Week 1: Network Security Monitoring

Date: 26/10/2024

1. Introduction to Network Security Monitoring:

NSM forms part of cybersecurity, whereby professionals exercise the value of network traffic to find out suspicious activities that may indicate some form of security threats. Here is a breakdown:

1. Network Traffic Monitoring

Network traffic monitoring means continuous surveillance of data as it moves across a network; actually, packets—small units of data that travel over a network—are monitored. In so doing, unusual activities or patterns indicative of a potential security threat can be noticed by a security team, such as a cyberattack.

2. Key Concepts in Network Security Monitoring

To get a feel for network security monitoring, several key concepts bear becoming familiar with:

Packet: A small unit of data transmitted across a network. Each packet includes information such as sender and receiver addresses, the data payload, and errorchecking information. Therefore, packet analysis is one of the fundamentals of the anomaly detection process, since the appearance of unusual or unexpected behavior from packets may indicate malicious activity.

Protocols: Protocols are sets of rules through which data is exchanged over a network. Examples include HTTP (used for websites), FTP (used for file transfers), and SMTP (used for email). Each protocol has a unique behavior pattern; hence, monitoring these patterns helps identify normal versus abnormal activity. For instance, if there is an unusual amount of FTP requests emanating from a single computer, that might signal an attempt to steal files.

Firewalls: These, in a way, act like sentinels between the internal network and the outside world. Firewalls segregate incoming or outgoing network traffic by analyzing data packets that fit predetermined security rules. Firewalls are usually the first layer of defense for any network against unauthorized access. They may not inspect packet content in depth, but they can block known unauthorized IP addresses or suspicious protocols from reaching the network.

IDS: Intrusion Detection Systems are specialized tools, whose role is to monitor the traffic of networks for suspicious or malicious activity. IDS works either by matching patterns in network traffic against known attack patterns or by identifying traffic as abnormal, which could be a signal of an impending attack. Mainly, there are two types of IDS:

SignatureBased IDS: Such an intrusion detection system finds the traces of wellknown attack patterns or "signatures." It is good for detecting known types of threats but possibly will not detect unknown ones.

AnomalyBased IDS: Departs from normality of network behavior. This may be useful to detect new or unknown threats but could result in false alarms if the normal activities change too frequently.

3. How Anomaly Monitoring of Network Traffic is Done

Network traffic anomaly detection copes with the identification of patterns within data packets and protocols. That mostly means that security teams or automated systems would look for patterns which go against the grain of the usual look and feel, such as:

Increase in traffic due to a particular IP address.

Unusual patterns of requests, such as a huge amount of login attempts within a very short period of time.

Protocol behavior not being one of expected use, such as inappropriate use of a nonstandard protocol.

An alert is then issued to the security team; investigation of the incident is subsequently carried out. Some of these alerts, however, emanate from automated means such as IDS, which monitors traffic for realtime threats.

4. Why Network Security Monitoring Matters

Monitoring network traffic and detecting anomalies is crucial for:

Threats are caught early stage: It is at an early stage that unearns exceptions can be prevented before they lead to potential damage.

Data Leak Protection: The flow of data in NSM is tracked to prevent sensitive data leakage.

Building Resilience: The security team will analyze the incidents in order to build better defenses over time and deter similar types of attacks in the future.

Network security monitoring is one of the most important features of cybersecurity, with investigative elements blocking organizations from different manners of attack and moving swiftly against emerging threats.

2. Setting Up the Environment:

Task: Set up a lab environment for network monitoring.

Description: Install Wireshark and Snort. Configure a virtual or physical machine to simulate a network environment.

Setting up a lab environment for network monitoring is a fantastic way to get handson experience with monitoring tools like Wireshark and Snort. Below are the steps to set up a basic lab environment that will allow you to simulate a network, monitor traffic, and analyze it for anomalies.

