Network Security

Homework 2

Name: Mahdi Aghajanian

Student number:

Theoretical Questions

1)

- IDS:
    1. Network Intrusion Detection Systems (NIDS):
        - NIDS monitors network traffic through sensors placed on the network
        - It provides a comprehensive view of network activity and can detect attacks targeting the entire network
        - NIDS is commonly used in large organizations to monitor and protect their network infrastructure.
    2. Host Intrusion Detection Systems (HIDS):
        - HIDS is installed directly on individual devices to monitor their traffic.
        - It offers more control and flexibility, but can be burdensome to manage in large organizations.
        - HIDS is often used to protect critical servers and workstations from unauthorized access or malicious activity.
    3. Protocol-based Intrusion Detection Systems (PIDS):
        - PIDS is placed at the front of a server and monitors traffic to and from devices.
        - It is commonly used to secure users browsing the internet and detect attacks targeting specific protocols.
    4. Application Protocol-based Intrusion Detection Systems (APIDS):
        - APIDS is similar to PIDS but monitors traffic across a group of servers.
        - It is often used to monitor specific application protocols and helps in segmenting and classifying network monitoring activities.
    5. Hybrid Intrusion Detection Systems:
        - Hybrid IDS solutions combine multiple types of intrusion detection to cover various systems in one interface.
        - They offer a more comprehensive approach to detecting and preventing attacks.
- IPS:
    1. Network-based Intrusion Prevention Systems (NIPS):
        - NIPS monitors and protects the entire network from anomalous or suspicious behavior.
        - It can be integrated with other monitoring tools to provide a comprehensive view of network security.

    - NIPS is commonly used to prevent attacks from reaching the network and to contain threats.
2. Wireless Intrusion Prevention Systems (WIPS):
    - WIPS monitors wireless networks owned by an organization.
    - It provides targeted detection and response for wireless networks.
    - WIPS is commonly used to protect wireless networks from unauthorized access and attacks.
3. Host-based Intrusion Prevention Systems (HIPS):
    - HIPS is deployed on key devices or hosts to monitor traffic and detect malicious behavior.
    - It offers protection at the host level and can prevent attacks from compromising the host.
    - HIPS is commonly used to secure critical servers and workstations.
4. Network Behavioral Analysis (NBA):
    - NBA looks for anomalous behavior within network patterns.
    - It is effective in detecting incidents such as DDoS attacks and other types of malwares.
    - NBA is commonly used to identify and respond to abnormal network behavior.

---

2)

1. Sniffer Mode:

   In Sniffer Mode, Snort acts as a packet sniffer similar to tcpdump. It reads IP packets and displays them in the console application. This mode allows users to analyze live network traffic and gain insights into the packets flowing through the network.

2. Packet Logger Mode:

   Packet Logger Mode enables Snort to log all IP packets, both inbound and outbound, that traverse the network. This mode is useful for network traffic debugging and analysis. It allows users to capture and store packet data for later analysis and investigation.

3. Network Intrusion Detection System (NIDS) and Network Intrusion Prevention System (NIPS) Modes:

   In NIDS and NIPS Modes, Snort functions as a full-blown network intrusion detection and prevention system. It logs and/or drops packets that are identified as malicious based on user-defined rules. NIDS mode is used to detect threats on a local network, while NIPS mode is used to stop threats on a local network by terminating connections.

---

3)

- The main rule types supported by Snort are:
  1. Traditional Rules:
     - Traditional rules are the most common type of rules used in Snort.
     - They consist of a rule header and a rule body.
     - The rule header specifies the action, protocol, IP addresses, ports, and direction to filter the traffic.
     - The rule body contains the content, options, and modifiers to match against the network traffic.
  2. Service Rules:
     - Service rules are used to detect specific network services or protocols.
     - They are designed to match against the application layer protocols, such as HTTP, FTP, SMTP, etc.
     - Service rules can be used to detect anomalies or malicious activities related to specific services.
  3. File Rules:
     - File rules are used to detect specific file types or file-related activities.
     - They can be used to identify files based on their content, extension, or other file attributes.
     - File rules are often used to detect malicious files or file-based attacks.
  4. File Identification Rules:
     - File identification rules are a specific type of file rule used to identify file types based on their content.
     - These rules define a file type that can be used in subsequent rules with the file_type option.
     - File identification rules have a rule header consisting of only file_id, which indicates that the rule is a file type definition.
     - They also include a file_meta rule option to set the file metadata for a given file identification rule.
- Differences between the rule types:
  - The main differences between these rule types are their purpose and the elements they match against. Traditional rules are used to detect general network traffic anomalies or malicious activities. Service rules focus on specific network services or protocols. File rules are used to detect file-related activities or specific file types. File identification rules specifically identify file types based on their content.

