**Week 2: Network Security Report**

**1. Detailed Network Scan Using Advanced Nmap Features:**

**Task Description:** To perform an in-depth network scan using advanced features of Nmap, the command nmap -A -T4 192.168.1.0/24 was utilized to conduct an aggressive scan with service detection.

**Problems Encountered:**

1. **Scan Performance Issues:**
   * **Problem:** The aggressive scan took a significant amount of time to complete, impacting network performance and resulting in delays.
   * **Solution:** To improve performance, the scan was split into smaller segments by reducing the target range and running multiple scans in parallel. Additionally, adjusting the timing template to -T3 optimized performance without overwhelming the network.
2. **Incomplete Scan Results:**
   * **Problem:** Some devices were not fully detected or displayed incomplete information due to network filtering.
   * **Solution:** To address this, more granular scanning options were used, such as --script for specific service detection and --unprivileged to avoid potential detection issues caused by elevated privileges.
3. **False Positives in Service Detection:**
   * **Problem:** The scan results included several false positives for service versions, leading to inaccurate assessments.
   * **Solution:** Cross-referencing with additional tools like netcat and telnet helped verify the accuracy of service information, and updated Nmap scripts were used to improve detection reliability.

**2. Identifying and Analyzing Network Anomalies with Wireshark:**

**Task Description:** A Wireshark capture session was conducted to identify and analyze network anomalies by starting a capture and reviewing the resulting file.

**Problems Encountered:**

1. **High Volume of Data:**
   * **Problem:** The capture file contained a large amount of traffic data, making it challenging to identify anomalies.
   * **Solution:** Applied display filters in Wireshark to narrow down the data to relevant protocols and traffic types, which facilitated more focused analysis. Filters like ip.addr==192.168.1.1 were used to isolate traffic from specific IP addresses.
2. **Difficulty in Identifying Anomalies:**
   * **Problem:** Distinguishing between normal and anomalous behavior was challenging due to the complexity of network traffic.
   * **Solution:** Utilized Wireshark’s built-in protocol analysis tools and expert systems to highlight unusual patterns. Consulting with network traffic analysis best practices and reference guides also provided additional insights.
3. **Performance Issues with Capture File Size:**
   * **Problem:** Large capture files caused performance issues when analyzing in Wireshark, leading to slow response times.
   * **Solution:** Reduced the capture file size by using capture filters to exclude irrelevant traffic during the capture process. Regularly saved and rotated capture files to manage file size and improve performance.

**3. Configuring a Basic Firewall on a Linux System Using iptables:**

**Task Description:** Configured a basic firewall using iptables by allowing SSH traffic and saving the configuration.

**Problems Encountered:**

1. **Misconfiguration of Rules:**
   * **Problem:** Initial firewall rules were too restrictive, causing unintended access issues and blocking legitimate traffic.
   * **Solution:** Reviewed and adjusted the rules to ensure correct access permissions. For example, added specific rules to allow traffic on required ports and tested configurations incrementally to verify functionality.
2. **Persistence of Firewall Rules:**
   * **Problem:** Firewall rules were not persistent across reboots, requiring manual reconfiguration after each restart.
   * **Solution:** Implemented iptables-persistent package to save and restore rules automatically upon reboot. Used the iptables-save and iptables-restore commands to ensure that the rules were properly preserved.
3. **Conflicts with Existing Rules:**
   * **Problem:** Conflicts arose with pre-existing rules, leading to unintended blocking of necessary traffic.
   * **Solution:** Carefully merged new rules with existing configurations, ensuring that rules did not overlap or conflict. Used iptables -L to review current rules and make necessary adjustments.

**4. Setting Up a Simple VPN and Testing Connectivity:**

**Task Description:** Installed and configured Proton VPN on a virtual machine, connected to a VPN server, and tested connectivity.

**Problems Encountered:**

1. **VPN Connection Drops:**
   * **Problem:** VPN connection intermittently dropped, affecting connectivity.
   * **Solution:** Checked VPN logs for errors and ensured the VPN client was up-to-date. Configured automatic reconnect settings in Proton VPN and tested different VPN servers to identify stable connections.
2. **IP Address Not Changing:**
   * **Problem:** After connecting to the VPN, the IP address did not change as expected.
   * **Solution:** Verified the VPN configuration and reconnected to the VPN server. Used external IP-checking tools to confirm the change and adjusted VPN settings to resolve IP leakage issues.
3. **Network Connectivity Issues:**
   * **Problem:** Experienced connectivity issues when accessing certain websites or services via the VPN.
   * **Solution:** Adjusted VPN settings such as DNS configuration and split tunneling options. Consulted Proton VPN support for configuration guidance and tested different VPN protocols to enhance compatibility.

**5. Implementing Basic Intrusion Detection System (IDS) Rules Using Snort:**

**Task Description:** Installed and configured Snort to use default or custom IDS rules, then monitored network traffic for intrusions.

**Problems Encountered:**

1. **Rule File Errors:**
   * **Problem:** Encountered errors related to missing or incorrect rule files, such as /etc/snort/rules/attack-responses.rules and /etc/snort/unicode.map.
   * **Solution:** Updated the Snort configuration file to include available rule files and corrected the paths. Installed any missing rule files from the Snort community repository and ensured the configuration referenced the correct file locations.
2. **Configuration Errors:**
   * **Problem:** Snort failed to start properly due to configuration errors, resulting in warnings and fatal errors.
   * **Solution:** Validated and tested the Snort configuration using snort -T to check for syntax and configuration issues. Corrected any identified errors and ensured that all required configurations were properly set.
3. **Performance and Resource Issues:**
   * **Problem:** Snort caused high CPU usage and memory consumption, impacting system performance.
   * **Solution:** Optimized Snort configurations, such as adjusting logging levels and tuning performance parameters. Implemented resource management practices to mitigate performance issues, including allocating additional system resources and optimizing rule sets.

**Conclusion:**  
Throughout Week 2, I successfully completed the network security tasks while overcoming various technical challenges. Each task provided valuable insights into network scanning, traffic analysis, firewall configuration, VPN setup, and IDS implementation. The solutions implemented not only resolved immediate issues but also enhanced my understanding of network security principles and tools.