INTRUSION DETECTION: FROM DETECTION TO RESPONSE USING SNORT

A THESIS SUBMITTED TO
THE FACULTY OF ARCHITECTURE AND ENGINEERING
OF
EPOKA UNIVERSITY

BY

AMELA RAHIMI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

MARCH, 2023

Approval sheet of the Thesis

This is to certify that we have read this thesis entitled **"Intrusion Detection: from detection to response using Snort"** and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Dr. Arban Uka
Head of Department
Date: 10/03/2023

Examining Committee Members:

Prof. Dr.Betim Cico	(Computer Engineering)	________________
Dr. Arban Uka	(Computer Engineering)	________________
Dr. Florenc Skuka	(Computer Engineering)	________________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Surname: AMELA RAHIMI

Signature:

ABSTRACT

INTRUSION DETECTION: FROM DETECTION TO RESPONSE USING SNORT

Rahimi Amela

M.Sc., Department of Computer Engineering

Supervisor: Dr. Arban Uka

Modern corporate networks are targeted by attacks from the Internet. The consequences of cyberattacks can be devastating, including loss of business information, theft of money, the cost of repairing affected systems, and possible damage to an organization's reputation. With the right devices, security can detect suspicious traffic.

With proper network security techniques in place, its security analysts get early warning of emerging problems. This research sought to explore and build a basic, robust system that could be used to distinguish between suspicious practices in network traffic.

In my tests I tried:

Discuss and analyze network traffic and gadget suspicious conspiracies. Analyze current techniques used to detect suspicious activity in network traffic. Development of systems to detect suspicious conspiracies in network traffic. Approve the proposed system. After review, the study plan was approved. The experiment was run in Virtual Box with Windows 7 and Snort and Metasploit's web GUI.

Snort had the ability to intercept and report large packets sent to this machine.

Network traffic was the subject of this study. Researchers sent packets over the network. Network traffic was analyzed using network security tools analyzed by researchers and selected for accessibility and similarity to each other for the desired deployment.

By providing precise critiques of what network administrators at various organizations can identify as questionable practices within their networks, the research has resulted in significant improvements.

Keyword: Snort, Nmap, Metasploit, cyberattacks, malware, network traffic.

ABSTRAKT

DETEKTIMI I NDËRHYRJES: NGA DETEKTIMI NË PËRGJIGJE DUKE PËRDORUR SNORT

Rahimi, Amela

Master Shkencor, Departamenti i Inxhinierisë Kompjuterike

Udhëheqësi: Dr.Arban Uka

Modelet moderne të korporatave janë të shënjestruara nga sulmet nga interneti. Pasojat e sulmeve kibernetike mund të jenë shkatërruese, duke përfshirë humbjen e informacionit të biznesit, vjedhjen e parave, koston e riparimit të sistemeve të prekura dhe dëmtimin e mundshëm të reputacionit të një organizate. Me pajisjet e duhura, siguria mund të zbulojë trafikun e dyshimtë.

Me teknikat e duhura të sigurisë së rrjetit në vend, analistët e saj të sigurisë marrin paralajmërim të hershëm për problemet në rritje.

Në teste u arrit:

Diskutimi dhe analiza e trafikut të rrjetit dhe komplotet e dyshimta. Analizimi i teknikave të tanishme që përdoren për të zbuluar aktivitetin e dyshimtë në trafikun e rrjetit. Zhvillimi i sistemeve për të zbuluar komplote të dyshimta në trafikun e rrjetit. Aprovimi i sistemit të propozuar. Pas shqyrtimit, plani i studimit u miratua. Eksperimenti u mbajt në Virtual Box me Windows 7 dhe web GUI të Snort dhe Metasploit.

Trafiku i rrjetit ishte subjekt i këtij studimi. Trafiku i rrjetit u analizua duke përdorur mjetet e sigurisë së rrjetit të analizuara nga kërkuesit dhe të zgjedhura për aksesueshmëri dhe ngjashmëri me njëri-tjetrin për vendosjen e dëshiruar.

Duke ofruar kritika të sakta se çfarë administratorët e rrjetit në organizata të ndryshme mund të identifikojnë si praktika të dyshimta brenda rrjeteve të tyre, kërkimi ka rezultuar në përmirësime të rëndësishme.

Fjalë kyçe: Snort, Nmap, Metasploit, sulmet kibernetike, malware, trafiku i rrjetit.

… to the ones that believed in me!

Thank you!

ACKNOWLEDGEMENTS

This thesis is dedicated to my beloved parents who showed continuous support during my master studies. Great gratitude to my thesis advisor Dr.Arban Uka for all the dedication, motivation and kindness that followed during these semesters.

To my friends and classmates for the wonderful time passed in the auditorium.

The last thank goes to God, for the guidance, strength, and protection.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

OVERVIEW

1. Introduction

The focal point of extraordinary exploration throughout the previous few decades has been intrusion detection. Ordinarily, intrusion detection is used to break down data about the framework activity or its information to identify any unauthorized activity. Network-based Intrusion Detection Systems (NIDs) and Host-based Intrusion Detection Systems (HIDs) are two types of intrusion detection systems.

A system layer assault is a progression of bundles that abuse a weakness in the system. NIDs look at the system traffic information to see if there is any activity between the two systems. A system-based intrusion detection framework is called Grunt. On the other side, HIDs screen, and break down the internals of a figuring framework like memory, record framework, the client instead of the outside interface to discover anomalies. There are loads of open-source HIDs accessible on the market and OSSEC is a case of an open-source have based intrusion detection framework.

Any kind of intrusion detection framework will usually cause a caution or log the activity. It's tempting to take equivocal or remedial activities to stop the assault and guarantee the wellbeing of the processing condition after an intrusion detection framework has distinguished a malignant activity. An intrusion reaction is a counter-measure.

When an intrusion detection framework is setting off a caution, the framework executive needs to experience everything about the alarm and convey a reasonable reaction. Framework overseers can't keep up with the pace of the framework nor can they respond to cautions inside a time limit. These manual reactions aren't effective. The framework head is used in these manual reactions.

If there should be an occurrence of many conveyed systems to react to a ready all the more rapidly and precisely, computerized reaction systems can assume control over the undertaking. The intrusion reaction framework is regularly used. The unpredictability of creating and sending reaction in a robotized design makes it less considered than intrusion detection research.

A computerized intrusion reaction is a choice of a reaction. A decrease in the reaction from the hour of detection is the principle bit of leeway of computerized reaction. Predictable and exact reaction are given by a robotized reaction. The way in which framework overseers neglect to consider the expense related to the reaction conveyed is wiped out by these mechanized reactions. Although the mechanized reactions have a ton of focal points, it's delayed to be received because its usage is mind-boggling in any event, for experienced experts as it despite everything needs normalized execution measurements and expanded robotization.

1.2. Motivation

The best way to assemble a framework model is to worry about the majority of the current mechanized intrusion reaction systems. The framework model considers intrusion cost, reaction cost, and reaction adequacy as assets. There are many chart-based framework models that have been proposed. The reliance chart, diagram models, and progressive tree model are included in a portion of the models.

The framework chairman needs to give out qualities to explicit assets of the framework in order to make the framework model. During the time spent assessing measurements like intrusion harm cost, reaction cost, and reaction adequacy, an incentive is used for the assets. Any framework can be partitioned into segments that can be used for either help or an asset. It gets obstinate for the framework executive to dole out qualities for a particular asset. It may not be a precise gauge for that framework, as it is not just a difficult undertaking for the framework chairmen.

The venture directors are more willing to relegating the qualities to the administrations of the framework. This can be obtained from the association's expenses when they stop working. The harm cost of an intrusion is known by most of the current robotized reaction systems and they just give approaches to assess the reaction cost.

To take care of the apparent multitude of issues we've proposed and actualized a nonexclusive reaction model that will be a potential answer. We give a method to assess the framework and a procedure to choose a reaction. The harm cost, reaction cost, and reaction adequacy of an intrusion are assessed with the help of a reliance chart.

1.3. Contribution

We plan and create a framework for intrusion detection. The reliance diagram model speaks to the interdependency between elements of a framework. The framework is made up of various substances. A substance is an overall name used to speak to an asset. The four layers of the reliance chart are the application administrations, part benefits, framework/uphold administrations, and assets of the framework. Virtual assets and physical assets are included.

Every one of these substances is a hub associated with edges. The conditions between substances are related to the security objectives. The reliance weight is depicted by the data on each edge. The general estimation of the framework is measured by the business examiner. The absolute estimation of the framework is shared among high-level application administrations. A worth spread strategy is used to spread the estimation of the high-level application administrations to all different substances of the framework. Data given by the reliance chart is used to do the worth spread.

After the engendering, every substance in the framework gets an incentive of its own. We use this method to gauge the number of elements in a framework. When an assault happens, we use the reliance chart to locate the damaged cost. We have capacities to assess responses like response cost and response viability.

After giving a technique to choose the best response for an intrusion, we convey it. When the response is assessed to accomplish more advantages than harm, we convey it. We use Linux shell contents to send responses, for example, a firewall rule, restart a cycle, or suspend a cycle.

1.4. Road map

The thesis is organized as follows.

• Chapter 1.

The problem that we will discuss is the introduction to the tools, the motivation to write the paper

. • Chapter 2.

I speak about open-sourced tools that can be used in a wide range of attacks.

• Chapter 3.

Snort is the most important tool used in this thesis and all the benefits that we get from using it.

• Chapter 4.

It talks about the attack that I've designed and how I used the tools to mitigate and get answers in chapter 5

. • Chapter 5.

I talk about the results that I received from the tests.

CHAPTER 2

OPEN SOURCE SOFTWARES AND RELATED WORK

The work done in the zone of the computerized response system is summarized in this chapter. There are some open-source intrusion detection systems and response apparatuses that are accessible to alleviate system and network attacks.

2.1. Intrusion detection systems (IDS)

Unauthorized use of a PC system is known as intrusion detection. The demonstration of intrusion is a way to compromise the PC system by breaking the security or making it go into a shaky state. An instrument is required to monitor and ready system administrators. IDSs are systems with methods and techniques to recognize an unauthorized activity based on rules and marks.

System administrators can use these intrusion detection systems to monitor their systems and give them cautions. Administrators can find unauthorized use of their systems when utilizing these systems.

Along with the advancement of PC systems, intrusion detection systems have a decade-long history. The beginning of IDS history was followed by Khan Pathan in the chapter.

2.1.1. The systems' advancement as a subject of academic exploration

The situation in system architecture and post-detection actions are some of the ways in which intrusion detection systems can be sorted. According to Khan Pathan, there are mark and anomaly-based detection methods, have based, network-based, or crossbreed systems utilizing sensor location and intrusion detection systems, and intrusion prevention systems based on post-detection actions.

There are different methods for detecting anomalies in the way the system assesses traffic. Mark based systems recognize intrusions by putting away the marks of attacks and the conduct of known intrusion methods and contrasting these marks with actions, commands, and network traffic that these known intrusion methods use.

When a match is discovered, the occasion is reported. A case of mark detection is monitor network traffic to system service port sending information packets that are attempting to misuse a known bug.

The system monitors the actions, commands, and network traffic and knows about satisfactory conduct. The occasion is reported when the conduct is not the same as a benchmark. A case of anomalies would be a server trying to take an outbound connection to the internet when it is not allowed by the company.

Host-based IDS (HIDS) are introduced on each host that requires intrusion detection and they report all occasions that happen on the host they're introduced on, for instance, document changes or dubious commands. Network-based IDSs monitor network traffic and report all occasions concerning the network traffic they monitor, for instance, dubious connections or information packets containing realized attack designs. A combination of both host and network-based IDS methods is used to find intrusions.

There are two distinct classes of IDS. The evolution of an intrusion detection system is called an intrusion prevention system. According to NIST-800-94, "software that has all the abilities of an intrusion detection system and can likewise endeavor to stop possible episodes" is what IPS is. By using the abilities of IDS to find possible intrusions, theIPS systems make a dynamic move to prevent the intrusion. After detecting a case of malicious traffic to a network service port, theIPS will block the network traffic from the source to the network service port to prevent any intrusion. The IPS should be in a position to perform preventive actions, for instance as a pass-through substance in-network or as HIDS introduced on the host which is the target of the intrusion. There is a distinction between IDS andIPS in network security.

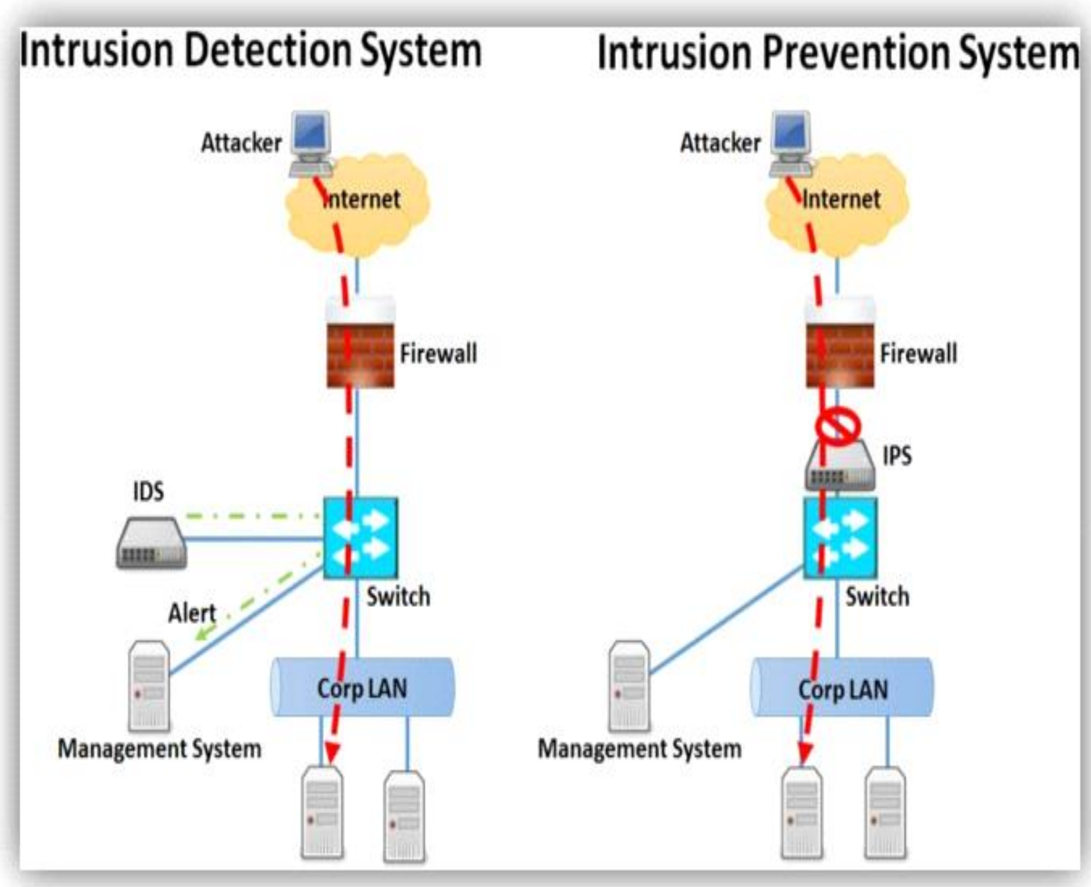


Figure 1 Difference between IPS and IDS in network security

CHAPTER 3

DETECTION AND SNORT

3.1. A multi-source intrusion detection module

There are many individual intrusion detection sources in the intrusion detection module. A variety of techniques are used for detecting attacks as a malicious activity. There are three different approaches to intrusion detection. They're Misuse-based, Anomaly-based, and Specification-based. The misuse-based strategy depends on pre-indicated assault marks and execution techniques that coordinate these marks. Any deviation from the ordinary examples will be distinguished as malicious because of the Anomaly-based methodology. The Specification-based procedure works along these lines as the Anomaly-based technique. It distinguishes deviations from genuine conduct. It requires client direction to build a model of substantial program conduct. We used intrusion detection sources for the execution. Snort is an open-source network intrusion detection system which is a sort of intrusion detection system that recognizes malicious activity by monitoring the network traffic.

