Global Information Assurance Certification Paper

Offensive Intrusion Analysis: Uncovering Insiders with Threat Hunting and Active Defense

GIAC (GCIH) Gold Certification

Author: Matthew Hosburgh, matt.hosburgh@gmail.com
Advisor: Christopher Walker
Accepted: July 6, 2017

Abstract

Today's adversaries are advanced and more capable than ever before. Passive defensive tactics are no longer viable for pursuing these attackers. To compound the issue, the existence of an insider threat creates a challenging problem for the passive defender. One of the largest breaches of classified information was carried out by an insider. Months after the incident had occurred, the Department of Defense (DoD) only began to realize the implications of the leak. The damage did not solely rest with the United States. A cascade of consequences was felt in many parts of the world, resulting from this breach. Techniques like Threat Hunting, attempt to diminish this problem by combating advanced threats with people, also known as Threat Hunters. Although Threat Hunting is proving to be invaluable for many organizations there remains a chasm between detection and disclosure. Offensive Countermeasure tools such as the Web Bug Server and Molehunt can be leveraged as a means to proactively hunt insider threats. To keep up with the continually evolving human adversary, defenders must employ these offensive tactics to annoy and attribute their adversaries.

1. Introduction

The words on a slide describing WikiLeaks 1.0 in late 2009 foreshadowed a grave and inevitable future. "…The leading disclosure portal for classified, restricted or legally threatened publications. We provide an anonymous safe harbour for the submission and uncensorable provisioning of documents" ("WikiLeaks Release," 2009). At the time, this statement was alarming to many individuals and organizations within the Department of Defense. In 2010, over 391,000 classified U.S. documents were leaked by WikiLeaks which was the largest unauthorized disclosure of classified information to date (Romero, 2010). After the initial shock and awe of the leak had subsided, specific documents and information surfaced out of the massive trove of documents. In Tunisia, the U.S. information pointed to greed and corruption of the Tunisian government, which helped fuel the Arab Spring (Bachrach, 2011). The effects did not stop there. The protests in Tunisia had a cascading effect felt around the world. In New York, protestors were galvanized by the actions in Northern Africa and eventually Occupy Wall Street was sparked (Saba, 2011). But why and who was to blame? Months after the information was posted to WikiLeaks, an Army private was indicted as a suspect and sole actor. At the time, his privileged access to the material enabled his actions to expose the wrong doing, and much more, by the U.S. military during the Iraq War. This disparity from the point of breach to the moment of detection is still problematic. Techniques like Threat Hunting, attempt to diminish this problem by combating advanced threats with people, also known as Threat Hunters. Although these techniques are proving invaluable to many organizations, there remains a delta between detection and compromise. Attribution is an Active Defense technique that, when combined with Threat Hunting, is a method to drastically reduce the detection delta and to minimize the effects of a targeted attack. Tools such as the Web Bug Server and Molehunt can be leveraged as force multipliers when hunting insider threats.

2. The Detection Delta

Detecting threats and adversaries on networks continues to be a problem for many organizations. In the 2017 M-Trends report by FireEye, "the global median time from compromise to discovery has dropped significantly from 146 days in 2015 to 99 days in

2016" ("M-Trends," 2017). This disparity is known as the detection delta. Although positive, the number still indicates that it takes over three months before an organization realizes they have been breached. Significant damage and data exfiltration can happen in 99 days. Put another way, 99 days is equal to 8.554e+6 seconds. At dial-up speeds of 56Kbps, that means an attacker could transfer approximately 59.87GB of data, assuming a constant bandwidth and connection. If an average customer record is 2KB in size, the total records lost would equate to 29,935,000—even at low and slow speeds. Adding bandwidth or multiple avenues for the attacker to exfiltrate the data only exacerbates the loss to the organization. These numbers are daunting and almost impossible to comprehend. Traditional alerting further adds to the exhausting task of reactive detection techniques.

2.1. Alert Fatigue

Alert fatigue is an enemy to detecting or hunting real, human adversaries on an organization's systems. In a recent study on Computer Security Incident Response Teams (CSIRT), researchers discovered that many operators or analysts are not well prepared in terms of tooling: "All are uniformly unhappy with current solutions for forensics and incident response. Commercial solutions like Security Information and Event Management (SIEM) systems do not address operational needs, probably because vendors and researchers do not understand how analysts think and work" (Sundaramurthy, McHugh, Rajagopalan, & Wesch, 2014). This discontentment erodes at the trust of the alerts that an analyst receives. The alerts produced by varying tools are not useless; however, they can be overwhelming and time consuming. The study went on to discover that repeatable tasks were not being automated. The perpetual cycle erodes at the analyst's mental well-being: "Receive an alert, scan the logs (three minutes), look up an address (one minute), find the user information (another minute), repeat" (Sundaramurthy et al., 2014). The argument can be made that all work and no mental stimulation can make the analyst a dull boy, or girl. This might not be such a problem if all of an organization's adversaries were robots. The reality is that there are human adversaries with human behaviors and human flaws attacking organizations.

2.2. The Human Adversary

At the other end of any bot, virus, or targeted attack there is a human. Someone to code an action, someone to conduct reconnaissance on a target, and often, someone to exfiltrate an organization's protected information. According to research from Carnegie Mellon University, "the human domain is complex, and as a result the reasons behind certain behaviors are inherently complex" (Costa et al., 2016). This problem that many detection systems try to solve is the automated detection of these complex actions. Some of the actions are obvious, like an NMAP port scan. Others are less overt, such as valid credentials used for nefarious purposes. To compound the issue, not all humans or analysts use the same techniques or methods to achieve their goals. For example, a nation state actor could have a set of known techniques tactics and procedures (TTPs) that could potentially be detected. What if those TTPs change mid-mission? Or even more frustrating, what if an insider was operating in the parameters of a company policy to exfiltrate data? The detection delta grows and might even be non-existent in the case of an insider leaking information until the damage is done.

2.3. Common Threads via an Intrusion Model

Leveraging known data on attack techniques is an excellent starting point for advanced adversary detection. One such example is the MITRE Adversarial Tactics, Techniques & Common Knowledge (ATT&CK) matrix. MITRE's claim is that "ATT&CK is a constantly growing common reference for post-compromise techniques that brings greater awareness of what actions may be seen during a network intrusion. It enables a comprehensive evaluation of computer network defense (CND) technologies, processes, and policies against a common enterprise threat model" ("Adversarial Tactics," n.d.). Common attack patterns provide a start; however, they are still too broad to begin a Threat Hunt. Fortune might favor a pattern search to uncover an attacker, but the advanced adversary's actions will more than likely remain undetected. Historical references are another key area to investigate what is known about insider attacks.

Research by the Carnegie Mellon University provides an additional resource for developing patterns for hunting Insiders. Costa et al., (2016) analyzed data from the

MERIT insider threat database, which contains instances of insider incidents. The research illustrated the insider's actions mapped out in an ontology model. Similar to developing patters, this method hones in on the actual human behaviors. Each of the scenarios could be used to develop additional patterns to match on. Figure 1 is an example provided by Costa et al., (2016) which models the unauthorized exporting of confidential data by an insider with a laptop.

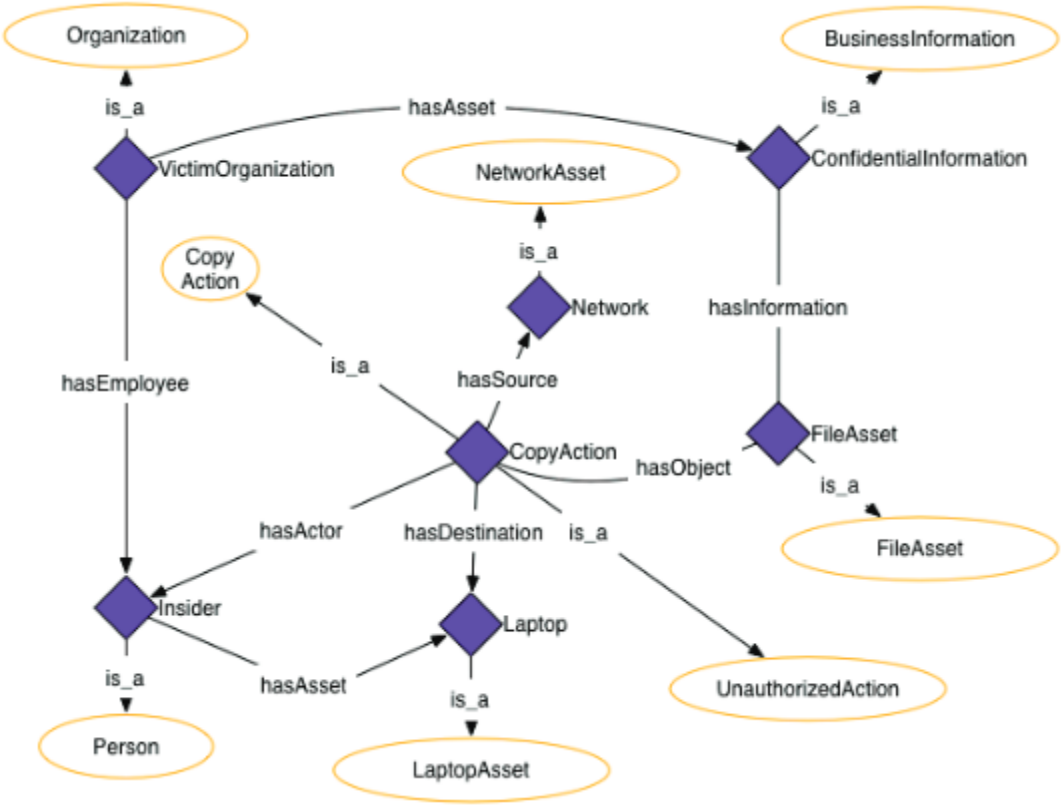


Figure 1. Ontology Model for Unauthorized Data Export

Because each organization is unique, a look at who the adversaries are and what their goals are is necessary in prioritizing the work of a Threat Hunter.