----

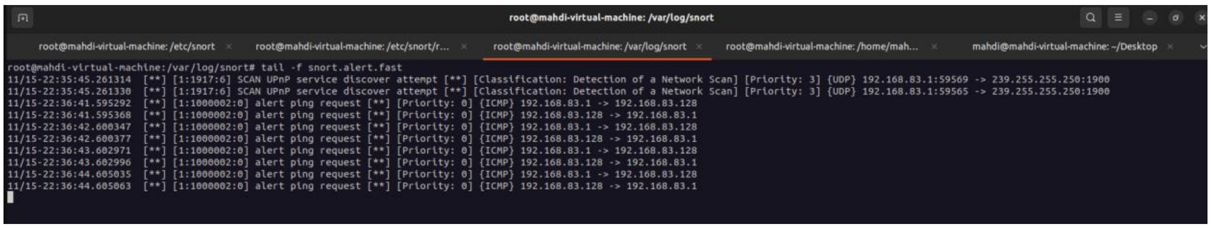
4)

1. Distance:
    - The distance keyword is used to specify where to start searching for a pattern relative to the previous content match.
    - It allows you to skip a certain number of bytes after the last content match before looking for the next one.
    - The value specified with distance can be positive or negative and can also be set to a string value referencing a variable extracted by the byte_extract keyword in the same rule.
2. Within:
    - The within keyword is used to specify how far into a Snort packet or buffer to look for a pattern relative to the previous content match.
    - It tells Snort to look for the content match within a certain number of bytes of the last one.
    - The value specified with within must be greater than or equal to the length of the content string.
    - It can also be set to a string value referencing a variable extracted by the byte_extract keyword in the same rule.
3. Replace:
    - The replace keyword is used to replace a specified pattern with a different string.
    - It allows you to modify the content of a packet by replacing a specific pattern with another string.
4. http_stat_code:
    - The http_stat_code option is used to match against the HTTP status code in an HTTP response.
    - It allows you to create rules that trigger based on specific HTTP status codes.
5. Metadata:
    - The metadata keyword is used to add metadata to a Snort rule.
    - It allows you to provide additional information about the rule, such as the author, reference links, or any other relevant details.

Practical Questions

1)

alert icmp any any -> $HOME_NET any (msg:"alert ping request"; itype:8; sid:1000003; rev:1;)


root@mahdi-virtual-machine: /var/log/snort
root@mahdi-virtual-machine: /etc/snort
root@mahdi-virtual-machine: /etc/snort/r...
root@mahdi-virtual-machine: /var/log/snort
root@mahdi-virtual-machine: /home/mah...
mahdi@mahdi-virtual-machine: ~/Desktop

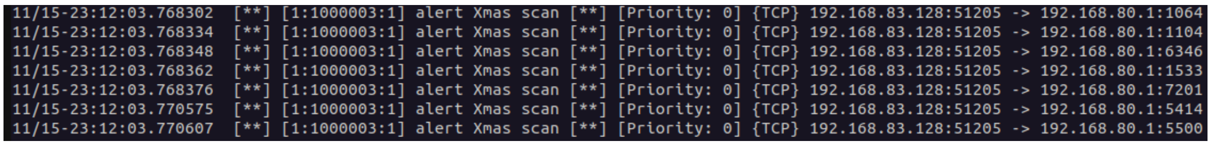
```
root@mahdi-virtual-machine:/var/log/snort# tail -f snort.alert.fast
11/15-22:35:45.261314  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.83.1:59569 -> 239.255.255.250:1900
11/15-22:35:45.261330  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.83.1:59565 -> 239.255.255.250:1900
11/15-22:36:41.595292  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.1 -> 192.168.83.128
11/15-22:36:41.595368  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.128 -> 192.168.83.1
11/15-22:36:42.600347  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.1 -> 192.168.83.128
11/15-22:36:42.600377  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.128 -> 192.168.83.1
11/15-22:36:43.602971  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.1 -> 192.168.83.128
11/15-22:36:43.602996  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.128 -> 192.168.83.1
11/15-22:36:44.605035  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.1 -> 192.168.83.128
11/15-22:36:44.605063  [**] [1:1000002:0] alert ping request [**] [Priority: 0] {ICMP} 192.168.83.128 -> 192.168.83.1
```

Nmap test: : nmap -PE IP –disable-arp-ping

Nmap bypass: nmap -Pn IP –disable-arp-ping

2)

alert tcp any any -> $HOME_NET any (msg:"alert Xmas scan"; flags:FPU; sid:1000004; rev:1;)