This is an intrusion detection system. This sort of IDS attempts to distinguish the intrusion by coordinating the assault example to the rundown of all conceivable realized assault designs. An alarm is produced when there's an example of a signature. We have a custom composed Perl code that reads the log documents and sends a custom intrusion message to the engine. When the mark of the assault is obscure yet the influence in our system is known, this type of intrusion detection source is helpful.

A resource is an assistance in a system. In our usage, we have built up a custom intrusion detection source that will continue searching for a particular resource in our system. There is a chance that a potential assault may have occurred when the particular resource shows abnormal conduct. A particular intrusion caution is conveyed to the response engine by the intrusion detection source. The system administrator can make many custom intrusion detection sources. It is possible to make an IDS that can be used to monitor explicit parts of the system.

3.2. Snort

There are systems in place to detect and prevent network intrusions. This program analyzes network traffic to detect intrusion attempts. The rules used by the program are constantly updated. The program is free, but the policy is limited to paid subscriptions. Customers can download it for free, only one month after the release of the program. The next section describes Snort in detail.

Snort IDS and IPS systems have become well-known elements for securing networks. It is based on 5 modules.

Intercepting information as it travels over a network is called packet sniffing. This is done with the help of libraries. Sniffers can read network information from records and work with gaps.

This module manages decoding intercepted packets, detecting anomalies and deviations from RFCs, analyzing TCP banners, and other comparison tasks. The stack is centered around the decode stack.

The preprocessor is designed for more detailed analysis and normalization protocols at the 3rd, 4th and 7th levels while the decoder analyzes the traffic at his 2nd and his 3rd levels of the reference model. I'm here. The most common preprocessors are frag3 (works on split traffic), stream5 (reconstructs TCP streams), http_inspect_ (normalizes HTTP traffic), DCE/RPC2, and sfPortscan. The engine consists of two parts. The rule constructor bundles many important principles into her single set for further use in subsystems that inspect intercepted and processed traffic.

yield module. Snort can provide messages in a variety of formats after detecting an attack.

Snort earned its place as a well-known and accepted standard after many different intrusion detection system modifications. It has been downloaded over 4 million times from the www.snort.org website. There was a love for Snort. The main problem is the language used to describe network security policy violations. On the other hand, the language is very easy to understand. Detections for attacks and other security breaches can be created in minutes. Edge rules and opening considerations allow customers to create 8 complex network event handlers.

3.3. Principles of operation of IDS and IPS.

In this section, you will learn about working standards for intrusion detection and prevention systems. This section is based on publications from the National Institute of Standards and Technology.

An IDS is a software tool that can be used to detect unauthorized access to a PC system or network over the Internet. This provides a high level of security for your PC system.

The IDS architecture includes:

The sensory subsystem was developed for collection. The analysis subsystem is designed to detect attacks and suspicious activity. Storage and donation of the main causes and results of the analysis. The operator console is used for IDS configuration.

There are many types of IDS. The most common are:

A network-based her IDS is used to identify intruders by inspecting network traffic and mostly monitoring hosts.

NIDS can access network traffic by connecting it to a hub or switch configured to mirror its ports. A protocol-based IDS is a system that detects and analyzes communication protocols. IDS monitors the protocols for a web-server. The IDS must be assigned in the interface where it can monitor the packets before they are sent to the network. Application Protocol-based IDS is a system that monitors and analyses information using explicit protocols for specific applications. Host-based IDS is a system that recognizes intrusions and bases its analysis on system reasons for living, application logs, document modifications, have statements, and other sources. There are at least two different ways of IDS in the hybrid IDS.

Host operators' information is combined with network information to create an away from network security.

3.4. Passive and active intrusion detection systems.

Information about security issues is composed into the application log record and cautions are sent to the comfort as well as framework administrator by coordinated link. The Intrusion Prevention System (IPS) is a dynamic framework that can be used to block traffic from the evildoer. Administrator commands can run this activity.

A product or equipment arrangement of network and PC security that recognizes intrusion and noxious action will prevent it. The framework resembles an augmentation of IDS as it has the same reason as attacks. Their distinction is thatIPS must respond to attacks immediately.

There are many ways to group the systems. The following groupings are used by Scarfone and Mell. Network-basedIPS screens traffic in the network and squares dubious information stream. The Wireless Intrusion Prevention Systems screen activities in the wireless networks.

Mac addresses spoofing and recognizes wrong arranged wireless access points. Network behavior analysis breaks down the network traffic and searches for untypical streams. Host-based Intrusion Prevention (HIPS) distinguishes dubious activities on the PC.

3.5. Methods of reacting to attacks.

There are many different approaches to respond when an attack is perceived. The most common techniques are:.

After the attack start.. Techniques execute well after an attack is recognized. The situation of a fruitful security framework can be hurt.

Association blocking. If an attack uses a TCD association, it will be shut down by sending it to everyone or someone with the RST banner. The evildoer loses potential outcomes by using this network association. Normally, this strategy works with network sensors. It has two significant cons:. The convention that it does not uphold is the one that requires a pre-installed association. The strategy can be used after the person gets unauthorized access.

Client's record blocking. If different client accounts have been undermined because of the attack or their sources, then played out there blocking through the sensor host framework. To bolt it, the sensors need to be running for the benefit of the record that has administrator rights. The blocking can happen at a given time.

PC network host blocking. Host sensors can block an attack if one of the hosts is identified as an attack. Network interface blocking of this host or a switch associated with the network is1-6556. It can be done by the activation of the security administrator. The lock won't be dropped if the host is restart or disconnected. The attacks can be stopped by the host PC network.

Attack blocking with Firewall. New configurations are sent to the Firewall. It will be used to channel traffic from the criminal. This reconfiguration can act on its own. SAMP, CPMI). The connector module can be used for interaction with the IPS. It's going to get commands about the changes. Boundary modification will be altered by it.

The communication gadget has different configurations. The MIB database boundaries are examined and changed. An attack is expected to be stopped by the gadget's specialist. TFTP, Telnet, and other conventions can be used.

Dynamic concealment of attack source. This strategy can be used when others are pointless. The bundles of the villain are impeded by theIPS. If his location is known, the attack will not upset other lawful hubs.

Such a technique has been used. NetBuster is designed to prevent intrusion of the "Trojan pony".

It could be used as a fool-the-one-tried-to-NetBus-you. NetBuster finds the malware and identifies the PC that started it, then sends it back. Tambu Scrambler works with UDP ports.

The Elements demonstration allows the programmer's gadgets to be "disabled" via UDP flooders.

at the start of the attack. Techniques are used before the attack arrives.

3.6. Sensors

Intrusion prevention systems are based on sensors. They can be called network and host-based alternatives.

There is a network sensor in the communication hole. Two network adapters operating in mixed mode are equipped with network sensors. All packets passing through are in the backing store. Under attack, packets may be dropped. Packets are analyzed using a signature or behavior method.

Host sensors can be used to detect attacks from a distance. The culprit sends a series of packets.

Packets are analyzed by sensors at different interaction layers. Securing your connection helps prevent attacks. Local attacks compromise system security.

System calls are intercepted, parsed, and threatening fields are invoked.

3.8. IDS is compared with others.

The intersection of two innovations was intrusion prevention systems. The first can only pass through itself. The second is ready to analyze traffic, but is built the same way, so it doesn't pass the traffic itself. IPS systems have brought out the best in each innovation. The network does not monitor for intruders within the network, but monitors certain types of traffic to prevent intruders.

The emergence of today's systems has gone in four different directions. Progression of IDS is the main direction. It was important to build the system with the network configuration of IDS. The solution was simple and practical.

Firewall evolution is the second direction of IPS. There was no detailed analysis of the traffic flowing through itself. Adding functional deep penetration into the dataset and understanding the transmission protocol allowed us to turn these IPS systems into firewalls. A third source of progress was antivirus solutions. It was very close to fighting 'worms', 'Trojan horses' and other malware. This direction may have started HIPS rolling.

Creating a system "from scratch" was the fourth direction.

Three major issues were identified. There are many false positives. • Response automation. There are many administrative tasks.

These issues were understood as the system evolved. An “order by chance” opportunity correlation system was used to reduce the level of false positives. This has led to the rise of state-of-the-art IPS systems. NGIPS requires minimal functionality.

1. Work gradually without affecting your company's network activity.

2. To work as a single platform that combines all the benefits and new features of the previous generation IPS:

Application control and monitoring, use of information from external sources, and analysis of recordings.

3.9. Alternative to Snort (IDS/IPS provider).

Techtarget mapped his top 5 IDS/IPS devices for free enterprise networks. • Security Onion. •OSSEC. • Open WIPS-NG. • Scraping. • Bro IDS.

Safety Onion is the most flexible system and allows you to work collaboratively.

OSSEC is an intrusion detection system for host systems. If you want to check document integrity monitoring on the server, log various actions on the server, retrieve security events from the server (or others) and report on those events, and view various reports and so on using HIDS OSSEC. According to the agent server plan, OSSEC can work in mixed mode. Like most IDS vendors, they offer several devices for system security that can be used in the central function of the IDS.

We have free wireless IDS/IPS. It depends on many things. works with hardware equipment. The creator he created two systems, of which he received scanning, detection and prevention from one. Extensions for more customizability can be found in WIPS-NG. Not unknown, not created or huge compared to other systems. A wireless intrusion prevention system can be done with little financial planning effort.

Another open source system is called Suricata. The designer tried her IPS version of his Snort. The main difference between these two is that he is more efficient and on legacy hardware he can handle 10Gb traffic and fully supports the Snort rule format. Suricata competes with Snort in providing IDS/IPS.

Snort consists mostly of modules (Capturing, Collection, Decoding, Detecting, and Yield) that intercept traffic occurring in streams before decoding. This is the ideal path for decoding, but puts a lot of strain on the system. Compared to Snort, you can capture and specify how streams are separated between processors and then change it by configuring individual streams. It offers many opportunities to improve traffic handling.

The HTP library was written by Ivan Ristic, creator of ModSecurity. Request and response bodies are supported. This feature is used on certain networks to log undetected data traffic. The content of the stream is separated behind a veil and can be checked for validity based on the recording.

Initially it supported decoding, but we'll also dig into IPv4-in-IPv6, IPv6-in-IPv6, Teredo, and more. The special form of the motor allows quick connection of additional components. Traffic uses multiple interfaces for interception. Can automatically detect PCAP entries caught by another program in Unix socket mode. .

Snort originally did not have IPS. IPS works with its main version and with regular devices in the batch channel of the operating system. Linux had two types of IPS. Anything that can be handled with zero duplicates at the client level. The mode appeared in version 1.4.

The system should act as a gateway to achieve high speed.

The main difference with Suricata is that you can use both your own farming and Snorts crafting. For example, Sourcefire, Open Source Emerging Threats, and Business Emerging Threats Pro are good choices. Results can be analyzed with common backends (Barnyard2, Snortsnarf, Snorby, Aanval, BASE, FPCGUI, Sguil, Squirt NSM system). This has potential implications for syslog, documentation, and PCAP. The key and certificate will appear in the connection. A recent delivery showed an Eve log that produced alert occurrences in JSON format. Integration with external applications, such as system monitoring and log visualization, is greatly enhanced by accessibility. Suricata's settings and rules are written in the form of YML documents.

Layer 7 OSI improves the ability to identify malicious applications. Because the engine automatically identifies and parses logs, policies cannot be as carefully associated with port numbers as Snort. The module then sorts the traffic and finds the logs. Snort's followed Native's guidelines. The standard has components and descriptions such as Action, Pass, Reject, Reject, and Caution. With Native's current respite a bit, and some compromised in the documentation, it's time to focus more on his Snort ruleset.

Some applications are still open. Some IDSs can't see the big picture. The power bit concept was made worse by Suricata. We tracked the number of rule operations using specific session factors that can create counters and banners. It's hard to adapt to Brother has the following features:

• Adaptability. It uses a scripting language that allows you to set monitoring rules for each protected object. Additionally, Bro initially did not focus on detecting specific attacks and does not rely on signatures.

• Effectiveness. Brother is designed to work on networks with huge amounts of traffic and can be used for a wide variety of large-scale activities. In particular, it supports different architectures.

• Thorough analysis of traffic. Supports a multi-protocol his analyzer that does high-level semantic analysis even at the application level.

Brother is a framework for creating network IDS/IPS with a tiered, metered billing structure.

Packet capture mechanism. Indeed, this mechanism very often uses data for libpcap purposes, so Bro is platform and underlying network layer agnostic. The Occasion Mechanism (also known as the Event Engine, Core) converts incoming packet sequences into real opportunities. These opportunities reflect important information about network activity. This is intended to create an opportunity for each HTTP request to represent the HTTP protocol address, port, and URL version mentioned above. However, this mechanism does not determine shadowing opportunities. That is, whether it is ambiguous or malicious at this level.

A high-level content interpreter (policy script interpreter - reacts to all necessary opportunities and registers handlers according to specific content. Events are defined in his FIFO lines. The content also includes the means used to identify malicious traffic and the scripting language Bro.

Gradually, this platform will be available for building IDS and other traffic analysis. In principle, it could replace (and/or extend) Wireshark, highlighting and analyzing only the important traffic. The current release includes testing support for ElasticSearch, a full-text web crawler. (Brother, 2016)

Table 1 Advantages and disadvantages of snort

Advantages:	**Disadvantages:**
• Free for use • Available for Linux and Windows platforms • Flexible customization for many environments • Can be utilized in a decentralized model • Support auto-update rules • Opportunity to utilize MySQL database • Can be covered up in the network • Exist web-interface • Low system requires (in comparison with Suricata)	• Intrusion Detections happens behind the firewall. • Should be configured precisely for a specific environment to evade "bogus positives". • Not so natural for beginners. • Unidirectional Ethernet links ought to be utilized for sensors installation to evade security concerns.

3.7 Snort rules

Snort rules utilize lightweight and basic guideline language. Most Snort rules are composed on a single line. (Snort manual 2016)

It is isolated into two legitimate sections:

the standard header and the standard options. The standard header, the initial segment of Snort rule, contains:

• The standard's action

• Protocol

• Source and destination IP

• Addresses and NetMasks

• The source and destination ports information

The standard option section contains ready messages and information on which parts of the packet ought to be inspected to determine if the standard action ought to be taken.

Rule's actions have 5 default functions, yet on the off chance that you run Snort in inline mode there are additional 3 functions:

• alert - generate an alarm using the chose ready strategy, and then log the packet

• log - log the packet

• pass - disregard the packet

• activate - ready and then turn on another powerful standard

• dynamic - remain inert until activated by an activate rule, then go about as a log rule

• drop - square and log the packet

• Reject - block the packet, login, and then send a TCP reset, if the protocol is TCP or an ICMP port inaccessible message if the protocol is UDP. • Drop - Block the packet, but do not log it.

The next section of the rule is the protocol. Currently there are only four log types.

TCP, UDP, ICMP, and IP. An IP address defines a standard source or destination IP. Usually single addresses and ranges of addresses. Port numbers have similar functionality to IP addresses and are customizable and configurable.

Territory, Single and Negative. Directional operators are used to indicate the direction or direction of traffic to which a standard applies. This operator uses the following image:

'<', '>' and '<>'.