2.4. Prioritization of Adversaries

Two of the most fundamental questions an organization can ask are: what are we protecting and who are our adversaries? These two questions help to shape the larger security strategy, but can especially hone the focus of a Hunt Team. Because not all organizations are created equally, the answers will vary from industry to industry and

even organization to organization within a common commerce. One of the most rapid and effective means to capture who the adversaries are, is via threat modeling. In the most rudimentary example, a simple survey polling the current staff can illuminate a solid list of potential, or known adversaries. The tribal or tacit knowledge is powerful because it is the collective body of knowledge that has been learned over many years. Often it is the assumed knowledge, or information that did not make it into a formal document. One such example of this analytical model is the Crown Jewels Analysis (CJA) Process. CJA "can lead the hunter to think about the most useful types of data to collect in the environment (and the locations from which it should be collected) to be able to begin hunting for types of adversary activity that might be especially important to detect" (Lee & Bianco, 2016). Knowing what requires protection ensures the focus is on the most meaningful areas of the organization. Looking at who the adversaries are can also be extracted from the tacit knowledge and reporting from the larger community. Based on these findings, the hunt priorities or intrusion analysis focus can be set forth. Geopolitical and other market factors help to further paint the adversary picture by helping to understand the actor's motivation. Figure 2 illustrates a hypothetical model based on survey results:

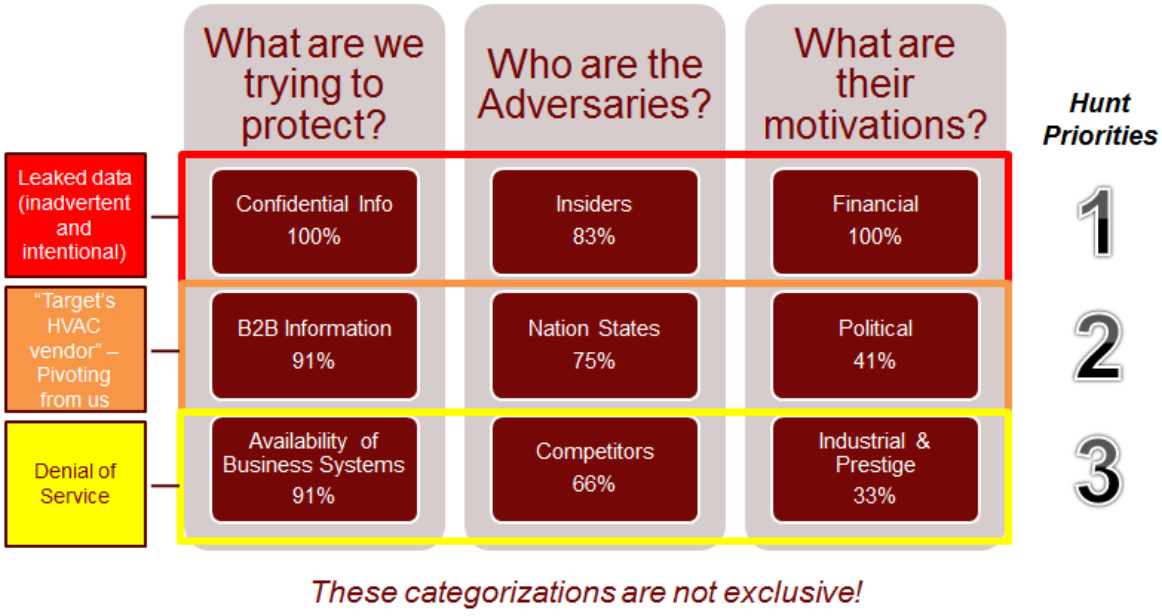


Figure 2. Adversary Survey Results to Begin to Prioritze Hunting Actions

The more of a cross-section within the organization, the comprehensive the results will be. Appendix A lists a series of questions that can be used as a basis for a survey.

3. Wait, What is Threat Hunting?

Threat Hunting can be defined as "the [proactive] pursuit of abnormal activity on servers and endpoints that may be signs of compromise, intrusion, or exfiltration of data [--both from external and internal entities]" (Gregory, 2017). To note, servers can include Windows, Linux, appliances, network devices, or modules that are acting to serve up a resource. An endpoint can be a laptop, mobile device, or other system that the proverbial user interacts with. True Threat Hunting is the area just beyond the automated detection capabilities of an organization. Simply put: it is the point where the human analyst or Threat Hunter must make the call on whether or not there has been a compromise, devoid of a definitive alert. Figure 3 illustrates the entire detection strategy that can be utilized. The more manual the detection area, the more skilled the Hunter must be.

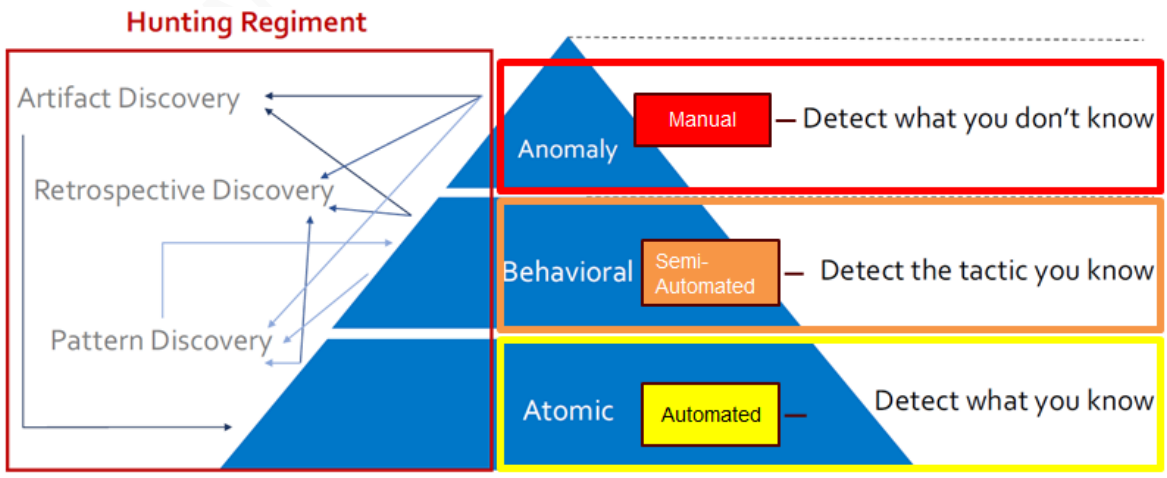


Figure 3. The Hunting Regiment in Relation to The Organization's Detection Strategy (Merritt & Concannon, 2017).

Not all hunts can produce indicators of compromise, but when possible, it is the area where the human Hunter leverages automation to assist with both behavioral and atomic types of detection. For the biggest return, hunting and incident response need to work together.

3.1. A Note on Incident Response

Incident response (IR) is a necessary component of Threat Hunting. According to Gartner, "Hunting success relies on a mature security operations center (SOC) and cyberincident response team (CIRT) functions" (Chuvakin, 2017). This is often true; however, it is not an absolute requirement to hunt. A mature organization might boast in having a robust set of procedures on how to handle malware, Denial of Service, and other attacks in place. A new organization, or a new response team, might have only a generic response plan. Regardless of the level of maturity, without some processes in place, hunting becomes a high fidelity alerting regime. The bigger value is achieved with hunting and IR working harmoniously. In Figure 4, the relationship between IR and Threat Hunting is shown:

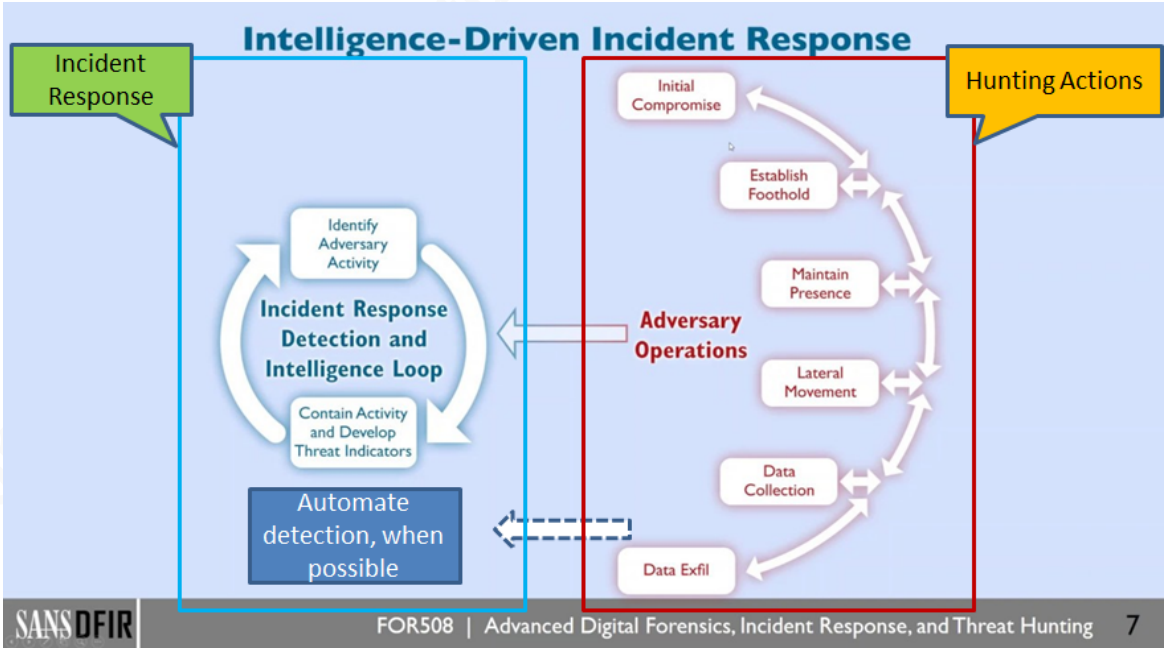


Figure 4. Threat Hunting and the Incident Response Relationship (Lee, 2016).

