Global Information Assurance Certification Paper

Don't Always Judge a Packet by Its Cover

GIAC (GCIA) Gold Certification

Author: Gabriel Sanchez, gmgsanchez@gmail.com
Advisor: Stephen Northcutt
Accepted: 01/21/16

Abstract

Distinguishing between friend and foe as millions of packets traverse a network at any given moment can be a very tedious and trying objective. Packets can contain viruses, malware, and botnets which necessitates the need to detect them fast. However, chasing every packet often becomes unmanageable and can often lead to many dead ends. Traditional approaches to this problem rely on heuristics or signatures with a known bad which tend to be ineffective to the advanced attacker. Instead, this paper will go beyond the known bad and describe a general approach of honing in on packets of interest utilizing the behavior and profiling of a network. The use of behavior analysis and profiling for packets that ordinarily traverse a network can shine light on the shadows that the enemy lurks in that bypass traditional detection. This behavior analysis and profiling is especially imperative since knowing the characteristics of your packets can certainly reveal their true intentions.

1. Introduction

Attackers are constantly inventing new ways to attack and compromise networks in order to gain supremacy over cyber space. These tactics include ways to evade intrusion detection systems, firewalls, web-filters, spam filters, and just about any device put in place to protect a network. Today, the innovative race between adversaries and security vendors is only accelerating, and organizations risk becoming more vulnerable to attack if they sit back and watch (Cisco 2015 Midyear Security Report, 2015). As millions of packets traverse a network, it can become a daunting task to differentiate friend from foe, which is why security defenders often use signature-based detection and heuristics to counteract these packets.

Signature detection involves searching network traffic for a series of bytes or packet sequences known to be malicious (Foster, 2005). However, attackers and defenders both have access to the criteria used by signature detection to determine whether a packet is malicious. This shared knowledge puts the defenders at a severe disadvantage, since their game plan is essentially laid out for the attackers whom can slightly modify packets and test them against signatures to ensure a bypass is possible. This approach is feasible, since the actual attacker can rewrite the exploit code from any public code and use simple mutations of the exploits to fly under the IDS/IPS radar (Coty, 2012). Before a malware campaign is launched, cybercriminals will usually pre-scan their malicious executables against all popular antivirus engines to successfully bypass the signature-based malware scanning used by these engines (Danchev, 2012). This pre-scan technique gives a high probability of success for the attacker to infect a system and gain a foothold into a network. Once an attacker has a foothold inside a given network, they can expand control to complete their ultimate goal such as stealing of personally identifiable information (PII), trade secrets, or credit cards. A signature-based detection approach will therefore only assist with a low level attack and the normal drive-by scans trying to infect a network. As opposed to signature-based scanning, which seeks to match signatures found in files with that of a database of known malware, heuristic scanning uses rules and algorithms to search for commands that may indicate malicious intent (Cade, 2015).

Gabriel Sanchez, gmgsanchez@gmail.com

Unlike signature-based detection, the attacker doesn't know the entire defensive game plan of the defender in heuristic scanning. Heuristics can offer a greater advantage over the signature-based approach in catching malicious activity from determined attackers, since the rules and algorithms used by heuristics differ by product, which creates an additional layer of protection to keep attackers out of a network. However, even with this advantage, many heuristics have an analysis of code ingrained inside its algorithm which is created by a specific product.

Because the engines are looking for specific pieces of code that indicate a malicious action, this can lead to two possible limitations:

- If the vendor has not built detection for a particular action, then the malware will evade detection.
- If the malicious action is obfuscated successfully (e.g. within an encrypted file), it will evade detection (Cade, 2015)

The limitations that heuristics may have does not necessarily indicate that it is a useless tool. Rather, people need to understand the importance behind the concept of heuristics which is one of the many necessary layers of defense that should be implemented in order to successfully protect a network against bad packets. However, people also need to understand the limitations of heuristics in order to accurately protect their networks from exploitation of compromise.

Using behavior analysis and profiling in networks need to go beyond signature or heuristic based detection and hone in on packets of interest. Advanced attackers go to great lengths to evade heuristics and signature-based detection through the use of devices that do not ordinarily reveal their true intentions, behaviors, or characteristics. In other words, simple malware is like a bad poker player with "tells" that give away its malicious intent right off the bat with executables that blatantly set out to extract and transfer personal data, keystroke loggers, and so forth (Andreassen, 2015). Advanced malware, however, has a better poker face (Andreassen, 2015). It hides its intent by going on loop, stalling, or otherwise cloaking itself while being analyzed by security tools at network perimeters (Andreassen, 2015). In order to deter advanced attacks, both behavior analysis

and the profiling of a network needs to be utilized as an additional defense against attackers, and not as the two primary, or only, lines of defense against bad packets.

2. Behavior Analysis and Profiling

Every attack on a network exhibits a different behavior, which creates the need to take into account multiple characteristics of a packet or device. Therefore, there isn't a one-size-fits-all approach to behavior analysis. However, it is crucial that people become intimately familiar with their network when investigating information that will provide insight into analyzing behavior or profiling a network with network flow, full packet capture, and establishing a baseline.

Two virtual machines will be utilized in order to analyze the above detection methods. The first virtual machine will utilize an IP address of 192.168.238.129 running on Windows 7 Professional Version 6.1.7600 while the second virtual machine will be the Ubuntu version 14.04.3. The setup of both virtual machines is shown below in Figure 1.

