https://documentation.wazuh.com/current/installation-guide/installing-elastic-stack/elastic_server_deb.html#elastic-server-deb
- [98] (2018) Upgrading the Elastic Stack. Elastic. Accessed 2018-08-23. [Online]. Available: <https://www.elastic.co/guide/en/elastic-stack/6.4/upgrading-elastic-stack.html>

- [99] (2018) Cisco SG350-10 10-Port Gigabit Managed Switch. Raspberry Pi Foundation. Accessed 2018-08-26. [Online]. Available: <https://www.raspberrypi.org/downloads/noobs/>
- [100] SeppBender. (2017, Aug) Can I Install on Debian 9? Accessed 2018-08-26. [Online]. Available: <https://github.com/TravisFSmith/SweetSecurity/wiki/FAQ>
- [101] cloudstrifeedge. (2018, Aug) For those who want to install this project to one single Raspberry Pi. Accessed 2018-08-26. [Online]. Available: <https://github.com/TravisFSmith/SweetSecurity/issues/48>
- [102] (2018) Install Wazuh agent on Windows. Wazuh, Inc. Accessed 2018-08-23. [Online]. Available: https://documentation.wazuh.com/3.x/installation-guide/installing-wazuh-agent/wazuh_agent_windows.html
- [103] (2018) Register Agent. Wazuh, Inc. Accessed 2018-08-23. [Online]. Available: <https://documentation.wazuh.com/3.x/user-manual/agents/command-line/register.html>
- [104] (2018) Capabilities. Wazuh, Inc. Accessed 2018-08-23. [Online]. Available: <https://documentation.wazuh.com/3.x/user-manual/capabilities/index.html>
- [105] R. Cenit. (2018, Aug) Monitoring Windows System. Accessed 2018-08-26. [Online]. Available: https://groups.google.com/forum/?utm_medium=email&utm_source=footer#!msg/wazuh/27-9odTV4xA/0nv_A5ApAAAJ
- [106] K. Elatov. (2016, Nov) pfsense logging with elk. Accessed 2018-08-23. [Online]. Available: <http://elatov.github.io/2016/11/pfsense-logging-with-elk/>
- [107] M. Walter. (2018, Apr) ELK + PFSENSE 2/2. Accessed 2018-08-23. [Online]. Available: <http://www.hs-x.ch/index.php/elasticsearch-informationen/20-elk-pfsense-2-2>
- [108] R. Verhoef. (2018) HoneyTrap Documentation. Accessed 2018-08-23. [Online]. Available: <http://docs.honeytrap.io/docs/home/>
- [109] (2018) Cisco SG350-10 10-Port Gigabit Managed Switch. Cisco. Accessed

- 2018-08-23. [Online]. Available: <https://www.cisco.com/c/en/us/support/switches/sg350-10-10-port-gigabit-managed-switch/model.html>
- [110] (2018) Removal of Mapping Types. Elastic. Accessed 2018-08-23. [Online]. Available: <https://www.elastic.co/guide/en/elasticsearch/reference/current/removal-of-types.html>
- [111] (2018) EICAR Test File Intended Use. Accessed 2018-09-26. [Online]. Available: <http://www.eicar.org/86-0-Intended-use.html>
- [112] M. Barbero. (2017, Aug) Get Windows Defender logs on Wazuh Manager. Accessed 2018-09-26. [Online]. Available: https://groups.google.com/forum/#!topic/wazuh/2_b1KRVKT6o
- [113] J. Andrea Bichsel. (2018, Sept) Review event logs and error codes to troubleshoot issues with Windows Defender Antivirus. Accessed 2018-09-26. [Online]. Available: <https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-antivirus/troubleshoot-windows-defender-antivirus>
- [114] F. Roth. (2018) APT Simulator. [Online]. Available: <https://github.com/NextronSystems/APTSimulator>
- [115] AlphaSOC. (2018) AlphaSOC Network Flight Simulator. Accessed 2018-09-16. [Online]. Available: <https://github.com/alphasoc/flightsim>
- [116] W. Inc. (2018) Compatibility matrix. Accessed 2018-09-29. [Online]. Available: https://documentation.wazuh.com/current/installation-guide/compatibility_matrix/index.html
- [117] (2014) Using the Bro SSL analyzer. Accessed 2018-09-23. [Online]. Available: https://www.bro.org/brocon2014/ssl_exercise.pdf
- [118] T. Radivilova, L. Kirichenko, D. Ageyev, M. Tawalbeh, and V. Bulakh, “Decrypting SSL/TLS Traffic for Hidden Threats Detection,” in *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, May 2018, pp. 143–146.