When possible, Indicators of Compromise (IOCs) should be worked back into the automated detection system. Future alerts and detection patterns would trigger the IR process and not necessarily the Hunter. One such means to identify active adversaries is with the application of Active Defense, or Offensive Countermeasures.

4. Offensive Countermeasures in the Hunt

In the pursuit of a human adversary, Offensive Countermeasures can act as a force multiplier in traditional Threat Hunting operations. Offensive Countermeasures are a set of techniques that can be leveraged to proactively pursue adversaries. The countermeasures focus on three Active Defense categories, referred to as the three A's (AAA). They are: Annoyance, Attribution, and Attack (Strand, Asadoorian, Robish, & Donnelly, 2013). Attribution will be the focus and primary method to hunt for active insider threats. Strand et al. (2013) provides its definition when they say, "Attribution is focused on knowing who is attacking you" (2013). As simple as it may sound, illuminating who is attacking an organization is a challenging endeavor. Challenges such as virtual private networks (VPNs), compromised hosts being used as an attack platform, proxies, and other obfuscation techniques help adversaries hide their identity. From an insider perspective, attribution might seem easier because within the enterprise network, hosts, software, and the users of those services should be known. A lack of security or detection capabilities could leave blind spots. Split tunneling for web traffic and lack of an always-on VPN solution are just a few areas where monitoring the behavior of a user can be degraded. On the endpoint, Data Loss Prevention (DLP) and other Endpoint Detection and Response (EDR) agents attempt to bring light to the poorly lit areas of the organization. Often these platforms do not (or cannot) account for encrypted or obfuscated data. In the case of DLP, alerting on encrypted files often yields noise and creates alert fatigue. Active Defense techniques are a great way to reduce alert noise; however, consultation of the legal department is a must before going live.

4.1. Legal Advice

The organization's appetite for implementing Offensive Countermeasures will vary. Before actively engaging any adversaries, both internal and external, an organization should obtain guidance on the limits of the Active Defense techniques. Similar to any information security program, both the legal and management buy-in is a key to success. A simple mechanism for preparing the environment is to review the logon and warning banners for the organization. According to the authors of Offensive Countermeasures, "Warning banners are key because they allow [the organization] to

define the boundaries of [the] networks and the actions [an organization] may take to verify the security of the networks" (Strand et al, 2013). Put another way, warning banners can notify any user, including an insider, that they are closely being monitored for any leaked information, both production and test data. By stating this upfront, the argument of entrapment could be mitigated. Seek legal and management counsel prior to hunting insiders with Offensive Countermeasures. Technology such as the Web Bug Server is a means to hunt the intentional leak of data from an organization.

4.2. The Web Bug Server

The Web Bug Server is essentially a command and control (C2) server for the defender. In its most rudimentary form, the server is a collector for the call back traffic. This server is best utilized when set up outside of the organization's infrastructure. One example is Amazon's Web Services (AWS), or other Infrastructure as a Service (IaaS) provider. Attributing the server back to the organization could alert the attacker that the document is not only bugged, but being monitored by the organization. The second part to the server is the bugged document itself. This document contains a simple web bug that is not seen by the attacker ("Web Bug Server, n.d.). It can be "embedded inside word processing documents. These bugs are hidden to the casual observer by using things like linked style sheets and 1 pixel images" (Smith, 1999). The important note is that the bug can be placed inside of any document that can process Hyper Text Markup Language (HTML). The primary target file for these bugs would be Office documents, such as .doc, .docx, .xls, .xlsx, and even HTML formatted emails.

Now that both the C2 server and bugged document are in play, the attacker must be enticed with the bugged document. It can be placed in a common share or location the insider might only have access to. Ideally, this share should take effort to access so the argument of accidental disclosure can be lessened. Regardless of how the document makes it out of the organization, when it is opened, a simple callback is sent to the Web Bug Server from the device or host that opens the document. This callback contains identifying information. "Each entry includes the document id which can change by editing the .doc file, the type of media request that was triggered, the IP address the

connection came from, and the time the connection was made" ("Web Bug Server, n.d.). The document ID can be made unique to the user or area it came from to help with attributing where it came from, or who accessed it. Figure 5 illustrates the conceptual infrastructure for using the Web Bug Server:

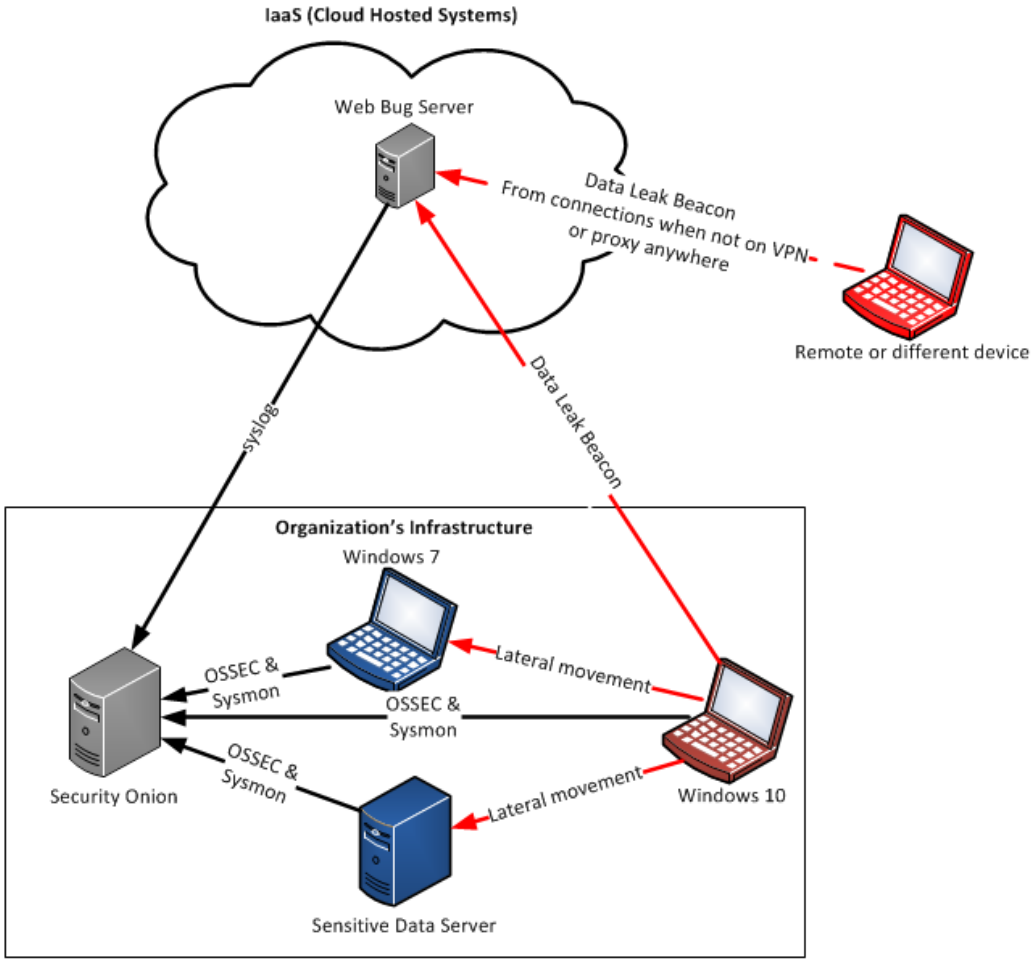


Figure 5. Web Bug Server Conceptual Infrastructure

This indicator works to suggest that there may be active insiders in an organization. From that knowledge, a more succinct list to identify the insider threat can be formulated.

Although excellent for pinpointing insider threats, the leaked document could also indicate an adversary has made it through the network successfully and achieved his or her actions on the objective. In the case of the leaked document the action or goal is data

exfiltration. If the Threat Hunter has a suspicion that there are leaks happening or potentially happening, Mole Hunt helps to narrow the focus.

4.3. Molehunt

In some cases, the insiders might already be known, so Molehunt can be used for further attribution. Molehunt takes the simple Web Bug concept to the next level. By leveraging a list, an insider hunt campaign can easily be built by feeding the list to a Python script. Molehunt.py takes the list of insiders and automatically generates unique and bugged documents. Since Molehunt relies on the Web Bug Server for collecting responses, one can easily dive deeper into the insider hunt, if required ("Molehunt," n.d.). Figure 6 highlights the features of Molehunt with a recommended configuration.