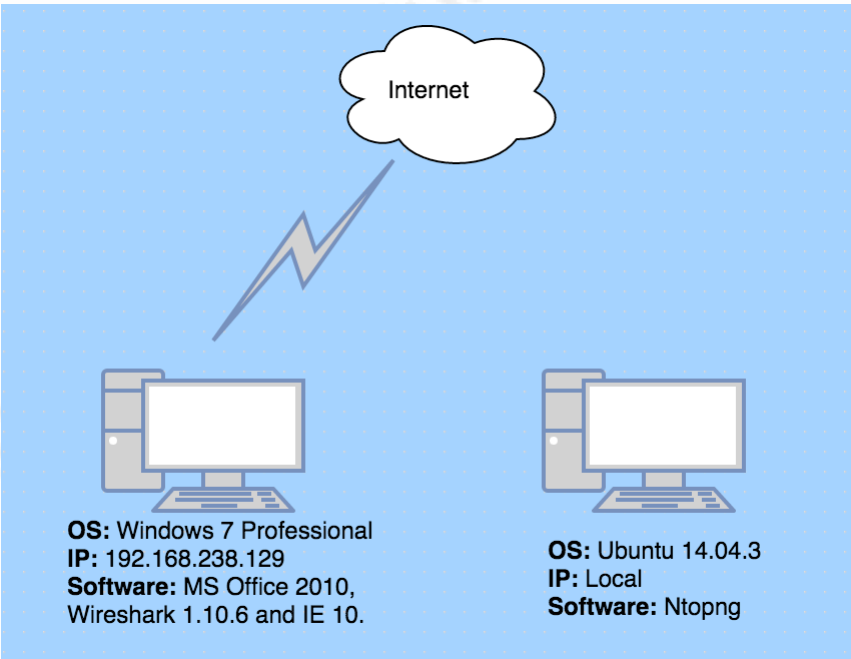

Figure 1 - Experiment Network

The Windows 7 machine will impersonate a user of a fictitious organization running Microsoft Office 2010. Wireshark will also be installed on the Windows 7 machine as a straight forward way of gathering full packet capture, although, in larger environments, other enterprise solutions may be used. The Ubuntu machine will be fitted with the Ntopng tool suite which will be utilized for analyzing network flow after the packet capture is imported into Ntopng. The scope of testing will be limited to simple actions performed on the Windows 7 machine in order to analyze network flow, full packet capture, and baseline analysis. Actions performed on the Windows 7 machine will include:

1. Open Wireshark software to capture packets.
2. Open Internet Explorer 10 browser to default page google.com.
3. Open word document named "test.doc" which contains malicious content.
4. Ping cnn.com.
5. Close both Internet Explorer 10 and test.doc after 5 minutes.
6. Save Wireshark capture results to fullpacket.pcapng for full packet analysis.

The packet capture file fullpacket.pcapng will then be imported into Ntopng in order to utilize network flow analysis. The following actions will then be performed on the Ubuntu machine:

1. Copy fullpacket.pcapng to Ubuntu machine.
2. Use the command "ntopng –i fullpacket.pcapng" to import packet capture.

Full packet analysis will be accomplished on the Windows 7 machine using the Wireshark software.

2.1. Network Flow

Network flow records are high-level descriptions of Internet connections that offer information about the endpoints and volume of data involved, but does not offer access to the actual data transferred (Meiss, 2009). Network Flow is similar to a phone

conversation between two people where a third person gathers information about the call, but cannot hear what is being said, which will be useful when the baseline section is discussed. Many network protocols exist that collect the same type of flow data utilized on a network, which typically consists of the source IP, destination IP, source port, destination port, protocol, type of service (TOS), and input interface. Below is Figure 2, which demonstrates a visual representation of the network flow data collected by an enabled device.

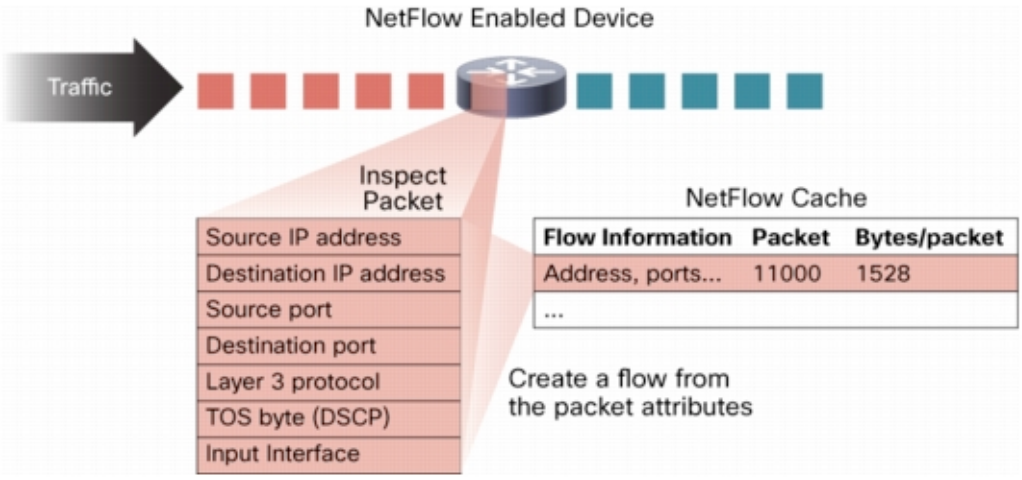

Figure 2 (Cisco, 2012) - Netflow

Both open source and commercial tools can assist with the gathering of network flow data. Regardless of the tool, the same characteristics of packets typically will be gathered. Ntopng, an open-source traffic monitoring application designed for high-speed networks and key features include real-time analytics and the ability to characterize application protocols and user traffic behavior (Deri, 2015). An advantage of using network flow records in Ntopng is that they are kept in binary format, which allows for quick access and a longer retention of data. The Ntopng tool will be used for viewing network flow records in the Ubuntu virtual machine and to provide an array of network utilization graphs, live maps, and current and past network traffic, including protocol, source, destination, and hosts involved in specific transactions (Arianto, 2013). The ability to view network flow records is what makes Ntopng advantageous since a summary of data can save a tremendous amount of time when trying to oversee millions of packets that traverse a network. The Ntopng tool was used to analyze the fullpacket.pcapng file with

information about the actions performed on the machine. In Figure 3, the Ntopng tool analyzed the application, protocol, source IP, source port, destination IP, destination port, duration, client/server breakdown, and total bytes of fullpacket.pcapng.

