Chip Gardiner

WMT Cohort Capstone Project

**Using Snort as an Intrusion Detection System for a SOHO Network**

**Investigation**

In a good defense-in-depth, there are many layers to a security policy, but one component that my project will be looking at is the Intrusion Detection System or IDS. Most organizations using cloud-based solutions and having multiple connections from third-party services and other devices require a well-configured IDS that will alert security professionals to impending attacks based on specific rules set up in the IDS. For large-scale organizations, an IDS is often commonplace because of their budgets but an IDS like Snort can be beneficial to smaller businesses.

An IDS is only one part of a layered security system but many recent cyber attacks would have benefited from having a well-configured IDS. Snort is an open-source IDS/IPS that is signature and anomaly-based (Snort).

Dollar Tree was affected by a data breach of a third-party data service on August 7 where they were unable to determine the data stolen in the breach but did have confirmation of a breach on that day (Kass). This third-party service had weakened security that could detect a breach but did not have prevention tools in place that would have detected the connection as anomalous and alerted the security team to begin reacting to it. Cloud data provider Snowflake had large amounts of data stolen from multiple customers from its databases after a breach was performed with stolen credentials and had valid logins and improper security controls where they didn’t have whitelisted connections to prevent access from abnormal locations (Mandiant). An anomaly-based IDS could have helped prevent or deter a more widespread data breach.

As ransomware attacks continue to be prevalent an IDS won’t be able to stop them completely but is a tool that detects when there is activity happening from ransomware being activated on a device. Allan Liska, a threat analyst at Recorded Future says, “We’re definitely not winning the fight against ransomware right now” (Pearson). Volexity detected a zero-day vulnerability in the Ivanti VPN that became a widespread exploit in their Connect Secure VPN. The exploit was disclosed in January and allowed them to deliver the mitigation days later (Alspach). The detection of this allowed for quick mitigation and later patching of this vulnerability which affected over 1,700 users. This was detected through network monitoring that an IDS can provide to organizations to find active exploits and perform forensics to work towards a patch or mitigation for the exploit.

**Planning**

The plan is to create a SOHO-style network setup that uses Snort as a host-based IDS. A tool like Snort can be useful in smaller offices to provide intrusion detection without a lot of overhead and this project will demonstrate some of the uses of Snort.

A virtual network will be set up with 3 devices on this network. The 3 devices are a Windows 10 device running Snort 2.9, Metasploitable, and Windows Server 2022. This virtual network will be purposely vulnerable with no network or host-based firewall, or anti-virus protections. This will make for a large attack plane but also allow the IDS to be able to detect anything fully. The Windows devices will still have any major updates applied to them at installation but will not have extra security measures installed.

Virtual Box will be used for all virtual machine installations. An evaluation copy of Windows 10 will be used for that device as well as an evaluation version of Windows Server 2022. Metasploitable’s virtual hard disk will be used for the Metasploitable virtual machine. Metasploitable is a purposely vulnerable Linux system that can be used for testing many of the features of Metasploit.

Snort 2.9 on a Windows 10 Virtual Machine. This version of Snort is only an IDS and does not function as a way to deter or prevent anything except some unwanted packets that may be sent to any system on the network. Setting up the rules will have some ease with several community rules around, but also having to modify many of the file paths built into the configuration being set for Linux and not for Windows. Some will require changing to Windows file methods while others require Linux-style file pathing. Finding the appropriate places for a back slash versus a forward slash. Some local rules will be made for initial testing and the community rules will be used for identifying malware. Blacklist rules will be made to test remote sign-ons from IPs not associated with the network.

Windows server 2022 to test legit remote sign-ons and non-white-listed IPs for sign-on. And act as another IP to perform packet scans against.

2 devices will act as attacker devices for testing. A Kali Linux device performing packet scans and using Metasploit to perform exploits on the Metasploitable device to test for Snort detection of malware and packet scans. A Windows 10 device performing a remote sign-on to the Windows server to test the white and black list capabilities of Snort.

