Global Information Assurance Certification Paper

Home Field Advantage: Employing Active Detection Techniques

GIAC (GCIH) Gold Certification

Author: Benjamin Jackson, bbj@mayhemiclabs.com
Advisor: Rich Graves

Accepted: 2013-10-27

Abstract

The defensive measures used today by most enterprises have been repeatedly proven ineffective by adversaries who are actively attempting to bypass them. A shift from simply responding to alerts to employing more active techniques for intrusion detection is needed. This paper will discuss methods to detect intruders via tools such as internally facing honeypots, darknets, and other electronic booby traps.

1. Introduction

In sports, the term "home field advantage" is often discussed; the home team often knows all the quirks oddities of their "home field" due to constant practice on the same field day after day. Similar concepts are also true in military infantry operations; defenders, when fighting from prepared positions, will also attempt to use any quirks in the battlefield to give themselves an advantage over the attacker.

However, in computer network defense (CND), the advantage almost always seems to go to the attacker. There are a multitude of hypotheses on why this difference exists, but it doesn't change the fact that most analysts tasked with CND have neither surveyed their network for quirks they can take advantage of, nor have they established any of their own. By failing to do so, they're ignoring one of the key advantages that the defender has.

By looking for and adding some "quirks" to their network "home field" defenders will have the advantage return to them. By knowing how their network works and establishing defensive positions that will actively detect the attacker, the defender can more easily discover anomalous behavior on their network. Also, by simply using techniques that are a bit unorthodox, it may cause attackers to slow down by simply keeping them off balance, thus giving the defender a longer period of time to detect the attacker before the attacker exfiltrates data.

2. The Intrusion Kill Chain

The SANS Institute has a six step incident handling process that fits well within the CND model. The phases of the incident handling process are: Preparation, Identification, Containment, Eradication, Recovery, and Lessons learned. These six phases do a very good job of surmising the lifecycle of any computer security incident from the point of detection to the complete recovery of business.

In a counterpoint to the incident response model, in 2010, researchers from the Lockheed Martin Computer Incident Response Team (LM-CIRT) developed a seven step

Author Name, email@addressbbj@mayhemiclabs.com

Computer Network Offense (CNO) "Kill Chain" that describes the steps an attacker uses to gain access to a network (Hutchins, Cloppert, & Amin, 2011):

- Reconnaissance – Research, identification, and selection of target(s).
- Weaponization – Finding a suitable exploit and combining it with malware that allows for remote control of the targeted system(s).
- Delivery – Transmission of the exploit and malware into the target(s).
- Exploitation – Compromise of the target(s) with the exploit.
- Installation – Placement of the malware onto the target(s).
- Command and Control – Establishing a connection to the malware that allows the attacker to exert their remote control.
- Actions on Objectives – Now that they have access to the network, the attacker can seek out and exfiltrate the data they want.

Current CND techniques focus heavily on detecting and disrupting the attack within the first four phases of the kill chain. Once the attacker has completed phase four, they have bypassed the majority of the defenses in use and have almost a free run of the network. This is not a new problem, it's been around for close to 25 years; Bill Cheswick described this same situation when he described Bell Labs' network security as "a sort of crunchy shell around a soft, chewy center" (Cheswick, 1990), meaning that its defenses were focused mostly on the perimeter and there were little to no defenses that would track an attacker once they had passed the perimeter. Sadly, this problem has not been solved or otherwise gone away in almost 25 years.

3. Why "Home Field Advantage?"

The ability to detect and identify security incidents within a network is key for any incident response team. Without the ability to detect a security incident, the entire incident response model falls apart. Sadly, most enterprises are still using the same techniques they were using over five years ago: Firewalls, Anti-Virus (AV), and Intrusion Detection and Prevention (IDP) systems have been a staple in network defense, in one way or another, since the early 2000s. Unfortunately, due to enterprises' reliance on these tools for so long, attackers have learned where the weak points exist in these tools and have adjusted their tactics to bypass and avoid detection.

Author Name, email@addressbbj@mayhemiclabs.com

These adjustments have allowed the attackers to achieve incredible success. With stunning regularity major corporations and government organizations have admitted that they have been breached and the attackers avoided detection for an extended period of time. In most data breaches the time from compromise to exfiltration of data is measured in hours while the time from compromise to discovery is measured in months (Verizon RISK Team, 2013). This statistic is borne out in anecdotal evidence as well: To use one well known example, despite their use of firewalls and anti-virus, attackers had penetrated and maintained access to the New York Times network for months without being discovered (Perlroth, 2013).

The idea behind home field advantage is to use your network design to increase the difficulty for attackers to successfully execute their objectives without being detected. In CND, like any other military engagement, the defender controls the battlefield; therefore they have the home field advantage. By having the ability to change their network the defender can make the network much more hostile to the attacker, but this ability is often not realized. By making some simple changes, the defender can greatly increase their ability to detect and respond to incidents.

4. Establishing Defensive Positions

As we discussed, the first phase of the Incident Handling process is preparation. This most often involves training, honing skills, and just simply being ready to handle the next incident that occurs. However, this time should also be used to establish defensive positions to detect intruders. By making preparations during the downtime between incidents, the defender can focus solely on establishing solid and robust defenses at strategic points on their network rather than quickly making ad-hoc fortifications around key systems when the defenders knows there is a hostile entity in the network.

4.1. Honeypots

Honeypots have been used since 1990 (Cheswick, 1992) primarily for two main reasons: Either to gather information about new attacks or to develop better situational awareness of attacks against a network by external actors. While both these reasons still are valid use cases, the use of externally facing honeypots within a modern environment

suffers from three main drawbacks: They generate a large amount of "noise" from automated scanners, they do not provide visibility into client side attacks, and there is always a chance that they can be compromised and used to launch attacks against another organization.

While the effectiveness of external honeypots can be debated, changing their positioning to make them internally facing within the environment will solve two of the three problems. It will both reduce the number of alerts and increase the amount of actionable data from them and, since they are positioned internally, with proper firewalling the possibility of them being used to attack another network is greatly reduced.

There are two types of honeypots, low-interaction and high-interaction. Low-interaction honeypots attempt to impersonate vulnerable network services on an otherwise secure system; high-interaction honeypots actually run those services (Provos, 2004). While high-interaction honeypots often provide better data, they require near-constant maintenance to monitor, triage, and restore due to the fact that they will actually become compromised. Low-interaction honeypots, on the other hand, shouldn't become compromised; however, if an attacker uses a technique that the honeypot isn't expecting, the tool emulating the vulnerable service may not know how to give the correct response.

Choosing what type of honeypot to use is left as an exercise to the reader as each situation and each organization's risk appetite is different. However, when establishing requirements for an internal honeypot, the knowledge of how an attacker compromises a system is often secondary to knowing that an attacker is present on the network. Therefore, choosing a low interaction honeypot over a high interaction one will often provide enough information to be useful with while limiting the chances of the honeypot possibly becoming compromised itself.

4.1.1. The Nova System

Early low interaction honeypots used simple scripts on physical hardware to emulate vulnerable services (Cheswick, 1992). Indeed, this technique can still be used today either with physical or virtualized machines. However, there are other strategies available as well: In 2003, Niels Provos, a graduate student at the University of Michigan, released Honeyd, a tool to quickly generate multiple low interaction honeypots

using only a single system. Honeyd could be configured to listen for traffic for multiple IP addresses and emulate replies based on how various network stacks would handle the traffic. For example, if Honeyd was running on Linux, "personalities" could be configured so that the Linux host could emulate a Windows host, a FreeBSD host, and a Linux host on the network simultaenously. This was a major advancement in honeypot research as it allowed the quick creation of multiple honeypots each responding in unique ways.

