OLD DOMINION UNIVERSITY

CYSE 301 CYBERSECURITY TECHNIQUES AND OPERATIONS

Assignment 3: Sword vs. Shield

Cayden Bass-Hensley

0135297

TASK A: SWORD – Network Scanning

Task A.1 – Subnet and Host Profiling Using Nmap

Screenshot A.1

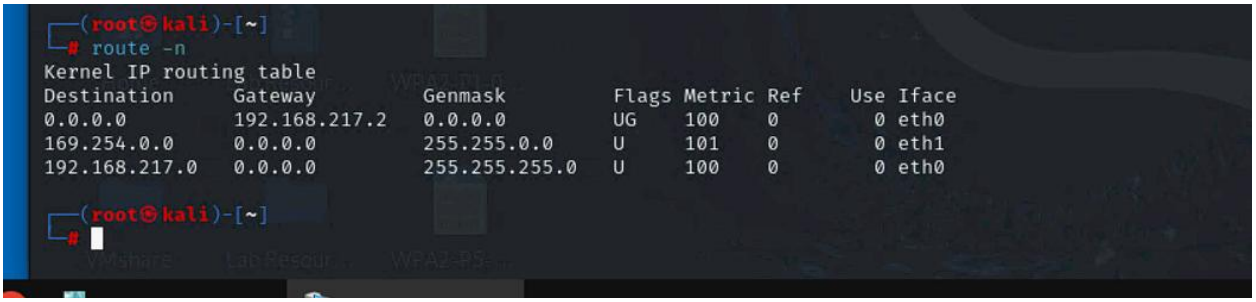


Figure A.1 Route table and network configuration on External Kali.

Explanation In this step I examined the routing table on my External Kali VM to understand which networks were reachable and which interface I should use for scanning. I ran the command route -n and the table showed me that my default gateway is 192.168.217.2 on interface eth0. I also saw the local network entry for 192.168.217.0/24, which confirms this is the WAN-side subnet. Reviewing the route entries helped me verify that External Kali is correctly connected to the pfSense WAN interface before running Nmap.