Testing will be done to see the alerting capabilities of Snort to the console and to log creation for packet scans, malware, and log-in with allowed credentials from an unrecognized IP.

**Implementation**

The network consists of 3 devices. A Windows 10 device, A Windows 2022 Server, and a Linux device running Metasploitable. All of these are in a virtual machine and virtual network. Configuring the virtual network required some changes to VirtualBox network settings from NAT which puts all devices on the same outward IP that is translated into a new IP by the host system to an internal network so that Snort can listen to all activity on this network and not only for the interface of the Windows VM it is installed on (VirtualBox).

The Windows 10 VM has Snort installed on it. The version of Snort being used is 2.9 which has a Windows binary installer (Snort). When installing Snort you must configure it in the snort.conf file to how your system is set up.

Many sections of the snort configuration file are already configured to work within the measures of the security policy such as having the ports assigned that are used by the TCP/IP protocols. Variables need to be assigned: RULE\_PATH which covers where the rules files are in the directory and PREPROC\_RULE\_PATH which are the rules that cover the preprocessor in Snort. The preprocessor handles the signature and anomaly detection and helps with the speed of the IDS by handling the processor disassembly (Beale). The paths for the white and black list variables (WHITE\_LIST\_PATH, BLACK\_LIST\_PATH) and the log directory. The Snort configuration file is set originally for the Linux installation of Snort so file paths have to be changed for the preprocessor libraries and engine. After that the rules files that will be accessed need to have their comment tag removed so they can be accessed.

Snort has a local rules file that was used for initial testing that sent an alert on every TCP, UDP, and FTP packet to confirm the system was working.

Snort offers community rules that were downloaded and configured. All of these are set to signatures based on ports or IPs from Common Vulnerabilities and Exposures listed on the Mitre website. The rules come commented out so the rules that are needed to match to specific CVE need to be located. This can take time but all the rules have the names associated with these CVEs listed in the files. This system is lacking in some security by these other rules being commented out, but they were left out to help cut down on any possible noise that could come from the alert system or the logs to allow me to better determine that the rules were working as expected. With all of these rules enabled it could become very noisy on the logs and console alerts during testing and I want to be able to show only the factors I am testing for simplicity. The community rules that cover malware will be used extensively. These rules cover known vulnerabilities and Snort will detect them by their signature, the port, and the protocol that they use to connect to their target device. Setting up the blacklist rules requires putting some rules into the local file that pull from the reputation preprocessor to have the alerts come up when the connection happens. It will also require discovering the IP address of the Windows 10 VM that will be performing the remote desktop connection.

The Windows Server is set up on a virtual machine with default settings and the desktop environment ISO. The only changed setting is enabling remote log-in to test logging in from the outside Windows machine for the white and black list rules.

Metasploitable is downloaded from the Metasploitable site and installed from the virtual hard disk. No settings are changed on this. This purposely vulnerable system is good for testing but allows for multiple vulnerabilities to enter the network so this must only be left on for testing and for not much else.

2 devices sit outside of the network in a normal configuration: a Kali Linux VM and a Windows 10 VM.

Kali Linux is configured with default settings and the virtual machine is configured at normal settings and not onto the network with the other 3 devices. This is an already existing system I have previously used. Kali has a wide range of tools that are useful for researching networks and devices for possible exploits and vulnerabilities (Kali). Metasploit will be used on this device to perform reconnaissance on the Windows server and the Metasploitable device and then perform different malware attacks against the Metasploitable device as well as other vulnerabilities through typical web-based servers such as Apache that are on the Metasploitable device.

The Windows 10 system is configured with normal settings and the remote desktop client. It will maintain the same IP and be outside of the network but the rules will be modified to white-list the IP and perform a remote login and then black-list the IP and perform a remote login.