While Honeyd was a major advancement it had two major shortcomings: The system had a high learning curve and the logs were difficult to interpret. In addition to this, development in Honeyd stopped in 2007 and the detection methods used by network scanners have continued to advance, making the personalities in Honeyd less effective. In 2012, a new project, the Network Obfuscation and Virtualized Anti-Reconnaissance ("Nova") system was released. Building upon the features pioneered by Honeyd, Nova provides a cleaner interface and decreases the learning curve in honeypot configuration and monitoring.

Let's go over how to use Nova to establish some defensive positions within an example network and how to use them to detect an intruder. In this example, we'll use the following simple network consisting of a workstation network segment, a server network segment, and a demilitarized zone (DMZ) all protected by a a firewall running the pfSense open source firewall. This network is shown in Figure 1.

Author Name, email@addressbbj@mayhemiclabs.com

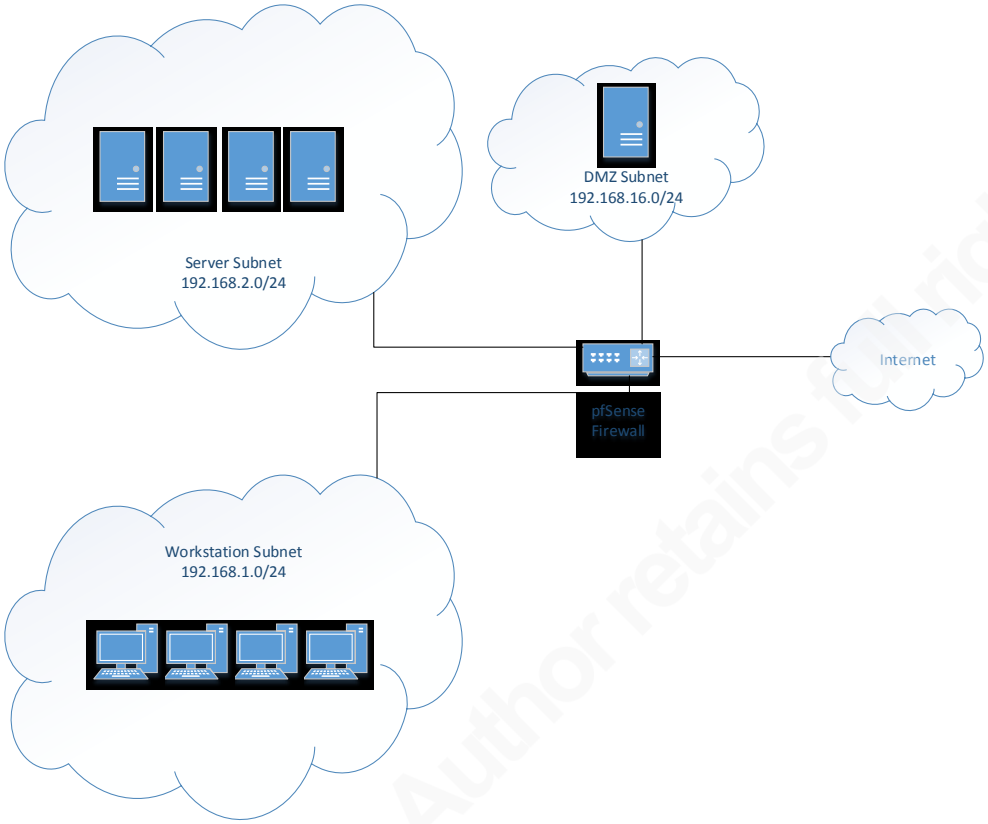


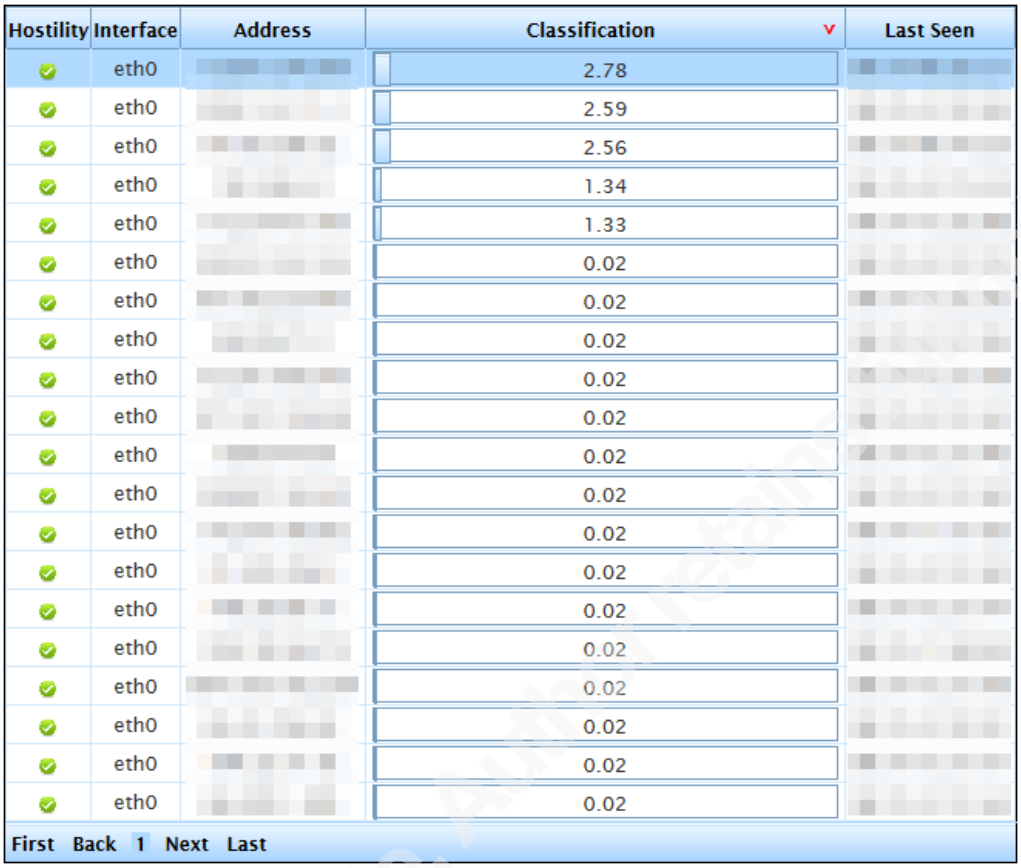
Figure 1 - Example Network

When deciding where to place honeypots, each situation is different. Since most attackers would likely attempt to locate and exfiltrate data from the servers, a good place to put a Nova installation in this network would be the server subnet.

Installing and configuring Nova is left as an exercise to the reader, but there is one major caveat to call out: When configuring, it is important to assign the honeypots personalities that match the operating systems already in use in your organization. If an attacker sees an operating system that looks different from the rest, while there is a chance that they may go after it first, it is more likely that they may ignore it as an oddity, which limits the honeypots use as decoys.

Once installed and running Nova should automatically start classifying hosts on the network that it can see traffic from. These hosts are shown to the user on the initial screen that contains the "suspects" table. This is a list of all the hosts generating traffic that is visible to Nova, whether Nova classifies them as "hostile" or "benign", when the host was last seen, and the interface that Nova saw it on.