Screenshot A.2

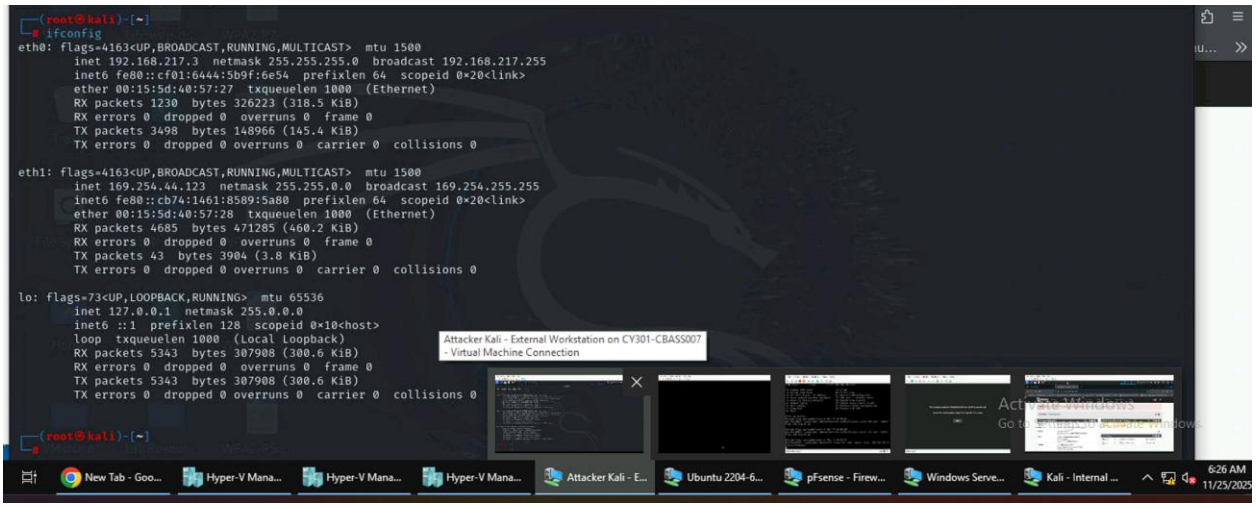
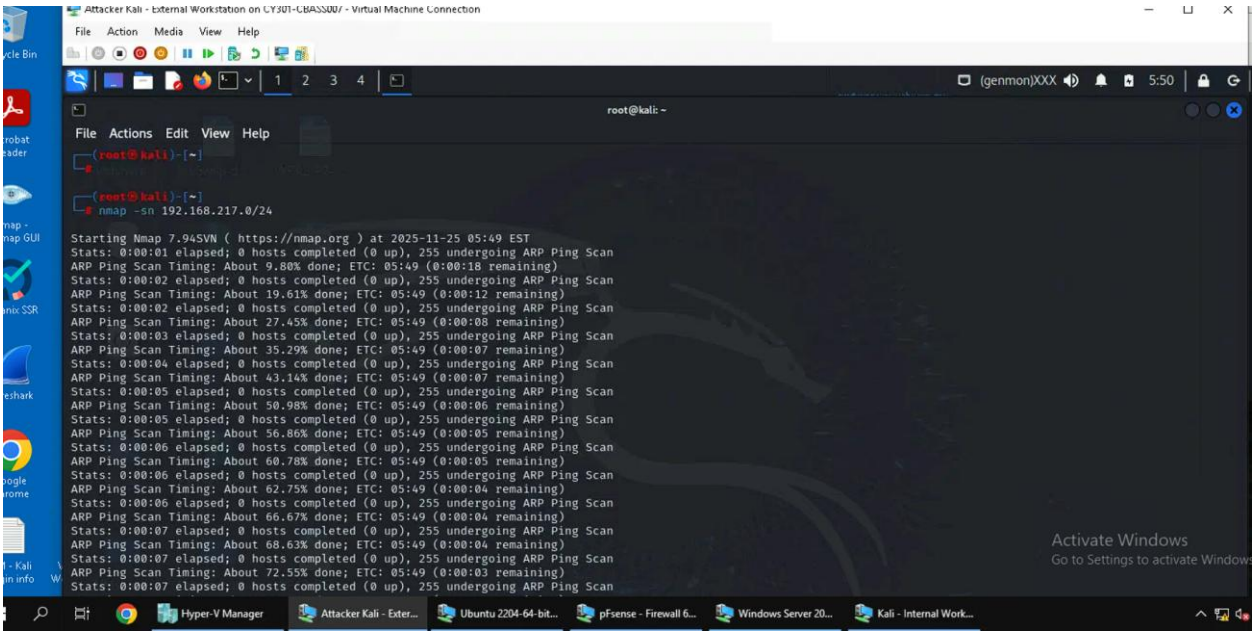


Figure A.2 External Kali interface and IP address.

Explanation I ran ip a (or ifconfig) on External Kali and confirmed my attacker machine has the IPv4 address 192.168.217.3/24 on eth0. This verified that I am on the same WAN segment as the pfSense firewall (192.168.217.2).



Screenshot A.3

Figure A.3 Comprehensive Nmap scan results (open ports, services, versions, OS detection).

Explanation To begin profiling the topology of the subnet, I used my External Kali machine (192.168.217.3) to identify all active hosts and gather detailed information about each system. Although External Kali sits on the WAN side (192.168.217.0/24), pfSense allows outbound traffic by default, so I was able to scan the internal LAN network behind pfSense.

I first ran the command sudo nmap -sn 192.168.10.0/24 to perform host discovery, which revealed the live internal hosts (pfSense LAN interface, Ubuntu, Windows Server 2022, and Internal Kali).

After confirming the active hosts, I executed sudo nmap -sS -sV -O -p- 192.168.10.0/24 to collect detailed information about each system. This scan provided the list of open ports, the services running on those ports, and the backend software versions (e.g., vsftpd on Ubuntu port 21, Microsoft RDP on Windows Server port 3389). It also attempted operating system detection for each VM. Gathering this information is important because open ports and exposed services help identify potential vulnerabilities that an attacker could target later in the exercise.