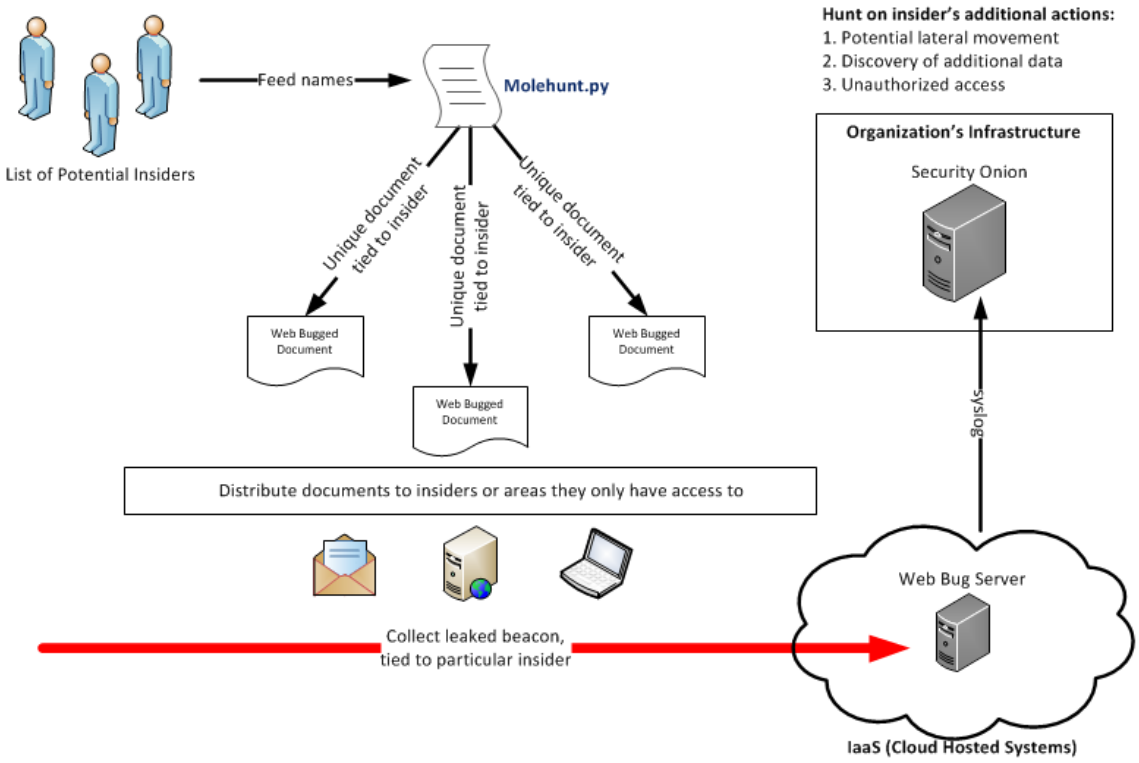


Figure 6. Molehunt Conceptual Infrastructure

This data, if received, would be a warning sign that leaks are taking place before any real damage occurs and can even implicate the insider. True to the Threat Hunting definition, this is indeed the proactive pursuit of abnormal, and unwanted, activity on the

organization's systems indicating data exfiltration. The operationalization of Threat Hunting, in particular Active Defense, is the next step in decreasing the detection delta.

5. A Threat Hunting Platform: Security Onion

Similar to a rifle or bow, the Threat Hunter requires a set of tools to accomplish the hunt. Commonly thought of as just a Network Security Monitoring (NSM) tool, Security Onion has one of the most expansive sets of security and intrusion detection tools around, including host monitoring. Furthermore, it is open source—free! The core tenants that make Security Onion an extensible platform for Threat Hunting are: full packet capture abilities, network and host –based intrusion detection, built-in analysis tools, and the ability to integrate with the Critical Stack Intel platform for threat feeds (Burks, 2017). All of these, combined with the ability to run in most virtual environments, lend it to being a necessary and vital tool for intrusion detection, both reactive and proactive. The fundamental problem that Security Onion addresses, at least from a Threat Hunting perspective, is the ability to centrally collect log data and network packet captures from nearly anything that can generate a log. New to the platform is the integration of Elasticsearch, Logstash, and Kibana (ELK), which expands the Threat Hunter's arsenal.

5.1. ELK Hunting

The Elastic Stack is now a feature of Security Onion, which enables the Threat Hunter like never before. From an insider hunting perspective, the alerts received by the Web Bug Server can be forwarded to Security Onion. A guide on how to set this up is located in Appendix C. Once ingested, the Threat Hunter can leverage Kibana to visualize the data from the leak, as well as, view the context around the systems or users who might be involved. Ultimately, the goal would be to determine if the insider is working alone, with other insiders, or even possibly if an advanced adversary is present and moving laterally. Once the insider or group of insiders has been identified, further hunting activities should be conducted. These activities could start with examining the insider's lateral movement, enumeration of additional services, or any unauthorized or denied access to data that the user should not be accessing. Preparing the environment

ahead of time is a crucial step in the hunting process. Kibana streamlines the searching and analysis of an intrusion, especially when fed with rich data from the organization's environment.

5.2. Windows Logging and Sysmon

To truly prepare the environment, several areas of logging should be considered, and especially for Windows hosts. In the most basic form, additional auditing for Windows hosts can yield the records required to hunt down human adversaries on an organization's network. As a more advanced configuration, the introduction of Sysmon, and OSSEC will add even more context to the hunt. Within Security Onion, the means to ingest these logs is built-in. This allows for organizations to more rapidly deploy a comprehensive solution, while maximizing the time the Threat Hunter can spend searching out the human adversaries. Figure 7 depicts a tiered approach to enabling the logging for an enterprise with Hunting in mind:

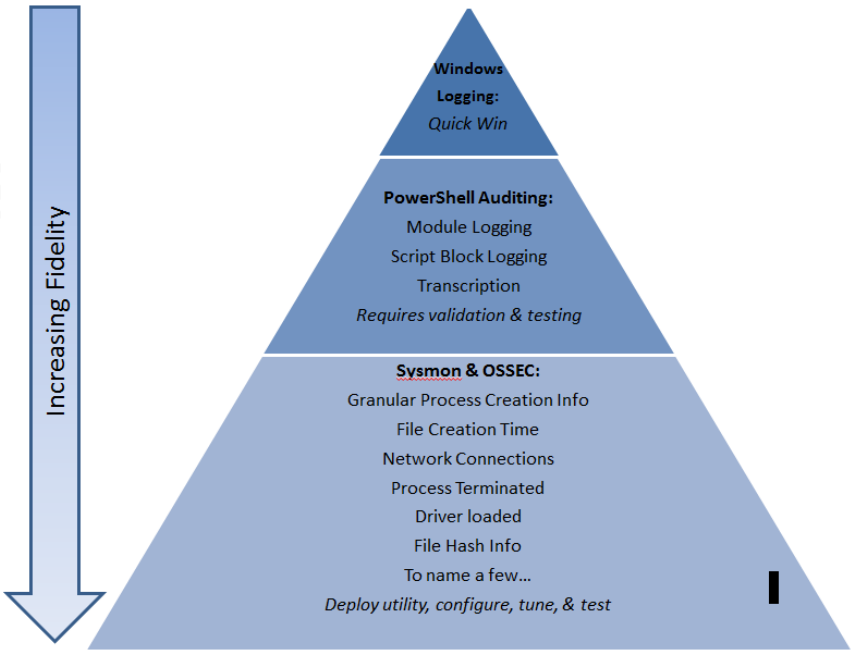


Figure 7. Tiered Top-Down Approach to Enabling Logging for Hunting ("Cheat Sheets," 2017)

The recommended log settings can be found in Appendix B. To note, even when logs and network traffic is being analyzed, there is still a possibility that an adversary can fool a system by leveraging a rootkit. Augmenting a platform such as Security Onion

with a live memory and disk acquisition capability, such as F-Response, is still recommended. This allows for further analysis by a malware analyst or forensic investigator, if the incident warrants a deeper look. Now that the environment is primed for the hunting season, the adversary's ability to remain undetected is diminished.

6. Open Season

With the environment prepped, the focus turns to identifying the active human adversaries. The Web Bug Server and Molehunt will be the primary means used for the active seeding and hunting of the insiders. But how can an organization be so sure that the attacker will go after the bugged documents? The answer might be simpler than expected. The first answer relates to the previously discussed threat modeling and setting the organization's hunt priorities based on the data that requires protection and its relevant adversaries. The second key stems from human nature.

6.1. Observed Human Behavior

Both the attackers and victims are fundamentally the same: they are human. When phishing attacks are conducted, the adversary is attempting to exploit the trust of a user. In many cases, a spoofed website or document is sent to lure the victim into clicking on a link or opening a document. The more authentic the email appears, the more likely the user is to act. From a phishing study of 15 participants, the following was observed: "six do not ever click links from email and five will click on a link if it looks interesting. Two only click on links if it from a web-site where they have an account. One will click on links from email only if associated with a specific transaction" (Dhamija, Tygar, & Hearst, 2006). Interestingly, nearly half of the participants would click on an interesting link. Because attackers and phished users are both people, the allure for an adversary to open or exfiltrate interesting data, if that is their intent, is more than likely a motivating factor. For example, if a web server hosted a public directory that contained 50 files and one of those contained a file that was named customer_data.docx and the rest of the files had a non-descript name like index.html, the likelihood that the customer_data.docx file would be stolen would be greater.

In a separate study conducted utilizing honey tokens, researchers discovered common motivations for data misuse. The scenario conducted by Shabtai et al. (2016) involved 173 participants who posed as bankers. Each banker's task was to approve loans by one of two means: The first was to approve the loan legally and the second method was to fund the loan via an outside source, illegally. The more loans and the higher the amount of the loan approved equated to more commission for the banker. Some of the loans were legitimate and some were actually seeded with honeytoken. If the loan was approved illegally, the banker risked being fired. What the study uncovered was that "attractive loans (i.e., loans at higher amounts) were more prone to illegal approval" (Shabtai et al., 2016). This means, there was a direct correlation between the amount of personal gain and data misuse. Everyone has their price. The second finding was around religion. "The Religiosity factor was also found to be statistically significant. More specifically, the more religious the participant was, the less illegal actions he or she performed" (Shabtai et al., 2016). Detection was conducted using the honeytokens, which enabled the researches to uncover when decoy data was used and by which banker (Shabtai et al., 2016). Because of the observations of the human behavior, the same tactic can be leveraged against an organization's adversaries when hunting for insiders.

6.2. Scenario One: Insiders or Actions on Objectives

The following scenario outlines a very real and probable incident that will be used as an example of hunting an internal, suspected, leaker.

