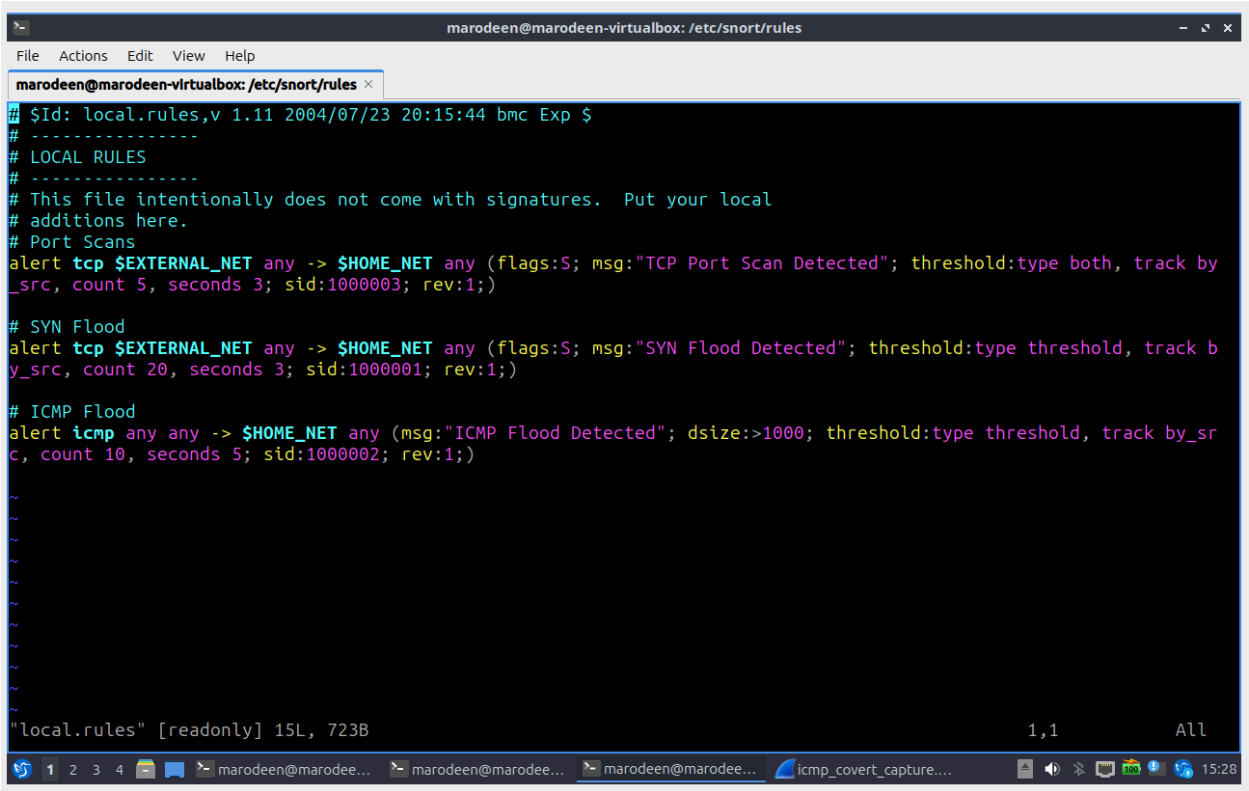
Marodeen Shamonei
14 November 2024
DAT 505 - Ethical Hacking

Assignment 2 - Bypassing Snort IDS

Snort Configuration

Two virtual machines were set up using Virtualbox, both running Lubuntu 24.04.1. One machine was designated the attacker machine and the other the victim machine. On the victim machine, Snort was configured with the following rules to detect basic network attacks like SYN Floods, port scans, and ICMP flooding:



Evasion Techniques

Packet Fragmentation for Evasion -

Usually, SYN Floods work by sending a large amount of SYN packets to ‘flood’ a target system. Using its signature based detection, Snort would be able to detect such an attack. However, if the packets are fragmented, then the pattern is broken. Each packet would not necessarily have a SYN flag, which Snort would be looking for. In this way, fragmenting packets helps bypass Snort’s detection.

Timing-Based Evasion -

Once again, Snort would normally detect a port scan by pattern - it sees a large amount of connection attempts to ports in rapid succession. Timing based evasion is a very basic technique,

it essentially introduces a random delay between the scanning of each port. As such, Snort is not able to detect any kind of malicious pattern. It would be unable to distinguish the slow port scan from genuine connections to the same ports.

