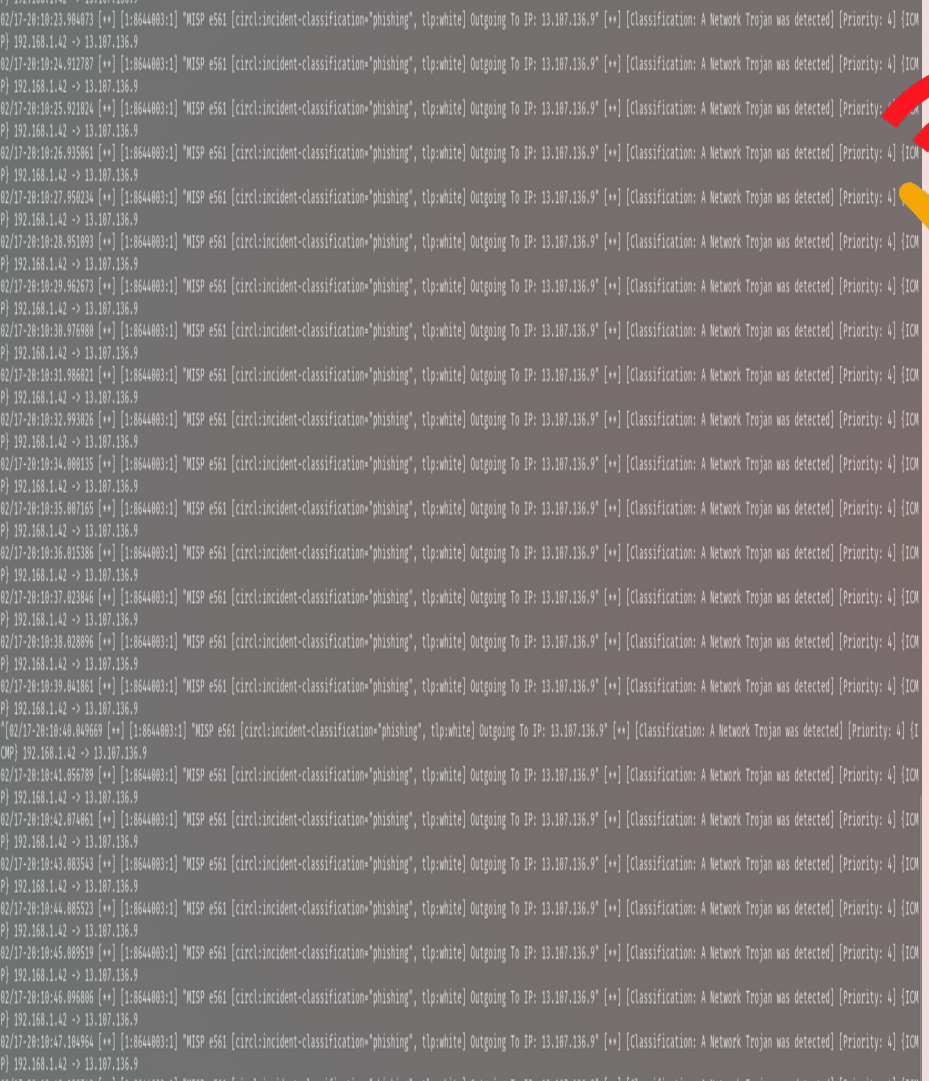


SAND5G

Snort IDS Training – Intrusion Detection in Action

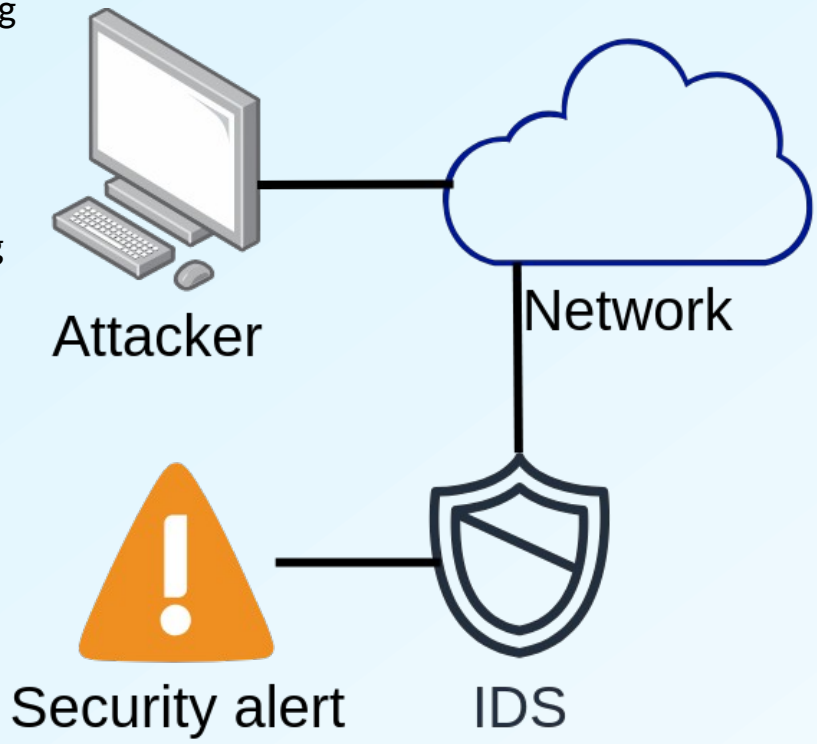
Understanding How Snort Detects Cyber Threats

Dionysia Varvarigou

Contents

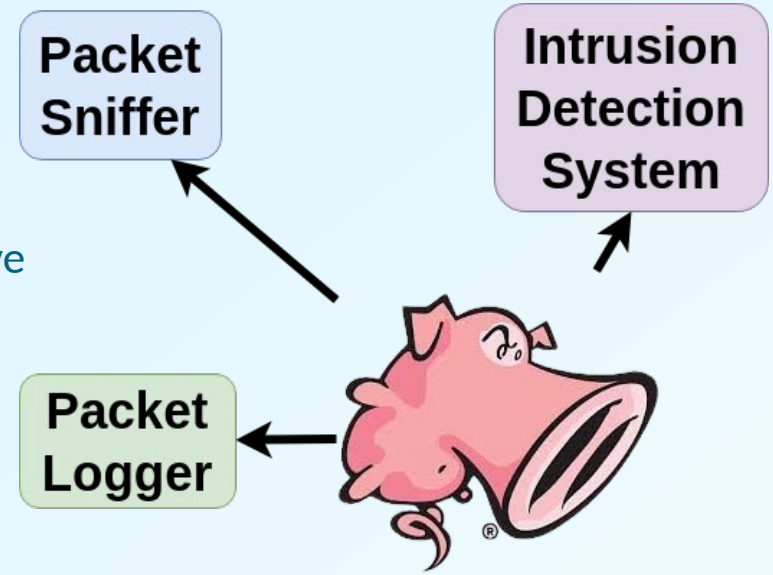
Introduction to IDS

- Definition : Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network [1]
- Example : A company has a network IDS (NIDS) monitoring all incoming traffic. An attacker