Application	L4 Proto	Client	Server	Duration	Breakdown	Actual Thpt	Total Bytes
SSL.Google	TCP	192.168.238.129:49699	216.58.219.228 :https	4 min, 2 sec	Server	0 bps	62.73 KB
SSL.Google	TCP	192.168.238.129:49700	216.58.192.67 :https	4 min, 1 sec	Server	0 bps	37.45 KB
SSL.Google	TCP	192.168.238.129:49701	216.58.192.78 :https	4 min, 1 sec	Client Server	0 bps	6.17 KB
HTTP	TCP	192.168.238.129:49702	91.219.192.253 :http	17 sec	Client Server	0 bps	3.58 KB
NetBIOS	UDP	192.168.238.129:netbios-ns	192.168.238.2:netbios-ns	4 min, 13 sec	Client	0 bps	3.17 KB
MDNS	UDP	192.168.238.1:mdns	224.0.0.251:mdns	5 min, 2 sec	Client	0 bps	1.61 KB
ICMP	ICMP	192.168.238.129	157.166.226.25	4 sec	Client Server	0 bps	592 B
NetBIOS	UDP	192.168.238.129:netbios-ns	192.168.238.255:netbios-ns	3 sec	Client	0 bps	276 B
Unknown	TCP	23.13.171.27 :http	192.168.238.129:49698	1 sec	Client Server	0 bps	228 B
Unknown	TCP	104.25.10.6 :https	192.168.238.129:49696	3 sec	Client Server	0 bps	199 B
DNS.Google	UDP	192.168.238.129:63429	192.168.238.2:domain	1 sec	Client Server	0 bps	198 B
DNS.Google	UDP	192.168.238.129:62324	192.168.238.2:domain	1 sec	Client Server	0 bps	166 B
DNS	UDP	192.168.238.129:61683	192.168.238.2:domain	1 sec	Client Server	0 bps	166 B
DNS.Google	UDP	192.168.238.129:56249	192.168.238.2:domain	1 sec	Client Server	0 bps	164 B
DNS	UDP	192.168.238.129:62433	192.168.238.2:domain	2 sec	Client Server	0 bps	158 B
Unknown	UDP	fe80::8cb8:8502:dcbf...:dhcpv6-client	ff02::1:2:dhcpv6-server	1 sec	Client	0 bps	157 B

Figure 3 - Ntopng Net Flow Summary

The traffic above shows communication with multiple IP addresses and protocols including TCP, UDP, and ICMP with their respective source and destination ports. Traffic to, or from, the Windows 7 machine with the IP 192.168.238.129 shows ports that include 443 (HTTPS), 80 (HTTP), and 53 (DNS). The source and destination addresses provide tremendous value in network flow threat detection when compared to an updated list of known areas of threat or concern (Patterson, 2012). For the Windows 7 machine, Figure 4 shows further analysis of the fullpacket.pcapng file with the Ntopng tool filtered to show "Top Peers" that the Windows 7 machine, with IP address 192.168.238.129, communicated with.

Gabriel Sanchez, gmgsanchez@gmail.com

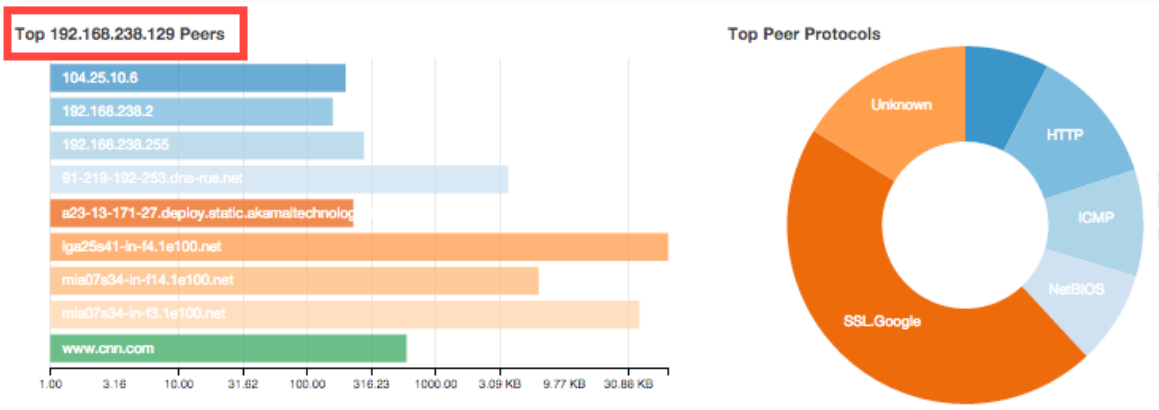

Figure 4 - Top Peers for 192.168.238.129

The summary above, in Figure 4, also shows communication with 9 different IP addresses, which were then utilized with the Ntopng tool, GeoMap, to map a list of IP addresses on a map. Codes are assigned to IP addresses to designate the country to whom the IPs are registered (Weaver, 2009). On the Ubuntu machine, a map file was downloaded from maxmind.com to obtain the country codes while the Ntopng tool was used to provide the list of IP addresses. In Figure 6 the Hosts GeoMap shows the IP addresses from Ntopng and their location on a map.