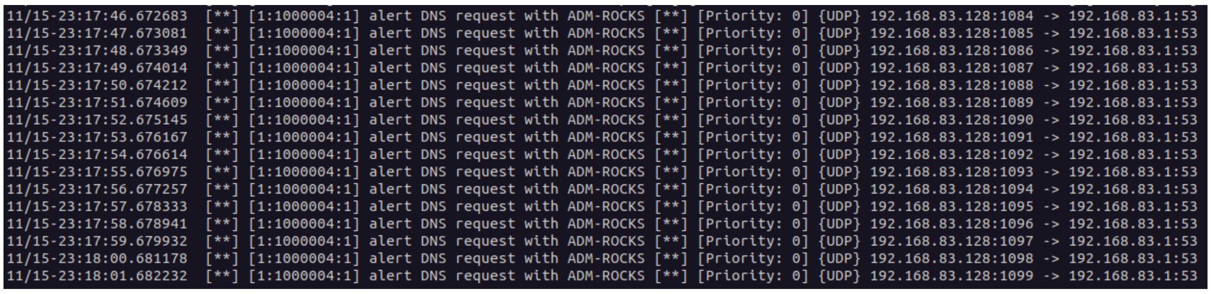


```
11/15-23:12:03.768302  [**] [1:1000003:1] alert Xmas scan [**] [Priority: 0] {TCP} 192.168.83.128:51205 -> 192.168.80.1:1064
11/15-23:12:03.768334  [**] [1:1000003:1] alert Xmas scan [**] [Priority: 0] {TCP} 192.168.83.128:51205 -> 192.168.80.1:1104
11/15-23:12:03.768348  [**] [1:1000003:1] alert Xmas scan [**] [Priority: 0] {TCP} 192.168.83.128:51205 -> 192.168.80.1:6346
11/15-23:12:03.768362  [**] [1:1000003:1] alert Xmas scan [**] [Priority: 0] {TCP} 192.168.83.128:51205 -> 192.168.80.1:1533
11/15-23:12:03.768376  [**] [1:1000003:1] alert Xmas scan [**] [Priority: 0] {TCP} 192.168.83.128:51205 -> 192.168.80.1:7201
11/15-23:12:03.770575  [**] [1:1000003:1] alert Xmas scan [**] [Priority: 0] {TCP} 192.168.83.128:51205 -> 192.168.80.1:5414
11/15-23:12:03.770607  [**] [1:1000003:1] alert Xmas scan [**] [Priority: 0] {TCP} 192.168.83.128:51205 -> 192.168.80.1:5500
```

Nmap test: nmap -sX IP

3)

alert udp $HOME_NET any -> any 53 (msg:"alert DNS request with ADM-ROCKS"; content:"ADM-ROCKS"; sid:1000005; rev:1;)

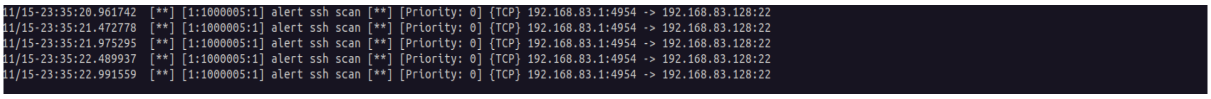


```
11/15-23:17:46.672683  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1084 -> 192.168.83.1:53
11/15-23:17:47.673081  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1085 -> 192.168.83.1:53
11/15-23:17:48.673349  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1086 -> 192.168.83.1:53
11/15-23:17:49.674014  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1087 -> 192.168.83.1:53
11/15-23:17:50.674212  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1088 -> 192.168.83.1:53
11/15-23:17:51.674609  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1089 -> 192.168.83.1:53
11/15-23:17:52.675145  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1090 -> 192.168.83.1:53
11/15-23:17:53.676167  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1091 -> 192.168.83.1:53
11/15-23:17:54.676614  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1092 -> 192.168.83.1:53
11/15-23:17:55.676975  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1093 -> 192.168.83.1:53
11/15-23:17:56.677257  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1094 -> 192.168.83.1:53
11/15-23:17:57.678333  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1095 -> 192.168.83.1:53
11/15-23:17:58.678941  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1096 -> 192.168.83.1:53
11/15-23:17:59.679932  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1097 -> 192.168.83.1:53
11/15-23:18:00.681178  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1098 -> 192.168.83.1:53
11/15-23:18:01.682232  [**] [1:1000004:1] alert DNS request with ADM-ROCKS [**] [Priority: 0] {UDP} 192.168.83.128:1099 -> 192.168.83.1:53
```

test : hping3 –udp -p 53 –sign "ADM-ROCKS" IP

4)