tries to scan the company's servers for vulnerabilities. The IDS detects unusual traffic patterns (e.g., multiple connection attempts from the same IP). Alert is generated, notifying the security team.
- Types of IDS:
  - Network-Based IDS (NIDS) [2] – Monitors traffic across a network (nids reference)
  - Host-Based IDS (HIDS) [3] – Monitors activity on a specific device (hids reference)



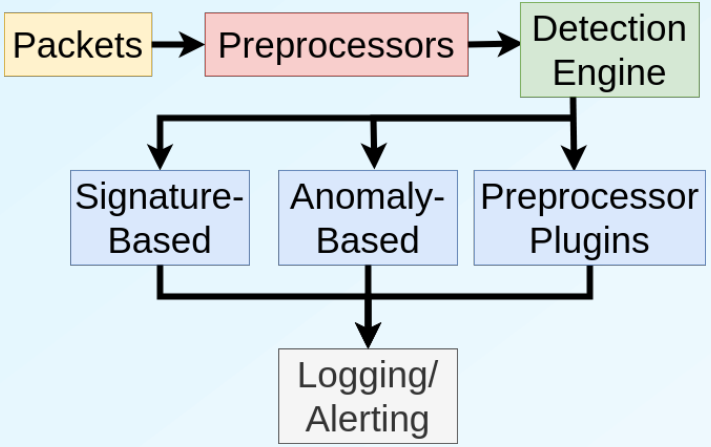
What is Snort?