An IDS is only one part of a layered security approach and can only help identify anomalous or already identified behaviors in a network. Without other layers to this defense-in-depth strategy, a system will be lacking in further hardening against attacks which this system demonstrates. The IDS will be able to alert and log activity that matches the rules defined, but it has no means to stop these attacks. When these tests are performed these systems will be completely vulnerable and unable to stop any attack. An IDS will alert to incoming attacks and allow the security team to perform countermeasures but none of those will be demonstrated here, but will show the importance of an IDS to alert and log anomalous activity for security professionals to research, harden or perform countermeasures against.

Installing Snort on Windows also requires npcap, a replacement version for Wincap, and an update to some Visual Studio drivers that help with the running of Snort in the command line. In my implementation of Snort, I ran into some problems with the setup of the snort.conf file. I had the pathing wrong on the white and black list files where I left a forward slash instead of a back slash. After finding the slash snort was able to run properly.

**Testing**

I am testing for three main criteria: detection of network reconnaissance through a port scan, malware types, and using the black list to alert to a flagged IP that is using a non-malicious connection and credentials.

The first test is the scan from the Kali device. Snort is configured and running on the Windows VM. It is configured to alert to console for this test. Nmap is performed on the IP for the Windows VM running Snort. The alert shows a TCP port scan detected from the IP of the Kali VM outside of the network to the Windows VM.

The second test involves the Metasploitable VM. I look through the exploits available on Metasploit and research what vulnerabilities are purposefully enabled on Metasploitable. Each of these different vulnerabilities is among the community rules downloaded from Snort. I can search on the Snort site by the name of vulnerability and it shows which rule set contains alerts to the vulnerability. Vulnerabilities tested are a vulnerability in Apache to get a shell login on port 80, the VSFTPD backdoor on port 21, MySQL account check on port 3306, and PostgreSQL on port 5432.

After finding the corresponding rules on the Snort site I looked through the rules files according to the names provided, rules such as Malware-CNC for the VSFPTD vulnerability, MySQL for the MySQL vulnerability, PostgreSQL for that vulnerability, Apache for the Apache server vulnerabilities

Testing for the blacklist rules proved difficult. My initial implementation of the blacklist appeared to be working to deny packets from the IP for the Remote Desktop Connection, but that was only an error of the Windows Server connecting. Researching more showed that my blacklist rules were incorrect. I was putting an entire alert method into the blacklist rules file when I only needed an IP address. I needed to access the preprocessor and have an alert that required a different rule going to my local file. Once that was configured properly I tested again but I was not receiving an alert to the IP being blacklisted. I blacklisted based on the IP address being shown when I did the ipconfig command on the Windows 10 RDP VM. I created a local rule to alert for any traffic coming across port 3389 which is the port that is used by Remote Desktop Connection. This alert shows the IP that I know is for the Windows Server and one other which must be the IP for the Windows 10 VM that is being used by my own computer and ISP to provide the internet connection to connect back to the Windows server.

With that IP now blacklisted the test is running as expected and sending the alert to the console for the connection happening from a now blacklisted IP address.

**Documentation**

Virtual Machines were all created on VirtualBox (Oracle VirtualBox).

Windows 10 Enterprise, version 22H2 was used for both Windows 10 Virtual Machines (“Windows 10 Enterprise”).

Kali Linux version 2024.2 was used for the Kali Virtual Machine (Kali).

Metasploitable 2 was used for the Metasplolitable VM (Moore).

Windows Server 2022 with Desktop Experience was used for the Windows Server VM (Windows Server).

The Metasploitable 2 VM is from a Virtual Hard Disk downloaded from their site. The Windows, Kali, and Windows Server VMs were all installed by ISOs.

Snort 2.9.20 was the version of Snort being used. The executable installer was downloaded from the Snort website (Snort).

Npcap is needed to allow for packet capture (Npcap).

Visual C++ runtime library was needed to run Snort (Microsoft Learning).

Snort was installed from the executable. The snort.conf file was modified to work with Windows. The direct file paths are needed in specific areas listed in Implementation.