Obfuscating Attack Payloads -

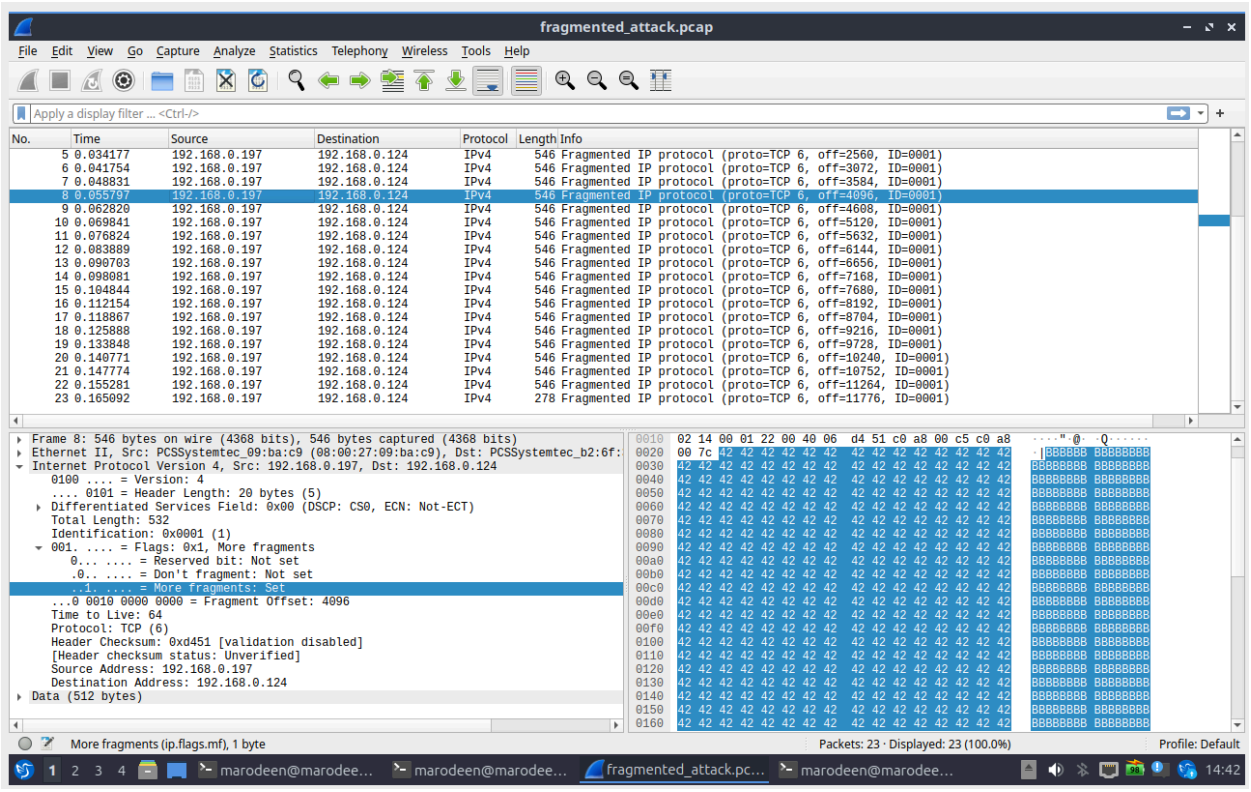
In this attack, we disguised a malicious reverse shell within an HTTP request. HTTP requests are extremely common and Snort generally sees them as harmless. Malicious code can be hidden within the different segments of the request which Snort has a difficult time distinguishing between a legitimate request. Given Snort's signature based detection, there is no real 'signature' to this type of attack, no type of pattern that look for to immediately identify an attack.

ICMP Tunneling for Covert Communication -

This method is similar to the above. Harmful code is hidden within a protocol that is usually considered innocuous. The ICMP protocol is very common and used regularly to ping devices. As a result, Snort does not prioritize it. However, malicious code can be hidden within different contents of the ICMP packet and sent to the target device.