- Snort is an open source NIDS that is specifically used for scanning data flowing on the network [4]
- Real-time packet analysis to detect attacks [5] : monitors network traffic in real time, analyzing packets as they pass through the system
- Functions as:
  - Packet sniffer [6] (like tcpdump) : acts as a passive network listener, capturing and displaying network packets without logging or analyzing them
  - Packet logger [7] : records captured packets and stores them for later analysis
  - Full IDS [7] : actively compares network traffic against predefined attack signatures and anomaly detection rules to detect and alert on threats



How Snort Works

- Captures network packets and analyzes them against rules [8] : Snort continuously monitors network traffic, examining each packet in real-time and it compares packet contents to predefined Snort rules. When a packet matches a rule, Snort generates an alert or log entry for security analysis.
- Example: If a packet originates from a suspicious IP address or contains malicious payloads (e.g., SQL injection), Snort can flag it as an attack.
- Detection Methods:
  - Signature-based [9] : works by comparing packets against a database of known attack patterns. This method is effective for detecting well-documented threats, such as SQL injection attempts or specific malware signatures.
  - Anomaly-based [10] : focuses on identifying unusual network behavior that deviates from normal traffic patterns. For example, if a system that typically sends 100 packets per minute suddenly begins sending 10,000, this could indicate a DDoS attack.
  - Preprocessor plugins [11] : allow for deep analysis of specific network protocols such as HTTP, FTP, and DNS. These plugins enhance Snort's ability to detect more complex attack patterns, such as buffer overflow attempts in web servers.



Syntax of Snort rules

- Alert Type → alert
- Protocol → tcp
- Source/Destination → any any -> 192.168.1.1 80
- Message → "SQL Injection Attempt"
- content:"UNION SELECT"; → Looks for the string "UNION SELECT", commonly used in SQL injection attacks
- nocase; → Makes it case-insensitive, so union select and UNION SELECT are both detected
- sid:1000002; → Assigns a new unique Signature ID

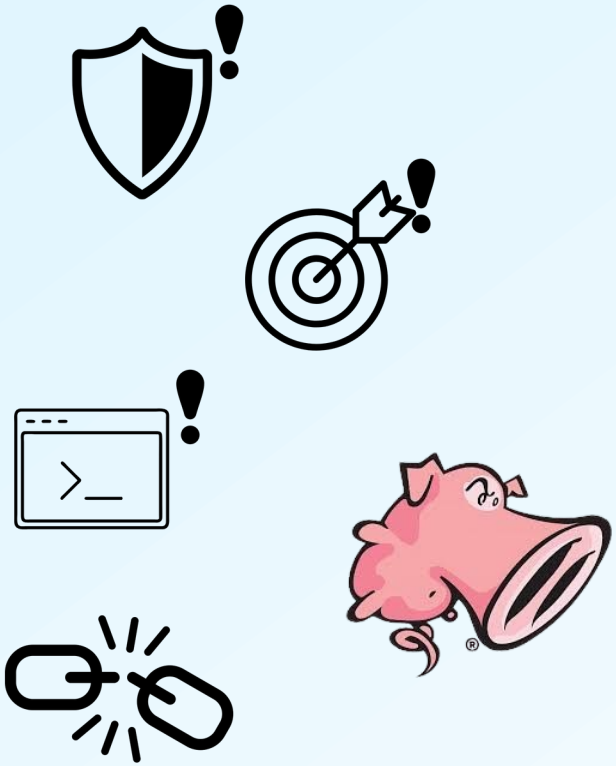
```
alert tcp any any -> 192.168.1.1 80 (msg:"SQL Injection Attempt"; content:"UNION SELECT"; nocase; sid:1000002;)
```