1. Prepare the Environment

Choose Your Virtualization Software: For lab purposes, virtual machines (VMs) are ideal. Install a virtualization software like VirtualBox or VMware Workstation. These will allow you to create multiple VMs on a single physical machine, simulating a network.

Select Operating Systems:

Use Linux for network monitoring tools (e.g., Ubuntu or CentOS), as most tools are optimized for Linux.

Use Windows for simulating client activity, which is common in realworld networks.

2. Set Up Virtual Machines

Create a Monitoring VM:

Install a Linux distribution (such as Ubuntu or CentOS) on a VM to act as your primary network monitoring station.

Ensure that this VM has at least 2GB RAM and 2 CPU cores for smooth operation.

Create a Target VM:

Set up another VM (could be Windows or Linux) that will act as a target for generating traffic. You'll use this to test the monitoring setup by sending pings or other types of traffic.

3. Install Wireshark

Download and Install Wireshark:

Open a terminal on your monitoring VM and install Wireshark. For Ubuntu, you can use:

bash

sudo apt update

sudo apt install wireshark y

Grant the necessary permissions by allowing nonroot users to capture packets, if prompted.

Configure Wireshark:

Open Wireshark and select the network interface connected to your virtual network.

Start capturing packets to observe live traffic. You can filter by protocol (like HTTP, ICMP, etc.) for specific types of traffic.

4. Install Snort

Download and Install Snort:

Snort can also be installed from the command line. For Ubuntu:

bash

sudo apt install snort y

During installation, it will prompt you to configure some network settings. Input the IP range for your simulated network.

Configure Snort Rules:

Snort operates on a set of predefined rules that tell it what to watch for. By default, it will come with a set of basic rules, but you can find more rules on [Snort's official site](https://www.snort.org/).

Create a new directory to store custom rules. For example:

bash

sudo mkdir /etc/snort/rules/local.rules

Add a simple custom rule to detect ICMP (ping) traffic, which will generate an alert for every ping request:

bash

alert icmp any any > any any (msg:"ICMP test"; sid:1000001; rev:1;)

Test Snort:

Run Snort in network intrusion detection mode:

bash

sudo snort A console q c /etc/snort/snort.conf i eth0

From the target VM, try pinging the monitoring VM to see if Snort generates an alert based on your ICMP rule.

5. Simulate Network Traffic

To fully test your setup, generate various types of traffic:

Ping from one VM to another.

Use a web browser on the target VM to access external websites.

Run a file transfer using a protocol like FTP if possible.

Monitor the traffic in Wireshark and observe Snort's output in the console. You should see alerts triggered by any custom rules you've set in Snort.

6. Review and Analyze Data

Wireshark: Look at the details of captured packets, noting protocols, source/destination IPs, and packet contents. Apply filters to look for specific types of traffic.

Snort: Review the alerts in the Snort console or logs, seeing what triggered them and adjusting rules as needed.

7. Optional: Set Up a Central Logging Tool (e.g., Splunk or ELK)

Integrate with a Security Information and Event Management (SIEM) tool to centralize logs from Wireshark and Snort. This can help in organizing alerts and analyzing trends.

This setup provides a solid foundation for network monitoring. You'll get practice

capturing and analyzing network traffic with Wireshark, detecting and responding to network anomalies with Snort, and refining your rules as you experiment with different types of traffic.

3. Packet Capture with Wireshark:

Task: Capture and analyze network traffic using Wireshark.

Description: Focus on capturing HTTP, DNS, and ICMP packets. Explore Wireshark's interface for filters and statistics.

Capturing and analyzing network traffic with Wireshark is a great way to understand how different protocols work. Here's a stepbystep guide focused on capturing and analyzing HTTP, DNS, and ICMP packets and using Wireshark's interface to filter and analyze this data.

1. Open Wireshark and Select an Interface

Launch Wireshark and choose the network interface you'll be capturing traffic on (e.g., eth0, wlan0, etc.).