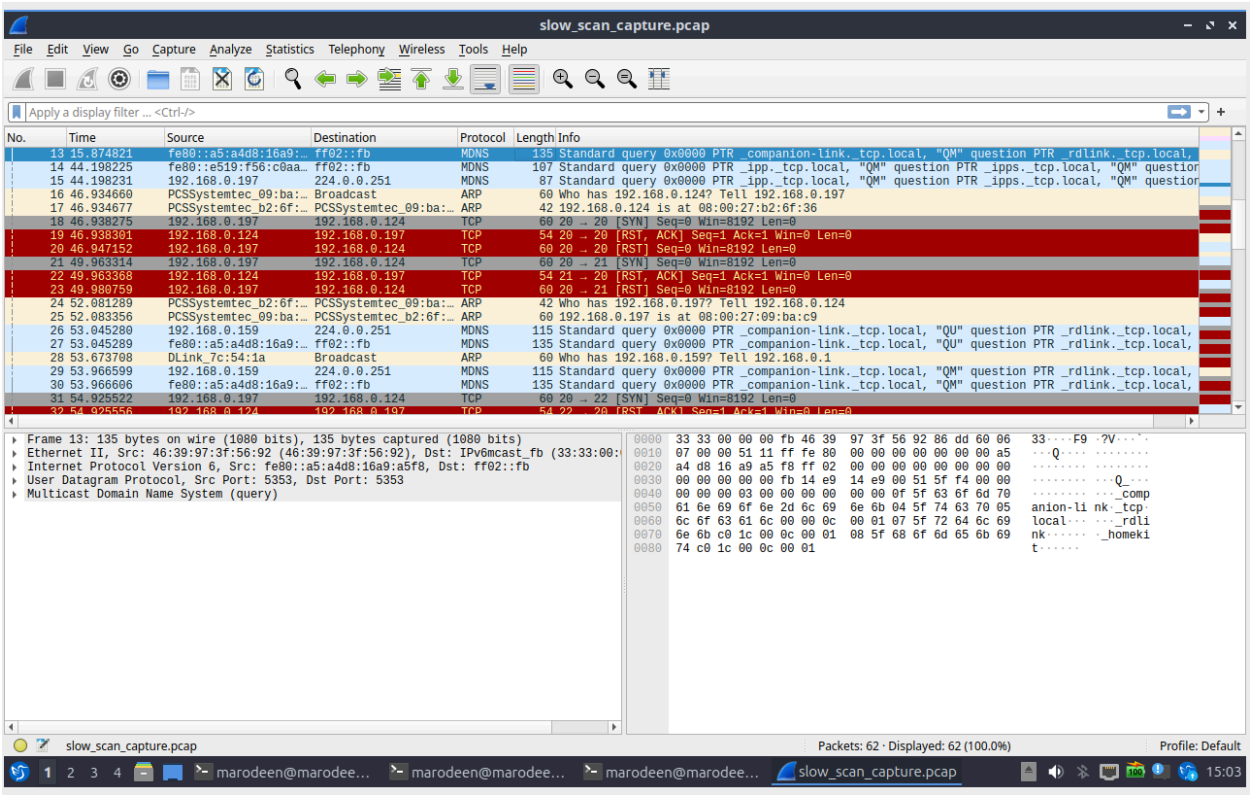
Traffic Captures

Packet Fragmentation for Evasion -



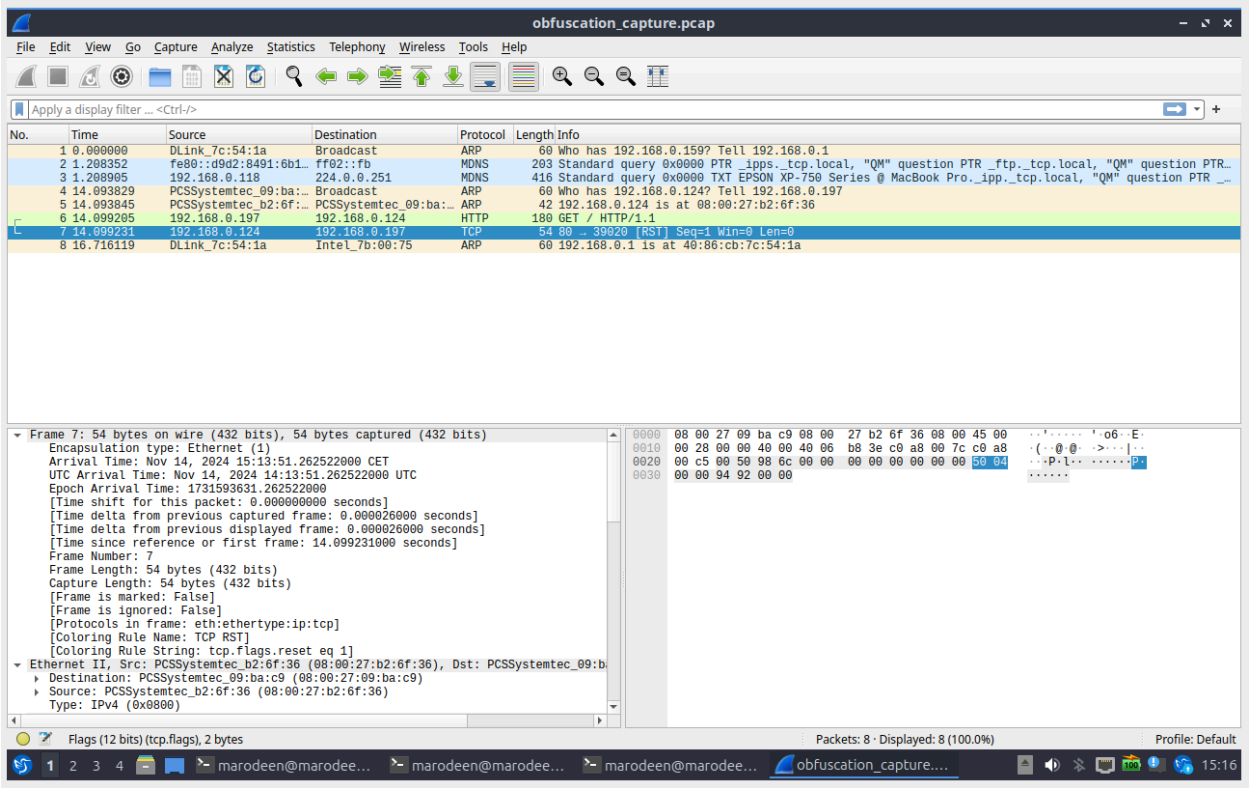
Analysis of the Wireshark packets from the packet fragmentation attack showed that the packets were indeed all fragmented. We can see the ‘More Fragments: Set’ flag as well as the ‘Fragment Offset’ line.