The dynamic/dynamic principle makes the system more adaptable, taking advantage of explicit options to allow clients to activate other policies when certain alerts appear.

Rule options are the second aspect of the standard, with her four categories:

• general

• Payl

• Non-payload

• After detection

Table 2 General options

msg	tells the analyzer what to write in alert message
reference	allows including references to external attack identifications
gid	identifies what part of Snort identifies the event
sid	identical number of rule
rev	identifies a revision of Snort rule
classtype	categorizes an attack type
priority	sets priority to the event
metadata	allows to add additional information

3.8 snuff rules

Snort rules use a lightweight, basic policy language. Most Snort rules consist of one line. (Snort Handbook 2016)

This is divided into two legitimate sections of his:

Default headers and default options. The first segment of a Snort rule, the standard header, contains:

CHAPTER 4

ATTACK IMPLEMENTATION AND EXPERIMENTAL EVALUATION

This chapter describes the research techniques and strategies I have chosen for this work. It describes the quests and worldviews and strategies used in collecting and examining quests and quest information.

4.1 Technology used

4.1.1 Map

Nmap ("Network Mapper") is an open source device for organization, research, and security auditing. It works fine for single hosts, but I need to check large organizations quickly. Nmap uses raw IP packets in novel ways to determine which hosts are accessible within an organization, what controls these hosts offer (application names and formats), what working environments they run (and operating system variations), Identifies the packet channel type. /Firewalls are used and come in different grades. Nmap is typically used for security checks, but many frameworks and organizational leaders find it useful for everyday tasks such as: B. Monitoring network inventory, management update schedules, and checking host or management uptime. Various highlights are disclosure, port filtering, variant identification, operating system detection, and scriptable mapping to targets. The output of Nmap is a summary of filtered foci, each containing additional data depending on the selections used. The key to this data is the "Attractive Harbor Table". This table records port numbers and conventions, administrative names, and status. The state can be open, screened, closed, or unfiltered. Open means the application on the target machine will be tuned to this port association/bundle. Disconnected means that a firewall, duct, or other organizational obstruction is blocking the port, so Nmap cannot determine if the port is open or closed. A closed port has no applications listening to it, but can be opened at any time. A port is said to be unfiltered if it is susceptible to Nmap's probes, but Nmap cannot determine if the port is open or closed. Nmap reports a mixed state of open|filtered and closed|filtered when it cannot determine which of the two states represents a port. Port tables may also include programming adjustment subtleties when variant locations are mentioned. At the point where an IP rule filter is mentioned (-sO), Nmap outputs data about the IP rule being followed, not the listening port. Despite the interesting port table, Nmap can provide additional data about the target, such as reverse DNS name, working framework inference, gadget type, MAC address, etc. In [11] it is written:

"Again, it shows that little is understood about how tests such as Nmap affect common elements of SCADA and ICS organizations." indicates a representation of - good.

This selection tells Nmap not to perform a port sweep after host publication, and only output reachable hosts that have responded to host publication probes.

This is often called a "ping filter", but it can also require traceroutes and NSE content to run. This is of course a more curious level than rundown exams and is often used for similar purposes. You can easily monitor the desired tissue without worrying too much. Knowing the number of hosts, you rely on is more important to an attacker than summarization, which provides a summary output for each IP and hostname. Even those responsible for frameworks often find this choice important. You can use it effortlessly to check accessible computers in your company or check employee accessibility. This is often called a ping clear and is more reliable than pinging the broadcast address because many hosts will not respond to communication requests. The default disclosure ends with - sn consists of an ICMP reverberation request.

TCP SYN on port 443, TCP ACK on port 80, and of course ICMP timestamps. When run by an unprivileged client, only SYN packets are sent (using interface calls) to ports 80 and 443 of the target. When a preferred client seeks a connection to a nearby

Ethernet network, an ARP request is used unless "Send-IP" is specified. The -sn alternative can be attached to any disclosure test type (-P* options except -Pn) for even more remarkable customizability. Using any of these test type and port number alternatives will override the default test. We recommend using these hard-nosed practices at points where there is a sophisticated firewall between the source and target organizations running Nmap. Also, hosts can be overlooked when the firewall is performing tests or its responses.

-sp

In previous versions of his Nmap, -sn was called -sP, and this order still works.

-p/-p

This is used to determine which port to target. -p- means all possible ports

-v

This is used to get a detailed yield of more data. – cent

TCP Associate Sweep is the default TCP probe type when a SYN probe is not possible. This is if the customer does not have a total package benefit.

Instead of building rough packets like most other sweep types, Nmap asks the underlying working framework to connect to the target his machine and port by making an interface framework call . This is the same level of framework call that Internet browsers, P2P clients, and most other organization-aware applications use to set up mappings. This is integral to the programming interface known as the Berkeley Sockets API. Instead of reading raw burst responses from the wire, Nmap uses this API to get status data for each connection attempt. Once SYN checking is available, it's usually a good decision. Nmap is less effective at calling higher-level interfaces than coarse-grained packages. The framework call ends the association with the open target port instead of performing the half-open reset that the SYN check does. However, collecting similar data takes longer, requires more bundles, and requires the target machine to log the association. A traditional IDS gets both, but most machines don't have such an alarm framework. If you're doing a lot of maintenance with the normal Unix framework, when Nmap connects and then closes the connection without sending any information, it adds notes to syslog and cryptic error messages all over the place. . When this happens, a truly deplorable regime collapses, but this is an exception. Administrators logging a lot of linking activity from a single environment should be aware that the interfaces are being inspected.

-s.p

SYN probes are the standard and most common justification output option. It scans many ports per second in a fast organization that usually runs fast and is not hindered by firewalls that prohibit it. It's also usually subtle and unobtrusive, as it never terminates the TCP connection. SYN checks neutralize all acceptable TCP stacks, rather than relying on explicit levels of idiosyncrasies like Nmap's FIN/NULL/Xmas, Maimon, and idle sweeps. You can also clearly and reliably distinguish between open, closed and screened states. This strategy is often called half-open probing because it does not open a full TCP connection. Send a SYN packet and wait for a response as if opening a real association. SYN/ACK indicates the port is tuned (open) and RST (reset) indicates private. If there is no response after several retransmissions, the port is marked as disconnected. Ports are additionally stamped individually if an ICMP inaccessible error (type 3, code 0, 1, 2, 3, 9, 10, or 13) is received. A port is also considered open if a SYN packet (without the ACK banner) is received accordingly.

-O

Nmap's Distant OS Discovery uses TCP/IP stack fingerprinting. Nmap sends a series of TCP and UDP packets to a remote host and analyzes each part of the response for all intents and purposes. After performing many tests such as TCP ISN checks, TCP alternate backing and requests, IP ID checks, underlying window size checks, Nmap has compiled an nmap-os-db information database of over 2,600 realized OS fingerprints. Examine and output the result of . OS

-s TV

Form recognition looks at the port to determine what is actually running. The Nmap Dosage Test Information Base contains tests to explore different dosages and match articulations to recognize and analyze responses. Nmap will try to set the management rules, variant number, hostname, gadgets, and OS family. Standard Procurement allows recovery of highly unique variant numbers. [Five]

-Tx (-T1, -T2, -T3, -T4, -T5)

With six planning layouts, Nmap provides an easy way to handle timing control. Can be specified by -T option and its number (0-5) or name. The formal names are nervous (0), tricky (1), graceful (2), typical (3), powerful (4), and crazy (5). The first two are for IDS evasion. Pleasant mode prevents the data transfer and target machine resources used by the output from getting low. So standard mode is the default. T3 is idle. Force mode speeds up the investigation by assuming you're in a fairly fast and solid organization. Finally, in wacky mode, we happily expect to be in a blazing fast org, or sacrifice some accuracy for speed. While choosing a value, you can specify how energetic you want it to be. The format also makes some minor speed changes for which there is currently no alternative for finer control. For example, T4 prohibits TCP port dynamic output delays from exceeding 10 ms, and T5 exceeds this estimate by 5 ms. Styles can be used in conjunction with fine-grained controls, and the specified fine-grained control will override the defaults for its border context layout.

4.1.2 Ness

Nessus is Tenable's professional vulnerability assessment tool. Leverage the Common Vulnerabilities and Exposure architecture to facilitate networking between compliant security devices. It has a closed architecture consisting of an integrated server that performs scanning and a remote client that allows for administrator interaction. Administrators can create custom scans by integrating Nessus attack script language descriptions of all published vulnerabilities. Key features of Nessus include:

• Compatibility with PCs and servers under the same conditions.

• Detection of security holes in nearby or distant hosts.

• Detect missing security updates and patches.

• Simulated attacks to identify vulnerabilities. • Running security tests in a closed environment.

• Regular security reviews.

4.1.3 Metasploit overview

The Metasploit project [14] is an open source downloadable exploit framework designed to be a comprehensive extension environment for penetration testing. Metasploit 1.0 first shipped in 2003 and quickly spread to penetration testing and information security networks. Since being acquired by industry leader in vulnerability scanning, Rapid7, in 2009, the Metasploit project has grown exponentially. The latest version (4.11.5), which had 11 trials in early 2003, now contains over 1500 adventures [33]. Besides exploits, Metasploit includes other devices that support different stages of penetration testing. These devices include scanners, payloads, and session handlers. Various network and port scanners are built into Metasploit to aid in host and vulnerability discovery. Metasploit also offers a huge number of payloads (bundles of code that are transferred to the target computer throughout the adventure). The payload, once delivered through a successful adventure, provides current usage functionality in the analyzer. One model is a terminal session, such as the Converse shell, which restores the command shell from the target machine to the analyzer [38]. Session handlers are used to handle sessions returned from successful ventures and payload deliveries. Interactions include granting or leaving multiple sessions, executing code remotely, and using highlighting. B. Routing, Hash Offload, Privilege Escalation. 4.2 Snort Implementation

Depending on your operating system and CPU architecture, getting started with Snort can be as simple as three words and pressing enter, or as fun as an oral treatment without the benefit of anesthesia. Snort does not come with an installer application. And the 250-page installation, administration, and customer manual isn't in a curvy hardcover book with glossy pages.

```
###################################################
# Step #1: Set the network variables.  For more information, see README.variables
###################################################

# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.0.0/16

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET

# List of sip servers on your network
ipvar SIP_SERVERS $HOME_NET
```

Figure 2 Setting the network variables

```
# List of ports you run web servers on
portvar HTTP_PORTS
[80,81,311,383,591,593,901,1220,1414,1741,1830,2301,2381,2809,3037,3
128,3702,4343,4848,5250,6988,7000,7001,7144,7145,7510,7777,7779,8000
,8008,8014,8028,8080,8085,8088,8090,8118,8123,8180,8181,8243,8280,83
00,8800,8888,8899,9000,9060,9080,9090,9091,9443,9999,11371,34443,344
44,41080,50002,55555]

# List of ports you want to look for SHELLCODE on.
portvar SHELLCODE_PORTS !80

# List of ports you might see oracle attacks on
portvar ORACLE_PORTS 1024:

# List of ports you want to look for SSH connections on:
portvar SSH_PORTS 22

# List of ports you run ftp servers on
portvar FTP_PORTS [21,2100,3535]

# List of ports you run SIP servers on
portvar SIP_PORTS [5060,5061,5600]

# List of file data ports for file inspection
portvar FILE_DATA_PORTS [$HTTP_PORTS,110,143]

# List of GTP ports for GTP preprocessor
portvar GTP_PORTS [2123,2152,3386]
```

Figure 3 List of ports that run web servers

```
# other variables, these should not be modified
ipvar AIM_SERVERS
[64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200
.0/24,205.188.3.0/24,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,20
5.188.153.0/24,205.188.179.0/24,205.188.248.0/24]

# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute
path,
# such as:  c:\snort\rules
var RULE_PATH c:\Snort\rules
#var SO_RULE_PATH ../so_rules
var PREPROC_RULE_PATH c:\Snort\preproc_rules

# If you are using reputation preprocessor set these
# Currently there is a bug with relative paths, they are relative to
where snort is
# not relative to snort.conf like the above variables
# This is completely inconsistent with how other vars work, BUG
89986
# Set the absolute path appropriately
var WHITE_LIST_PATH c:\Snort\rules
var BLACK_LIST_PATH c:\Snort\rules
```

Figure 4 Other non-changeable variables

```
C:\Snort\bin>snort -W

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.16.1-WIN32 GRE (Build 140)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2020 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using PCRE version: 8.10 2010-06-25
           Using ZLIB version: 1.2.3

Index   Physical Address        IP Address      Device Name     Description
-----   ----------------        ----------      -----------     -----------
    1   00:00:00:00:00:00       0000:0000:fe80:0000:0000:0000:bce1:79dc \Device\NPF_{2DFE82C4-FF9D-4390-B07B-E73E0AF535C
B}      Microsoft
    2   00:00:00:00:00:00       0000:0000:fe80:0000:0000:0000:c132:1e83 \Device\NPF_{6A3D99AF-78BB-491C-B7B4-31477DBB6E8
6}      VMware Virtual Ethernet Adapter
    3   00:00:00:00:00:00       0000:0000:fe80:0000:0000:0000:349a:3a25 \Device\NPF_{DBF25F52-94B2-4820-B0A2-C7EA6BBC4C1
C}      Broadcom NetLink (TM) Gigabit Ethernet Driver
    4   00:00:00:00:00:00       0000:0000:fe80:0000:0000:0000:094f:4961 \Device\NPF_{CA33AC30-4CA6-4F58-9474-9D103A2650C
7}      Microsoft
    5   00:00:00:00:00:00       0000:0000:fe80:0000:0000:0000:041b:8aa6 \Device\NPF_{95C5D8FB-1C65-4F16-8E16-6C635D8E7DB
A}      Microsoft
    6   00:00:00:00:00:00       0000:0000:fe80:0000:0000:0000:ac2f:c1a2 \Device\NPF_{0473F337-772B-4075-9681-B5DB4C0BE2D
8}      Microsoft
    7   00:00:00:00:00:00       0000:0000:fe80:0000:0000:0000:95d4:d628 \Device\NPF_{8663BAB9-8E27-493C-A611-FEE2B4088D4
3}      VMware Virtual Ethernet Adapter
```

Figure 5 snort -w command

```
SSH config:
    Autodetection: ENABLED
    Challenge-Response Overflow Alert: ENABLED
    SSH1 CRC32 Alert: ENABLED
    Server Version String Overflow Alert: ENABLED
    Protocol Mismatch Alert: ENABLED
    Bad Message Direction Alert: DISABLED
    Bad Payload Size Alert: DISABLED
    Unrecognized Version Alert: DISABLED
    Max Encrypted Packets: 20
    Max Server Version String Length: 100
    MaxClientBytes: 19600 (Default)
    Ports:
        22
DCE/RPC 2 Preprocessor Configuration
  Global Configuration
    DCE/RPC Defragmentation: Enabled
    Memcap: 102400 KB
    Events: co
    SMB Fingerprint policy: Disabled
  Server Default Configuration
    Policy: WinXP
    Detect ports (PAF)
      SMB: 139 445
      TCP: 135
      UDP: 135
      RPC over HTTP server: 593
      RPC over HTTP proxy: None
    Autodetect ports (PAF)
      SMB: None
      TCP: 1025-65535
      UDP: 1025-65535
      RPC over HTTP server: 1025-65535
      RPC over HTTP proxy: None
    Invalid SMB shares: C$ D$ ADMIN$
    Maximum SMB command chaining: 3 commands
    SMB file inspection: Disabled
DNS config:
    DNS Client rdata txt Overflow Alert: ACTIVE
    Obsolete DNS RR Types Alert: INACTIVE
    Experimental DNS RR Types Alert: INACTIVE
    Ports: 53
```