Scenario One

- The organization is trying to protect trade secrets
- The organization's biggest threat is the insider (or an advanced adversary already in the network)
- A leak of the information could cause brand degradation and lost revenue
- Data is suspected as being exfiltrated from the organization
- The scope of insider's actions is unknown
- *Hunt the insiders!*

Based on the scenario, it is now up to the Threat Hunter to discover if this activity is occurring on the organization's systems. Included in Appendix E is the flowchart to

illustrate the process. To assist with the Hunt, Kibana and attribution via Active Defense techniques will be leveraged.

6.3. Discovering Insiders Using Kibana and Active Defense

With the priorities and environment set, the Threat Hunter can begin the pursuit of insiders within the organization's infrastructure. The procedure for this scenario can be referenced in Appendix E. The first phase of the scenario will take the generically bugged document and place it in areas that might either be near truly sensitive data, or in a common area where most users have access to, for example, an internal departmental share. This location should still take effort to access to diminish any accidental disclosure. The detailed steps on how to bug a .docx file are explained in Appendix C. Additionally, the document could be distributed to the company, or department within the company, via an official looking email. Both of these approaches are the wide net approach. Once seeded, the Hunter can watch for the trap to be tripped. In Figure 8, the alerts received in Security Onion from the Web Bug server are noted.

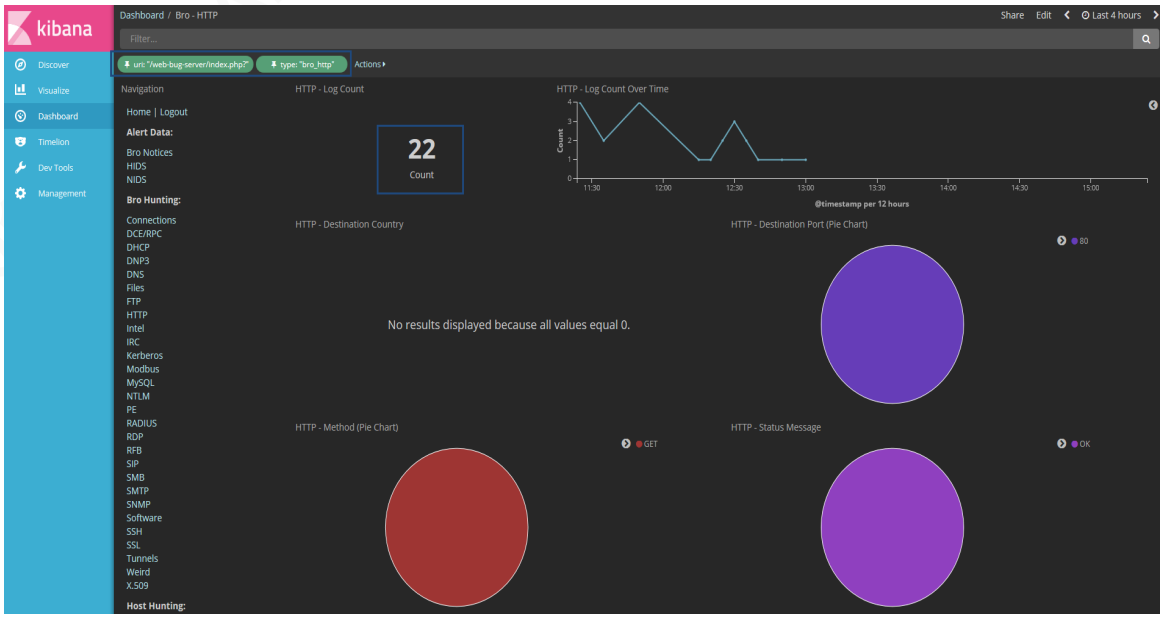


Figure 8. Kibana Dashboard for Viewing the Call Backs to the Web Bug Server

Based on the alerts, two conclusions can be drawn. The first shows that someone has opened the document. The second inference, and more importantly, is the source IP

address. If the IP is the Network Address Translation (NAT'd) IP, it might be assumed the document has not left the organization. In the event the IP is a public IP and not associated with the organization, the document has been opened from outside the organization. If the latter is true, it can be deduced that there is at least one insider leaking information. The question then becomes: has the insider purposely leaked the document or was it a mistake? To track down that answer, a more purpose- built tool can be utilized: Molehunt.

6.4. Pinpointing the Mole(s)

With the knowledge that documents are being leaked, the Threat Hunter must determine who the moles are and if the leaks are intentional with the use of Molehunt. Based on the unique document IDs, a list of potential insiders can be determined. Furthermore, if an administrative or Human Resources (HR) representative can be involved at this stage, they can help in narrowing the list down. Taking the public source IP from the Web Bug Server alert, a network WHOIS can be utilized to reference the geolocation of the IP. Taking the city list to HR, a list of employees or contractors who live in that area might help with attribution. Although not a perfect method of attribution, the technique removes more uncertainty from who is an insider. Armed with the list of individuals, Molehunt is now ready to accept submissions.

Feeding the list into Molehunt.py will produce uniquely bugged documents for each human. Once created, the Threat Hunter should rename each document to something enticing, while keeping the filename unique, mapped, and referenced so they do not get confused. Appendix C includes the detailed steps of creating the uniquely bugged documents. Distributing the documents is the next challenge. In this round, it is time to place the bugged document into a location that requires the insider a degree of work to access. For example, the analyst should create a directory on a share that the insider would have to actively search to discover. Additionally, an extra warning banner could be placed on the bugged directory, which might seem like a legitimate directory, to further warn the insider. Doing this reduces the case that the insider was ignorant to the fact they were in an area where they should not be. Within that directory, the bugged document, and some others to decrease suspicion, can be staged. Distributing the document will

require thought and preparation as to not alert the insider to the fact that they are on a watch list. Now, if the document is exfiltrated, it will have an alert tying directly to the user, or mole. As these alerts are generated, the hope is that the count is small. From there, additional context can be added to the incident.

The final step in the Hunt is to fully scope the adversary's actions. With a short list of insiders, the Threat Hunter can focus on the additional actions, if any, that were performed. At this point, it is a great hope that the insider has not leaked anything other than the bugged documents. Further Hunting on the actions, such as lateral movement, additional discovery, or additional sensitive data access can be explored. Because the environment is prepped, a historical search into the host logs (event logs, PowerShell, Sysmon, and OSSEC events) can piece the puzzle together. In a recent report by Eduard Kovacs from SecurityWeek, a National Security Agency (NSA) contractor was charged with leaking classified information. The investigation used similar Hunting techniques to hone in on the insider. "An internal audit showed that a total of six individuals had printed the leaked report and one of them was Winner. An analysis of the desk computers used by these six individuals revealed that Winner had contacted the news outlet via email" (Kovacs, 2017). Because the environment was primed, it was a relatively easy process to hunt down the leaker. At the end of the hunting phase, the incident should be scoped and ready to move into the capable hands of the IR Team. In some cases, it might be necessary to understand if the insider is working alone, or in collusion with others—either internal or external. In the case of the NSA contractor, the external communication was identified between her and the Intercept reducing any uncertainty that she was the sole proprietor of the leak (Kovacs, 2017). At this stage in the incident, it might be time to call upon the organization's IR retainer for additional incident handling support.

7. Conclusion

Large scale data breaches have occurred and will continue to occur unless the mindsets of security practitioners change. WikiLeaks, the Arab Spring, and the Occupy movements are significant examples of the damage leaked information can do to governments and organizations alike. Bots and machines are not the advanced

adversaries, humans are. Because of that reality, Threat Hunting should focus on going after, or hunting, the humans. Simply sifting through logs and alerts may be effective, but it does not lend to a proactive pursuit of intrusions within or against an organization. This is the way it has been done and it produces marginal results, while burning out the human analyst. Although the numbers are decreasing for the identification of a breach, they still lend to a ripe environment for an attacker to succeed. For that reason, Offensive Countermeasures and Threat Hunting must be synonymous. Each organization's appetite for the Active Defense spectrum of AAA will be different. Most can and should focus on the first two A's: Annoyance and Attribution. By determining what needs to be protected and who the adversaries are that the organization faces, lends itself to a strategy or prioritized Hunting program and application of these techniques.

With direction, the Threat Hunter can focus effort and prepare the environment for a successful Hunt. Boiling the ocean will not yield positive results, so an organization might need to start with a platform, such as Security Onion and basic logging. When complete, the next phase can be used to enable further logging, which increases the fidelity of the data a Hunter can analyze. Combined with Active Defense tools of Web Bug Server and Molehunt, the Hunter can go on the offense and proactively seek out insiders who might be leaking data, hopefully before any real data is leaked. Based on the results, Molehunt can help target and validate the moles on an organization's network. From discovery of a mole, additional context will help to scope the adversary's actions. Based on the organization's needs, this extremely rich data can be used to kick off an IR process or other actions as needed. It is time to let the machines hunt the machines and humans hunt humans. (Merritt & Concannon, 2017).