Author Name, email@addressbbj@mayhemiclabs.com



Hostility	Interface	Address	Classification	Last Seen
	eth0		2.78	
	eth0		2.59	
	eth0		2.56	
	eth0		1.34	
	eth0		1.33	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	
	eth0		0.02	

First Back 1 Next Last

Figure 2 - Nova Suspects Table

Nova classifies traffic on a scale from 0 to 100, with a number of factors to generate the score. Primarily, Nova uses the interaction of hosts with its low interaction honeypots as an indicator of hostility, but it also will make decisions on the amount of traffic that are TCP SYN, FIN, RST, or SYN/ACK packets. It will also track the number of ports a host is interacting with along with the overall amount of traffic.

Now, let's assume that an attacker has compromised a workstation in our example network. The manner of the compromise is unimportant, but the attacker now has full access to the workstation. Now that the attacker has a beachhead onto the network, they begin to scan the network including the server subnet.

Author Name, email@addressbbj@mayhemiclabs.com

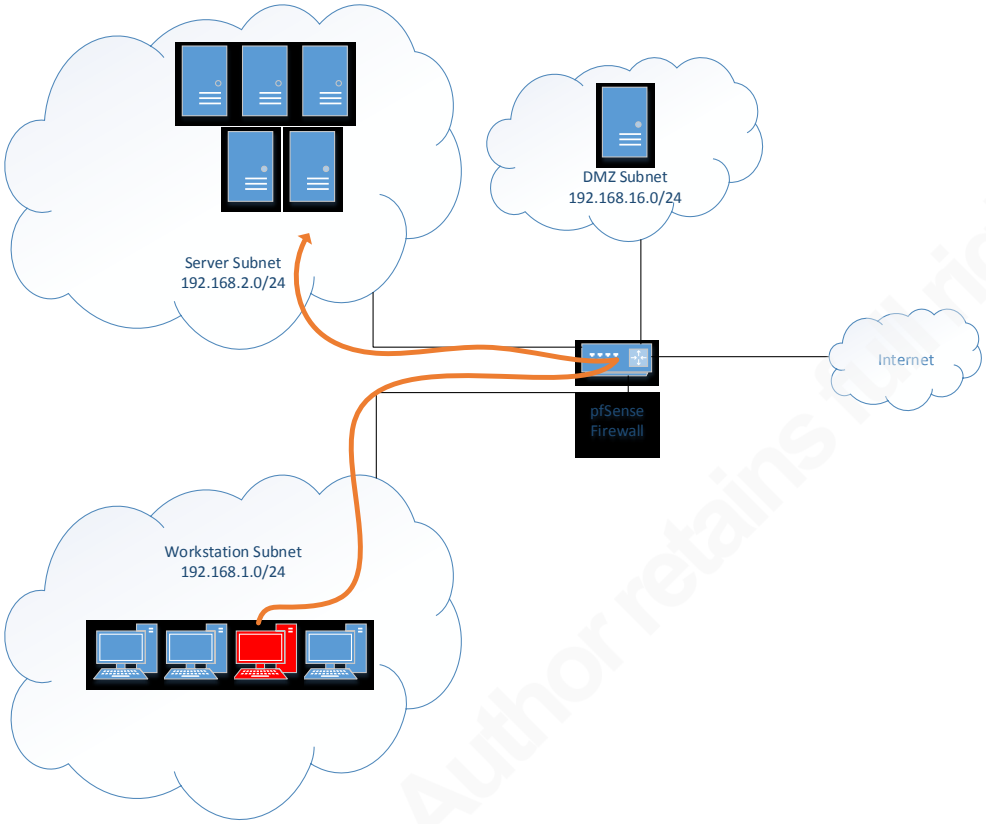
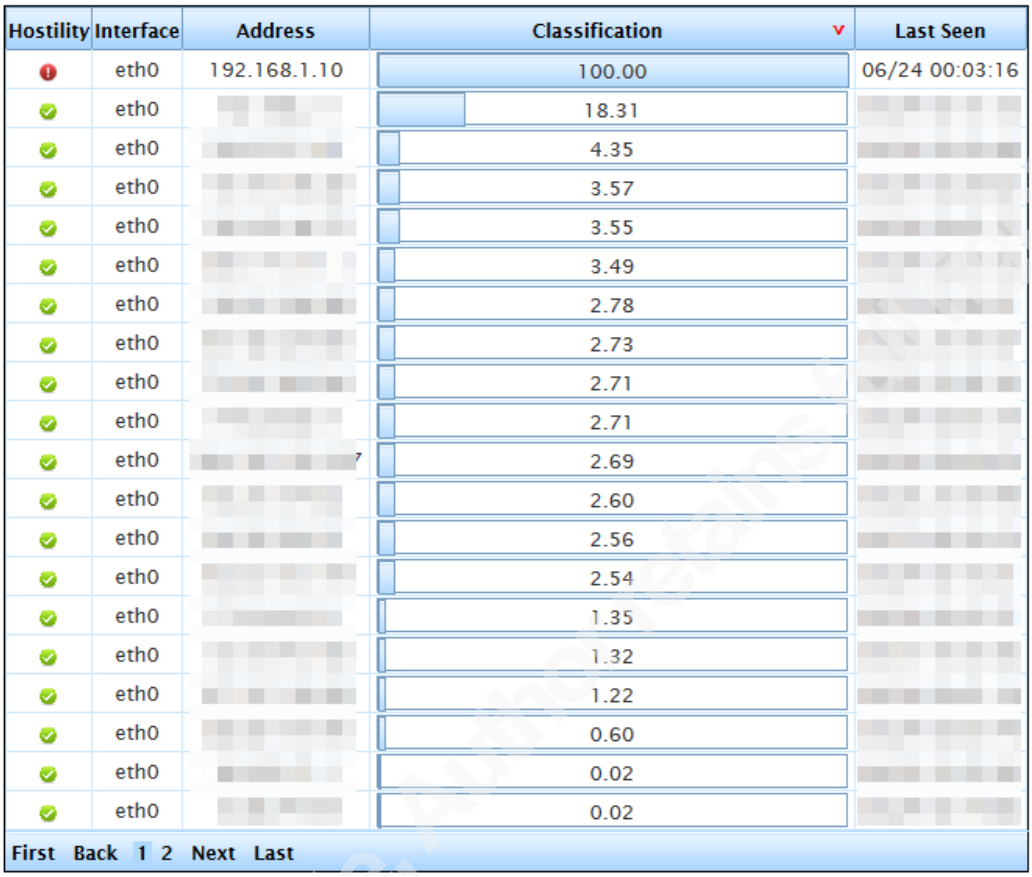


Figure 3 - Comprised Workstation Scanning Network

After the compromised workstation, indicated by the red workstation in Figure 3, starts scanning the server subnet, it will run into the Nova honeypots, indicated by the washed out servers. From the attacker's perspective, the scan is proceeding as expected and there are no oddities. Except instead of the two valid servers in this example, there are ten additional servers that the attacker has to spend their time probing.

As the workstation is interacting with the Nova honeypots, Nova will detect and classify that host as hostile. In this example, the compromised workstation has an address of 192.168.1.10. As Figure 4 shows, after the initial scan of the server subnet on 192.168.2.0/24, Nova has classified that host as hostile.