If you're on a virtual machine, select the interface that connects to the virtual network (usually a "Bridged" or "NAT" network).

2. Start Capturing Traffic

Click the Start Capturing Packets button (the blue shark fin icon) to begin capturing all network traffic on the chosen interface.

As traffic flows, Wireshark will display packets in realtime.

3. Capture and Filter Specific Protocols

To focus on HTTP, DNS, and ICMP traffic, you can use filters in Wireshark's interface to isolate packets from each protocol:

HTTP Traffic:

Open a web browser and visit any website (preferably a nonsecure HTTP site, as many websites now use HTTPS).

In Wireshark, type http in the Filter bar and press Enter. This filter will show only HTTP packets, including requests and responses.

Look for packets with details such as GET or POST requests. Click on these packets to explore headers and content.

DNS Traffic:

Type dns in the Filter bar to display only DNS packets. DNS is responsible for translating domain names (like www.example.com) into IP addresses.

Open your browser again and type a URL into the address bar, which should generate DNS packets as your device queries for the website's IP.

Look at the Queries and Responses fields in the DNS packets to see which IPs are resolved for specific domain names.

ICMP Traffic:

ICMP is the protocol used for ping requests. Type icmp in the Filter bar to isolate ICMP packets.

Open a terminal or command prompt on another device or VM and ping an IP address on the network. The ping command should generate ICMP Echo Requests and Echo Replies.

In Wireshark, you'll see these ICMP packets, which contain details about the request and reply, including the roundtrip time.

4. Explore Packet Details

Packet List Pane: Shows a list of all captured packets matching your filter criteria.

Packet Details Pane: Displays the protocol layers for each packet, from the physical layer up to the application layer. Here, you can see packet headers, source/destination IPs, protocolspecific fields, and more.

Packet Bytes Pane: Shows the raw hexadecimal data of the selected packet. This is useful for indepth analysis.

5. Use Wireshark's Statistics Tools

Wireshark provides several builtin statistics and analysis tools to help you get insights into your network traffic.

Protocol Hierarchy: Go to Statistics > Protocol Hierarchy to see a breakdown of all protocols captured in the session, including HTTP, DNS, and ICMP. This provides an overview of the traffic distribution.

Conversations: Select Statistics > Conversations to see conversations between IP addresses over the network. This is useful for identifying the main sources and

destinations of traffic.

IO Graphs: Navigate to Statistics > IO Graphs to create visual representations of network activity over time. You can add filters for specific protocols (like http, dns, or icmp) to see traffic patterns for each one.

6. Analyze Captured Traffic

HTTP Analysis: Look at specific HTTP requests (GET, POST) and responses to see how data is exchanged between a client and server. Analyze headers like Host, UserAgent, and ContentType.

DNS Analysis: Study DNS query and response times, the domain names requested, and the IP addresses returned. Pay attention to any DNS error codes, as they can indicate issues with name resolution.

ICMP Analysis: Examine roundtrip times for pings, which can help in assessing latency and connectivity issues. ICMP is also commonly used in scanning and reconnaissance, so it's good to recognize any unusual patterns.

7. Save and Export Captures (Optional)

After you've finished your capture session, you can save the data for future analysis. Go to File > Save As to store the capture in a .pcap file.

You can also export specific packets or data in CSV or other formats for external analysis.

This setup gives you a handson understanding of how HTTP, DNS, and ICMP protocols function on a network and how Wireshark can be used to monitor, filter, and analyze this traffic. Over time, you'll become familiar with normal traffic patterns, making it easier to spot anomalies or signs of potential security issues.

4. Analyzing Captured Traffic:

Task : Analyze packets for anomalies.

Description : Identify suspicious activities and document findings like unusual IPs or ports using Wireshark.

Analyzing packets for anomalies with Wireshark is an essential step in identifying suspicious activities and potential security threats on a network. Here's a structured approach for identifying unusual patterns, such as strange IP addresses, uncommon port usage, or unexpected protocol behaviors.