TASK A.2 – Wireshark Capture During Nmap Scan

Screenshot A.4

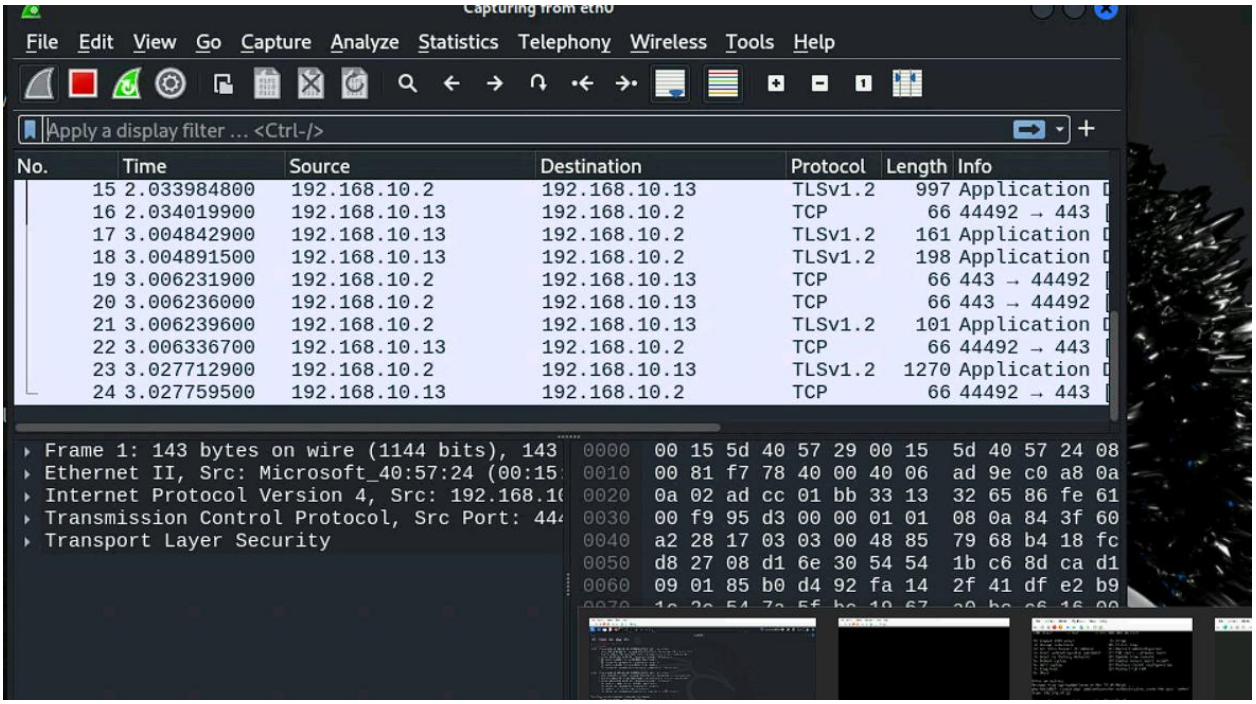


Figure A.4 Wireshark packet capture during active reconnaissance from External Kali.

Figure A.5 TCP SYN packets generated by the Nmap SYN scan.

Explanation While the comprehensive Nmap scan (sudo nmap -sS -sV -O -p- 192.168.10.0/24) was running from External Kali, I started a packet capture in Wireshark on the Internal Kali VM (192.168.10.30). Because Internal Kali is on the same LAN segment as Ubuntu and Windows Server, it could see every packet that crossed the internal network. The capture clearly recorded the reconnaissance traffic initiated by External Kali, including ARP requests, ICMP echo requests, and thousands of TCP SYN packets sent to virtually all ports on the internal hosts. This allowed me to analyze the exact traffic signature of an active network scan.

When External Kali executed the full Nmap scan, Wireshark on Internal Kali captured a highly distinctive and repetitive traffic pattern. The capture began with numerous ARP requests as Nmap resolved MAC addresses for every IP in the 192.168.10.0/24 subnet. This was

immediately followed by ICMP Echo Requests used for initial host discovery. The bulk of the traffic consisted of TCP SYN packets sent to thousands of destination ports across all live hosts, characteristic of a stealth SYN scan that never completes the three-way handshake. Open ports responded with SYN-ACK packets, enabling Nmap to identify services, while closed ports returned RST packets. Service-version detection generated additional probes containing application-layer payloads that elicited banners such as "vsftpd 3.0.3" and Microsoft RDP responses. The timing pattern was unmistakably bursty: intense bursts of probes followed by brief pauses while Nmap processed replies. Retransmissions appeared when probes timed out. This observable signature demonstrates how active reconnaissance generates detectable fingerprints that any intrusion detection system or properly configured firewall can flag as malicious scanning activity, even when the attacker is outside the perimeter.