Figure 6 SSH Config

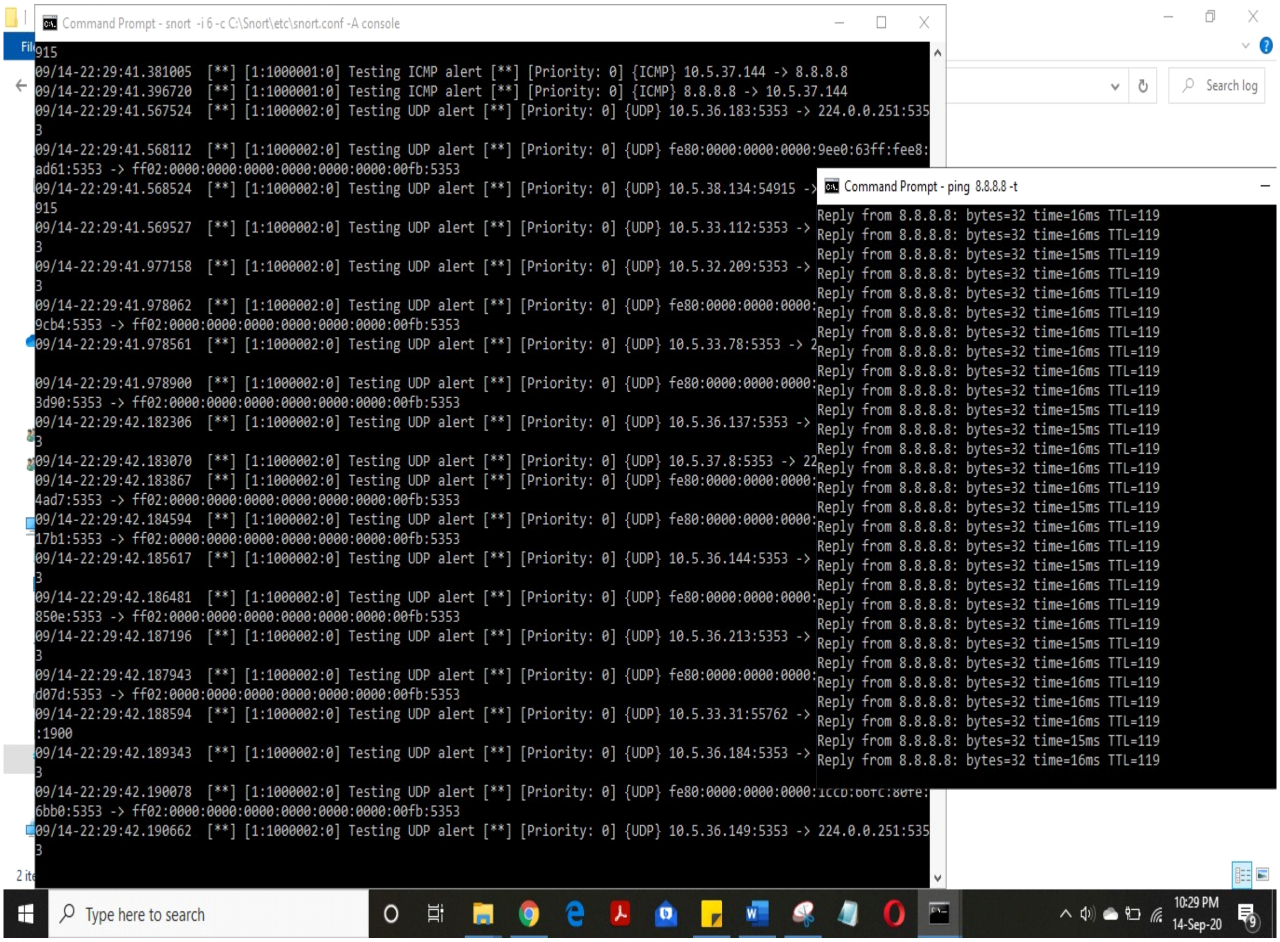


Figure 7 snort.conf –A console command

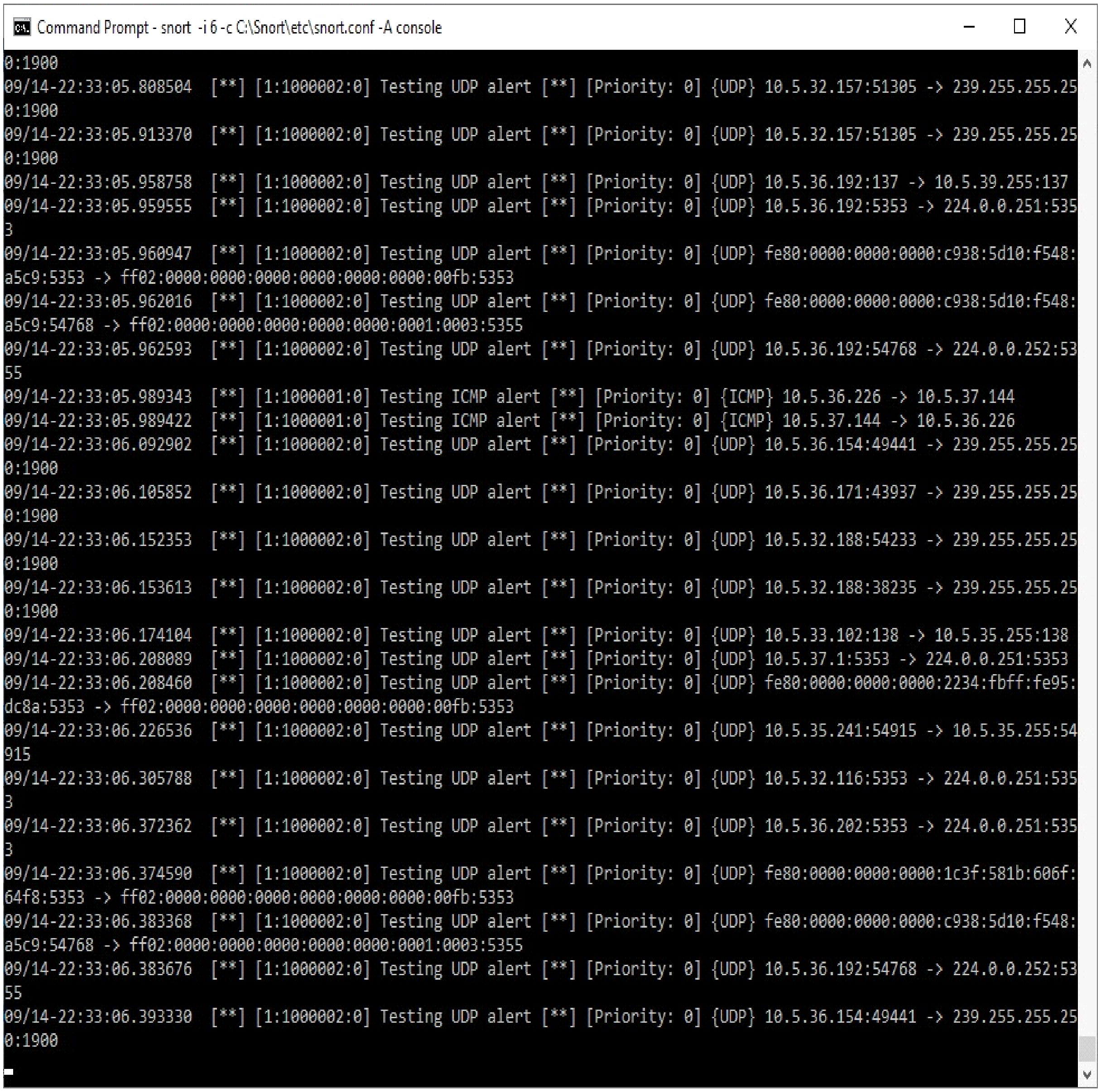


Figure 8 snort.conf –A console command

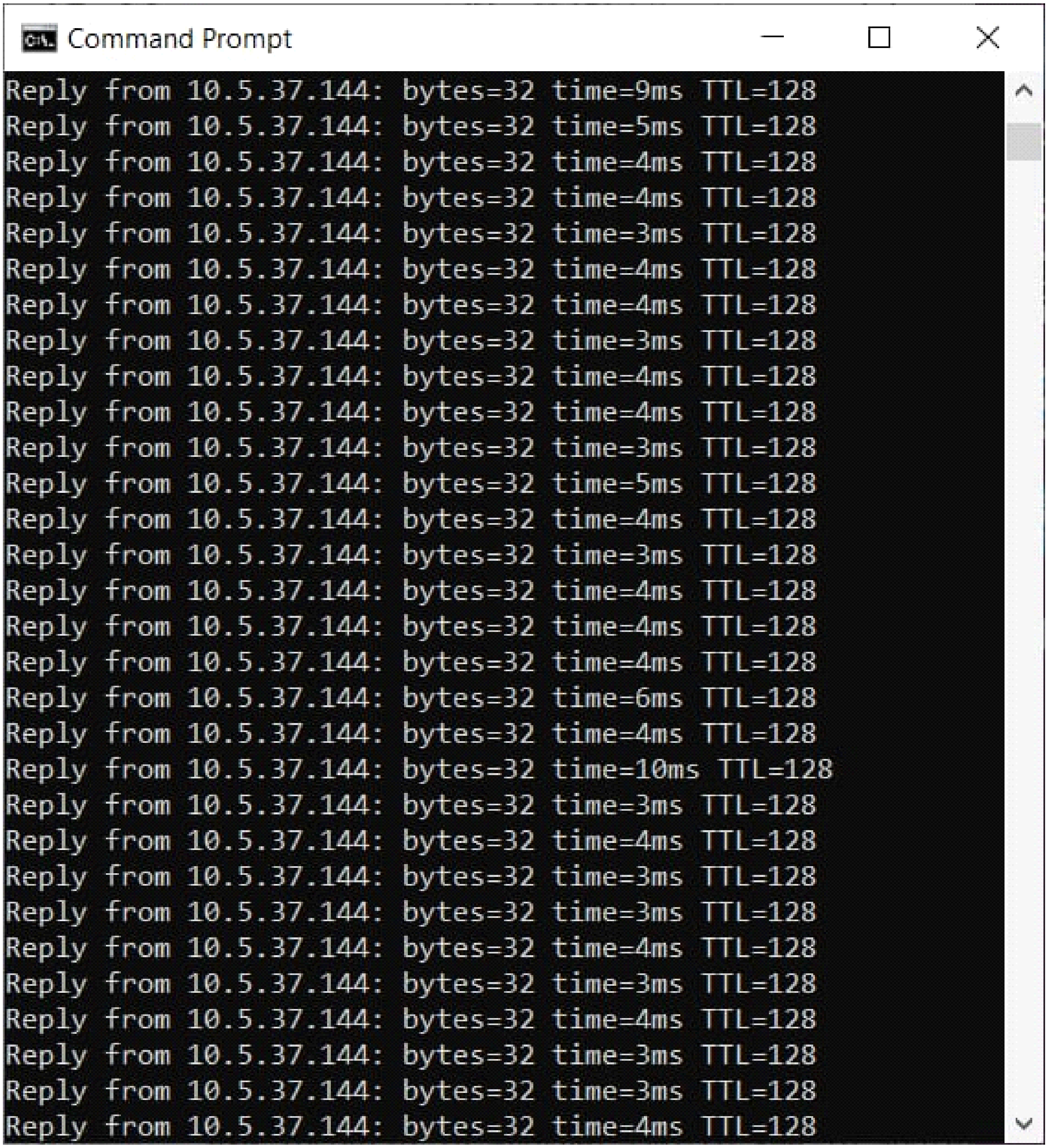


Figure 9 snort.conf –A console command

Command Prompt

```
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 13:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Ethernet 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::39f8:b8d6:5bf8:d8be%21
   IPv4 Address. . . . . . . . . . . : 192.168.195.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet8:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::9854:87e1:1aea:f723%22
   IPv4 Address. . . . . . . . . . . : 192.168.58.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::2894:8062:1a5f:8447%14
   IPv4 Address. . . . . . . . . . . : 192.168.100.4
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::1%14
                                       192.168.100.1

C:\Users\olikaj>
```