Hostility	Interface	Address	Classification	Last Seen
	eth0	192.168.1.10	100.00	06/24 00:03:16
	eth0		18.31	
	eth0		4.35	
	eth0		3.57	
	eth0		3.55	
	eth0		3.49	
	eth0		2.78	
	eth0		2.73	
	eth0		2.71	
	eth0		2.71	
	eth0		2.69	
	eth0		2.60	
	eth0		2.56	
	eth0		2.54	
	eth0		1.35	
	eth0		1.32	
	eth0		1.22	
	eth0		0.60	
	eth0		0.02	
	eth0		0.02	

First Back 1 2 Next Last

Figure 4 - Nova Suspects Table with Hostile Scanner

To show the effectiveness of Nova against Nmap, compare Figures 5 and 6. The figures show two Nmap scans of the example server subnet. Both scans were run with the default options for the "Intense" scan setting. In Figure 5, the scan was run with Nova's honeypots enabled. This scan shows twelve servers in the subnet, most of which are running Linux, and two running "other" operating systems.

Author Name, email@addressbbj@mayhemiclabs.com

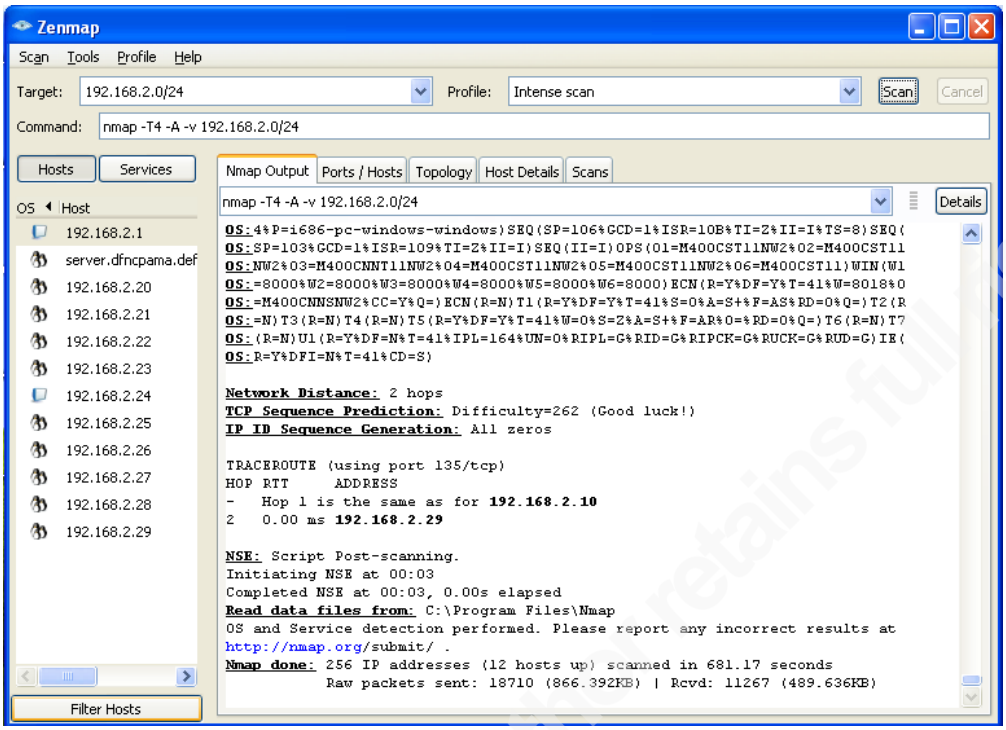


Figure 5 – Nmap scan with Nova enabled

In Figure 6, another scan with the same settings was run, with the Nova honeypots disabled.

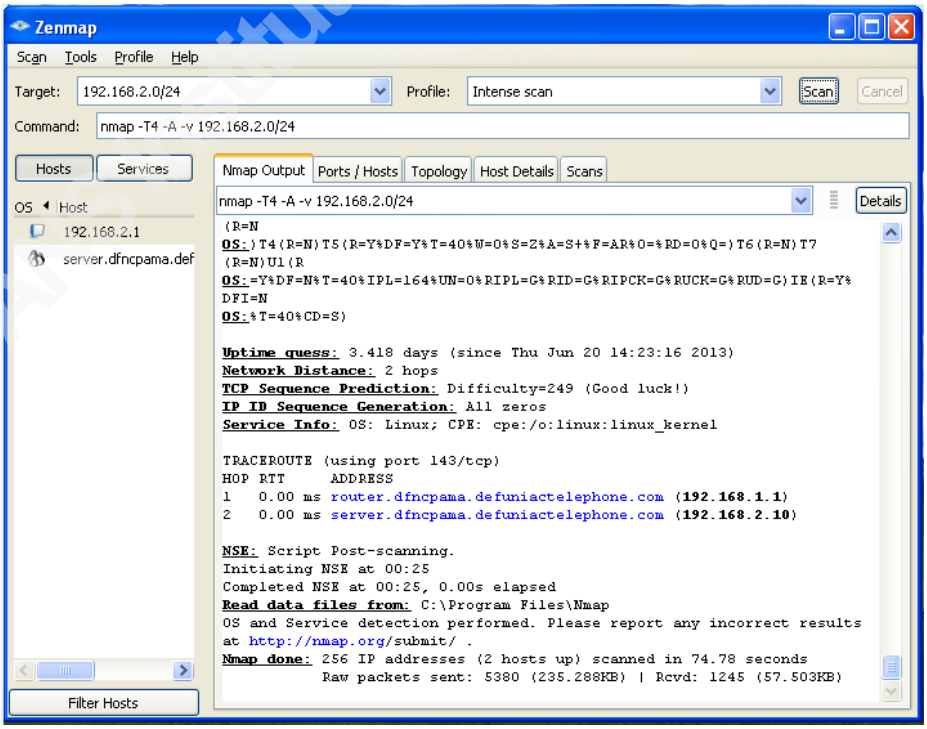


Figure 6 – Nmap scan with Nova disabled

Author Name, email@addressbbj@mayhemiclabs.com

Note the disappearance of the servers between 192.168.2.20-29 in Figure 6. These were the honeypots generated by Nova.

4.1.2. Decoy Services

In addition to honeypots, you can also use existing infrastructure to establish decoy services. Decoy services are similar to honeypots insomuch that no valid user should ever use them legitimately, but they are different as they would not be on a dedicated system. Two examples of tools that do this are SpiderTrap and WebLabyrinth. SpiderTrap, written by Ethan Robish of Black Hills Information Security (Strand, 2011), is a Python script that starts a web server and serves up a web page with four links leading back to the same server; when those pages are requested, the server serves up a similar page with four more links. WebLabyrinth, written by the author, behaves similarly but it is written in PHP in order to be used on an existing web server. WebLabyrinth also provides some additional features such as increased page randomization, support to trigger an IDP alert, and better logging.

Both tools are designed to make any web page spider become stuck in endless loop of meaningless web pages. Some commercial web scanners, in an attempt to map the entire website, will often crash or hang due to the endless pages. The advantage of this is twofold: beyond the ability to detect someone running a web spider within your environment, if the web spider becomes stuck in one of the traps and crashes or hangs, it the portions of the scan that it has already completed will not be saved. This will require the attacker to start the scan from the beginning.

Setting up additional decoy services can also be as simple as installing additional packages on existing servers. If an attacker is looking for a SQL server with customer data on it, and there is one SQL server in your environment, 100% of the attackers' effort will be focused on that single SQL server. However, if three additional SQL instances are installed on servers within the environment, the attacker's workload jumps fourfold. It can be said that this is nothing more than another version of security through obscurity and while that argument can be made, by adding that obscurity will increase the attackers' workload. Doing this will result in an increase in the amount of time expended

to achieve their objectives and by keeping them active on the network for a longer period, it increases the probability of detection.

There is some additional risk with running decoy services, as installing additional services on an endpoint invariably increases the attack surface of the network. However, if there is a robust patch management system already in place in the organization, adding additional services to patch should result in a negligible workload increase and the risk should be limited and the amount of effort to patch one versus multiple servers in the environment should be negligible.