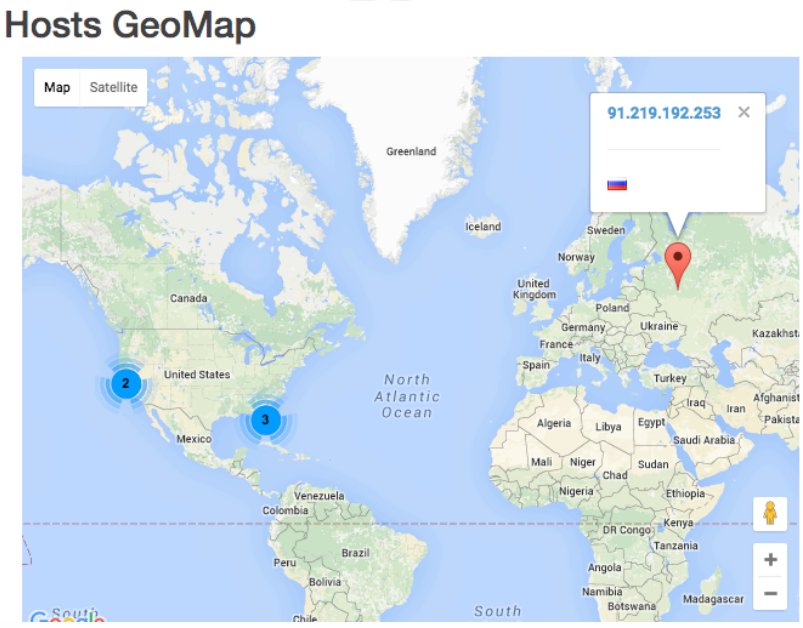

Figure 6 - Hosts GeoMap of IP addresses

Gabriel Sanchez, gmgsanchez@gmail.com

One IP address that stands out from the rest is the 91.219.192.253, which is located in Russia. Even though the network flow does not provide payload information, it can be indicative of areas that require more in depth analysis as is the case with countries of interest such as Russia. After identifying an area of interest, a more in depth analysis can be completed with a full packet capture.

2.2. Full Packet Capture

A full packet capture is a method that is utilized for obtaining a copy of the entire packet, which includes the payload and header as it traverses a network. Referencing the phone conversation analogy from the Network Flow section, a full packet capture allows the third person, known as a sniffer, to gather information while hearing what is being said during the entire conversation. Different methods for capturing full packets include sniffer, span ports, port-mirroring, and network taps, however, regardless of the method utilized, the primary objective for security is to copy, or intercept, all packets to be analyzed at a later date. One main disadvantage of a full packet capture is the massive volume of data that needs to be analyzed, which makes the analysis of packets difficult and costly in regards to the storage of information. However, conducting an intelligent packet capture for select areas of a network where an issue is suspected can dramatically reduce storage requirements and costs while making it easier to extract actionable data for fending off sophisticated cyber attacks (Talbot, 2015). Full Packet Capture was utilized for the Windows 7 machine by analyzing areas of interest such as the network flow results, which yielded the IP address 91.219.192.253 mapping to Russia as is shown in Figure 7.

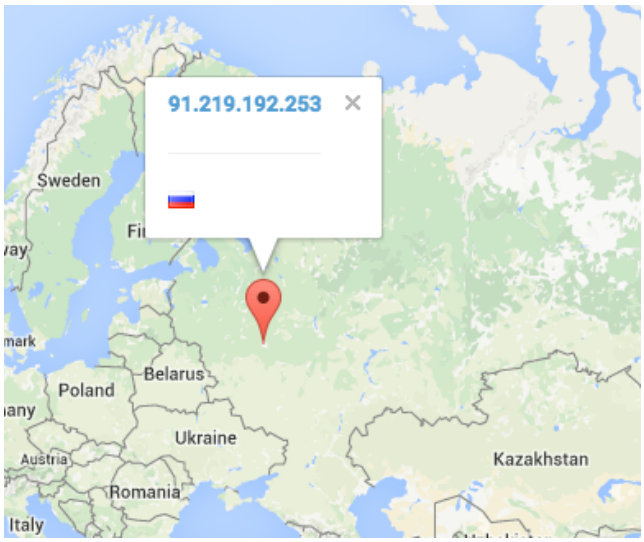

Figure 7 - GeoMap of IP address

For an in depth analysis, of the network flow results, a full packet capture with Wireshark was performed as a part of the Windows 7 machine test to allow the security analyst to view header and payload information associated with the IP address 91.219.192.253. Wireshark is a graphical user interface that allows for packet capture in real time and includes filters, color-coding and other features that allow for a deep analysis of network traffic and to inspect individual packets (Hoffman, 2014). A diagram of a full packet capture with Wireshark on the Windows 7 machine is shown in Figure 8.

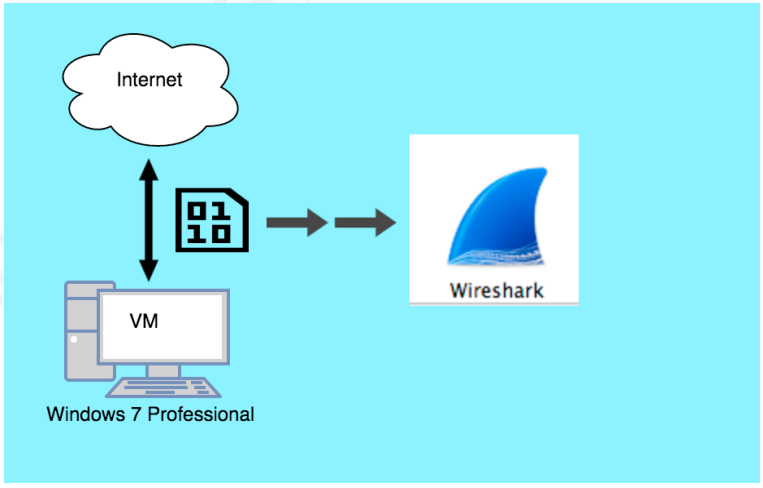

Figure 8 – Windows 7 Full Packet Capture

The results of running Wireshark for five minutes on the Windows 7 machine, as part of the experiment, produced 246 packets saved to a file named fullpacket.pcapng. Wireshark was used to open the file fullpacket.pcapng as shown in Figure 9.