Figure 10 IP Config

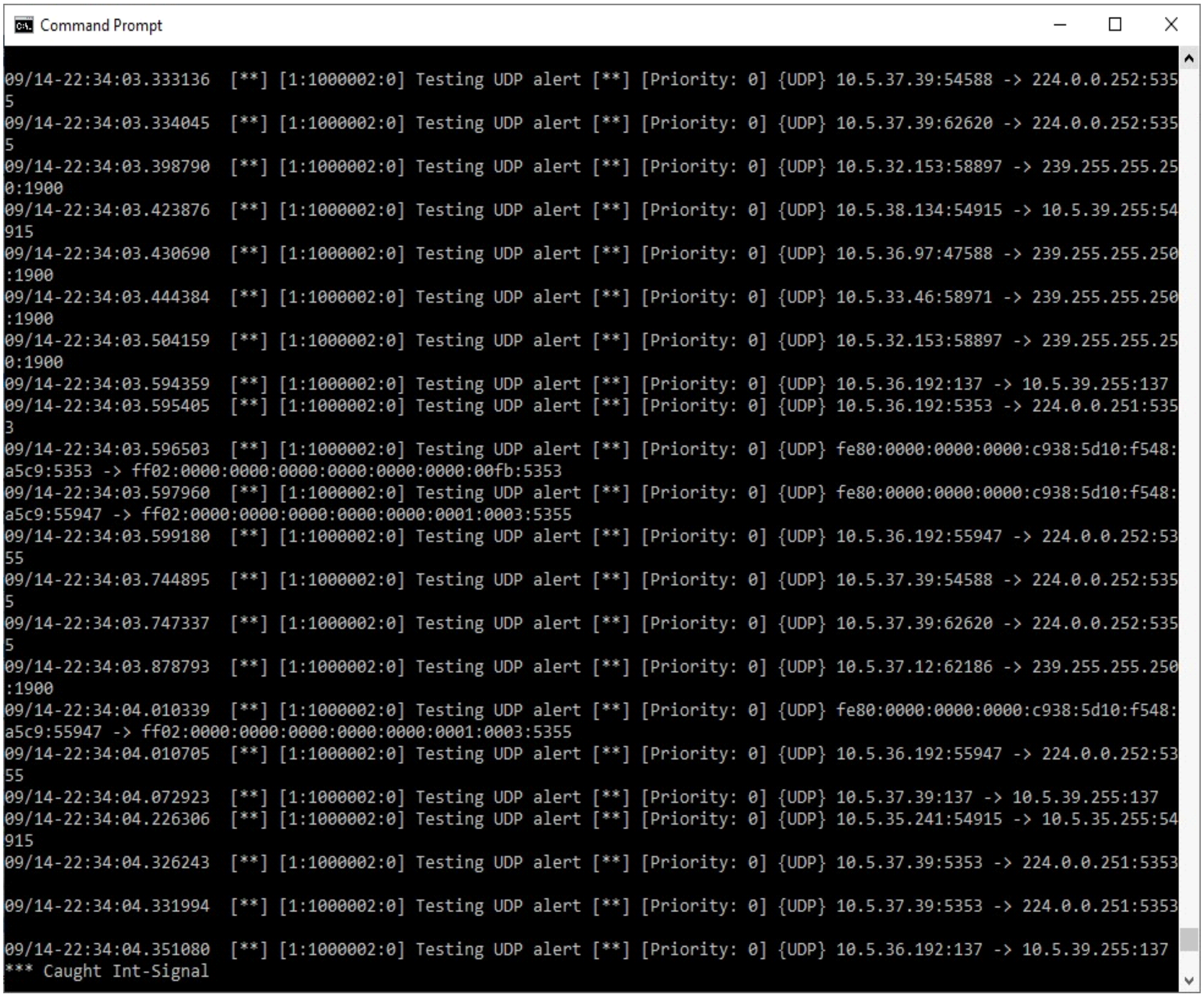


Figure 11 Sending Packets

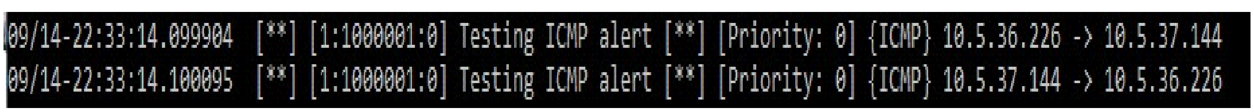


Figure 12 Testing ICMP alerts

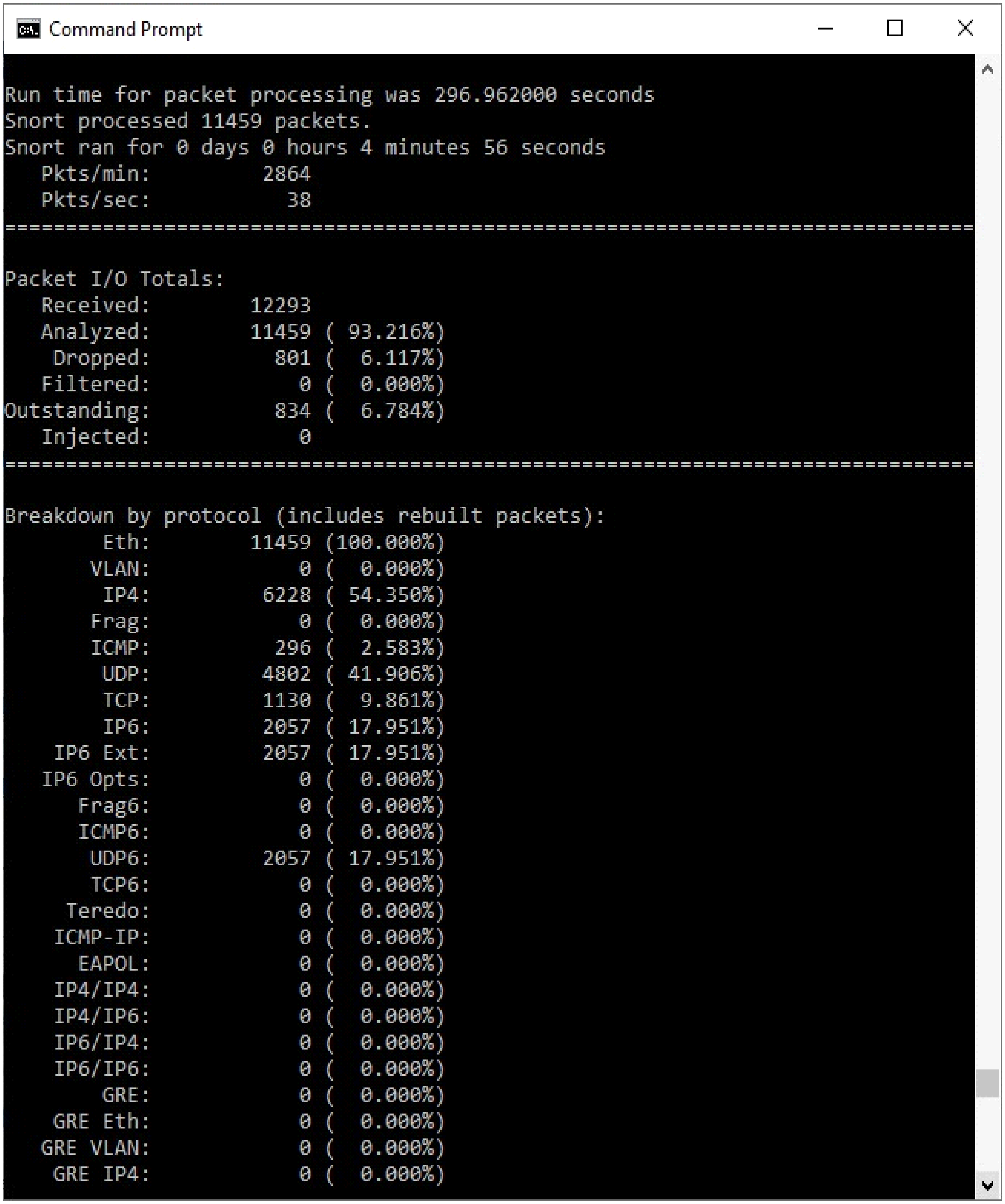
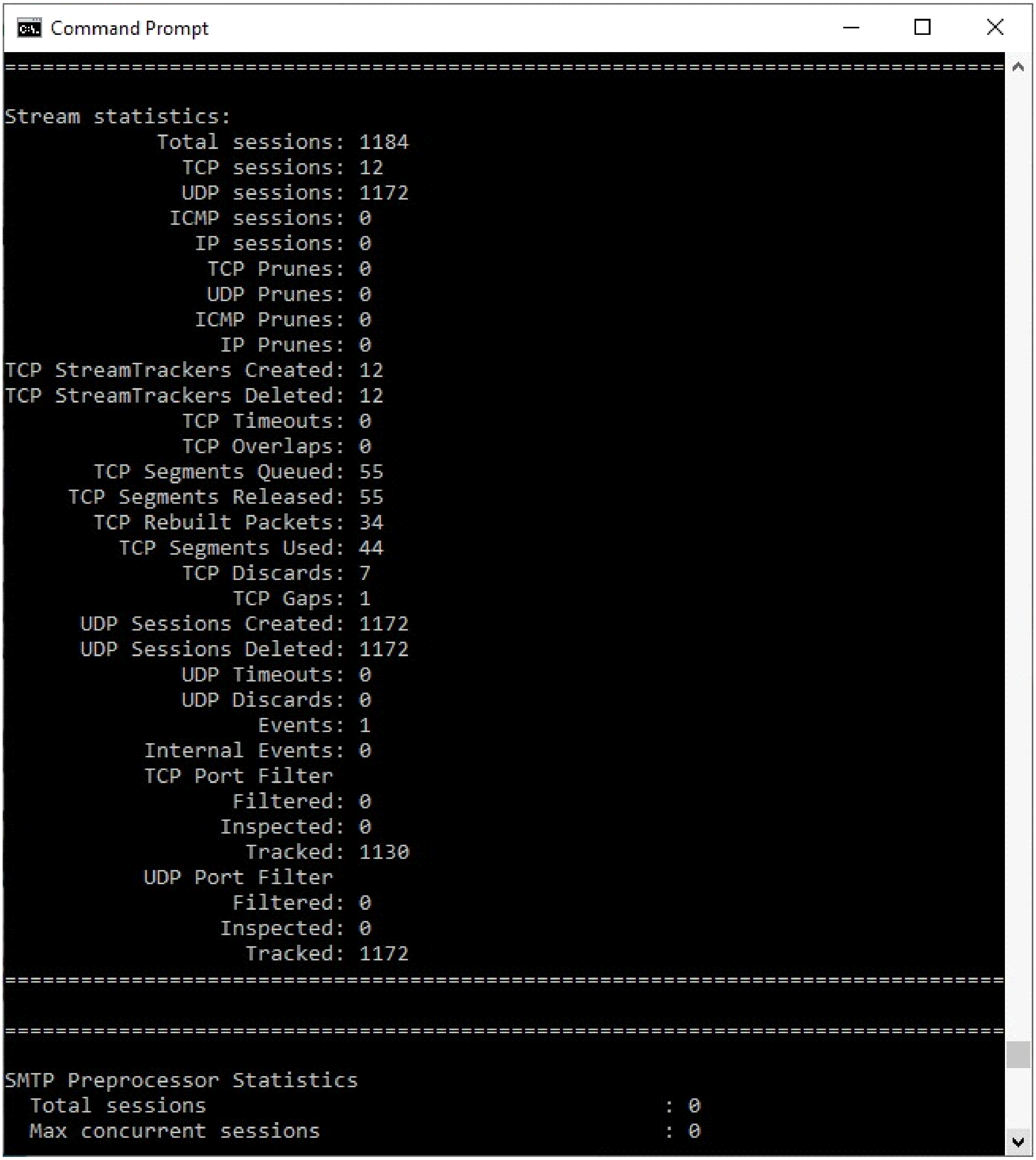


Figure 13 Snort Packet Processing

```
Command Prompt

Action Stats:
     Alerts:         8286 ( 72.310%)
     Logged:         8286 ( 72.310%)
     Passed:            0 (  0.000%)
Limits:
      Match:            0
      Queue:            0
        Log:            0
      Event:            0
      Alert:           34
Verdicts:
      Allow:        10426 ( 84.812%)
      Block:            0 (  0.000%)
    Replace:            0 (  0.000%)
  Whitelist:         1033 (  8.403%)
  Blacklist:            0 (  0.000%)
     Ignore:            0 (  0.000%)
     (null):            0 (  0.000%)
===============================================================================

Frag3 statistics:
        Total Fragments: 0
      Frags Reassembled: 0
               Discards: 0
          Memory Faults: 0
               Timeouts: 0
               Overlaps: 0
              Anomalies: 0
                 Alerts: 0
                  Drops: 0
     FragTrackers Added: 0
    FragTrackers Dumped: 0
FragTrackers Auto Freed: 0
    Frag Nodes Inserted: 0
     Frag Nodes Deleted: 0
===============================================================================

===============================================================================

Stream statistics:
            Total sessions: 1184
              TCP sessions: 12
```

Figure 14 Snort Packet Processing



```
Command Prompt
===============================================================================

Stream statistics:
            Total sessions: 1184
              TCP sessions: 12
              UDP sessions: 1172
             ICMP sessions: 0
               IP sessions: 0
                TCP Prunes: 0
                UDP Prunes: 0
               ICMP Prunes: 0
                 IP Prunes: 0
TCP StreamTrackers Created: 12
TCP StreamTrackers Deleted: 12
              TCP Timeouts: 0
              TCP Overlaps: 0
       TCP Segments Queued: 55
     TCP Segments Released: 55
       TCP Rebuilt Packets: 34
         TCP Segments Used: 44
              TCP Discards: 7
                  TCP Gaps: 1
      UDP Sessions Created: 1172
      UDP Sessions Deleted: 1172
              UDP Timeouts: 0
              UDP Discards: 0
                    Events: 1
           Internal Events: 0
           TCP Port Filter
                  Filtered: 0
                 Inspected: 0
                   Tracked: 1130
           UDP Port Filter
                  Filtered: 0
                 Inspected: 0
                   Tracked: 1172
===============================================================================

===============================================================================

SMTP Preprocessor Statistics
  Total sessions                                    : 0
  Max concurrent sessions                           : 0
```

Figure 15 Snort Packet Processing

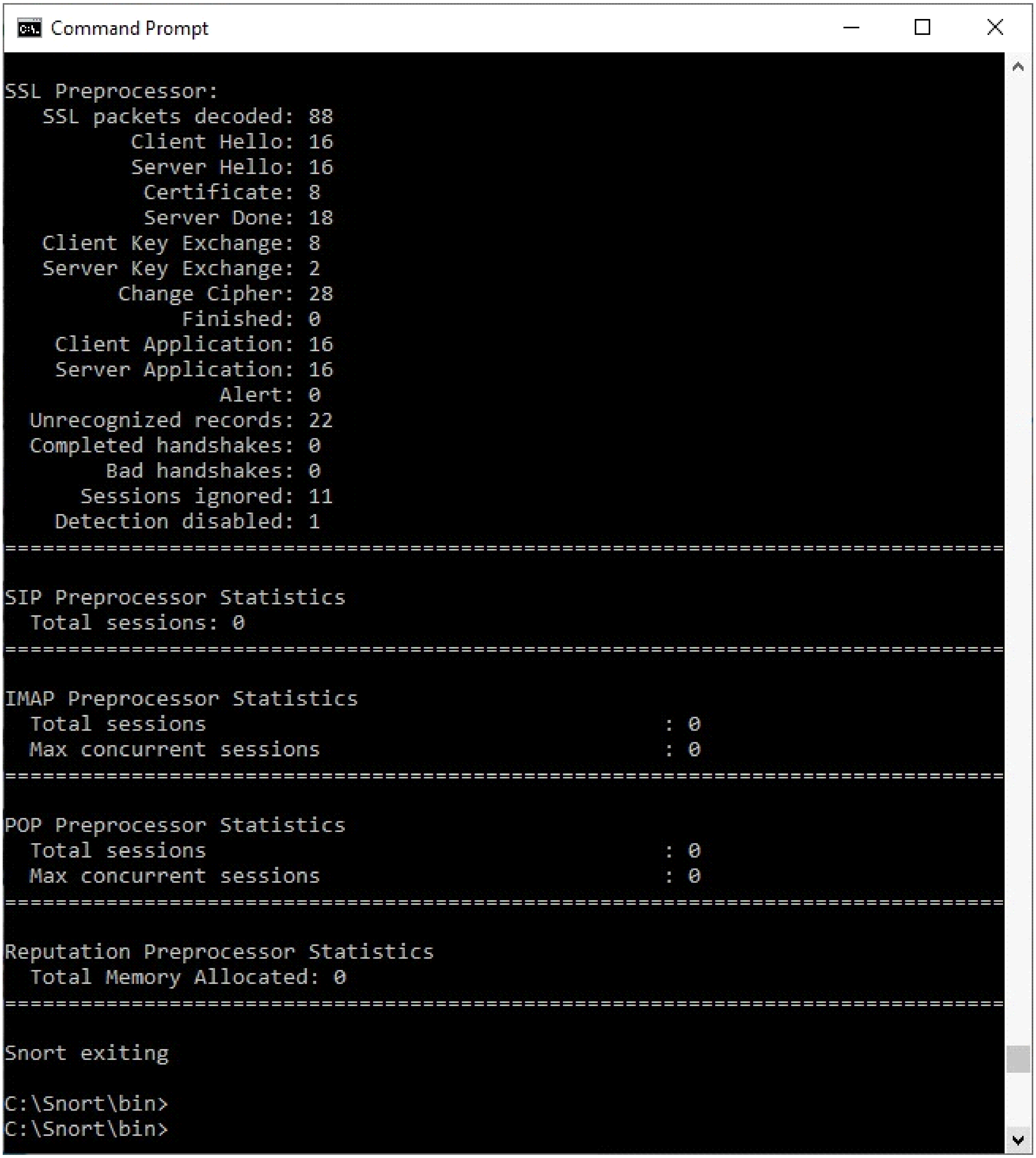


Figure 16 Snort Exiting

4.1.4 Metasploit

The Metasploit project [14] is an open source downloadable exploit framework designed to be a comprehensive extension environment for penetration testing. Metasploit 1.0 first shipped in 2003 and quickly spread to penetration testing and information security networks. Since being acquired by industry leader in vulnerability scanning, Rapid7, in 2009, the Metasploit project has grown exponentially. The latest version (4.11.5), which shipped 11 trials in early 2003, now has over 1500 adventures [33]. Besides exploits, Metasploit includes other devices that support different stages of penetration testing. These devices include scanners, payloads, and session handlers. Various network and port scanners are built into Metasploit to aid in host and vulnerability discovery. Metasploit also offers a huge number of payloads (bundles of code that are transferred to the target computer throughout the adventure). The payload, once delivered through a successful adventure, provides current usage capabilities in the analyzer. One model is a terminal session, such as the Converse shell, which restores the command shell from the target machine to the analyzer [38]. Session handlers are used to handle sessions returned from successful ventures and payload deliveries. Interactions include granting or leaving multiple sessions, executing code remotely, and using highlighting. B. Routing, Hash Offload, Privilege Escalation. 4.3 Attack simulation procedure

4.2 Finding host ports with Nmap

One of the first phases of a network reconnaissance mission is to reduce a (possibly huge) set of IP ranges to a dynamic or interesting set of hosts. Scanning all ports on all IP addresses is easy and usually unnecessary. Obviously, the attractiveness of a host is highly dependent on the purpose of the scan. A network administrator may want only certain services to run, while a security auditor may want all devices remembered by her IP address. While it is sufficient for administrators to examine incoming traffic to the network using ICMP pings, external attack analyzers may be using various probes to try to circumvent firewall restrictions. Disclosure requirements can vary greatly, so Nmap offers a wide variety of options for customizing the techniques used. Host Revelation is often referred to as a ping scan, but it actively works beyond the basic ICMP reverb request packets associated with universal ping devices. The client uses rundown scanning (-SL) or ping degradation (-Pn), or an aggressive combination of multiport TCP SYN/ACK, UDP, SCTP-INIT, and ICMP, and the network withdraws the probe and pings. Check - Avoid the step entirely. The purpose of these tests is to request a response (used by hosts or network gadgets) that indicates that the IP address is truly dynamic. In many networks only a few IP addresses are dynamic. This is especially common in private address spaces. for example:

10.0.0.0/8. This network has 16 million IPs, but I've seen machines used by less than 1,000 companies. Host Revelation can find these machines in a sparsely distributed sea of IP addresses.