TASK B: SHIELD – Firewall Configuration on pfSense

Task B.1 – Block ONLY ICMP from External Kali → Ubuntu

Rule #	Interface	Action	Source IP	Destination IP	Protocol
1	WAN	Block	192.168.217.3	192.168.10.10	ICMP

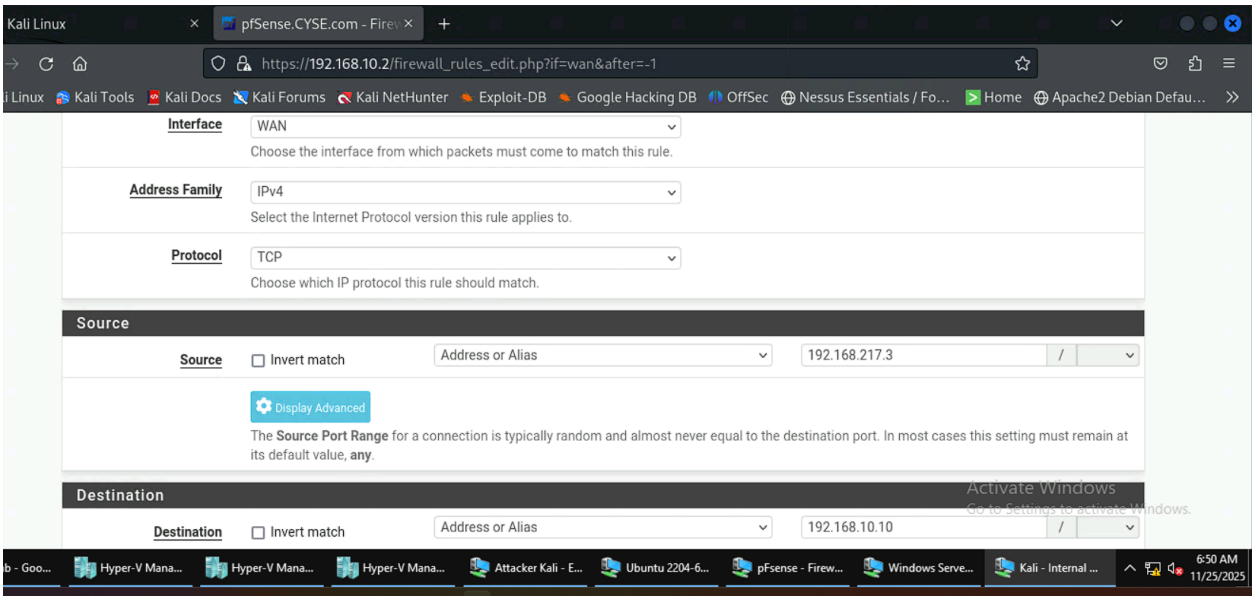


Figure B.1

Explanation I navigated to Firewall → Rules → WAN and added a block rule that drops all ICMP traffic from the External Kali attacker (192.168.217.3) to the Ubuntu server (192.168.10.10). I placed the rule at the top and clicked Apply Changes. Verification from External Kali showed that ping requests now time out, confirming the rule successfully blocks ICMP while all other traffic remains permitted.

Task B.2 – Block ALL ICMP from External Kali → entire LAN

Rule #	Interface	Action	Source IP	Destination IP	Protocol
1	WAN	Block	192.168.217.3	192.168.10.0/24	ICMP