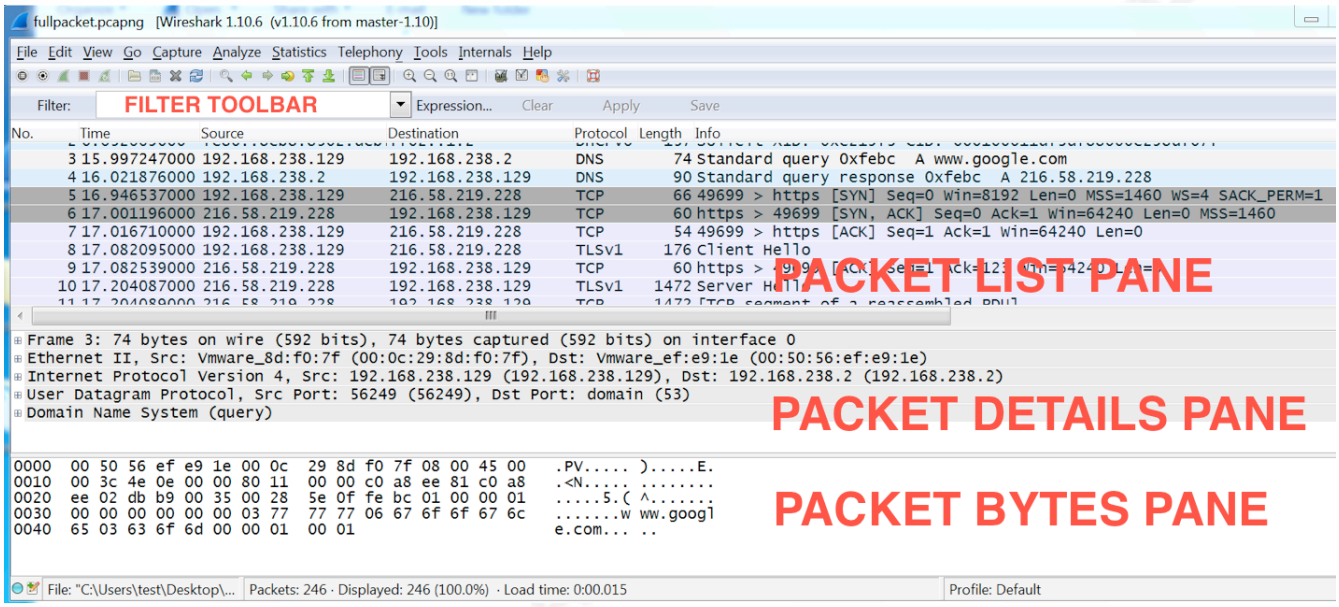

Figure 9 - Wireshark Interface

The four main components labeled in Figure 9 consist of the filter toolbar, packet list pane, packet details pane, and packet bytes pane. The primary function of each component is listed below.

- The filter toolbar provides a way to manipulate the display filter.
- The packet list pane displays a summary of each packet captured.
- The packet details pane displays the packet selected in the packet list pane in more detail.
- The packet bytes pane displays the data from the packet selected in the packet list pane, and highlights the field selected in the packet details pane (Ulf Lamping, Richard Sharepe, & Ed Warnicke, 2014).

The Network Flow section identified areas of interest such as the destination of certain packets, which, in this experiment, was Russia with an IP address 91.219.192.253 over

port 80. In Figure 10, the filter toolbar was used with Wireshark to narrow down the areas of interest to traffic going to IP address 91.219.192.253 over port 80.