If no host disclosure options are specified, Nmap sends ICMP reverberation requests, TCP SYN packets on port 443, TCP ACK packets on port 80, and ICMP timestamp requests. (IPv6 overlooks the ICMP timestamp requirement on the grounds that it is not important for ICMPv6.) These presets are the same as options - PE - PS443 - PA80 - PP. Exceptions to this are ARP (for IPv4) and Neighbor Discovery (for IPv6) scans, which are used for all targets on nearby Ethernet networks. For unprivileged Unix shell clients, standard probes are SYN packets to ports 80 and 443 using the connect system call. While this host disclosure is often sufficient to scan nearby networks, a more comprehensive set of Revelation probes is recommended for security audits.

The -P* options (which select the ping type) can be combined. Sending different types of probes with unique TCP ports/banners and ICMP codes increases your chances of getting through tough firewalls. Also note that ARP/Neighbor Discovery is performed for targets on nearby Ethernet networks regardless of whether you specify any other -P* options. Because it is often faster and more powerful.

Of course, Nmap has a revelation and performs a port scan of every host it finds online. This applies regardless of whether there are specific non-standard disclosure types. B. UDP probe (-PU). For more information on the -sn option, see How to do host disclosure only, or how to use -Pn to skip all target disclosures and port scans. 4.3.2 Vulnerability scanning with Nessus

Nessus scans and detects vulnerabilities using an engine that runs on every host on your network. Modules can be thought of as individual bits of code used by Nessus to perform individual types of scans on targets. The modules vary greatly in their capacity. For example, you can send your module to the host and send it to:

• Identify which live systems and services are running on which ports. • Id

entify software components that are vulnerable to attack (FTP, SSH, SMB, etc.). • Distinguish whether compliance requirements are met on different hosts.

The measures followed in the scan can be summarized in the image below.

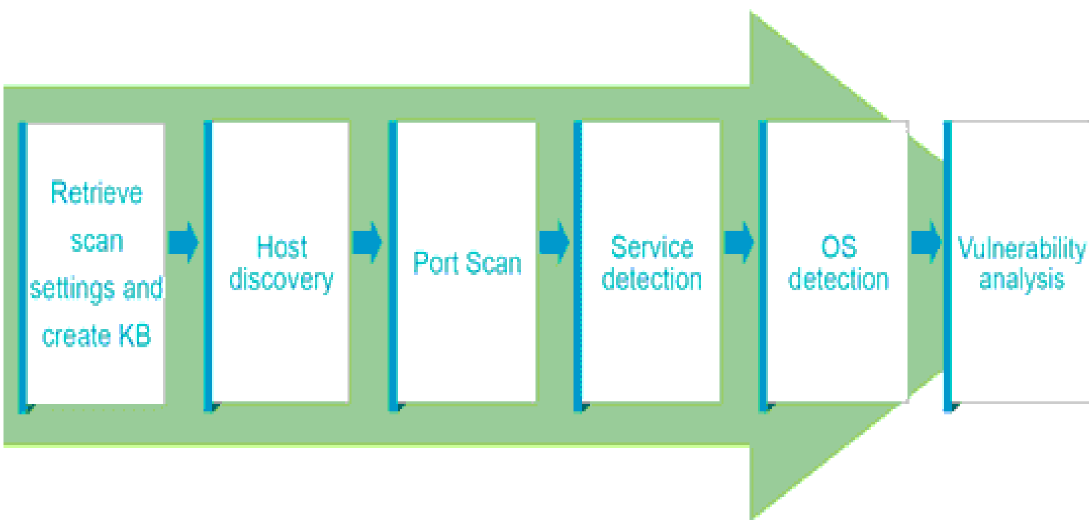


Figure 17 Nessus scan procedure

When you submit your scan, Nessus goes through a series of steps.

Stage 1:

Nessus will restore your scan settings. A configuration characterizes the ports to scan, the modules to enable, and the definition of policy settings.

Level 2:

Nessus then performs a publish to determine active hosts. The protocols used are ICMP, TCP, UDP and ARP. You can specify these as needed.

Level 3:

Nessus then performs a port scan of each host detected as active. You can also characterize the ports to scan. Ports can be characterized as ranges or exclusively with significant ports ranging from 1 to 65535.

Level 4:

Nessus then performs service discovery to identify services running behind each port on each discovered host.

Stage 5:

Nessus will perform a discovery of the system in action.

Stage 6:

Once all resources are exhausted, Nessus compares each host to its intelligence base of known vulnerabilities to determine which hosts have which vulnerabilities.

4.3 Attack by Metasploit

As a mainstream penetration testing framework, Metasploit is only perceived as a toolkit. It is usually an important tool for exploiting project security vulnerabilities and exploiting these vulnerabilities to control information systems. This allows Personals to go on their own adventures with security vulnerabilities and use them to attack machines.

It has become the most popular device for performing hacking operations, especially when it comes to computer-aided assessment of security gaps. In addition to this, it was a basic tool for securing an organization's network. It is a compelling device used to detect and exploit security vulnerabilities in an organization, with the majority of attackers using Metasploit as a means of attacking vulnerable systems.

4.3.1 Functionality

Metasploit is an assembly of several applications used to computerize several phases of penetration testing. Its use can be extended to a range of situations where it is intended to detect security breaches and use the control interface with post-use and reporting devices. Its framework detects vulnerabilities using information identified on vulnerable hosts, exploits vulnerabilities to launch attacks with payloads, and extracts information from vulnerability scanners that destroy systems. To do.

Attackers exploit isolated results from vulnerability scanners and import them into Armitage, the Metasploit project's graphical digital attack management appliance, to detect vulnerabilities in that module. After discovering the vulnerability, the attacker uses an executable adventure to influence the system, gain a shell, and dispatch a progressively upgradeable payload, Meterpreter, to control the system.

Payload refers to the commands used to execute on neighboring systems after gaining access through the enterprise. This may include technical documentation and information bases used to build practical adventures after identifying vulnerabilities. These payloads routinely contain components to isolate passwords from neighboring systems, introduce other software, or control the device in the same manner as late access devices such as BO2K. increase.

4.3.2 Defense against Metasploit-based attacks

As an information security device, Metasploit discovers its applications in both security protection and attack. Malicious programmers exploit vulnerabilities against organizations and use them to give unauthenticated access to networks, applications, and information systems. You can learn about the real causes of these attacks by attending a Metasploit course.

```
msf exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 10.0.2.4:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 10.0.2.4:21 - USER: 331 Please specify the password.
[+] 10.0.2.4:21 - Backdoor service has been spawned, handling...
[+] 10.0.2.4:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (10.0.2.15:34037 -> 10.0.2.4:6200) at 2018-06-12 23:57:21 -0400

id
uid=0(root) gid=0(root)
```

Figure 18 Exploit Command

Attacks orchestrated by Metasploit can be detected on your network unless you use the "encode" option to prevent network traffic from being monitored by an intrusion detection system. In addition to this, Metasploit activity can also be monitored using host-based detection tools that monitor executables running on nearby systems.

That said, you can build incredible security material, or you can tear it apart. Attackers also like to detect similar vulnerabilities, so this could be a problem for organizations that expect continuous security and use his Metasploit as a first line of defense device. Using Metasploit as a component of your organization's vulnerability management program can effectively control security attacks through patch and configuration updates. Not patching can prevent network abuse even if the system is compromised. If you want to learn all about hacking, join the Moral Hacking Course.

In particular, Metasploit can be used to organize patch or vulnerability management plans and practices within an organization. Once the Metasploit module is deployed, organizations can install patches on premises that are in high demand, especially given the restricted use of the system by contented young people of that age. If you're interested in the Morale Hacking certification, you'll know how vulnerabilities detected by Metasploit are placed at the top of your vulnerability list to remediate or mitigate threats to your organization.

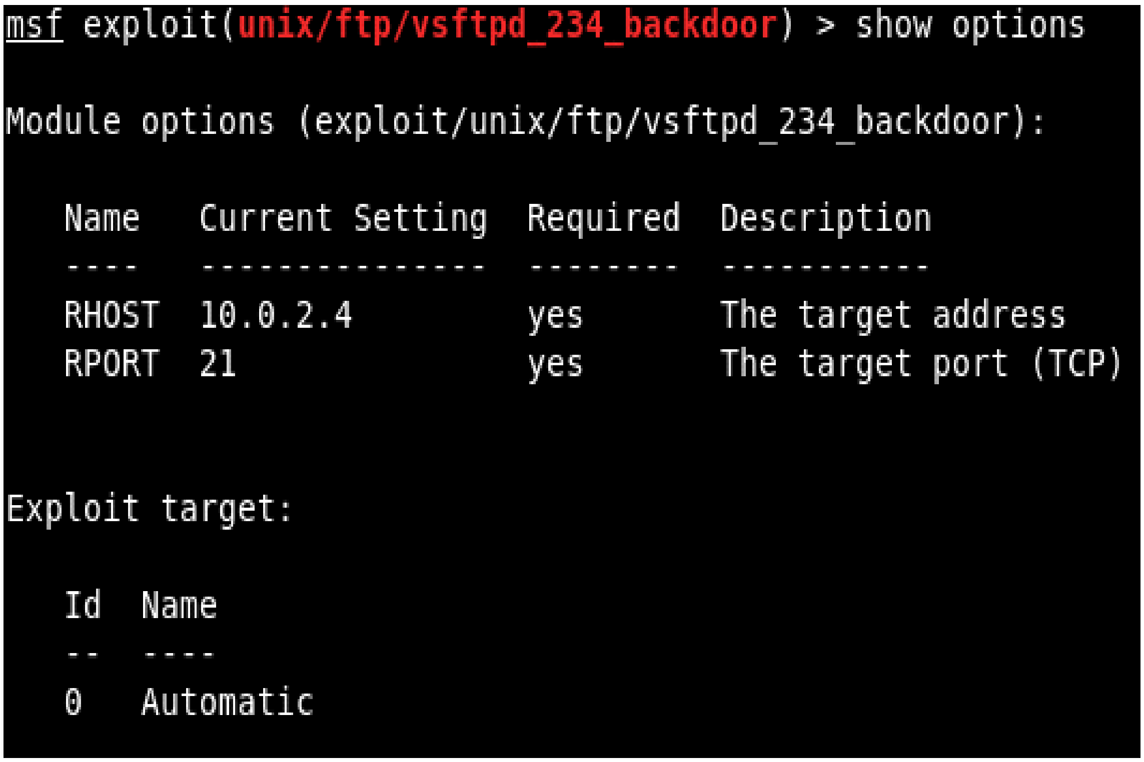


Figure 19 Show options command

4.4 Attack setup

Creating this attack required downloading a version of Windows that had a potential vulnerability and could be exploited using the methods described above. I decided to download Windows 7 and recently decided to end support for its users. This will allow users to pass and upgrade to a better version of Windows 10.

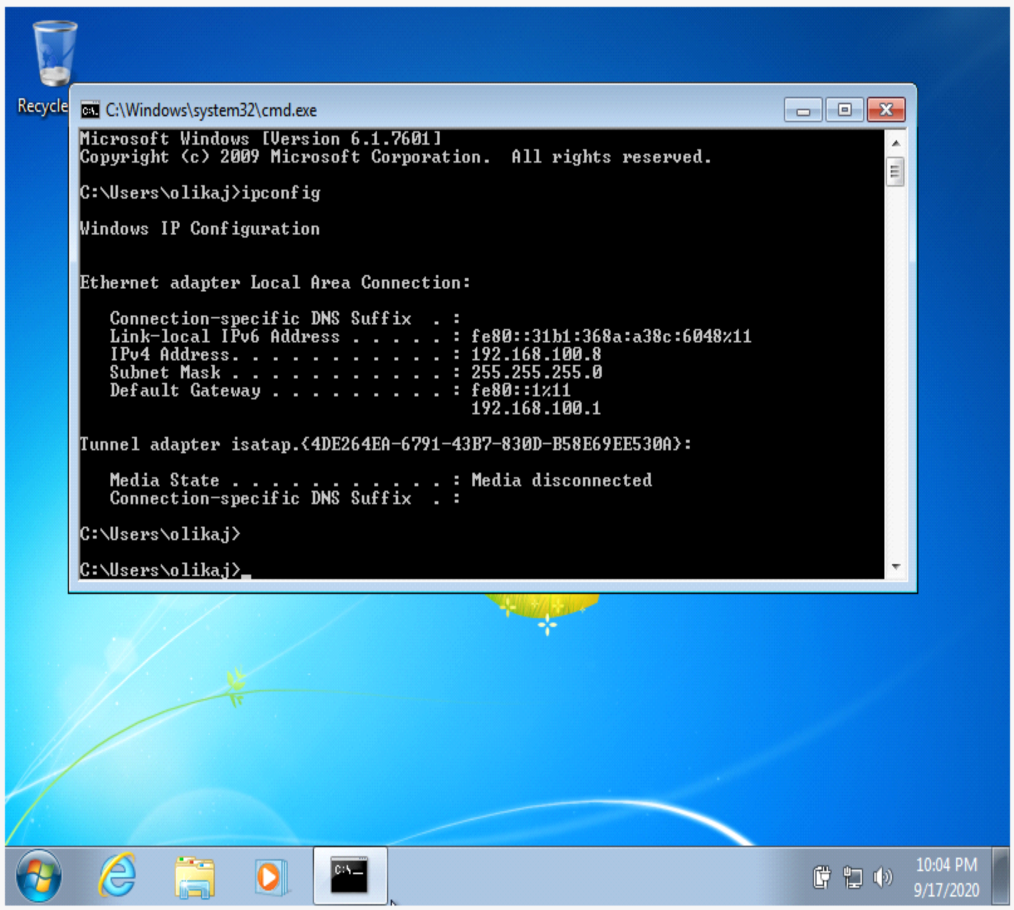


Figure 20 Virtual Machine created in VBox

After creating the virtual machine in Virtual Box, I also created network settings so that the new operating system would get an IP address as the host. Select the Bridged Adapter option.