1. Start by Capturing or Opening Network Traffic

Begin a new capture or open a preexisting capture file (.pcap) that contains network traffic to analyze.

Make sure you capture or analyze enough traffic to establish a baseline and detect any deviations.

2. Identify Common Indicators of Suspicious Activity

Unusual IP Addresses: Look for traffic from IP addresses outside of your expected network range or addresses that are known to be malicious (e.g., foreign, unrecognized, or blacklisted IPs).

Strange Port Activity: Check for uncommon ports being used for common services. For example, HTTP traffic typically uses port 80, and HTTPS uses port 443. Suspicious activity might include:

Common services running on nonstandard ports.

Traffic on ports commonly associated with known threats (e.g., port 4444, often used in exploits).

High Volume of Specific Protocols or Ports: A sudden spike in certain types of traffic (e.g., DNS, ICMP) can indicate abnormal behavior. For instance, multiple DNS requests within a short period could suggest a DNS tunneling attempt.

Anomalous Protocol Behavior: Look for uncommon or unexpected protocol usage, such as FTP or Telnet traffic in a network that doesn't typically use these protocols.

3. Set Filters in Wireshark for Common Anomalies

Use Wireshark's filters to isolate and examine suspicious traffic patterns.

Unusual IP Addresses:

Filter for external IP addresses not belonging to your network range. For instance, if your network uses the 192.168.1.0/24 range, you can use:

plaintext

!(ip.src == 192.168.1.0/24) && !(ip.dst == 192.168.1.0/24)

Identify any IPs flagged as potentially malicious by researching them in public threat intelligence databases.

Uncommon Port Usage:

Filter for unusual ports using tcp.port != 80 && tcp.port != 443 && tcp.port != 22, which will show traffic on ports other than common HTTP, HTTPS, and SSH ports.

Focus on ports associated with suspicious or unexpected services, like tcp.port == 23 (Telnet) or tcp.port == 445 (SMB).

ProtocolSpecific Filters:

DNS: dns filter to focus on DNS traffic. Look for a high number of DNS queries from a single IP, which can indicate DNS tunneling.

ICMP: icmp filter to focus on ICMP traffic. An unusually high number of pings can indicate a potential reconnaissance attempt.

HTTP: Use the http filter to examine HTTP requests, particularly looking at headers and payloads for potential signs of webbased attacks, such as suspicious URLs or requests to uncommon domains.

4. Analyze Suspicious Packets in Detail

Source and Destination IPs: In each packet, look at the Source IP and Destination IP. Document any IP addresses that seem out of place or are from regions associated with cyber attacks (use threat intelligence services like VirusTotal or AbuseIPDB to check suspicious IPs).

Ports and Services: Check the Source Port and Destination Port for services running on unusual ports. For example, if HTTP traffic is running on port 8080 or 1234, it might warrant further inspection.

Content and Payload: Inspect the Packet Details pane to review packet payloads. Look for signs of:

Malicious Commands: For HTTP, this might include suspicious GET or POST requests with strange parameters.

Encoded Payloads: Attackers often encode malicious payloads to evade detection.

Repeated Queries or Scans: Large numbers of ICMP packets or DNS queries from a single IP could indicate scanning or an attempt to exhaust resources.

5. Use Wireshark's Statistics and Analysis Tools

Protocol Hierarchy: Go to Statistics > Protocol Hierarchy to view a summary of protocols in the capture. This can reveal if any protocol is being used more than expected (e.g., an excessive amount of DNS or ICMP traffic).

Conversations: Select Statistics > Conversations to analyze which IP addresses are communicating with each other. This tool helps identify any IP addresses making an unusual number of connections.

Endpoint Analysis: Go to Statistics > Endpoints to see all unique IP addresses in the traffic. This allows you to quickly identify outliers and unusual IP addresses.