Timing-Based Evasion -



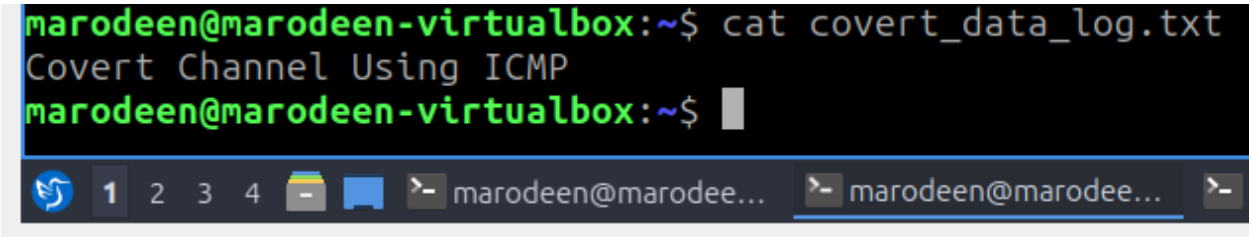
We can see here that using TCP, we were able to scan the different ports on the victim machine by implementing a random delay between the scanning of each port.

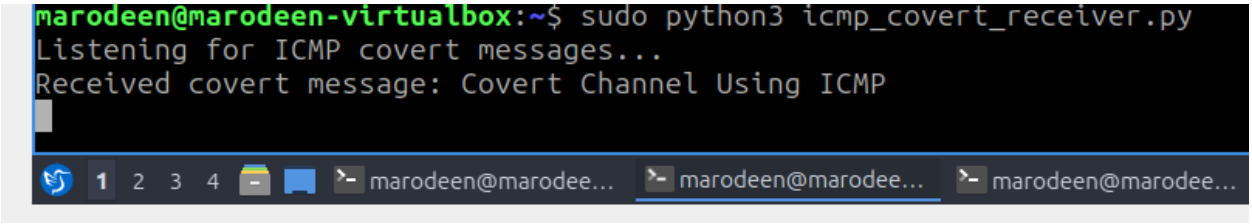
Obfuscating Attack Payloads -

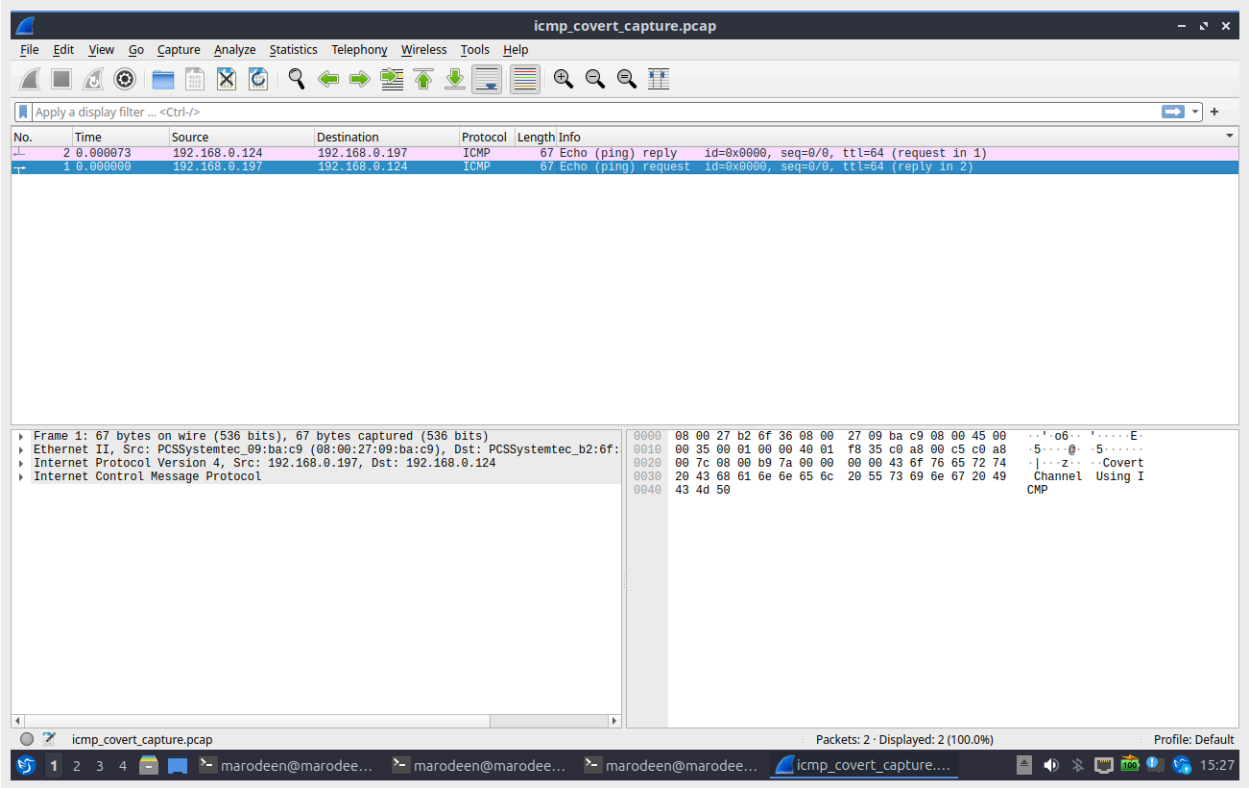


Shown is the TCP packet that contained the reverse shell.

ICMP Tunneling for Covert Communication -







Looking at the contents of the packet here, we were able to capture the hidden message within the ICMP payload: “Covert Channel Using ICMP”. Nothing was detected by Snort, but in this way we were able to hide a potential attack within an ICMP packet.

Snort Logs

The only attack which produced a Snort alert was the Timing-Based Evasion attack, and it was not detected by the custom rule we set to detect port scans. For this attack, Snort output the following:

```
11/14-15:01:29.545907  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:20
11/14-15:01:32.570949  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:21
11/14-15:01:37.533154  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:22
11/14-15:01:40.217492  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:23
11/14-15:01:45.941058  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:24
11/14-15:01:51.814417  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:25
11/14-15:01:57.429322  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:26
11/14-15:02:01.445877  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:27