4.2. Darknets

Darknets are portions of Internet Protocol (IP) address space that route to nowhere. These have been used by organizations to observe what is referred to as "Internet Background Radiation" (Wustrow, Karir, Bailey, Jahanian, & Huston, 2010) (IBN). IBN is generated by a number of events, such as misconfigured devices or faulty IP packets, but it is commonly a sign of spoofed network traffic. Groups that monitor IBN use it to detect events such as a Distributed Denial of Service (DDoS) attacks or changes in network scans. Doing this often requires large portions of routable IP space, but using the same techniques on a smaller scale to detect anomalies on your network is a simple exercise.

4.2.1. Establishing a simple Darknet

Although some people think that darknets are an almost magical art, most networks already have a darknet in them. Any kind of unused address space is, by definition, a "dark net". However, in a large network, subnets can be assigned and reassigned rapidly so it's important to explicitly designate subnets as permanently dark and not to be assigned. Otherwise, your monitored darknet could be "lit" with valid network traffic, resulting in false alarms.

Let's take our example network from section two and add a darknet to it. When setting up a Darknet, the only size limit is the available addres space. While one can make a darknet as small as a /30 network, the larger the addres space is the greater the chance of catching anomalous traffic. In this example we will establish a darknet of three separate subnets: 192.168.3.0/24, 192.168.15.0/24, and 192.168.17.0/24. These subnets

were chosen as they are adjacent to the network space already in use; if an attacker attempts to map the network via a sequential scan, they will likely end up scanning one, if not all, of these subnets due to their logical location. In Figure 7, the example network with the Darknet is shown. Since the darknet is just unused address space, it is represented by a dashed cloud.

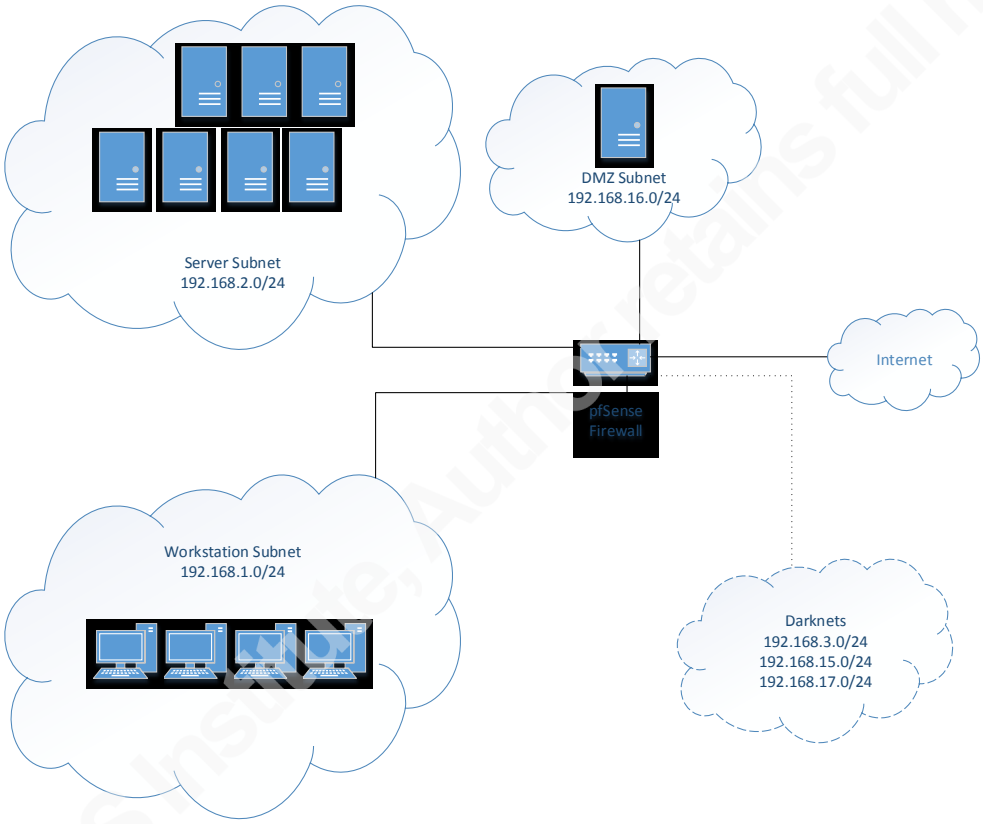


Figure 7 – Example Network with Darknet

As previously mentioned, any unused address space is a darknet. Therefore, once the subnets have been designated as not to be used, there is no further configuration required beyond monitoring. There are more advanced setups, specifically in networks that have non-RFC1918 address space, that specifically choose to null route internal darknets at their network edge. There are some arguments for this: It limits the possibility of a misconfiguration allowing hostile traffic out beyond their network and it also keeps the possibility of an inadvertently "lit" subnet from accidently leaking data beyond your network perimeter. However, this added step is completely optional.

4.2.2. Monitoring Darknets

Creating darknets is a fairly straightforward exercise; monitoring them, however, gets a little more complicated. Since the traffic gets forwarded into a black hole, there is nothing to receive the traffic. However, through the use of logs it is possible to observe the traffic and, if there is an IDS involved, possible to inspect it.

Traffic to an unused subnet on a firewall will often be allowed or denied based on the configuration of the catch all rule in the firewall. Often, that rule will not log the action taken on that traffic specifically, so the firewall needs to be explicitly configured to log traffic to the darknet. As shown in Figure 8, the example pfSense firewall is configured to explicitly log traffic to the darknet subnets from the Workstation subnet (192.168.1.0/24), similar rules would need to be created for each other subnets in order for the firewall to log the traffic. Once these rules are in place, any activity to the darknet will be allowed, but logged.

	ID	Proto	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description
		*	*	*	WORKSTATIONS Address	443 80 22	*	*		Anti-Lockout Rule
		*	WORKSTATIONS net	*	192.168.3.0/24	*	*	none		Darknet Logging Rule
		*	WORKSTATIONS net	*	192.168.15.0/24	*	*	none		Darknet Logging Rule
		*	WORKSTATIONS net	*	192.168.17.0/24	*	*	none		Darknet Logging Rule
		TCP	WORKSTATIONS net	*	*	80 (HTTP)	*	none		Allow LAN to HTTP any
		TCP	WORKSTATIONS net	*	*	443 (HTTPS)	*	none		Allow LAN to HTTPS any
		*	WORKSTATIONS net	*	SERVERS net	*	*	none		Allow LAN to any SERVERS
		*	WORKSTATIONS net	*	*	*	*	none		Deny all the things

Figure 8 - DarkNet Firewall Rules

Now, let's go back to our scenario that we used in 3.1.1 in which a compromised workstation is scanning the network. The attacker has finished mapping 192.168.2.0/24 and has moved to 192.168.3.0/24. As shown in Figure 9, the attacker is "routed" to one of the darknets. Since this traffic is not actually routed anywhere, it is shown as a dashed line going to the darknet cloud.