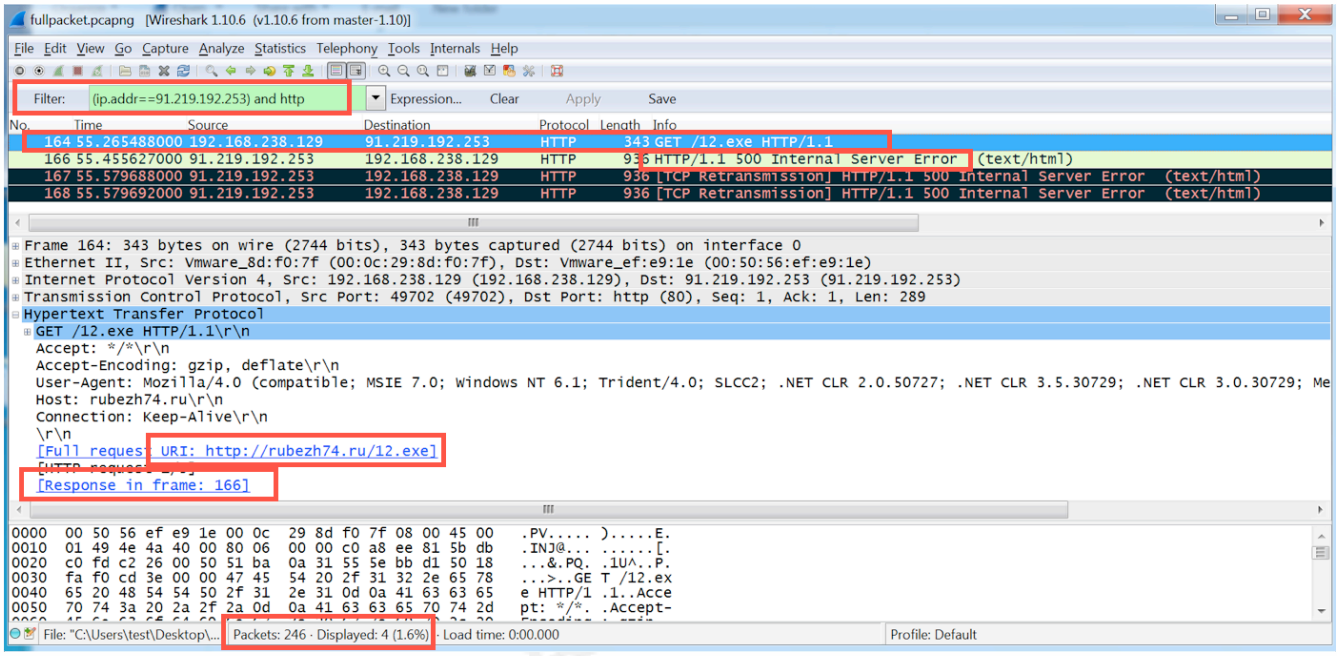

Figure 10 - Wireshark Filter

Results from Wireshark, displayed four packets out of 246 with the conditions specified in the filter toolbar. Packet number 164 shows the Windows 7 machine, with the source IP address 192.168.238.129, initiating a GET request to destination IP address 91.219.192.253 for a file named 12.exe. Viewing the packet details pane of packet 164 shows that the URI of rubezh74.ru contains the 12.exe file with the corresponding response in frame 166. In the packet list pane frame 166 shows a response of HTTP/1.1 500 Internal Server Error from 91.219.192.253. In addition, frames 167 and 168 shows a retransmission attempt from 91.219.192.253 with the same 500 Internal Server Error. The 500 Internal Server Error is a very general HTTP status code, which indicates that something has gone wrong on the web site's server, but the server could not be more specific on what that exact problem is (Fisher, 2015). In this experiment, the 500 Internal Server Error is indicative of an unsuccessful download of the 12.exe file to the Windows 7 machine. Although the file was not downloaded it still makes up part of the overall

behavior and profiling of the Windows 7 machine. Areas of interest from both network flow and full packet capture has provided the information listed below.

1. The Windows 7 machine with IP address 192.168.238.129 was communicating over port 80 to IP address 91.219.192.253.
2. The IP address 91.219.192.253 originates from Russia.
3. The 91.219.192.253 IP address translates to the http://rubezh74.ru website.
4. The Windows 7 machine with IP address 192.168.238.129, attempted to download a file named 12.exe from the http://rubezh74.ru website.
5. A 500 Internal Server Error was sent from the site http://rubezh74.ru to the Windows 7 machine with the IP address 192.168.238.129.

Armed with the previous information the established baseline for the Windows 7 machine needs to be taken into account to determine if the characteristics found in both the network flow and full packet capture are normal. If the characteristics found are not typical for the Windows 7 machine, then the system will continue to be analyzed.

2.3. Baseline

In order to avoid and detect threats to a system, it is imperative that security analysts utilize baselines, which are starting points that are used to draw critical observations or data for comparison (Merriam-Webster, 2015). These observations are accomplished by gathering information in regards to what is deemed normal behavior from characteristics such as IP addresses, acceptable software, typical bandwidth, web-browsing, acceptable file formats, and ports. Many of these characteristics can be gathered with network flow and full packet capture profiling with a process, which can be broken down into the following steps.

1. Gather available network information.
2. Select an initial data set.
   - Duration
   - Timing

- Direction
- Sampling
- Network size

3. Identify the active address space.
4. Catalog common services.
5. Catalog remaining active assets.
6. Maintain the profile.
7. Report on findings.

(Whisnant, 2013)

Creating a profile for a system will enable security analysts to view behaviors to search for any suspicious or unusual activity to pinpoint areas of interest. In Figure 11, is an example of a baseline for the total amount of network traffic expected over a 24-hour period for a specific system.

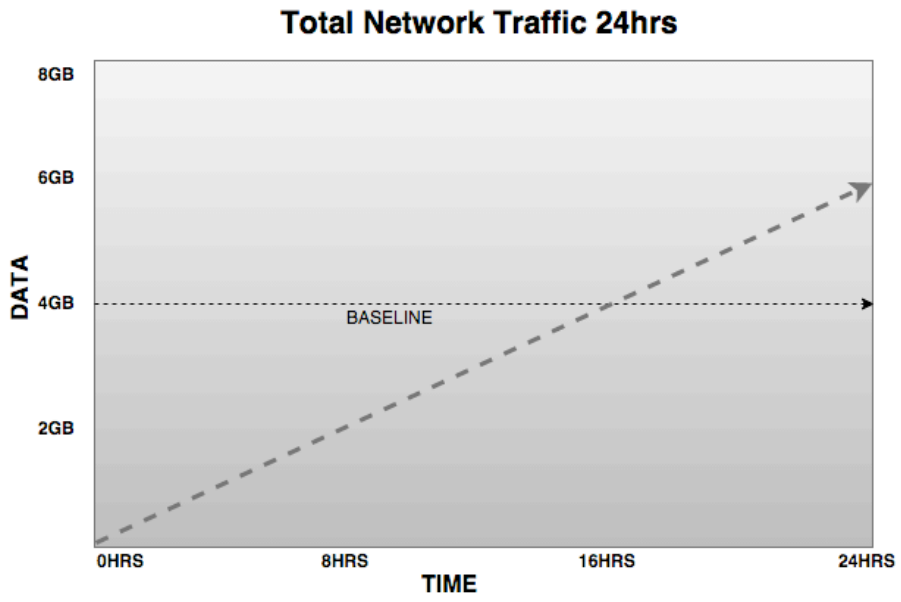

Figure 11 - Network Traffic

Figure 11 shows that the baseline of total traffic for the network was 4GB for a specific system over a period of 24-hrs, yet, the diagram above shows that the total traffic reached 6GB over 24-hrs, which is unusual for this type of system. However, total traffic is just

one of the many characteristics that can be taken into account by a security analyst to detect any compromises in a system. There are many approaches to profiling the characteristics of IP addresses, which may include blacklisting, whitelisting, location, and content. The right approach is largely dictated by the sensitivity of the system and compliance with company policies. Security analysts can identify assets that violate policy and engage in suspicious activity, while business administrators can use the profiles to help guide long-term decisions regarding network security (Whisnant, 2013). The experiment for the Windows 7 machine consisted of a profile that communicated with IP addresses that included internal IPs, acceptable domains, and whitelisted IP addresses. Below in Figure 12 is a basic visual representation of the Windows 7 machine and the communication it had with IP addresses.