References

Adversarial Tactics, Techniques & Common Knowledge. (n.d.). Retrieved June 15, 2017, from https://attack.mitre.org/wiki/Main_Page

Apache Logs. (n.d.). Retrieved June 20, 2017, from https://www.loggly.com/docs/sending-apache-logs/

Bachrach, J. (2011, July & aug.). WikiHistory: Did the Leaks Inspire the Arab Spring? Retrieved June 15, 2017, from http://www.worldaffairsjournal.org/article/wikihistory-did-leaks-inspire-arab-spring

Bejtlich, R. (1970, January 01). Try the Critical Stack Intel Client. Retrieved June 20, 2017, from https://taosecurity.blogspot.com/2015/01/try-critical-stack-intel-client.html

Burks, D. (2017, March 16). Introduction to Security Onion. Retrieved June 15, 2017, from https://github.com/Security-Onion-Solutions/security-onion/wiki/IntroductionToSecurityOnion

Carbone, R. (2015, March 19). Using Sysmon to Enrich Security Onion' s Host Level Capabilities. Retrieved June 15, 2017, from https://digital-forensics.sans.org/community/papers/gcfa/sysmon-enrich-security-onions-host-level-capabilities_10612

Cheat Sheets to help you in configuring your systems. (2017). Retrieved June 15, 2017, from https://www.malwarearchaeology.com/cheat-sheets/

Chuvakin, A. (2017, April 06). My "How to Hunt for Security Threats" Paper Published. Retrieved June 15, 2017, from http://blogs.gartner.com/anton-chuvakin/2017/04/06/my-how-to-hunt-for-security-threats-paper-published/

Costa, D. L., Michael, A. J., Matthew, C. L., Perl, S. J., Silowash, G. J., & Spooner, D. L.

(2016, May). An Insider Threat Indicator Ontology. Retrieved July 05, 2017, from https://resources.sei.cmu.edu/asset_files/TechnicalReport/2016_005_001_454627.pdf

Dhamija, R., Tygar, J. D., & Hearst, M. (2006, April). Why Phishing Works. Retrieved June 15, 2017, from http://people.ischool.berkeley.edu/~tygar/papers/Phishing/why_phishing_works.pdf

Gregory, P. H. (2017, April). Threat Hunting for Dummies. Retrieved June 15, 2017, from https://www.carbonblack.com/resource/threat-hunting-dummies/

Kovacs, E. (2017, June 06). NSA Contractor Charged With Leaking Russia Hacking Report. Retrieved July 06, 2017, from http://www.securityweek.com/nsa-contractor-charged-leaking-russia-hacking-report

Lee, R. (2016, November 28). FOR508 - Advanced Incident Response and Threat Hunting Course Updates: Hunting Guide. Retrieved June 15, 2017, from https://www.youtube.com/watch?v=C-0JD1Fwk7U

Lee, R. M., & Bianco, D. (2016, August). Generating Hypotheses for Successful Threat Hunting. Retrieved June 15, 2017, from https://www.sans.org/reading-room/whitepapers/threats/generating-hypotheses-successful-threat-hunting-37172

Merritt, K., & Concannon, B. (2017, May 16). Vector8 Threat Hunting & Advanced Analytics Course [PDF]. Denver.

Molehunt. (n.d.). Retrieved June 15, 2017, from https://github.com/adhdproject/adhdproject.github.io/blob/master/Tools/Molehunt.md

M-Trends 2017 A View From the Front Lines [PDF]. (2017). California.

Robish, E. (2016, November 7). Bugging Microsoft Files: Part 1 – Docx Files using

Microsoft Word. Retrieved June 20, 2017, from https://www.blackhillsinfosec.com/?p=5409

Romero, F. (2010, November 29). Top 10 Leaks. Retrieved June 15, 2017, from http://content.time.com/time/specials/packages/article/0,28804,2006558_2006562_2006567,00.html

Saba, M. (2011, September 17). Wall Street protesters inspired by Arab Spring movement. Retrieved June 15, 2017, from http://www.cnn.com/2011/09/16/tech/social-media/twitter-occupy-wall-street/

Shabtai, A., Bercovitch, M., Rokach, L., Gal, Y., Elovici, Y., & Shmueli, E. (2016). Behavioral Study of Users When Interacting with Active Honeytokens. ACM Transactions On Information & System Security (TISSEC), 18(3), 9:1-9:21. doi:10.1145/2854152

Smith, R. (1999, November 11). The Web Bug FAQ. Retrieved June 15, 2017, from https://w2.eff.org/Privacy/Marketing/web_bug.html

Strand, J., Asadoorian, P., Robish, E., & Donnelly, B. (2013). Offensive countermeasures: the art of active defense. Place of publication not identified: Publisher not identified.

Sundaramurthy, S. C., McHugh, J., Rajagopalan, X., & Wesch, M. (2014, Sept. & oct.). An Anthropological Approach to Studying CSIRTs. Retrieved June 15, 2017, from https://pdfs.semanticscholar.org/d31a/1c631d9a74f144c5291ed50765ac36e760ad.pdf

Web Bug Server. (n.d.). Retrieved June 15, 2017, from https://github.com/adhdproject/adhdproject.github.io/blob/master/Tools/WebBugServer.md

WikiLeaks Release 1.0. (2009, December). Retrieved June 15, 2017, from https://archive.org/details/26c3-3567-en-wikileaks_release_10

Appendix A
Threat Modeling Survey Questions

Build these into your survey platform of choice. The results will help to prioritize hunting within the organization.

1. What are we trying to protect?

- ☐ Confidential personal information of customers and employees
- ☐ Confidential system information (data & app security, infosec program)
- ☐ Confidential business information (third party info, business 2 business)
- ☐ Availability of the the systems that process customer information
- ☐ Integrity of the data that is processed via our information systems

Other (please specify) _____

2. What are the implications to the business if what we're trying to protect is exposed or impeded (items identified previously may map to one or more of the following)?

- ☐ Damage to the business reputation
- ☐ Loss of confidentiality (losing confidential records)
- ☐ Loss of trust from customers and employees
- ☐ Financial impacts
- ☐ Loss of availability of internal or external systems
- ☐ Loss of integrity of the data & information processed

Other (please specify) _____

3. Who are the actors or adversaries we face?

- ☐ Nation States
- ☐ Competitors
- ☐ Organized Crime
- ☐ Script Kiddies

Matthew Hosburgh, matt.hosburgh@gmail.com

- [ ] Terrorists
- [ ] Hactivists
- [ ] Insiders
- [ ] Auditors (I am my own worst enemy)

Other (please specify) _____

4. What are the motivations of the adversary?

- [ ] Financial
- [ ] Military
- [ ] Industrial
- [ ] Ideological
- [ ] Political
- [ ] Prestige

Other (please specify) _____

6. Do you think we are well prepared to detect and respond to an advanced adversary?

Extremely poor	Poor	Neutral	Good	Without a doubt!
○	○	○	○	◉

Additional comments _____

7. At what level of attribution do you think is required to combat our adversaries?

- ○ Individual (think of the individual hacker)
- ○ Group (think organized crime syndicate or hactivist group e.g. anonymous or NSA)
- ○ Nation (think China, Russia, USA)

Matthew Hosburgh, matt.hosburgh@gmail.com

Only attribute to the attack or technique i.e. no specific attribution is needed to who is doing the attacking.

Other (please specify)

8. What do you think the most likely avenue of attack is?

- Email
- Drive by downloads
- Privileged access (insider)
- Weak credentials or authentication (e.g. single factor)
- User error or mis-configuration
- Users (via vishing, phishing, or social engineering)
- Insecure public infrastructure
- Business partners or vendors.
- Lost or stolen device
- Mobile device
- Wireless
- VPN
- Wired, or physically at one of our locations
- Denial of Service (DoS or DDoS)

Other (please specify)

9. How can we become a hard target for an adversary?

- Deploy advanced malware protection
- More robust user awareness
- Non-traditional security means (active defense techniques, such as annoyance, decoys, etc)
- Getting back to the basics (CIS top 20 or CSF)

Other (please specify)

Matthew Hosburgh, matt.hosburgh@gmail.com

Appendix B
Recommended Log Settings

Windows Logging Cheat Sheet

https://static1.squarespace.com/static/552092d5e4b0661088167e5c/t/580595db9f745688bc7477f6/1476761074992/Windows+Logging+Cheat+Sheet_ver_Oct_2016.pdf

Windows File Auditing Cheat Sheet

https://static1.squarespace.com/static/552092d5e4b0661088167e5c/t/580596a8893fc021e944c4f9/1476761256829/Windows+File+Auditing+Cheat+Sheet+ver+Oct+2016.pdf

Windows Registry Auditing Cheat Sheet

https://static1.squarespace.com/static/552092d5e4b0661088167e5c/t/580596c2893fc021e944c5fe/1476761283602/Windows+Registry+Auditing+Cheat+Sheet+ver+Oct+2016.pdf

Windows PowerShell Logging Cheat Sheet

https://static1.squarespace.com/static/552092d5e4b0661088167e5c/t/578627e66b8f5b322df3ae5b/1468409832299/Windows+PowerShell+Logging+Cheat+Sheet+ver+June+2016+v2.pdf

Sysmon Config

https://github.com/SwiftOnSecurity/sysmon-config

Appendix C
Security Onion & ADHD Setup

There are several components to the setup of Security Onion to use Elastic Search and for ADHD to function properly. Below are the steps to rapidly set up Security Onion with the ability to ingest syslogs from ADHD for alerting purposes. The setup of the ADHD Web Bug Server and bugged documents will be demonstrated as well.

Security Onion with Elastic Search Setup

1. First, review the Hardware Requirements page.
2. Review the Release Notes page.
3. Download and verify our Security Onion ISO image.
4. Boot the ISO image and select the Install option.
5. Follow the prompts in the Xubuntu installer. If prompted with an encrypt home folder or encrypt partition option, **DO NOT** enable this feature. If asked about automatic updates, **DO NOT** enable automatic updates. Reboot into your new installation. Login using the username/password you specified during installation.
6. Verify that you have Internet connectivity. If necessary, configure your proxy settings.
7. Install updates and reboot.
8. Double-click the Setup icon. The Setup wizard will walk you through configuring /etc/network/interfaces and will then reboot.
9. TAKE A SNAPSHOT, OR BACKUP
10. After rebooting, log back in and start the Setup wizard again. It will detect that you have already configured /etc/network/interfaces and will walk you through the rest of the configuration. When prompted for Evaluation Mode or Production Mode, choose Evaluation Mode.