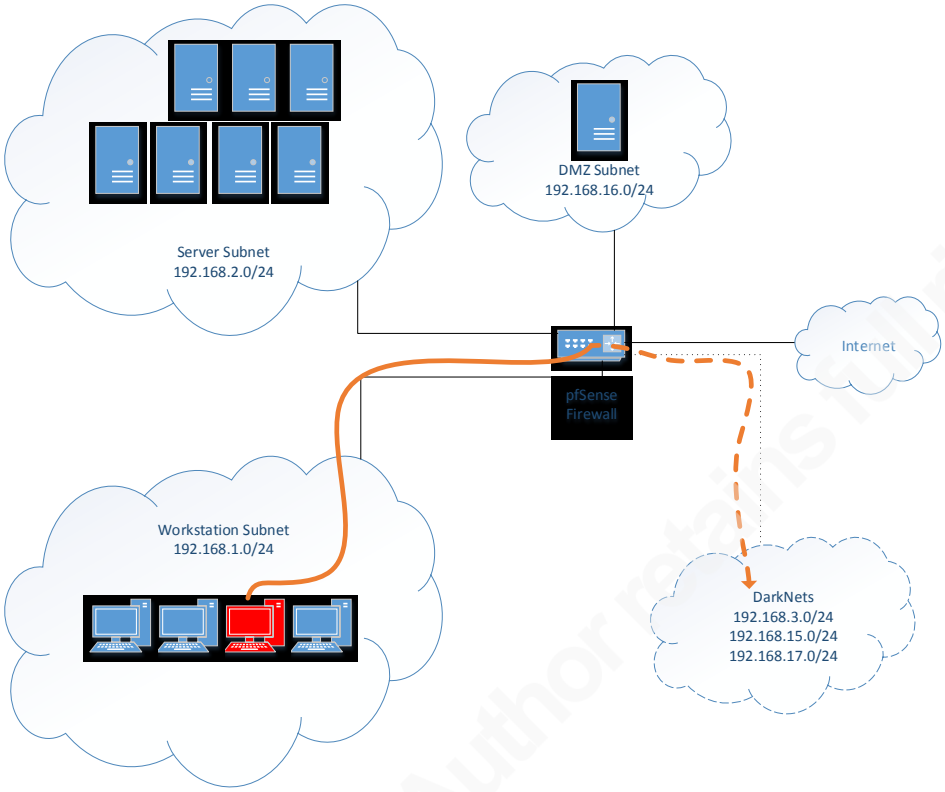


Figure 9 – Example Network with Darknet

As the attacker starts to scan 192.168.3.0/24, the firewall will start logging events due to the rules that were put in to allow and log the traffic. As one would expect, this causes quite a few events, as shown in Figure 10. An Nmap scan on default settings resulted in approximately twenty log messages per second just for the initial firewall scan.

Last 50 firewall log entries. Max(50)					
Act	Time	If	Source	Destination	Proto
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.75	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.76	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.77	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.78	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.79	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.80	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.81	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.82	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.83	ICMP
	Jun 26 22:09:01	WORKSTATIONS	192.168.1.10	192.168.3.84	ICMP
	Jun 26 22:09:02	WORKSTATIONS	192.168.1.10	192.168.3.87	ICMP

Figure 10 - Firewall Logs during scan

Author Name, email@addressbbj@mayhemiclabs.com

This method can be replicated on IDP systems as well. While IDP systems will give you slightly more information when compared to the firewall as they often will provide a packet capture of the traffic detected, if the attacker is aware that they are being monitored, IDP evasion techniques can be used. However, if a packet capture is available, it can allow a responder to make a better determination of what the traffic is and whether the traffic is a misconfiguration or is being generated with hostile intent.

In some cases, temporarily "lighting up" a darknet and routing it into a controled environment can provide additional data that firewall or IDP can't. After detecting a scan on the network, a high-interaction honeypot can be temporarily set up in a darknet that the attacker hasn't scanned yet. By knowing what the scan is looking for, the honeypot can be configured to specifically have those features, becoming a ripe target. The author has used this method multiple time to collect samples of malware that is attempting to propogate via open Windows file shares on the internal network. Completing this successfully requires a healthy amount of preperation, speed, skill, and a little bit of luck, if successfully completed, it can often provide a wealth of information regarding the tactics, techniques and procedures used by the attacker.

5. Detecting Unauthorized Data Access & Exfiltration

Data exfiltration is often one of the primary objectives of attackers when they again access to a network. Whether it is intellectual property, credit cards, or customer data, extracting that information and monetizing it are often the end goals. However, despite tools designed to detect and stop data exfilatration, such as data loss prevention (DLP) solutions, in 2012 69% of compomised businesses are told by an external party that they have suffered a data breach (Verizon RISK Team, 2013).

Tracking exfiltration of actual data is difficult and unfortunately setting up traps that one can generate alerts from isn't much easier. However, with the shortcomings in breach detection, reducing the time to detection even slighly is a major advantage.

5.1. Honeytokens & WebBugs

Honeytokens are objects whose use, like honeypots, indicate a possible network intrusion. However, instead of a physical device, it is a piece of data. This piece of data

can be anything from an e-mail address to a credit card number. Since this data is invalid, there should be no legitimate interaction with it. Normally, the interaction is caught either by an IDP system or, if the honeytoken is in a database, the application running as the database's front end.

Honeytokens are not limited to internal databases as well. Users often leave breadcrumbs on the Internet that can give an attacker who's looking for them data about your network. This can also be used against attackers: creating fake identities and positions within your business and building a believable background for them on various social networks allow you to monitor for attackers probing your network for possible social engineering victims or possibly competitors gathering information about your business in general. Probes to these fake identities can often indicate a precursor to a larger attack.

There are other ways to use and track honeytoken use within the environment: Fake customer records can be populated with valid e-mail addresses that are set up specifically for that purpose; any e-mail to those addresses could indicate a leak of that database. A small database containing a number of low-denomination pre-paid credit cards could be left on a SQL server; activity to those cards would indicate that someone has accessed that database and is using the data for fraud. An e-mail containing fictitious login credentials to a company system could be left in a senior manager's mail folder; those logins being used could indicate that the manger's e-mail was breached.

While honeytokens are a useful tool, detecting their exfiltration can be difficult if not impossible if they are encrypted while in transit. It would be advantageous if there were a way for the data to be able to be tracked once it was moved off network. Thankfully, there are way of doing this via web bugs. Web bugs are defined as "tracking devices embedded in web pages, executables or scripts that secretly monitor your activity on the web and send the information back to a 3rd party" (Nichols, 2003). However, they can also be used to monitor other data as well. Most people think web bugs are limited to web content, but they also can be used in specially crafted Microsoft office documents as well.

Microsoft Word and Microsoft Excel will render HTML as an office document even if the HTML is located in a .doc or .xls file. This means that a Word or Excel file

can use the same web bug techniques that are in use on the Web. For example, in Figure 11, we have a screenshot of a file called "CEO's Top Secret File.doc" visible in both a text editor and in Microsoft Word 2013. The HTML is rendered despite the file ending in .doc, making it appear to be a Word Document when opened in Word.