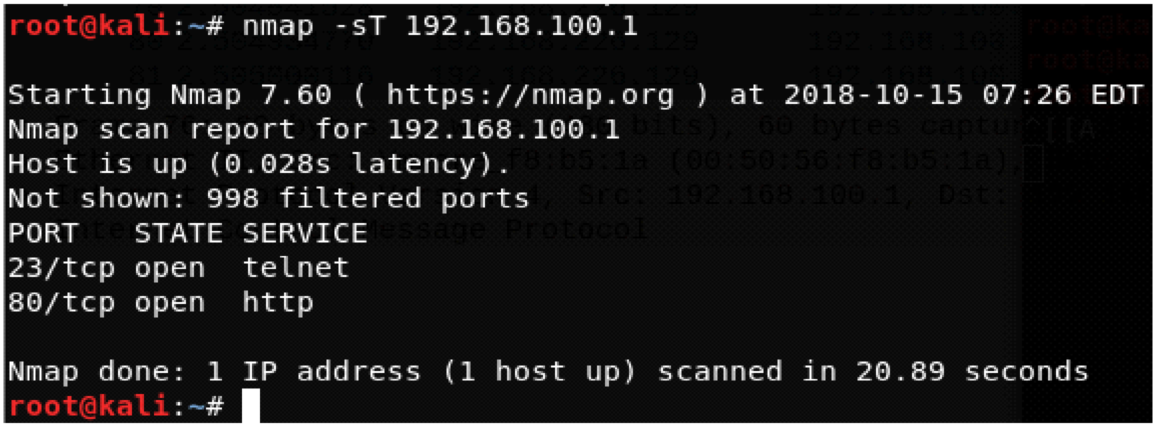


Figure 21 scan with Nmap

I ran some tests and found everything to be fine. I decided to use Nmap to see what ports might be open on this new operating system. The same process can be repeated for scans performed by Nessus and its plugins.

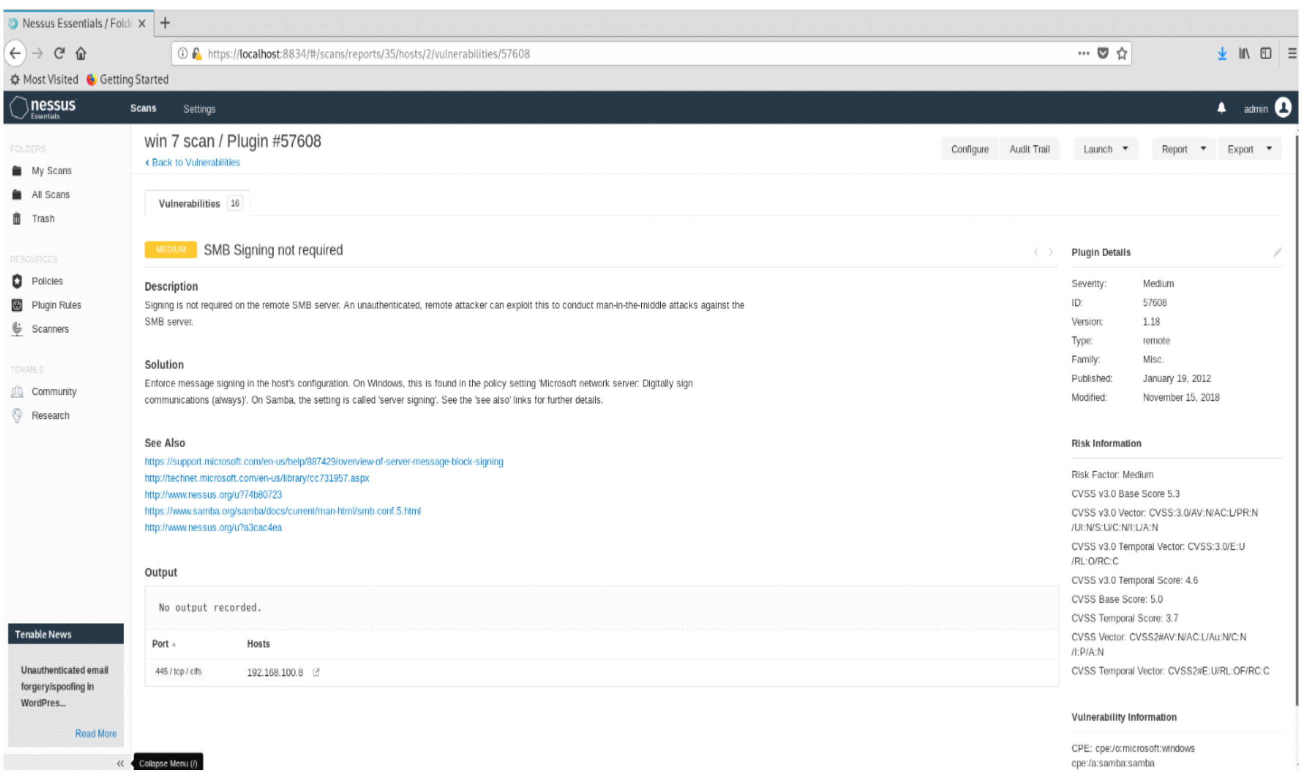


Figure 22 Scan with Nessus

After that, we ran some scans on Nessus and found some vulnerabilities that could help us exploit them in the Metalsploit framework.