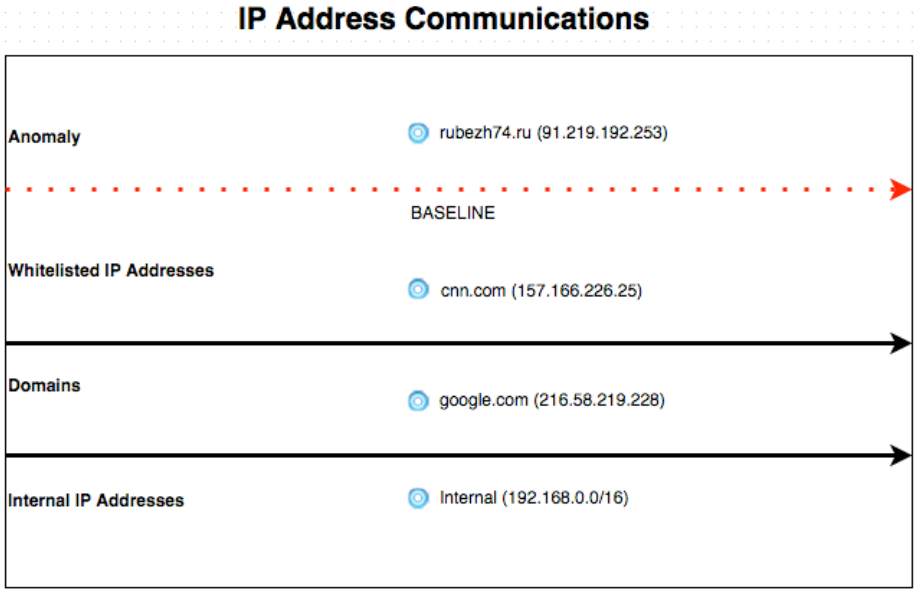

Figure 12 - Baseline of IP Communications

By utilizing the data obtained from the network flow, full packet capture, and behavior security analysts were able to notice that the IP address 91.219.192.253 surpassed the expected baseline for the Windows 7 machine, which indicated a need for further investigation. Communications with the IP address 91.219.192.253 was considered an anomaly because the behavior exceeded the baseline by deviating from the standard (Oxford Dictionary, 2007). In order to acquire an in depth analysis of the IP address

many resources, such as forensic tools, must be utilized to search for any compromises in the system. One resource used with the Windows 7 machine was the malware analysis tool, Payload Security, which can be used directly from their website. Upon searching for the file 12.exe, which was the attempted download from the Windows 7 machine, several results appeared that correlated with the network flow and full packet capture findings. Figure 13 shows one of these findings with the 12.exe file associated with the rubezh74.ru site.

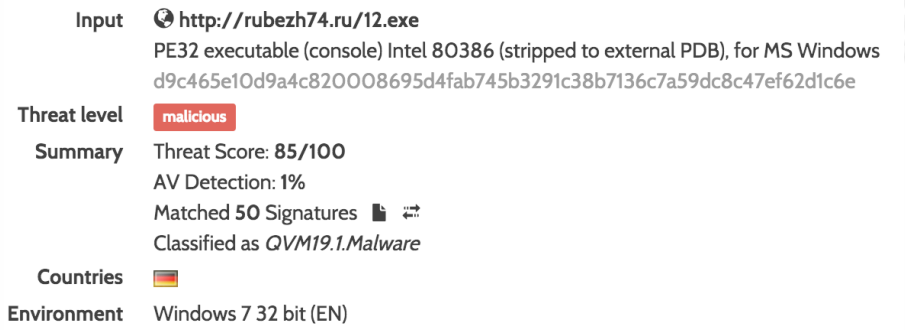

Figure 13 - Payload Security Report

The summary shows a threat score of 85/100, AV detection of 1%, and was classified as malware. This event solidifies that the Windows 7 machine was part of an incident that could have opened the system, and its information, to malicious malware and intent (NICCS, 2015). Analyzing the full report of the result revealed that the source of the malware was a phishing campaign with embedded macros inside a doc file. In addition, other IP addresses, processes, temp files, and extracted files were flagged as other areas of interest. This full report goes to show that security analyzers must begin to discover and investigate anomalies, which could be traced to the root cause of malware. In an uncontrolled environment, it is not uncommon for a system to have many documents, which necessitates the need to utilize the forensic tool, OfficeMalScanner, to scan for malicious code in Office documents as was the case with the Windows 7 machine. In Figure 14, OfficeMalScanner was used on test.doc to analyze any macros or malicious code.

```
C:\OfficeMalScanner>OfficeMalScanner.exe test.doc info

+-----------------------------------------+
|           OfficeMalScanner v0.61        |
|  Frank Boldewin / www.reconstructer.org  |
+-----------------------------------------+

[*] INFO mode selected
[*] Opening file test.doc
[*] Filesize is 264704 (0x40a00) Bytes
[*] Ms Office OLE2 Compound Format document detected

----------------------------------
[Scanning for VB-code in TEST.DOC]
----------------------------------
ThisDocument
--------------------------------------------------------------------------------
               VB-MACRO CODE WAS FOUND INSIDE THIS FILE!
              The decompressed Macro code was stored here:

------> C:\OfficeMalScanner\TEST.DOC-Macros
--------------------------------------------------------------------------------
```

Figure 14 - OfficeMalScanner Results

As Figure 14 shows, the forensic tool, OfficeMalScanner, found macros in the test.doc file and decompressed the code to the TEST.DOC-Macros directory. The files were then uploaded to the Payload Security site for further analysis to confirm whether or not the test.doc was the root cause for the incident and findings in both the network flow and full packet capture. Figure 15 shows a portion of the results found when the test.doc was uploaded to the Payload Security site.

Endpoint	Method/Response	URL/Code	
91.219.192.253:80 (rubezh74.ru)	GET	/12.exe	malicious Threat Score: 100/100 AV Detection: 40% W97M.Dropper

Figure 15 - Test.doc Payload Security Analysis

The results showed a threat score of 100/100, AV detection of 40%, W97M.Dropper malware, and a direct correlation of all the areas of interest found in the network flow and full packet capture. The test.doc file was identified as the root cause of the malicious behavior and incident. Even though the full packet capture indicated that there was an unsuccessful download of the 12.exe file, the test.doc was still able to extract VB scripts and drop other malicious artifacts on the Windows 7 machine. Therefore, security

analysts will need to expand the area of impact to any other systems correlating to the findings found from analyzing the Windows 7 machine due to environments having multiple endpoints.