6. Document Your Findings

Create a report detailing any suspicious activities, including:

Anomalous IP Addresses: List any IPs that appeared out of the ordinary or were flagged as potential threats.

Strange Port Activity: Note any unusual ports that were active and the types of traffic observed on them.

Protocol Anomalies: Describe any excessive use of specific protocols (like DNS or ICMP) and the associated risks.

Packet Contents: Document any unusual packet contents or payloads, such as encoded data or suspicious parameters in HTTP requests.

Sample Documentation Entry:

Anomaly Detected: High volume of ICMP requests

Description: Numerous ICMP echo requests originating from IP 192.168.1.50.

Action: ICMP activity recorded and flagged as suspicious due to its high frequency.

Potential Threat: Likely reconnaissance or denialofservice attempt.

Anomaly Detected: Unusual DNS traffic

Description: IP 10.0.0.10 was generating frequent DNS requests to unknown domains within a short timeframe.

Action: DNS traffic reviewed; domain names noted for further investigation.

Potential Threat: Possible DNS tunneling.

Using this approach, you'll be able to identify and document potential network threats, which is key for building skills in network security analysis and response.

5. Introduction to Snort :

Task : Install and configure Snort for intrusion detection.

Description : Write basic Snort rules to detect common attacks such as port scans.

Installing and configuring Snort for intrusion detection, as well as writing basic rules, will help you detect common network attacks like port scans. Here's a stepbystep guide to get Snort set up and running with custom rules.

1. Install Snort

Update the System: Start by updating your package list.

bash

sudo apt update

Install Snort: You can install Snort directly from the package manager on Ubuntu or Debianbased systems.

bash

sudo apt install snort y

During installation, you may be asked to configure the network range for Snort to monitor. Enter the IP range of your network (e.g., 192.168.1.0/24).

2. Verify Installation

Check if Snort is installed correctly by running:

bash

snort V

This command should output Snort's version, confirming the installation.

3. Basic Configuration

Edit the Configuration File: The main configuration file is located at /etc/snort/snort.conf.

Set HOME_NET: In snort.conf, set the HOME_NET variable to match your network's IP range. This tells Snort which IPs to monitor.

plaintext

var HOME_NET 192.168.1.0/24

Include Rule Files: Ensure that Snort is configured to include local rule files. Near the bottom of snort.conf, look for the line that includes local rules:

plaintext

include $RULE_PATH/local.rules

If local.rules is not specified, add this line.

4. Create and Edit the local.rules File

The local.rules file is where you'll define custom Snort rules for detecting specific types of attacks. Open or create this file:

bash

sudo nano /etc/snort/rules/local.rules

5. Write Basic Snort Rules for Common Attacks

Snort rules follow a general syntax that defines the action to take, protocol, source IP/port, destination IP/port, and additional options to specify conditions.

Port Scan Detection: This rule detects multiple connection attempts on different ports from the same IP address (a common indicator of a port scan).

plaintext

alert tcp any any > $HOME_NET any (msg:"Port scan detected"; flags:S; threshold:type both, track by_src, count 20, seconds 60; sid:1000001; rev:1;)

Explanation:

alert: Tells Snort to generate an alert.

tcp any any > $HOME_NET any: Monitors any source IP and port to any destination in HOME_NET on any port.

flags:S: Detects SYN packets, which are often used in port scans.

threshold: Triggers an alert if 20 SYN packets are received from the same source IP within 60 seconds.

ICMP Ping Flood Detection: This rule triggers if an IP sends multiple ICMP echo requests (pings) to your network within a short period.

plaintext

alert icmp any any > $HOME_NET any (msg:"ICMP flood detected"; itype:8; threshold:type both, track by_src, count 10, seconds 1; sid:1000002; rev:1;)

Explanation:

icmp any any > $HOME_NET any: Monitors any source IP sending ICMP requests to any destination in HOME_NET.

itype:8: Specifies ICMP type 8 (echo request).

threshold: Generates an alert if 10 pings are received from the same source in 1 second.