alert tcp any any -> $HOME_NET 22 (msg:"alert ssh scan"; flags:S; sid:1000006; rev:1;)



```
11/15-23:35:20.961742  [**] [1:1000005:1] alert ssh scan [**] [Priority: 0] {TCP} 192.168.83.1:4954 -> 192.168.83.128:22
11/15-23:35:21.472778  [**] [1:1000005:1] alert ssh scan [**] [Priority: 0] {TCP} 192.168.83.1:4954 -> 192.168.83.128:22
11/15-23:35:21.975295  [**] [1:1000005:1] alert ssh scan [**] [Priority: 0] {TCP} 192.168.83.1:4954 -> 192.168.83.128:22
11/15-23:35:22.489937  [**] [1:1000005:1] alert ssh scan [**] [Priority: 0] {TCP} 192.168.83.1:4954 -> 192.168.83.128:22
11/15-23:35:22.991559  [**] [1:1000005:1] alert ssh scan [**] [Priority: 0] {TCP} 192.168.83.1:4954 -> 192.168.83.128:22
```

nmap test: namp -sV -p 22 IP

5)

```
Alert tcp any any -> $HOME_NET any (msg:"alert dos attack"; flags:!S ;
detection_filter:track by_src, count 5, seconds 10; sid:1000007; rev:1;)
```

```
Test: hping3 --flood -p 80 IP -A
```

6)

```
alert tcp any any -> -> $HOME_NET any (msg:"alert SQL Injection"; content:"or 1 =
1−−"; http_uri; sid:1000008; rev:1;)
```

7)

```
alert tcp any any -> $HOME_NET any (msg:"alert HTTP Request to index.php";
content:"/index.php"; http_uri;  sid:1000009; rev:1;)
```

Bonus Exercise

1)

Suricata is an open-source intrusion detection system (IDS) and intrusion prevention system (IPS) that is designed to monitor and analyze network traffic for malicious activity. It was developed by the Open Information Security Foundation (OISF) and released in 2010 as an alternative to Snort, another popular IDS/IPS.

- Capabilities of Suricata:
  - Multi-threaded Architecture:

    Suricata's multi-threaded architecture allows it to process multiple tasks simultaneously, providing improved performance and scalability.
  - Packet Capturing:

    Suricata excels in packet capturing, enabling it to monitor and analyze network traffic effectively.
  - Signature-based Detection:

    Suricata, like Snort, implements signature-based detection, which involves comparing packets against a pre-defined ruleset to identify known threats with high accuracy.
  - Anomaly-based Detection:

    Suricata also supports anomaly-based detection, which uses machine learning to model baseline traffic patterns and identify unusual behavior that may indicate a potential threat.
  - Network-based Intrusion Detection:

    Suricata is a network-based IDS, meaning it can monitor and detect malicious traffic across an entire network, including cloud, virtual, and local network environments.
  - Intrusion Prevention:

    Suricata is equipped with intrusion prevention capabilities, allowing it to take action to stop potential threats detected by the IDS.
- Comparison with Snort:
  - Performance:

    Suricata's multi-threaded architecture provides improved performance compared to Snort, as it can process multiple tasks simultaneously.
  - Ruleset Compatibility:

    Suricata can use almost all of the same rules available in Snort, making it compatible with Snort's extensive ruleset.

  - Community Support:

    Both Suricata and Snort have active communities that contribute to the development and maintenance of rulesets.

  - Updates:

    Suricata receives regular updates from the OISF, while Snort's community ruleset is exclusively updated by the community.

---

2)

Zeek is a passive, open-source network traffic analyzer that is known as an anomaly-based IDS (Intrusion Detection System). It is designed to monitor network traffic and detect suspicious or malicious activity. Zeek is widely used by operators as a network security monitor (NSM) to support investigations of potential security threats.

- why it is known as an anomaly-based IDS:
  - Comprehensive Network Activity Logs:

    Zeek generates extensive logs that describe network activity, including a record of every connection seen on the wire and application-layer transcripts. These logs provide a comprehensive view of network traffic, including HTTP sessions, DNS requests, SSL certificates, SMTP sessions, and more.

  - Customizable and Extensible Platform:

    Zeek provides a domain-specific scripting language that allows users to express arbitrary analysis tasks. This scripting language enables users to customize and extend Zeek's functionality to suit their specific needs. Users can write custom code to perform various analysis and detection tasks, including anomaly detection and behavioral analysis.

  - Broad Spectrum of Detection Approaches:

    While Zeek supports standard signature-based intrusion detection functionality, its scripting language enables a much broader spectrum of detection approaches. These include semantic misuse detection, anomaly detection, and behavioral analysis. Zeek's flexibility allows users to implement different detection techniques based on their specific requirements.

  - Low-Cost Alternative:

    Zeek runs on commodity hardware, making it a cost-effective alternative to expensive proprietary solutions. It provides high-speed, high-volume network monitoring capabilities and can be deployed on 10GE and even

100GE networks. Zeek's scalability advantages allow administrators to scale the system as needed, adding more systems when necessary.