3. Conclusion

Attackers are relentlessly coming up with new methods to compromise networks and are overwhelming security experts who are not up-to-date with the latest methods of security protection such as firewalls, web-filters, and spam filters. The task of protecting networks has become an especially daunting task due to the need to differentiate friend from foe among millions of packets that traverse a network. This is the reason that it has become exceedingly imperative that security analysts become aware, and utilize, detection methods such as behavior analysis and the profiling of a network. In order to bring home the importance of these methods an experiment utilizing a Windows 7 machine was generated.

During the experiment common actions performed were web browsing, opening a word document, and the utilization of a ping command. Behavior analysis, with a known good baseline, of the Windows 7 machine was used and then compared to the network flow characteristics. This pointed to an area of interest where the Ntopng tool showed that communication was occurring with a suspicious IP address located in Russia. For a more in depth analysis a full packet capture, with Wireshark, was used to analyze payload and header information, which showed an attempted download of a file named 12.exe from an IP located in Russia. Finally, the forensic tool OfficeMalScanner traced the root cause of the incident to a macro hidden inside a document named test.doc.

There are also many other methods that include signature-based detection, heuristics, network flow, full packet capture, and forensic tools, which help to determine areas of interest for further investigation. However, the strengths of each tool needs to be known and adapted to the environment one tries to protect in order to maximize the potential for defending against attacks. As millions of packets traverse a network the key to

successfully identifying malicious activity is through the use of each tool’s strengths, with known good baselines, to quickly and easily identify anomalies in a given network. The use of these tools then allow security analysts to respond in a manner that protects the core business from intrusions or attacks. As the experiment exhibited, the ability to drastically reduce the amount of packets that must be analyzed, by the use of anomalies, greatly increases the chances of security analysts finding malicious packets and, therefore, being able to protect the integrity and contents of a given system.

Gabriel Sanchez, gmgsanchez@gmail.com

4. References

Andreassen/Lastline, J. (2015, March 9). Why Most High Stakes Cyber-Attacks Are Detectable - Redpoint Ventures. Retrieved from http://www.redpoint.com/blog/why-most-high-stakes-cyber-attacks-are-detectable/

Arianto, P. (2013, December 2). 7 free tools every network needs | Network World. Retrieved from http://www.networkworld.com/article/2825879/network-management/7-free-open-source-network-monitoring-tools.html

Cade, C. (2015, July 13). Understanding Heuristic-based Scanning vs. Sandboxing | OPSWAT Blog. Retrieved from https://www.opswat.com/blog/understanding-heuristic-based-scanning-vs-sandboxing

Cisco. (2012, May). *Introduction to Cisco IOS NetFlow - A Technical Overview - Cisco*. Retrieved from http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html

Cisco 2015 Midyear Security Report. (2015, July 1). Cisco 2015 Midyear Security Report.

Coty, S. (2012, September 27). IDS/IPS Signature Bypassing (Snort). Retrieved from https://www.alertlogic.com/blog/ids/ips-signature-bypassing-(snort)/

Danchev, D. (2012, February 23). Why relying on antivirus signatures is simply not enough anymore - Webroot Threat Blog. Retrieved from

Gabriel

http://www.webroot.com/blog/2012/02/23/why-relying-on-antivirus-signatures-is-simply-not-enough-anymore/

Deri, L. (2015). Realtime High-Speed Network Traffic Monitoring Using ntopng | USENIX. Retrieved from https://www.usenix.org/conference/atc15/technical-session/presentation/deri

Fisher, T. (2015, October 26). 500 Internal Server Error (What It Is and How To Fix It). Retrieved from http://pcsupport.about.com/od/findbyerrormessage/a/500servererror.htm

Foster, J. C. (2005, May). IDS: Signature versus anomaly detection. Retrieved from http://searchsecurity.techtarget.com/tip/IDS-Signature-versus-anomaly-detection

Hoffman, C. (2014, October 14). How to Use Wireshark to Capture, Filter and Inspect Packets. Retrieved from http://www.howtogeek.com/104278/how-to-use-wireshark-to-capture-filter-and-inspect-packets/

Meiss, M. (2009). Network Flow Analysis | Center for Complex Networks and Systems Research. Retrieved from http://cnets.indiana.edu/groups/nan/flowanalysis/

Merriam-Webster. (2015). Baseline | Definition of Baseline by Merriam-Webster. Retrieved from http://www.merriam-webster.com/dictionary/baseline

NICCS. (2015). Cyber Glossary | National Initiative for Cybersecurity Careers and Studies (NICCS) - Trademarked. Retrieved from https://niccs.us-cert.gov/glossary#response

Gabriel gmail.com

Oxford Dictionary. (2007). anomaly: definition of anomaly in Oxford dictionary (American English) (US). Retrieved from http://www.oxforddictionaries.com/us/definition/american_english/anomaly

Patterson, M. (2012, January 19). Threat detection with NetFlow: IP reputation - Mike Patterson. Retrieved from http://www.bradreese.com/blog/threat-protection-with-netflow.htm

Talbot, C. (2015, June 1). Lancope unveils PacketWatch intelligent packet capture solution - FierceEnterpriseCommunications. Retrieved from http://www.fierceenterprisecommunications.com/story/lancope-unveils-packetwatch-intelligent-packet-capture-solution/2015-06-01

Ulf Lamping, Richard Sharepe, & Ed Warnicke. (2014). 3.3. The Main window. Retrieved from https://www.wireshark.org/docs/wsug_html_chunked/ChUseMainWindowSection.html

Whisnant, A. (2013, February 18). Network Profiling Using Flow. Retrieved from https://insights.sei.cmu.edu/sei_blog/2013/02/network-profiling-using-flow.html

Gabriel Sanchez, gmgsanchez@gmail.com