Detecting HTTP Traffic to Unusual Ports: This rule detects HTTP traffic on ports other than the default port 80, which could indicate potential malicious activity.

plaintext

alert tcp $HOME_NET any > any !80 (msg:"Unusual HTTP traffic"; content:"GET"; http_method; sid:1000003; rev:1;)

Explanation:

tcp $HOME_NET any > any !80: Detects HTTP requests from any source in HOME_NET to any destination that isn't on port 80.

content:"GET"; http_method: Filters only GET requests.

6. Test Your Snort Configuration

Run Snort in Test Mode: Check for syntax errors in the configuration file and rules.

bash

sudo snort T c /etc/snort/snort.conf

Run Snort in IDS Mode: Start Snort in intrusion detection mode, specifying the configuration file and network interface.

bash

sudo snort A console q c /etc/snort/snort.conf i eth0

A console outputs alerts directly to the console.

q runs Snort in quiet mode to reduce nonessential output.

7. Generate Traffic to Test Rules

Port Scan: Use a tool like nmap to perform a port scan on the Snort machine's IP from another device. For example:

bash

nmap sS 192.168.1.5

ICMP Flood: Use the ping command to send multiple pings to your Snort machine:

bash

ping f 192.168.1.5

HTTP Request on Unusual Port: Use curl to make an HTTP request to a nonstandard port:

bash

curl http://192.168.1.5:8080

8. Review Alerts and Logs

If configured correctly, Snort will output alerts to the console or to a log file (usually located in /var/log/snort). Look for alerts that match the custom rules you created, which should trigger based on the simulated attacks.

By following these steps, you'll have a Snort setup capable of detecting port scans, ICMP floods, and unusual HTTP traffic. Adjust rules and thresholds as needed to reduce false positives and improve detection accuracy.

6. Monitoring and Reporting :

Task : Monitor network traffic using Snort and generate reports.

Description : Investigate Snort alerts and create a brief report of detected threats and their impact.

Monitoring network traffic using Snort and generating reports is crucial for identifying threats and understanding their potential impact. Here's how you can configure Snort to monitor network traffic, investigate alerts, and create a brief report summarizing detected threats:

1. Configure Snort for Logging Alerts

First, ensure Snort is set up to log alerts in a readable format (such as unified2, syslog, or text format).

Edit Snort Configuration File: Open the Snort configuration file (/etc/snort/snort.conf) and configure the logging settings. If you're logging alerts to a file, make sure the following line is enabled in snort.conf:

plaintext

output alert_fast: /var/log/snort/alert

This will generate a simple, human readable alert log.

Other Logging Methods:

output log_tcpdump: Logs packets in TCPDump format.

output syslog: Sends alerts to the system's syslog.

output unified2: A binary format that can be processed by other tools like Barnyard2.

2. Start Snort in IDS Mode

Run Snort in Intrusion Detection System (IDS) mode to capture and analyze network traffic. Use the following command to run Snort:

bash

sudo snort A fast c /etc/snort/snort.conf i eth0

A fast: Outputs alerts in a humanreadable format.

c /etc/snort/snort.conf: Specifies the configuration file.

i eth0: Indicates the network interface (replace eth0 with your actual interface).

3. Simulate Network Traffic for Testing

Simulate network traffic or attacks (like a port scan or ICMP flood) to generate alerts. For example:

Port Scan (using nmap):

bash

nmap sS 192.168.1.100

ICMP Flood (using ping):

bash

ping f 192.168.1.100

4. Investigate Snort Alerts

Once Snort has detected suspicious activity, alerts will be written to the log file, such as /var/log/snort/alert. Open this file to investigate the alerts:

View Alerts in Console: If running Snort in fast mode, alerts will be displayed in the console in realtime.