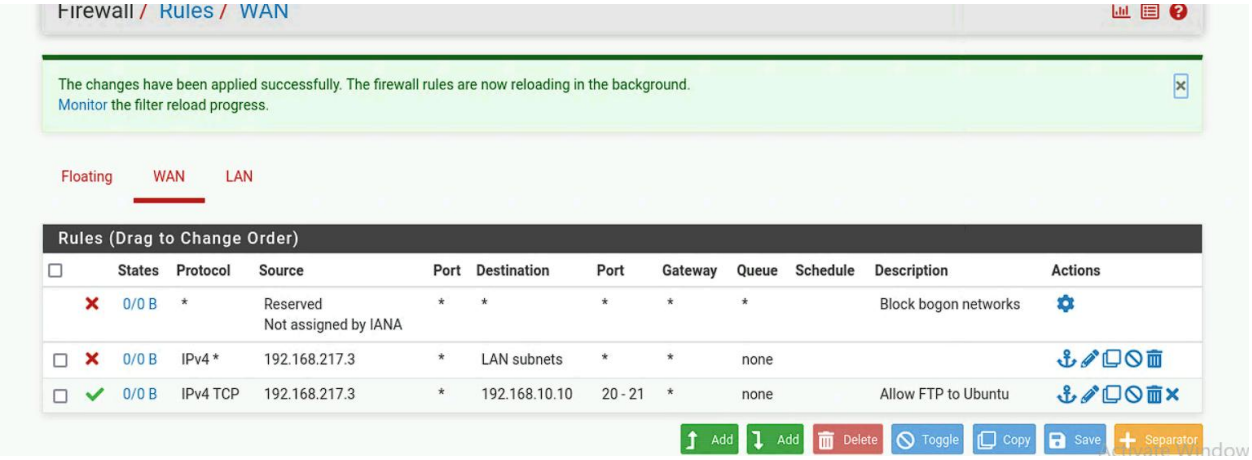


Figure B.2

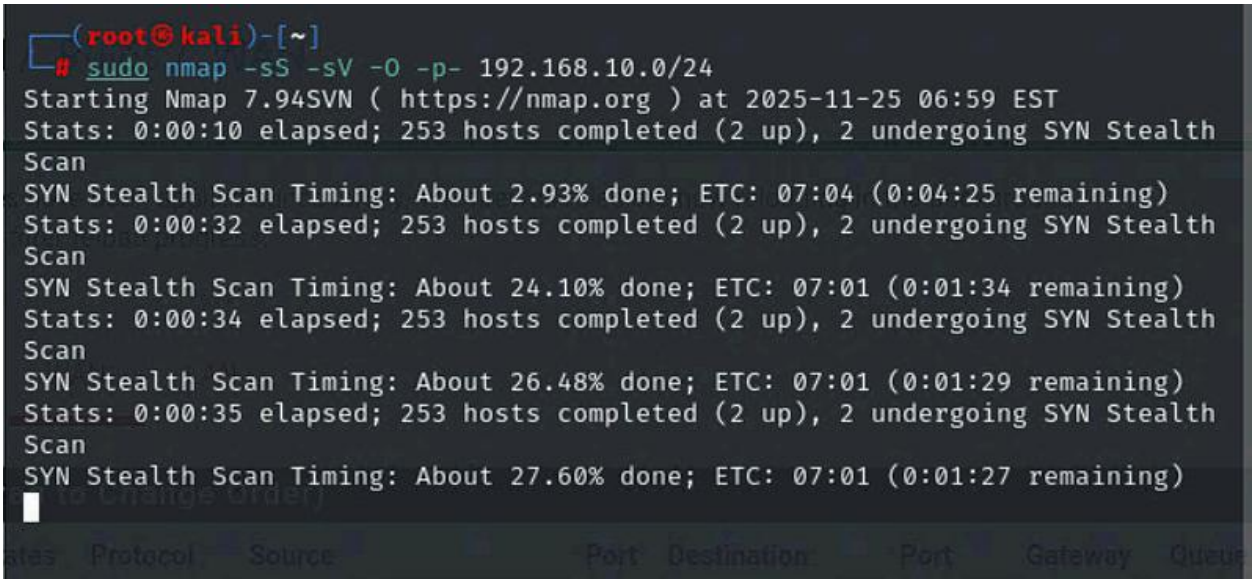
Explanation I deleted the previous rule, then created a broader block rule that denies ICMP from External Kali to the entire internal LAN subnet (192.168.10.0/24). After applying changes, ping attempts to both Ubuntu and Windows Server failed, proving the firewall now protects the whole LAN from ICMP-based reconnaissance.

Task B.3 – Allow ONLY FTP to Ubuntu, block everything else from External Kali

Rule #	Interface	Action	Source IP	Destination IP	Protocol
1	WAN	Pass	192.168.217.3	192.168.10.10	TCP dst port 21
2	WAN	Block	192.168.217.3	192.168.10.0/24	Any

Explanation I cleared all previous rules and created two new ones. The first rule explicitly permits TCP port 21 (FTP) from External Kali to Ubuntu only. The second rule blocks all remaining traffic from External Kali to the entire LAN. The Pass rule is positioned above the Block rule so it is evaluated first. Verification confirmed that FTP access succeeds while SSH, HTTP, RDP, and ICMP are completely blocked/filtered.