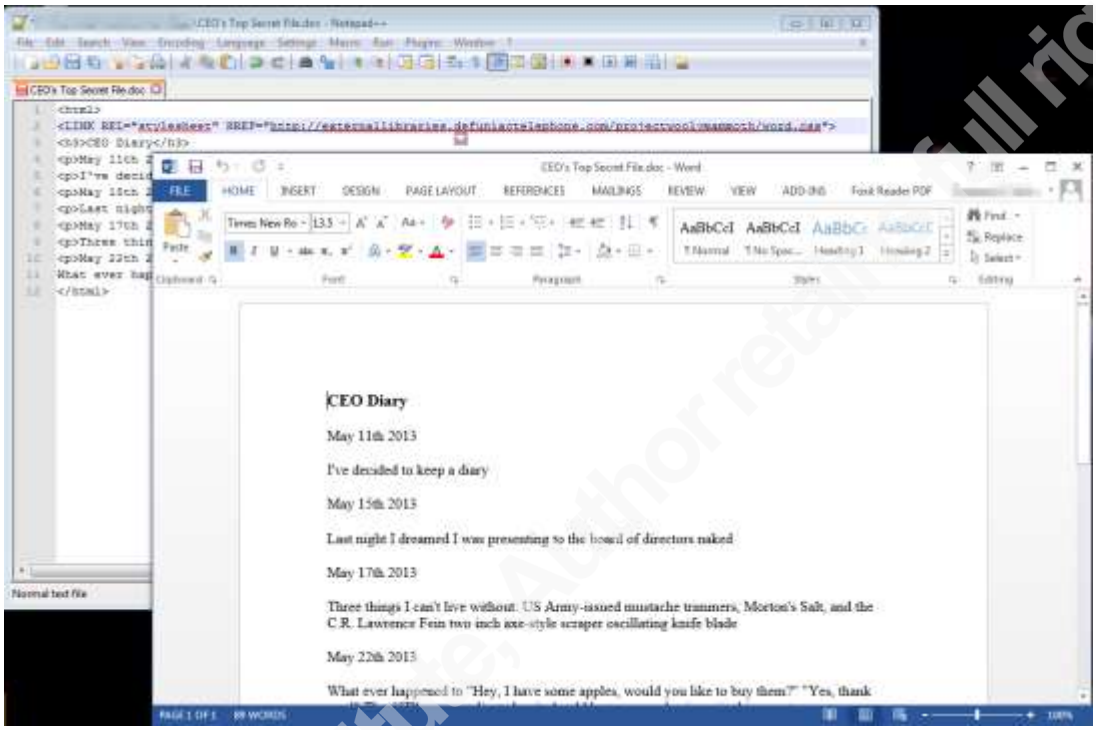


Figure 11

While a majority of web bugs are one pixel transparent GIFs, any kind of file can be used to bug a document. "CEO's Top Secret File.doc" uses a reference to an external Cascading Style Sheet (CSS), which is automatically downloaded by Word upon opening the file. This is visible if we load up a debugging proxy like Fiddler and monitor the web requests coming from the Word process, as shown in Figure 12.

#	Result	Protocol	Host	URL	Body	Caching	Content-Type	Process
1	200	HTTP	externallibraries.de...	/projectwoolymammoth/	0		text/plain	winword:7004
2	200	HTTP	externallibraries.de...	/projectwoolymammoth/word.css	104		text/css	winword:7004

Figure 12 – Fiddler log of Word 2013 grabbing the CSS file

Microsoft Excel will also render HTML content as an office document, however, in Excel 2007 and later, upon opening an XLS file that consists of HTML, it will display a warning, shown in Figure 13. Until the user confirms they want to open the file, the CSS file is not loaded, making the bug next to useless.

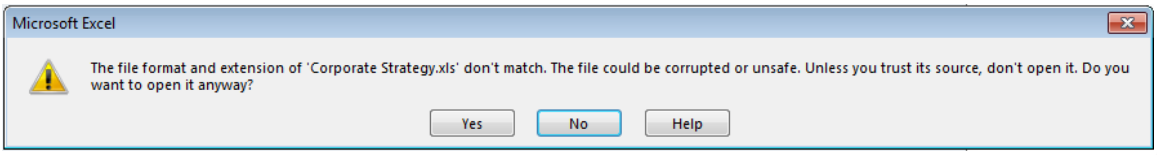


Figure 13 - Excel 2013 Warning

Using these web bug "features" in Word and Excel with honeytokens give defenders a powerful tool to detect data exfiltration and unauthorized access. Leaving files with interesting names such as "Employee Salaries" and "Upcoming Layoffs" on open shares that no one should access might detect an employee looking for information they shouldn't access. Similarly, generating files with names such as "Profit Forecast for next Quarter" or "Server Passwords" may attract an attacker who is looking to gain either Intellectual Property or further network access.

5.2. Aldrovandi

While web bugs are a powerful tool, there are limited options for tracking their use beyond manual log review and tracking. Generation of bugged files is also problematic and time consuming. The author, to address these issues, developed Aldrovandi, a web bug management system.

Aldrovandi consists of two main pieces: A management and logging console and a handler to receive requests from bugged documents and forward them to the management console. These pieces are separate as they are designed to be run two servers: an external server to handle the requests from the bugged documents and an internal server to manage the web bugs and handle the data forwarded from the external server.

The system automates the creation, management, and monitoring of web bugs. Creating a web bug on the administration page is a simple process of entering a unique identifier for the file, giving a description of the file, and hitting submit. Aldrovandi will then populate its tracking database, generate a skeletal file with the options given, and make it available to download. The file can then be downloaded and filled with fake data via a text editor. Figure 14 shows a generated file opened up directly from the download dialog.

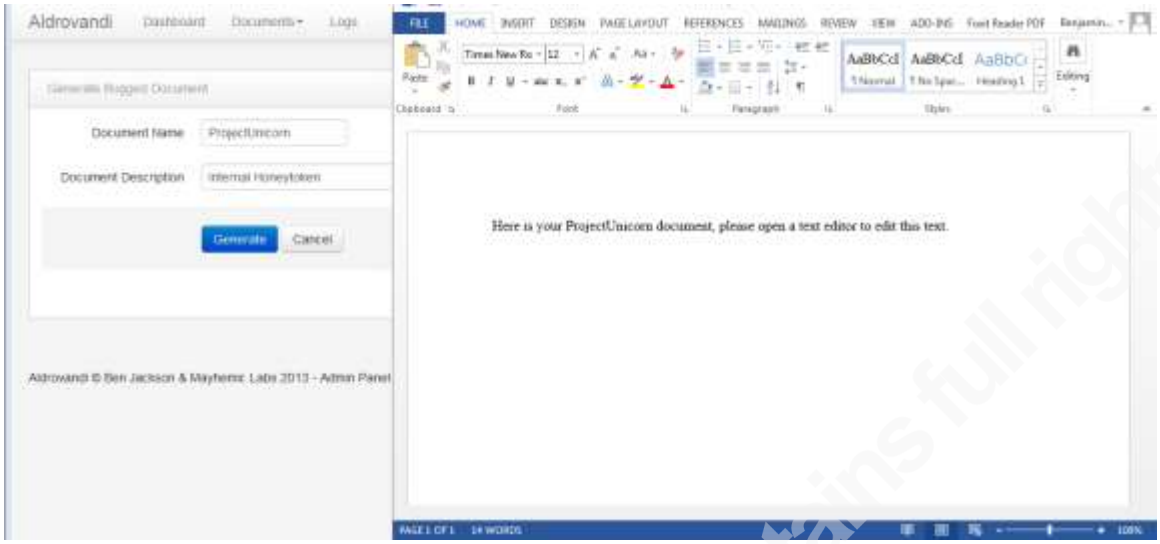


Figure 14 - Aldrovandi Generating a Document

When a bugged file is opened and the request is made, the handler parses the request, serves up a small CSS file to the requesting document, and forwards information about the request to the management console. The management console then checks the request against the database in order to make sure it's valid, and if so, logs the information about the request and generates an alert via e-mail.

Figure 15 - Example Aldrovandi Alert e-Mail

There are a number of possible issues with Aldrovandi and using web bugs for this purpose. Despite some protections, if the system is discovered, a skilled attacker could use it to generate disinformation and false alerts. In addition to this, if an attacker opens the bugged documents to a non-networked system, the document cannot report back into the server, rendering the tracking useless. While the non-networked system threat is beyond Aldrovandi's control, it does attempt to prevent false alerts by only generating alerts on valid webbugs. If an attacker attempts to generate false alerts, they would need to know what webbug identifiers are in use. If flooded by false alerts, by

tracking which webbugs an attacker knows about, inferences can be made as to where they gathered that information.