###Elastic Search Setup###

11. Download the script:

```
wget https://raw.githubusercontent.com/Security-Onion-Solutions/elastic-test/master/securityonion_elsa2elastic.sh
```

12. Run the script with sudo privileges:

```
sudo bash securityonion_elsa2elastic.sh
```

13. Access Kibana at https://localhost/app/kibana. Alternatively, you can run **so-allow** to permit your IP access to the web interface directly (as opposed to accessing it via your VM environment).

14. Once you've completed the Setup wizard, use the Desktop icons to login to Sguil, Squert, or Kibana.
15. Finally, review the Post Installation page for additional setup details

(Burke, 2017)

###Critical Stack Intel###

Consider augmenting Security Onion install with the Critical Stack Intel. This feature or hunting dashboard is pre-setup within Kibana. You will need to go to https://intel.criticalstack.com/user/sign_up and to sign up and configure your feeds and sensor before following the below steps.

1. First, from the sensor, download the script.deb script and execute it.

```
root@sensor:~# curl https://packagecloud.io/install/repositories/criticalstack/critical-stack-intel/script.deb.sh | sudo bash
```

2. Install the Critical Stack Intel Client:

```
root@sensor:~# apt-get install critical-stack-intel
```

3. Next configure the critical-stack-intel client with your API key.

```
root@sensor:~# critical-stack-intel api YOUR-API-KEY-GOES-HERE
```

4. Check Bro:

```
root@sensor:~# broctl check
```

5. Install the new policies:

```
root@sensor:~# broctl install
```

6. Use the command “critical-stack-intel list” command to show the active threat intelligence feeds.

```
root@sensor:~# critical-stack-intel list
```

(Bejtlich, 2015)

ADHD Setup: Web Bug Server

Before following the steps to configure the Web Bug Server, the Ubuntu system must be setup. Consider using AWS, or other IaaS provider. Alternatively, you can set many of the ADHD tools in an internal or private network as well.

1. Setup an instance of Ubuntu 16.04 server without any additional packages.
2. TAKE A SNAPSHOT OR BACKUP
3. Run the buildkit with the full install script:

```
bash -c "$(curl -sL
https://raw.githubusercontent.com/adhdproject/buildkit/master/adhd-install.sh)"
```

4. Download, extract, and move the webkit to the newly setup web root:

```
sudo wget https://github.com/adhdproject/webkit/archive/master.zip

sudo unzip master.zip

sudo mv webkit-master/* /var/www/
```

ADHD Setup: Bugging a .docx File

1. Create an enticing .docx file. Customer-data.docx, for example.
2. In the document, double-click to edit the header.
3. Select Insert > Quick Parts > Field
4. Click IncludePicture
5. Enter the specific URL for the Web Bug Server: **http://ec2-xx-xx-x-xx.us-east-2.compute.amazonaws.com/web-bug-server/index.php?id=marketing&type=img**
6. Check the box “Data not stored with document”
7. Check the box “Preserve formatting during updates”
8. Click Ok
9. Save the document
10. Now distribute or stage the bugged .docx file in a common area. This is the net casting phase to determine if leaks are happening. In this case, the target group is the Marketing Department in an attempt to see if the Marketing Department has insiders.
    (Robish, 2016)

ADHD Setup: Molehunt

1. If call backs were received from the previous document, a more comprehensive list of potential insiders can be built. Based on the list, specific documents can be tailored to the individual. If a call back is received, an insider has been implicated.
2. Configure the specific parameters within Molehunt that are specific to your Web Bug Server instance and database:

```
/opt/molehunt$ sudo nano molehunt.py

Replace the line BUILDER_STRING=None with
BUILDER_STRING="docz:/opt/docz.py/docz.py"

Edit the line SOURCE_STRING=None to (with your ADHD's IP address)
SOURCE_STRING="http://ec2-xx-xx-x-xx.us-east-
2.compute.amazonaws.com/web-bug-server/index.php?id=::ID&type=img"

Change the line MON_STRING=None to
MON_STRING="webbugserver:root:adhd:webbug"
```

3. Create the list of potential insiders:

```
/opt/molehunt$ sudo cat > marketing-targets.lst
Mae, Salley
Snowder, Edwardo
Manners, Bradly
```

4. Run and setup the Molehunt campaign:

```
/opt/molehunt$ sudo ./molehunt.py

>>> campaign
Campaign name: marketing-insiders

>>> honeyfile
Path to honeyfile: Customer-Data-Updated.docx

>>> targetfile
Path to targetfile: marketing-targets.lst

>>> generate
Generation complete...
Files saved to: campaign/marketing-insiders

>>> exit
```

5. Verify the results of the campaign output:

```
/opt/molehunt$ cd campaign/marketing-insiders

/opt/molehunt/campaign/adhd$ ls
Mae,_Salley.docx Snowder,_Edwardo.docx Manners,_Bradly.docx MAPPING.txt
```

"There it is, a file per target. Just send each target their corresponding version of the honeyfile (Don't forget to change the name back to something like 'Customer-Data-Updated.docx' right before you send it" ("Molehunt," n.d.). Additionally, check the MAPPING.txt to see which document is mapped to which ID. Be sure to update this if you change the name of the files.

ADHD Setup: Syslogs to Security Onion

1. Add the following configuration to your **/etc/rsyslog.conf** file on the Web Bug Server:

```
$IncludeConfig /etc/rsyslog.d/*.conf

#Only needs to be loaded once, like most rsyslog modules
$ModLoad imfile

#path to the file which you want to monitor
$InputFileName /var/log/apache2/access.log

#The tag apache can be changed to whatever you'd like
$InputFileTag apache:

#the name of file within rsyslogs working directory
$InputFileStateFile stat-apache-access

#By default this is set to 'notice'
$InputFileSeverity info

#This is necessary for file monitoring (no parameters)
$InputRunFileMonitor

#Set to how often the file should be polled. (default = 10s)
$InputFilePollInterval 10

if $programname == 'apache' then @x.x.x.x:514 #your syslog i.e. Kibana server
```

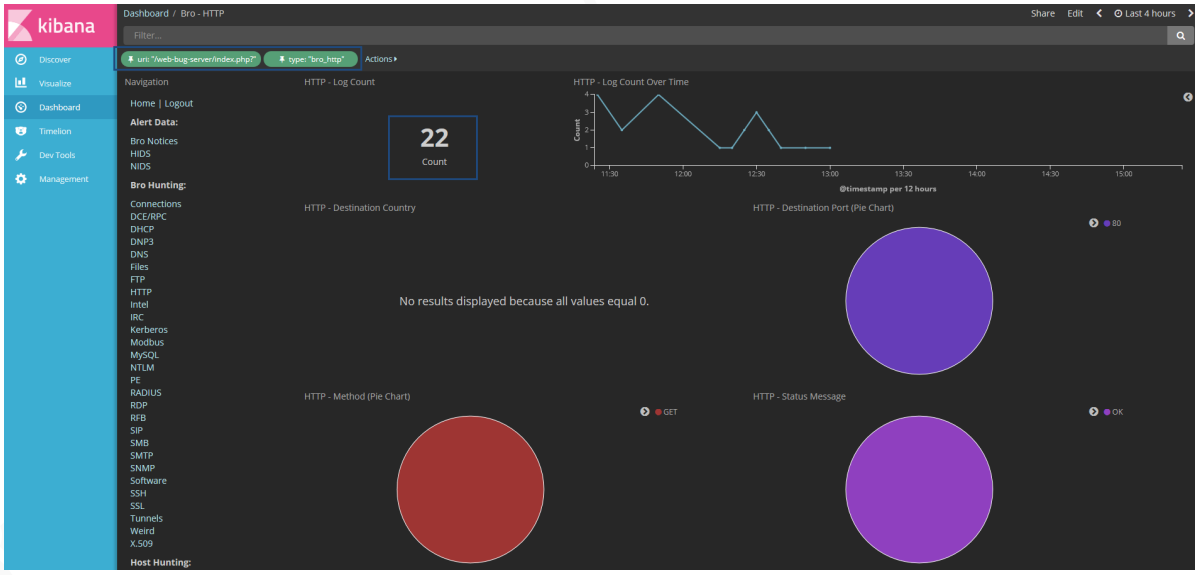
2. Restart the rsyslog service:

```
sudo /etc/init.d/rsyslog restart
```

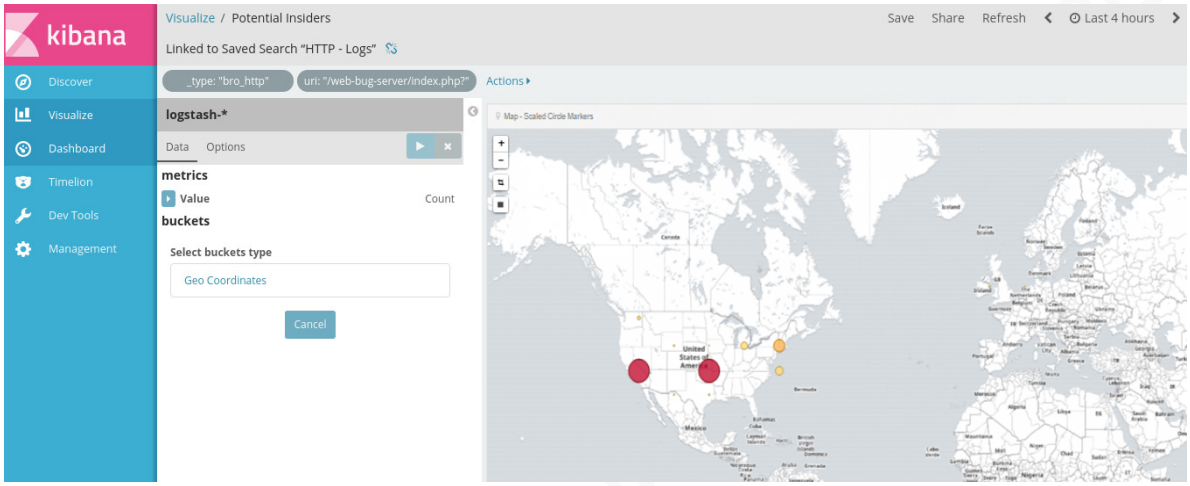
("Apache Logs," n.d.).