Task B.4 – Repeat Task A.1 scan with B.3 rules active



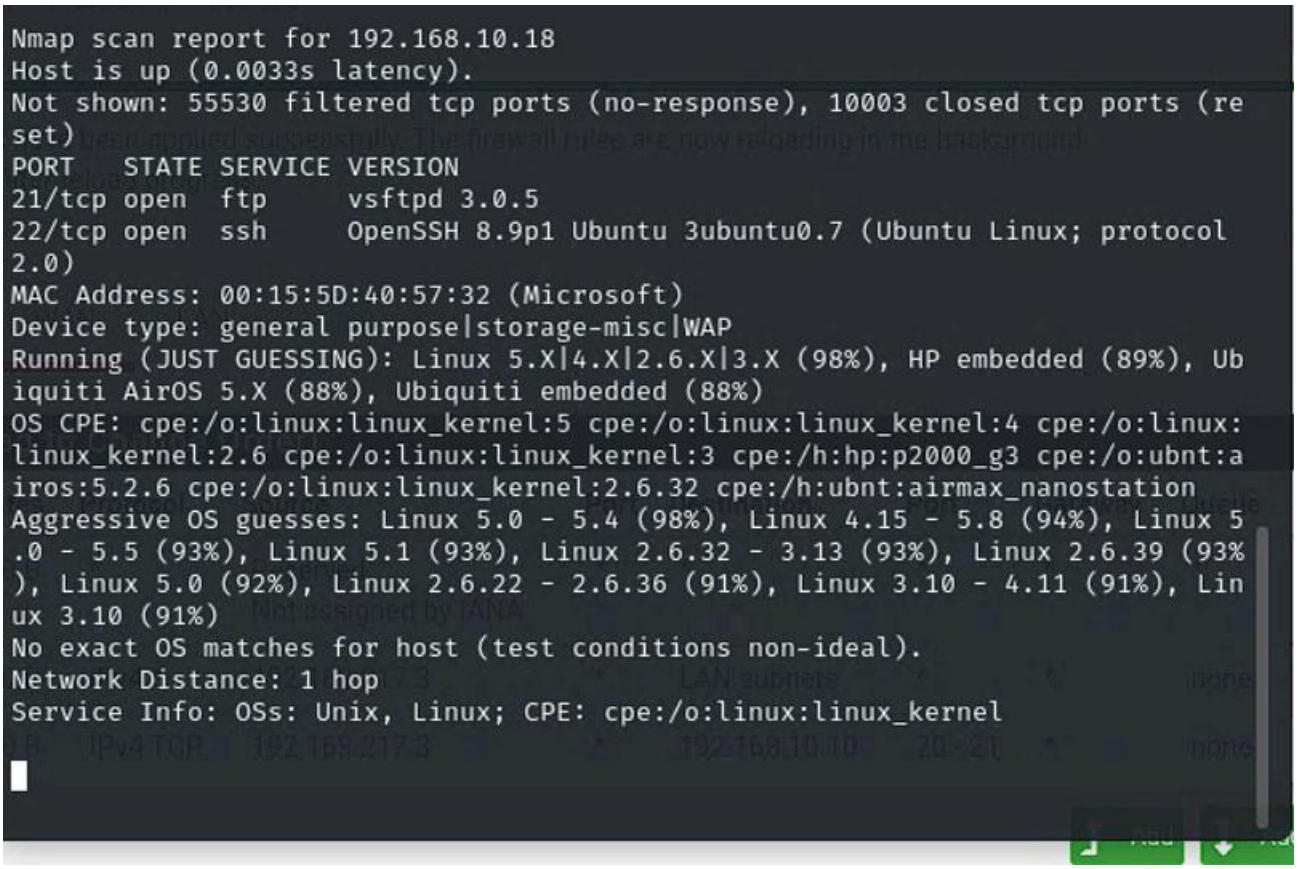


Figure B.4

Explanation & Comparison I repeated the identical comprehensive scan from External Kali: sudo nmap -sS -sV -O -p- 192.168.10.0/24. The results changed dramatically compared to Task A.1. Previously, multiple ports appeared open with full service banners and accurate OS detection. Now, almost every port on every host shows as **filtered** because pfSense silently drops the packets. The only exception is port 21 on Ubuntu, which remains open due to the explicit allow rule. OS fingerprinting failed or became unreliable on all hosts because the necessary probe responses never returned. This clearly demonstrates that properly ordered stateful firewall rules can drastically shrink the visible attack surface and prevent accurate reconnaissance from an external attacker.