Examine the Alert Log: Use cat or less to examine the alert log file. For example:

bash

less /var/log/snort/alert

Each alert will include:

Timestamp: When the alert was triggered.

Source and Destination IPs: The IPs involved in the suspicious activity.

Alert Type: A brief description of the detected activity (e.g., "Port scan detected", "ICMP flood").

Rule SID: The unique identifier for the Snort rule that triggered the alert.

Example of a log entry:

[] [1:1000001:1] Port scan detected []

[Classification: Attempted Information Leak] [Priority: 2]

03/1412:23:15.123456 192.168.1.100 > 192.168.1.50

TCP TTL:64 TOS:0x0 ID:12345 IpLen:20 DgmLen:60

SYN Seq: 0x12345678 Ack: 0x0 Win: 0x200

TCP Options: (0x02) Max Seg Size: 1460

5. Analyze the Alerts

Investigate each alert based on the rule that triggered it. Focus on understanding the nature of the threat and its potential impact:

Port Scan: A port scan typically means an attacker is probing your system to find open ports for exploiting services. It's often the first step in a larger attack.

Impact: A successful port scan can indicate that an attacker is preparing to launch a more targeted attack (e.g., exploiting a vulnerability in an open service).

Mitigation: Blocking or ratelimiting port scans using firewalls or intrusion prevention systems (IPS).

ICMP Flood: An ICMP flood can be a denialofservice (DoS) attack, where an attacker floods your network with pings, overwhelming network resources and potentially causing a service disruption.

Impact: Disruption of network services, leading to degraded performance or unavailability.

Mitigation: Ratelimiting ICMP requests or configuring firewalls to block excessive pings.

Unusual HTTP Traffic: Anomalous HTTP traffic on nonstandard ports (e.g., port 8080) may indicate an attempt to access web servers or other services in a way that wasn't anticipated.

Impact: Unauthorized access to web services or application vulnerabilities being targeted.

Mitigation: Blocking unusual HTTP requests and ensuring web services are secured and monitored.

6. Generate Reports on Detected Threats

Once you've reviewed the Snort alerts, you can document the detected threats and their potential impacts. A sample report structure could be:

Sample Report

Network Traffic Monitoring Report

Date: [Insert Date]

Objective: Monitor network traffic using Snort and investigate any detected threats.

1. Port Scan Detected

Timestamp: [Insert Time]

Source IP: 192.168.1.100

Destination IP: 192.168.1.50

Port(s) Scanned: Multiple ports (e.g., 22, 80, 443)

Alert Type: Port scan detected

Severity: Medium

Impact: The port scan indicates that the source IP is probing the system for open ports. This could be a prelude to a targeted attack.

Mitigation: Implement firewall rules to block or ratelimit port scans.

2. ICMP Flood Detected

Timestamp: [Insert Time]

Source IP: 192.168.1.200

Destination IP: 192.168.1.50

Type of Attack: ICMP flood

Severity: High

Impact: The ICMP flood could disrupt network services by overwhelming the target with ping requests, potentially causing denial of service.

Mitigation: Block excessive ICMP requests or configure ratelimiting.

3. Unusual HTTP Traffic on Port 8080

Timestamp: [Insert Time]

Source IP: 192.168.1.150

Destination IP: 192.168.1.50

Port: 8080

Alert Type: Unusual HTTP request

Severity: Low

Impact: Access to a web service on an unexpected port could indicate unauthorized access attempts or potential vulnerabilities in the service.

Mitigation: Block HTTP traffic on nonstandard ports and ensure that web services are secured.

7. Concluding Actions

Review Snort logs regularly to detect ongoing or new threats.

Configure automated alerts and monitoring tools for better detection of suspicious activity.

Update Snort rules to cover emerging attack vectors.

By following these steps, you'll be able to effectively monitor network traffic using Snort, investigate alerts, and create detailed reports to track and mitigate potential threats.