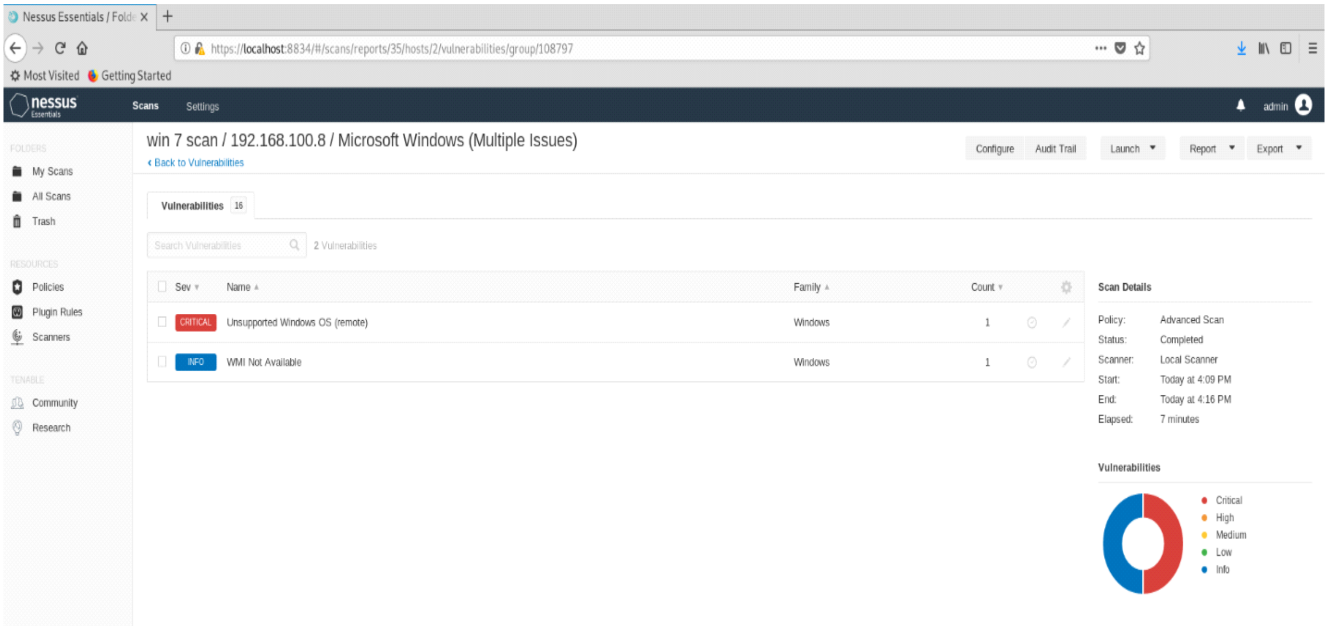


Figure 23 Scan with Nessus

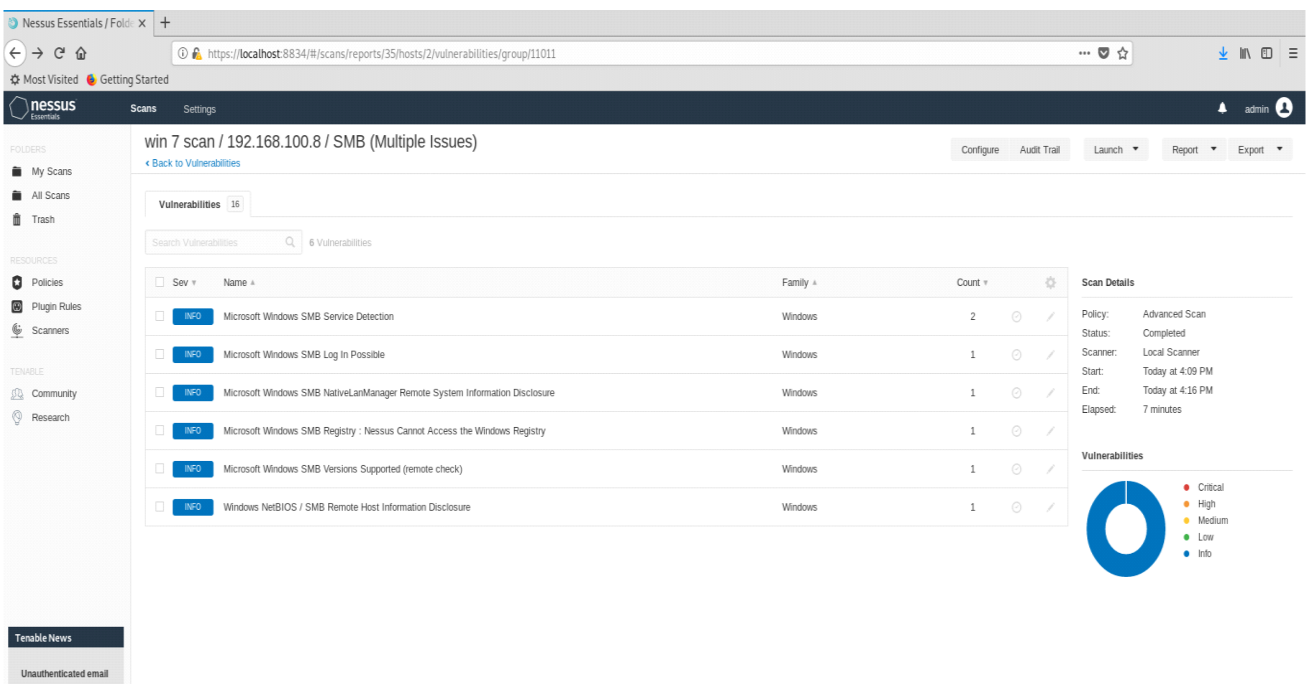


Figure 24 Scan with Nessus founded vulnerabilities

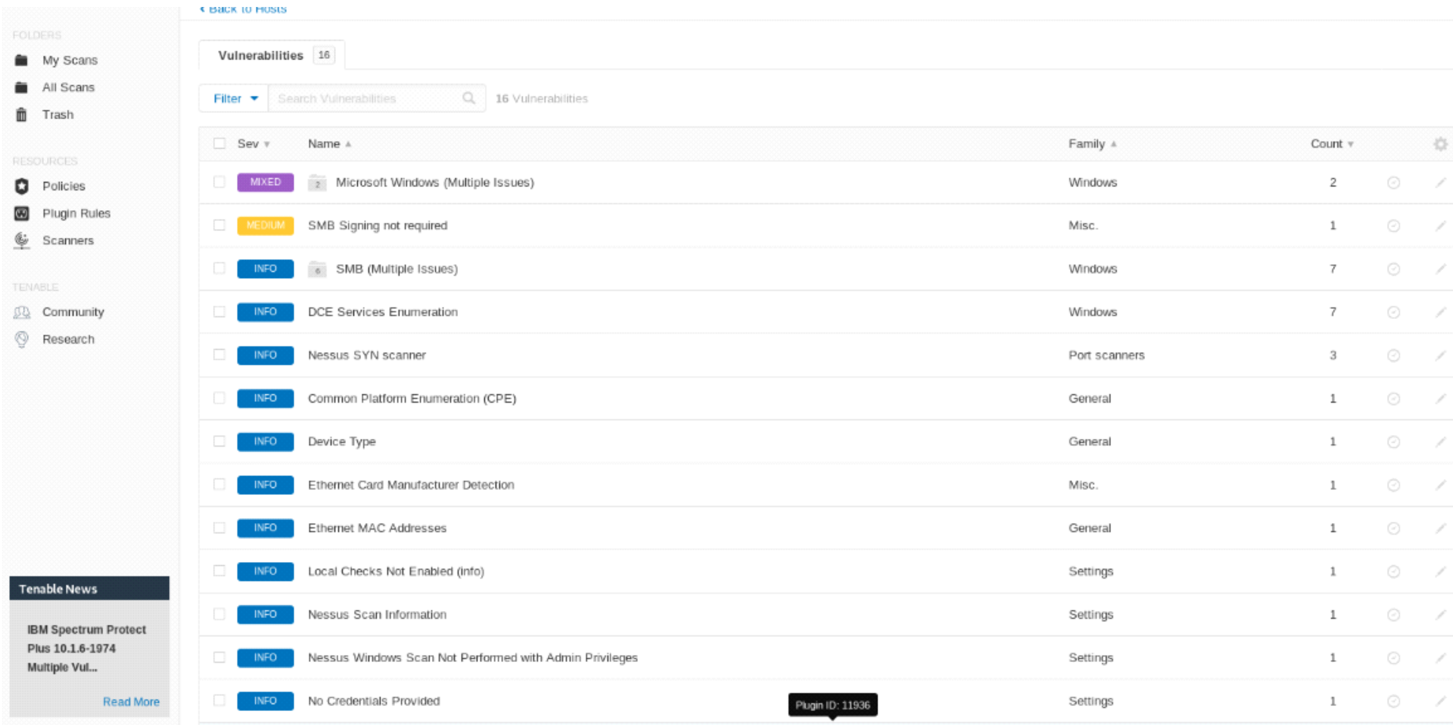


Figure 25 Scan with Nessus founded vulnerabilities

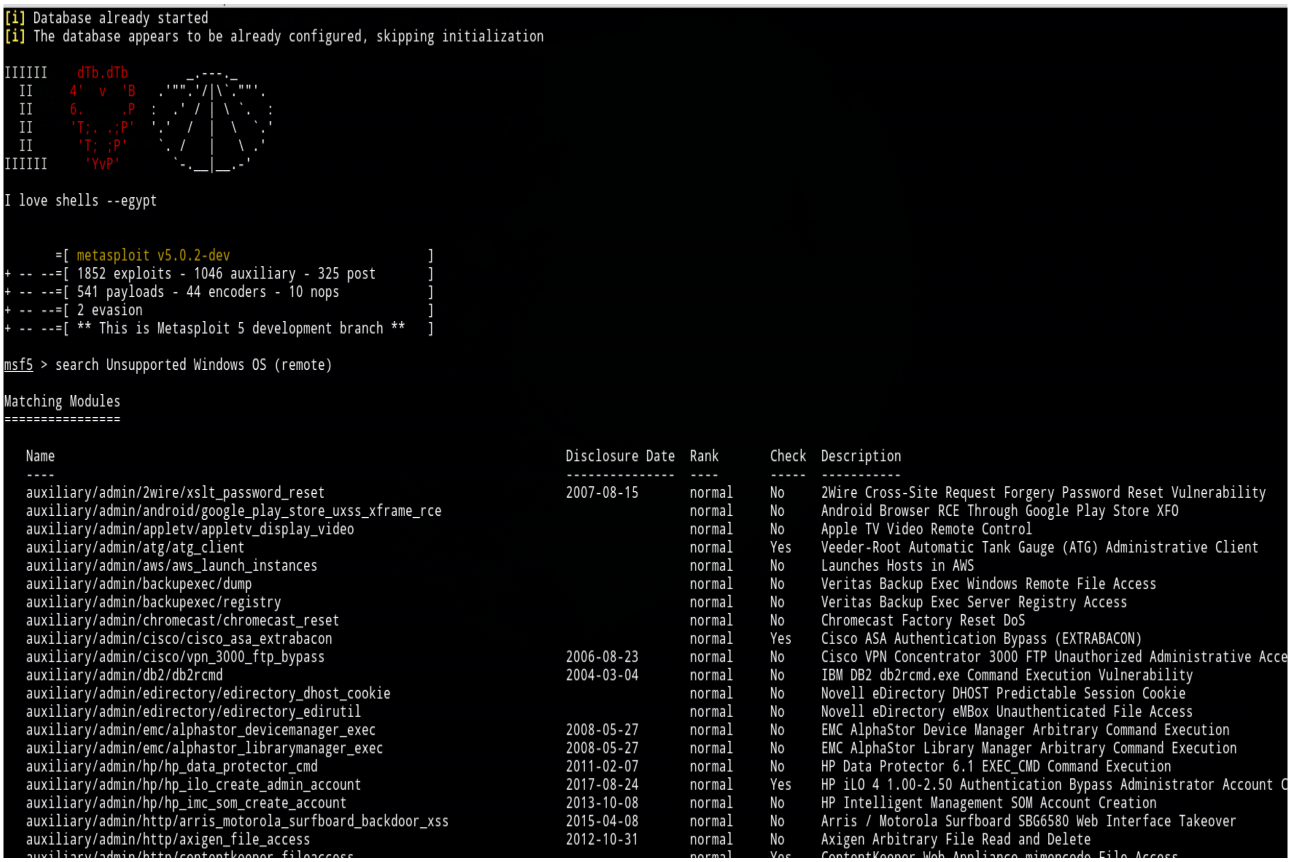


Figure 26 search command in Metalsploit

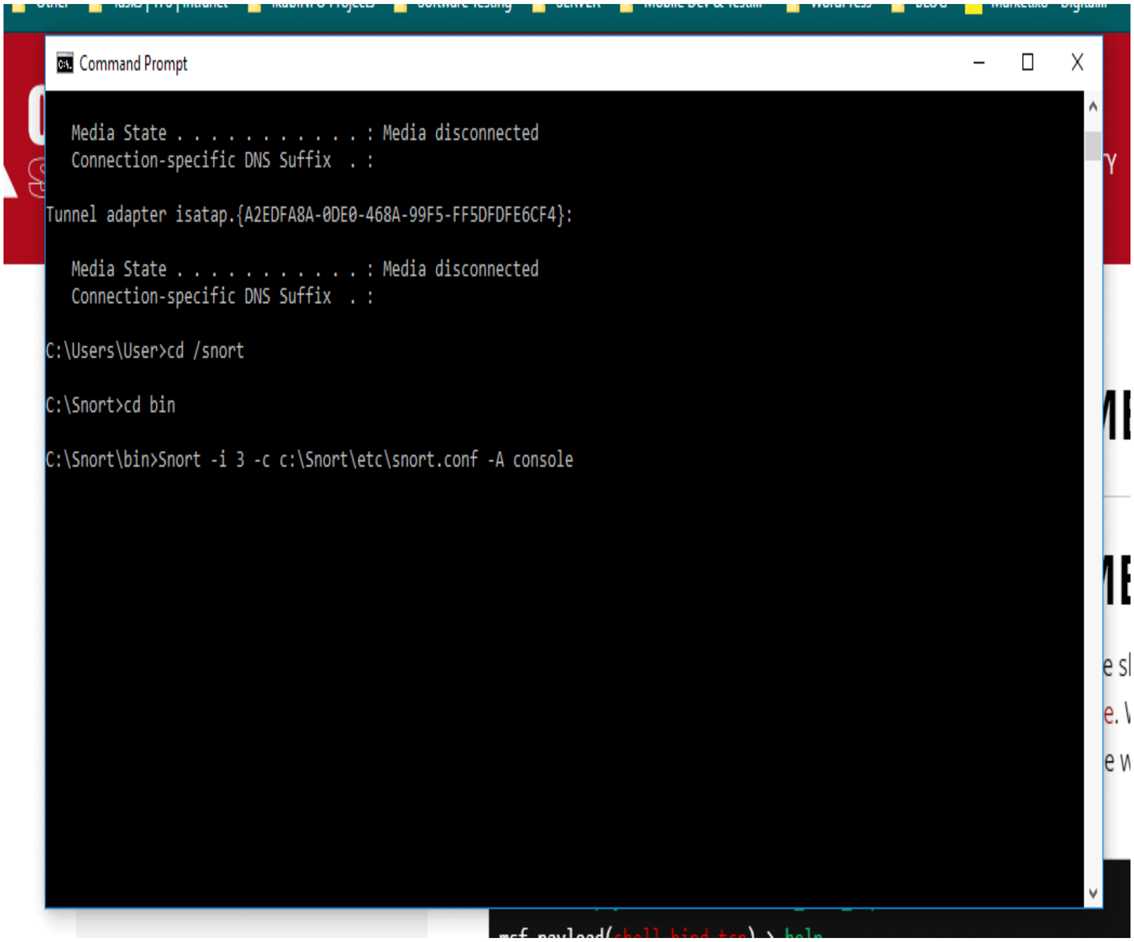


Figure 27 Running Snort for finding possible attacks

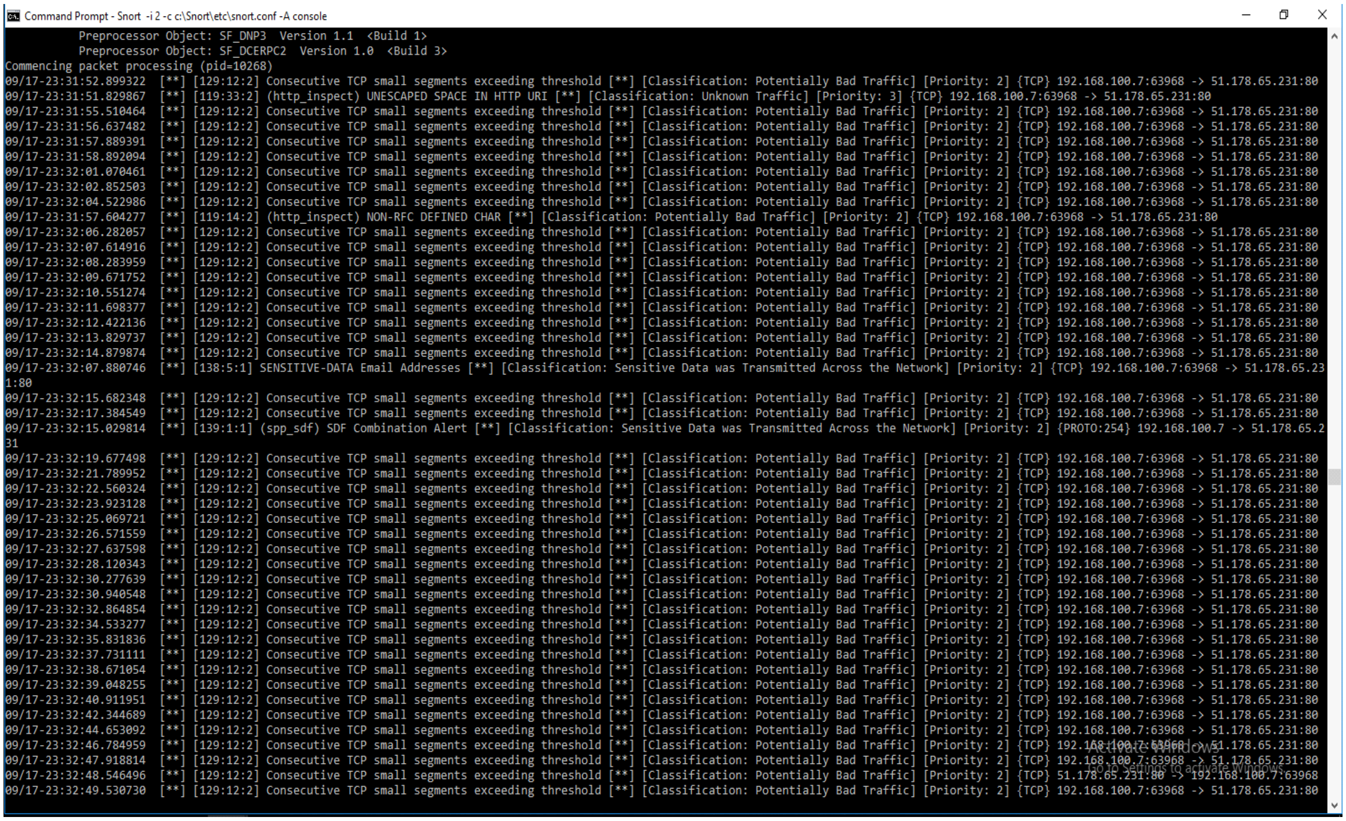


Figure 28 Logs collected from Snort

CHAPTER 5

CONCLUSIONS AND ANALYSIS

5.1 Conclusions

This chapter analyzes the insights introduced in the previous chapter. We discuss our test results and reveal insights between our research objectives and previously presented findings. In addition, this chapter looks at future exploration work that should be possible in similar zones, draws conclusions from findings and makes recommendations.

The information collected in the previous chapter makes it clear why integration with security operations (threat detection, alerting, analysis, and response) is so important. Based on the test results, the researchers were able to act as attackers within the network and send ICMP attacks similar to DoS attacks. The system had the ability to intercept these suspicious activities on the network and respond accordingly.

This paper is an analysis of his four IDS tools: Snort, Nmap, Nessus, and Metasploit.

They are as diverse as IDS, with comparable characteristics of architecture, network traffic, sensors, rule and recognition units, packet analyzers, etc. The goal of this work is to analyze the performance of the three proposed IDSs as a joint attack against Windows 7 machines. We are adjusting the environment accordingly.

From the completed experiments, we were able to obtain the following results:

• Snort was developed to meet the requirements of network intrusion detection systems. It turned out to be a small, highly adaptable and very good system to be used everywhere in networks big and small.

• Nmap is a powerful tool, but Nessus has many plugins and can do port scanning, so you don't need to use too many tools, so it's an easy replacement for Nessus.

• Nessus has a very interesting GUI and is very useful for first time users. I was able to exploit most of the vulnerabilities in Windows 7 computers. In some cases, I was even able to do the work assigned to Nmap.

• Metasploit is very powerful and in a completed experiment you can send packages and exploits to the created virtual machine. Very difficult for first-time users, but easy to master with practice.

5.2 Recommendations

The study recommends that security engineers upgrade their systems to Linux-based environments for similar arrangements. This is a result of the hardened nature of Linux-based operating systems. This meant researchers didn't have to worry about the security of their testbeds or networks.

Finally, the paper recommends that Snort rules pre-bundled with applications are a very good starting point for implementation outside of containers. However, security administrators are encouraged to make a special effort to create new rules that are tailored locally to the integration. This makes it easy to see what each alert means and why it was triggered. Apart from that, it is also important for security administrators to become familiar with existing Snort rules and understand their implications before invoking them.

5.3 Future research

This work focused on deploying Windows 7 in a network running in Virtual Box. Researchers recommend allowing devices to scan multiple networks or machines that may be on similar grids in the future. Additionally, the researchers included critical network hubs such as DNS, DHCP, email, and active directory in the snort.conf document for future implementations, allowing explicit rules to be set from open source networks, It is recommended to apply it explicitly. Prioritize these critical services. This means that each service has its own special rules associated with it and runs at the highest priority.

The study recommends that future work addressing this area should examine how multiple instances of Snort can exist on different LANs. This is claimed by similar organizations that submit (log submission and analysis) to information bases that these devices access. Research indicates that this implementation will facilitate further consolidation and provide benefits that come with this centralization.

REFERENCES

[1] T. Toth and C. Kruegel. “Evaluating the impact of automated intrusion response mecha- nisms”, in the 18th Computer Security Applications Conference (ACSAC02), Las Vegas,NV, 2002, p. 301C310.

[2] Balepin, S. Maltsev, J. Rowe, and K. Levit. “Using specification-based intrusion detec- tion for automated response,” in the 6th International Symposium on Recent Advances in Intrusion Detection (RAID) 2003, 2003.

[3] M. Jahnke, C. Thul, and P. Martini. “Graph based metrics for intrusion response measures in computer networks,” in 32nd IEEE Conference on Local Computer Networks (LCN), Dublin, Ireland, October 2007.

[4] C. Carver, J. M. Hill, and J. R. Surdu. A methodology for using intelligent agents to

pro- vide automated intrusion response. In Proceedings of the IEEE Systems, Man, and Cyber- netics Information Assurance and Security Workshop,West Point, NY, June 6-7, 2000, pages 110–116, 2000.

[5] W. Lee,W. Fan, M. Miller, S. J. Stolfo, and E. Zadok. Toward cost-sensitive modeling

for intrusion detection and response. J. Comput. Secur., 10(1-2):5-22, 2002.

[6] B. Foo, Y.-S.Wu, Y.-C. Mao, S. Bagchi, and E. H. Spaord. ADEPTS: Adaptive

intrusion response using attack graphs in an e-commerce environment. In Proceedings

of DSN, pages 508-517, 2005.

[7] Fwsnort, “Firewall snort,” [Website], Availalble:

HTTP://www.cipherdyne.org/fwsnort/

[8] Fwknop, “Firewall Knock Operator,” [Website], Availalble: HTTP://www.cipher-

dyne.org/fwknop/

[9] Psad, “Port scan attack detector,” [Website], Availalble: HTTP://www.cipherdyne.org/psad/

[10] C. Strasburg, "A framework for cost-sensitive automated selection of intrusion response," Master Thesis, 2009.

[11] N. Stakhanova "A framework for adaptive, cost-sensitive intrusion detection and response system" Phd thesis, 2007.

[12] Jacob Russell Lynch "Intrusion detection systems in wireless ad-hoc networks: detecting worm attacks" Master thesis, 2006.

[13] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and S. Zhou. Specifica- tion-based anomaly detection: a new approach for detecting network intrusions. In Pro- ceedings of the 9thACM conference on Computer and communications security, 2002.

[14] Snort, "Snort – the de facto standard for intrusion detection/prevention," [Website], 2006 Mar 23, [cited 2006 Apr 3], Available HTTP: http://www.snort.org

[15] N. Weaver, V. Paxson, S. Staniford, R. Cunningham, "A taxonomy of computer worms," in: Proceedings of the 2003 ACM Workshop on Rapid Malcode, 2003, pp. 11-18.

[16] S. Staniford, V. Paxson, N. Weaver, "How to own the Internet in Your Spare Time,"

Pro- ceedings of the 11th USENIX Security Symposium, 2002.

[17] "Tcp dump" [Online document]. Available:http://www.tcpdump.org/

[18] "Tinyxml" [Online document]. Availalble:http://www.grinninglizard.com/tinyxml/

[19] "Naval research laboratory" [Online document]. Available:http://cs.itd.nrl.navy.mil/ work/olsr/index.php

[20] "OSSEC" [Online document]. Available:http://www.ossec.net/