It is important to note that when Aldrovandi generates an alert, despite possibly having the IP address of someone opening a file exfiltrated from your network, it cannot be treated as attribution. Attribution of an attack is a very difficult process and it cannot be taken lightly. If the IP address traces back to a specific geographic location, the only evidence that provides is that the attacker routes their traffic through that IP address, nothing more.

5.3. Generating Fake Data

When creating fake data for an attacker to exfiltrate, it is important to keep the data believable. If you are a large Fortune 500 company, an attacker might be suspicious of a file labeled "Customer DB Backup.txt" that is only 200 records. Thankfully, there are numerous sources on the Internet that generate fake data suitable for databases. Three resources are http:// fakenamegenerator.com, http://mockaroo.com and http://generatedata.com.

All three services allow you to generate batches data, with each site focusing on a specific niche. Each include unique data types to choose from when configuring what fake data to generate, with mockaroo having the most extensive selection. When generating batches of data, generatedata limits you to 100 rows of data at a time, fakenamegenerator limits you to 50000, and mockaroo boasts an impressive 100000 records at time.

With these services it is trivial to generate fake data for fictitious customer databases, marketing lists, inventories, and almost anything else you can think to store in a database. With a small bit of effort, it is possible to use this data and push it into your existing database schemas and create multiple copies of valid databases, creating a digital shell game for an attacker who doesn't know which one is valid.

6. Correlation

While most of these tools are useful by themselves, tying them together to form a cohesive picture of the network allows the defender to quickly be alerted to an intruder

on the network and develop a plan of operation. The solution that does this correlation can be as simple or as complex as the defender wants it to be.

There are two routes that one can follow to establish a log server: Closed and Open source. Open source's main advantages are the fact that it's cheaper and it can run on a multitude of hardware platforms. However, there are also drawbacks: numerous projects do not have a commercial support option and there is usually a high learning curve to the software. Closed source is the ying to open source's yang: While the tools have commercial support and often have a lower learning curve, however the hardware they support is limited and the solutions are usually not cheap.

Whether the closed or open source route is followed, properly establishing a centralized log server is not a simple process. Hardware specifications, user roles, the position of the server within the network topology among other things are questions that need to be considered. It is important to remember that while a log server can provide you with a great view into your network, if an attacker gains access to it, they can wipe any fingerprints they left behind and tailor future attacks to avoid all your detection methods.

Once the log server has been established, it is simply a matter of configuring the various tools log to this server. This is commonly done with either syslog or by an agent that is pushed to the endpoint generating the logs. After the logs is are being pushed to the server, it is a simple task of configuring the system to recognize and alert on certain events. Closed source solutions have internal tools for this; some open source tools that can accomplish this are the Open Source SECurity tool (http://ossec.net) and the Sagan log analysis engine (http://sagan.quadrantsec.com). Both tools come configured to handle a multitude of events "out of the box" but are both able to heavily customizable to handle almost any kind of text log event.

7. Working Active Detection into the IR Process

Once set up on your network, some active detection techniques require no adjustment into the incident response process, they are simply just additional alerts to classify and analyze. However, with other techniques, the threshold for identifying an incident becomes a bit blurred. While a computer scanning multiple darknets for CIFS

ports may be a clear indication that something hostile is on the network, other indicators are not so simple to judge. Investigation of these events within that hazy grey area requires the same analysis as most other security events, however, the data will often be less complete than what an analyst may expect. This often requires reliance on external data and information from third parties. Determining if the event indicates an attack or is a false positive often boils down to a judgment call.

For example, if fake identities are seeded properly, one of the most common security events they see are phishing, malware, or both. While a receipt of a malware or phish to a regular user often escalates to an incident, a fake identity receiving the same does not automatically escalate to one. Simple questions need to be asked:

- Was this the only copy of the e-mail sent into your environment?
- Did other fake identities receive it was well? Was there a common thread (Social Media, Organization Unit, etc) between recipients?
- Was this tailored for this specific identity or does it have a more generic theme to it?
- Did the sending host attempt any other probes, e-mail or otherwise, within your environment? What about the host serving the phish or malware?
- Are other entities reporting the same e-mail or malware? Are they there any common threats between them?

These questions can expand a single data point: "Our fake identity received a malicious e-mail." to a much clearer picture: "Our fake identity received a malicious e-mail that was also received by three other unrelated people within our environment. It was also discussed on Twitter and a spam blog indicating that other entities received it as well. While we need to implement containment procedures, this was not an attack directed specifically against us." – The level of response between the two being significant.

8. Conclusion

In today's world of computer network defense, the advantage almost always seems to go to the attacker. Partially this is because the defenders have not adapted to the attackers techniques nor have they moved beyond the usual suite of tools that have been

around for over five years. However, as this paper demonstrates, completing some simple tasks and employing some different strategies can greatly alter how a network "responds" to an attacker's movement and actions.

By changing the network to make it have some oddities and quirks, along with setting some traps that are designed to be hostile to attackers it becomes easier to detect attackers' movements and activities. Once the attackers are detected, the incident process can begin and they can be denied access to their objectives. In general, these techniques makes the network stop playing nice when the attacker interacts with it. These techniques allow the network and the defender to start playing dirty.

9. References

Cheswick, B. (1990). The Design of a Secure Internet Gateway. *Proceedings Summer USENIX Conference*, (pp. 233-237). Retrieved from: http://www.cheswick.com/ches/papers/gateway.ps

Cheswick, B. (1992). An Evening with Berferd in Which a Cracker is Lured, Endured, and Studied. *Proceedings of the Winter 1992 USENIX Conference* (pp. 163-174). San Francisco: USENIX. Retrieved from: http://web.cheswick.com/ches/papers/berferd.pdf

Hutchins, E. M., Cloppert, M. J., & Amin, R. M. (2011). Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. *The 6th International Conference on Information-Warfare & Security* (pp. 113-125). Washington, DC: Academic Conference International. Retrieved from: http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/LM-White-Paper-Intel-Driven-Defense.pdf

Nichols, S. (2003). *Big Brother is Watching: An Update on Web Bugs.* SANS Institute. Retrieved from: https://www.sans.org/reading_room/whitepapers/threats/big-brother-watching-update-web-bugs_445

Author Name, email@addressbbj@mayhemiclabs.com

Perlroth, N. (2013, January 30). Hackers in China Attacked The Times for Last 4 Months. *New York Times*.

Provos, N. (2004). A virtual honeypot framework. *Proceedings of the 13th USENIX security symposium.* Retrieved from: http://www.citi.umich.edu/u/provos/papers/honeyd.pdf

Strand, J. (2011, January 6). *Episode 225 - Tech Segment: SpiderTrap*. Retrieved from Pauldotcom Security Weekly: http://pauldotcom.com/wiki/index.php/Episode225#Tech_Segment:_Spider Trap

Verizon RISK Team. (2013). *2013 Data Breach Investigations Report.* Retrieved from Verizon Enterprise Solutions Worldwide: http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2013_en_xg.pdf

Wustrow, E., Karir, M., Bailey, M., Jahanian, F., & Huston, G. (2010). Internet background radiation revisited. *IMC '10 Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (pp. 62-74). New York: ACM. . Retrieved from: http://www.eecs.umich.edu/techreports/cse/2010/CSE-TR-564-10.pdf

Author Name, email@addressbbj@mayhemiclabs.com