The community rules are downloadable from the Snort website (Snort).

The rules files used in this test were malware-cnc, bad-traffic, malware-backdoor, server-apache, and server-mysql.

The following rules were put into the local.rules file to alert for black and white listed IPs (Dietrich).

alert ( msg: "REPUTATION\_EVENT\_BLACKLIST"; sid: 1; gid: 136; rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )

alert ( msg: "REPUTATION\_EVENT\_WHITELIST"; sid: 2; gid: 136; rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )

This rule was placed in the local rules to help me find the IP address of the Windows 10 VM performing the Remote Desktop Connection

alert udp any any -> $HOME\_NET 3389 ( msg: "Remote Desktop Connection alert"; sid: 4; gid: 136; classtype:bad-unknown; )

Works Cited

Alspach, Kyle. “Ivanti VPN Vulnerabilities Seeing 'Mass Exploitation': Researchers.” *CRN*, 16 January 2024, https://www.crn.com/news/security/2024/ivanti-vpn-vulnerabilities-seeing-mass-exploitation-researchers. Accessed 1 August 2024.

Beale, Jay. *Snort 2.0 Intrusion Detection*. Elsevier Science, 2003. Accessed 10 August 2024.

Dietrich, Noah. “The Reputation Preprocessor in Snort – Blacklists and Whitelists – Sublime Robots.” *Sublime Robots*, 10 December 2015, http://sublimerobots.com/2015/12/the-snort-reputation-preprocessor/. Accessed 13 August 2024.

Kali. “Get Kali.” *Kali Linux*, 2024, https://www.kali.org/get-kali/#kali-platforms. Accessed 10 August 2024.

Kass, Howard. “Dollar Tree Supply Chain Attack Could Affect Millions of People.” *MSSP Alert*, 1 December 2023, https://www.msspalert.com/news/dollar-tree-supply-chain-attack-could-affect-millions-of-people. Accessed 14 July 2024.

Mandiant. “UNC5537 Targets Snowflake Customer Instances for Data Theft and Extortion.” *Google Cloud*, 10 June 2024, https://cloud.google.com/blog/topics/threat-intelligence/unc5537-snowflake-data-theft-extortion. Accessed 14 July 2024.

Microsoft Learning. “Latest supported Visual C++ Redistributable downloads | Microsoft Learn.” *Learn Microsoft*, 28 May 2024, https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170. Accessed 13 August 2024.

Moore, HD. “Metasploitable 2 | Metasploit Documentation.” *Docs @ Rapid7*, 2024, https://docs.rapid7.com/metasploit/metasploitable-2/. Accessed 13 August 2024.

Npcap. “Npcap.” *Npcap: Windows Packet Capture Library & Driver*, 19 January 2024, https://npcap.com/. Accessed 13 August 2024.

Oracle VirtualBox. “Oracle VM VirtualBox.” *Oracle VM VirtualBox*, 2024, https://www.virtualbox.org/. Accessed 13 August 2024.

Pearson, Jordan. “Ransomware Is 'More Brutal' Than Ever in 2024.” *WIRED*, 10 June 2024, https://www.wired.com/story/state-of-ransomware-2024/. Accessed 1 August 2024.

Snort. *Snort - Network Intrusion Detection & Prevention System*, 2024, https://snort.org/. Accessed 1 August 2024.

VirtualBox. “Chapter 6. Virtual Networking.” *VirtualBox*, 2024, https://www.virtualbox.org/manual/ch06.html. Accessed 10 August 2024.

“Windows 10 Enterprise.” *Microsoft*, 2024, https://www.microsoft.com/en-us/evalcenter/evaluate-windows-10-enterprise. Accessed 10 August 2024.

Windows Server. “Windows Server 2022.” *Microsoft*, 2024, https://www.microsoft.com/en-us/evalcenter/evaluate-windows-server-2022. Accessed 13 August 2024.