1. Test to ensure you are receiving logs from your Web Bug Server.
2. Within the **Bro Hunting: HTTP** dashboard, filter on the following:

The results will show any GET requests going to the Web Bug Server, which indicates an insider or that the document has been opened. You will only see results if Syslog is functioning and the bugged document has been opened.



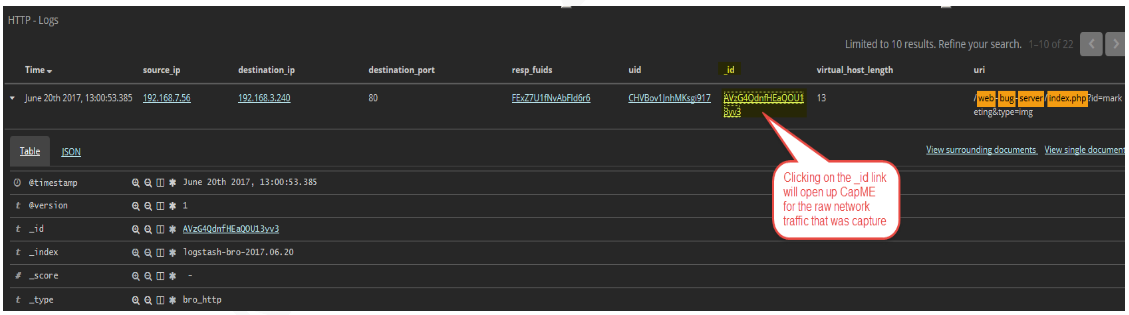
3. Additionally, a visualization can be configured to show the geolocation (IP geolocation) of the insider:

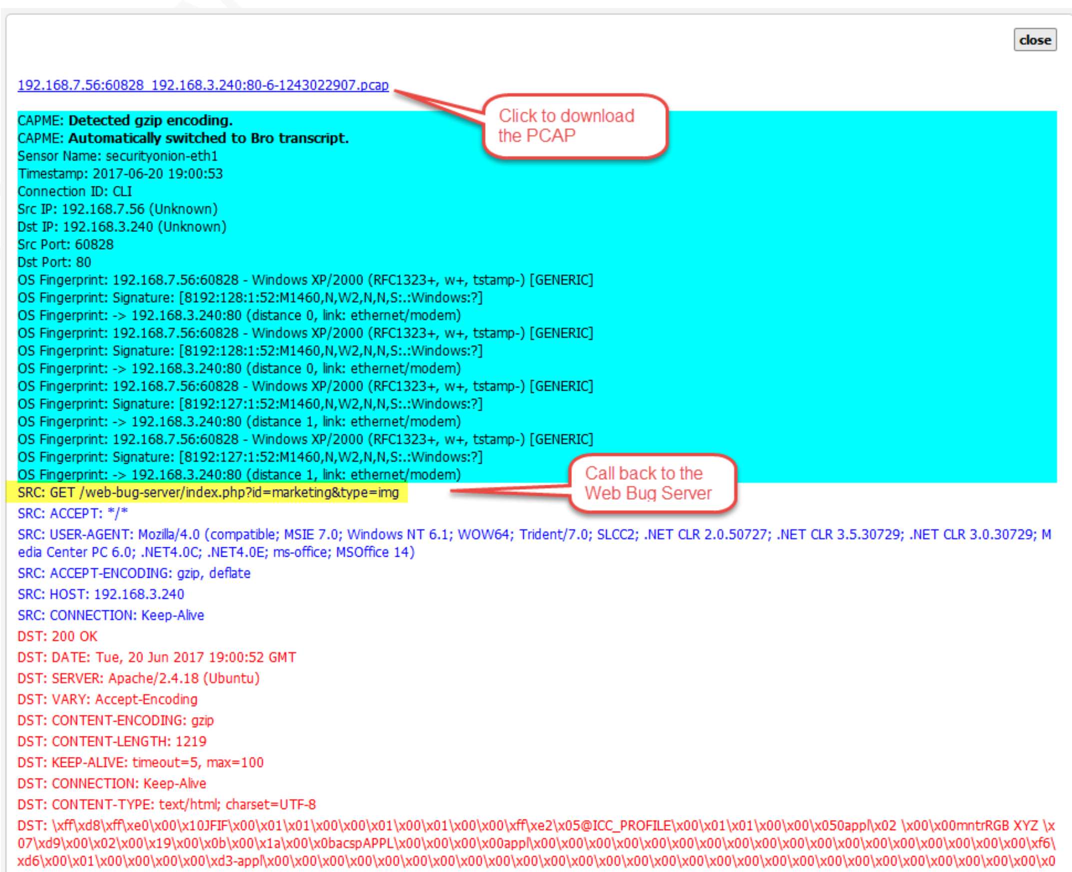


Matthew Hosburgh, matt.hosburgh@gmail.com

Appendix D
Network Captures

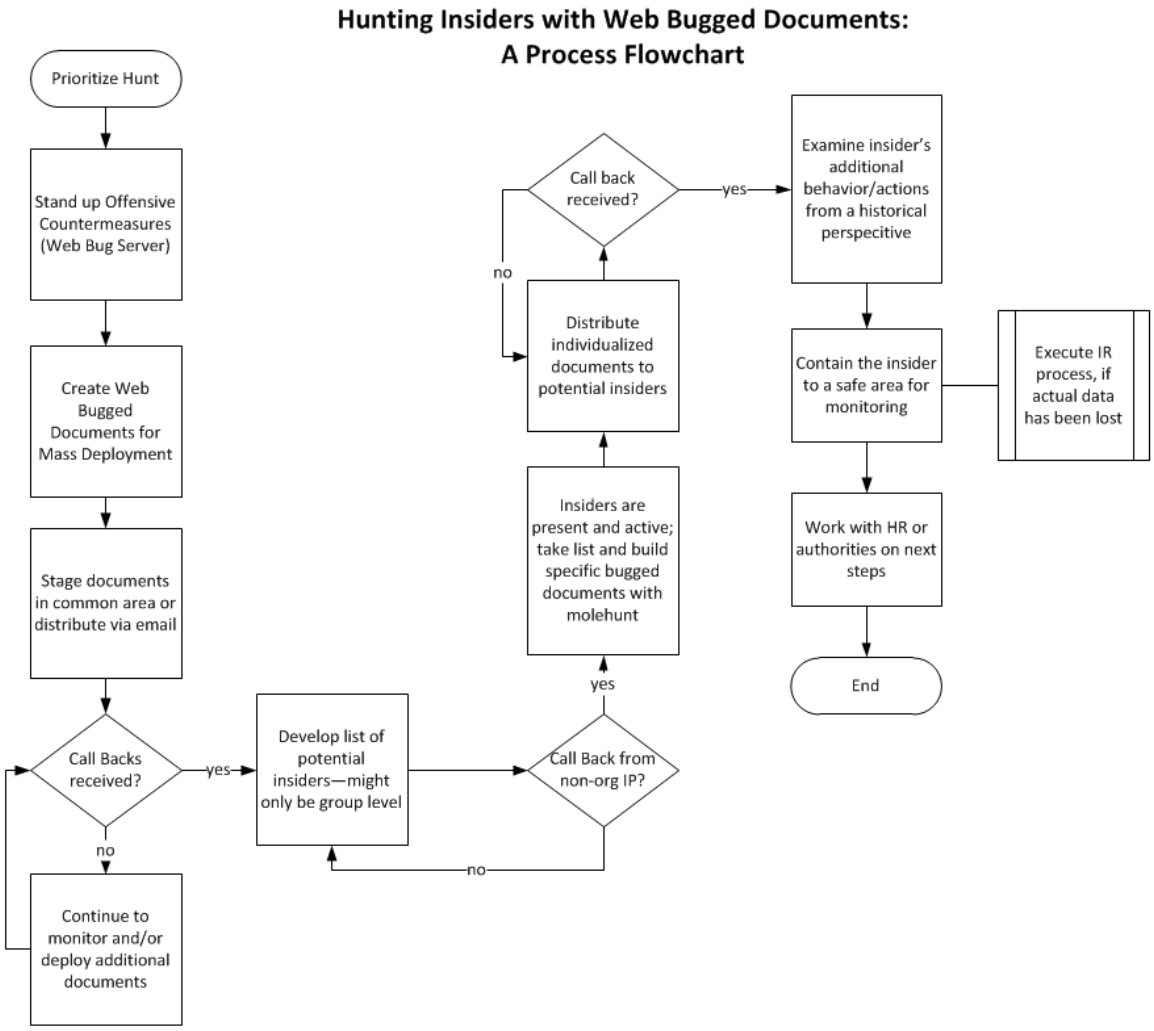
Pivoting from Kibana to CapMe is a great feature of Security Onion. This function is best served if there is a Security Onion sensor monitoring all network traffic to and from the Web Bug server. Clicking on the _id field within Kibana will bring up the network stream, which is similar to “Follow Stream” in Wireshark. Further, if the raw packet capture (PCAP) is desired, it can be downloaded for further analysis.





Matthew Hosburgh, matt.hosburgh@gmail.com

Appendix E
Threat Hunting Scenario Flowchart



Matthew Hosburgh, matt.hosburgh@gmail.com