11/14-15:02:06.951296  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:28
11/14-15:02:11.553841  [**] [1:503:7] MISC Source Port 20 to <1024 [**] [Classification: Potentially Bad Traffic]
[Priority: 2] {TCP} 192.168.0.197:20 -> 192.168.0.124:29
```

Presumably, our script used port 20 to do the port scan and the use of port 20 to scan ports below 1024 is able to be detected by Snort.

Mitigation Suggestions

Packet Fragmentation for Evasion -

To mitigate this attack, we would need to configure Snort to put together fragmented packets and do analysis of its contents. The Snort ‘frag3 preprocessor’ can be used to enable it to put together fragmented packets. As well, we may be able to set a rule that detects any unexpected fragmentation of packets.

Timing-Based Evasion -

To prevent timing-based evasion, we could fine tune the time-based thresholds that Snort uses to detect such scans. Another technique would be to use network behavior anomaly detection (NBAD) which does a broader analysis of traffic from a source. Such analysis would have been able to detect that the traffic from the attacker was anomalous.

Obfuscating Attack Payloads & ICMP Tunneling for Covert Communication -

Both of these attacks would be easily prevented by Snort doing a deeper analysis of the payloads within the packets. Instead of considering HTTP or ICMP as harmless and deprioritizing them, Snort could comb through all the contents of the packets to determine if any malicious code is hidden inside. Of course, such comprehensive scanning would be costly in time and computational power.