[119] (2018) Critical Stack Client not Fetching Lists. Accessed 2018-09-23. [Online]. Available:
<https://groups.google.com/forum/#!topic/security-onion/axOCfBgjva4>

APPENDIX A

Raw JSON Log Files

This appendix contains all the raw JSON log files from the ELK stack which are cited as reference examples for our test cases. We have included each one as a separate listing framed with line numbers on the left margin for easy reference.

Listing Appendix A.1: Wazuh-alerts-3.x Log File Results for EICAR Test File

```
1 {
2   "_index": "wazuh-alerts-3.x-2018.09.27",
3   "_type": "wazuh",
4   "_id": "KAq_GGYBI-9SBrwSyob1",
5   "_score": 1,
6   "_source": {
7     "path": "/var/ossec/logs/alerts/alerts.json",
8     "manager": {
9       "name": "fmsrv"
10    },
11    "predecoder": {
12      "program_name": "WinEvtLog",
13      "timestamp": "2018 Sep 26 19:58:34"
14    },
15    "rule": {
16      "description": "Windows Defender: error when taking action on
17        potentially unwanted software",
18      "id": "83002",
19      "mail": false,
20      "groups": [
21        "windows",
22        "windows_defender"
23      ],
24      "gdpr": [
25        "IV_35.7.d"
26      ],
27      "level": 7,
28      "firedtimes": 1
29    },
30    "decoder": {
31      "parent": "windows",
```

```

31     "name": "windows"
32 },
33 "id": "1538013513.436444",
34 "full_log": "2018 Sep 26 19:58:34 WinEvtLog: Microsoft-Windows-Windows
    Defender/Operational: INFORMATION(1117): Microsoft-Windows-Windows
    Defender: SYSTEM: NT AUTHORITY: win81-client.corp.local: Windows
    Defender has taken action to protect this machine from malware or
    other potentially unwanted software. For more information please see
    the following: http://go.microsoft.com/fwlink/?linkid=37020&name=
    Virus:DOS/EICAR_Test_File&threatid=2147519003&enterprise=0 \tName:
    Virus:DOS/EICAR_Test_File \tID: 2147519003 \tSeverity: Severe \
    tCategory: Virus \tPath: file:_C:\\Users\\Administrator\\Downloads\\
    Unconfirmed 965242.crdownload \tDetection Origin: Local machine \
    tDetection Type: Concrete \tDetection Source: Real-Time Protection \
    tUser: NT AUTHORITY\\SYSTEM \tProcess Name: C:\\Program Files (x86)\\
    Google\\Chrome\\Application\\chrome.exe \tAction: Quarantine \tAction
    Status: No additional actions required \tError Code: 0x80508023 \
    tError description: The program could not find the malware and other
    potentially unwanted software on this computer. \tSignature Version:
    AV: 1.275.1349.0, AS: 1.275.1349.0, NIS: 119.0.0.0 \tEngine Version:
    AM: 1.1.15200.1, NIS: 2.1.14600.4",
35 "location": "WinEvtLog",
36 "@timestamp": "2018-09-27T01:58:33.776Z",
37 "data": {
38     "id": "1117",
39     "srcuser": "NT AUTHORITY\\SYSTEM"
40 },
41 "agent": {
42     "id": "001",
43     "name": "win81-client"
44 }
45 },
46 "fields": {
47     "@timestamp": [
48         "2018-09-27T01:58:33.776Z"
49     ]
50 }
51 }

```