Advantages of Snort

- Snort advantages as found in [12]:
  - Free for Use : Snort is open-source software, meaning it is completely free to download, install, and use. Unlike commercial IDS solutions, there are no licensing costs, making it a cost-effective option for organizations of all sizes
  - Available for Linux and Windows Platforms : Snort is cross-platform, allowing users to run it on both Linux and Windows operating systems. This flexibility ensures compatibility with various network environments
  - Flexible Customization for Many Environments : Snort provides extensive configuration options and allows users to create custom rules, making it adaptable to enterprise, small business, and personal networks
  - Supports Auto-Update Rules : Snort allows for automatic rule updates, ensuring that new attack patterns and security threats are continuously added to its detection system without manual intervention
  - Can Be Hidden in the Network : Snort can be configured in stealth mode, making it invisible to attackers. Since it does not respond to traffic, it can monitor network activity without revealing its presence
  - Web Interface Availability : There are various third-party web interfaces (such as Snorby and BASE) that allow users to manage Snort more easily with a graphical user interface (GUI), rather than using only the command line
  - Low System Requirements (Compared to Suricata) : Snort is lightweight and consumes fewer system resources compared to Suricata. This makes it suitable for low-power devices and environments where computational efficiency is a priority.

Limitations of Snort

- Snort limitations as found in [12] :
  - Intrusion Detection Occurs Behind the Firewall : Snort operates inside the network, meaning it detects attacks only after they have passed through the firewall. While Snort can operate as a Intrusion Prevention Systems (IPS), it needs inline deployment and appropriate configuration.
  - Requires Precise Configuration to Avoid False Positives : To function effectively, Snort must be fine-tuned for each environment. Incorrect configurations can lead to false positives (detecting legitimate traffic as an attack) or false negatives (failing to detect real attacks)
  - Not Beginner-Friendly : Snort is command-line-based and requires manual configuration of rules, making it difficult for new users or those without cybersecurity expertise. Graphical interfaces exist but often lack full functionality
  - Requires Unidirectional Ethernet Cables for Sensor Installation : To prevent security risks, Snort sensors should be installed using unidirectional Ethernet cables (receive-only mode). This prevents attackers from sending malicious traffic back into the system, but it adds complexity to the deployment



Snort vs. Other Security Tools

Feature	Snort (IDS)	Honeypot	Firewall
Detects Attacks	✅	✅	❌
Prevents Attacks	❌	❌	✅
Requires Rules/Signatures	✅	❌	✅
Logs Traffic	✅	✅	❌

Future of Snort

- Future insights according to [13] :
- Key Areas for Evolution:
  - Real-Time Detection Capabilities : Enhance Snort's ability to detect emerging threats in real-time and focus on low-latency processing for high-speed networks
  - Integration with Advanced Tools : Strengthen collaboration with tools like Wireshark for detailed packet inspection and event correlation and support for additional security tools like Syslog for more comprehensive threat analysis
  - Scalability and Usability : Overcome current performance limitations (e.g., slow processing, high memory usage) and improve user interface for easier management and interpretation of alerts
  - Signature-Based Detection & Heuristic Enhancement : Develop hybrid detection mechanisms combining signature-based and heuristic analysis and regularly update signature libraries to stay ahead of evolving attack patterns
- Proposed Enhancements:
  - Alert Generation : Integrate proactive alert systems directly into Snort for faster threat response
  - Heuristic Detection : Enhance Snort's heuristic capabilities to detect previously unknown threats using advanced analytics
- Conclusion: The future of Snort relies on ongoing research and development to maintain its relevance in an evolving cybersecurity landscape. By focusing on real-time detection, tool integration, and scalability, Snort will continue to be a critical component in safeguarding network infrastructures.

References

1 Bace, Rebecca Gurley, and Peter Mell. "Intrusion detection systems." (2001)

2 Ring, Markus, et al. "A survey of network-based intrusion detection data sets." Computers & security 86 (2019): 147-167

3 Liu, Ming, et al. "Host-based intrusion detection system with system calls: Review and future trends." ACM computing surveys (CSUR) 51.5 (2018): 1-36

4 Rehman, Rafeeq Ur. Intrusion detection systems with Snort: advanced IDS techniques using Snort, Apache, MySQL, PHP, and ACID. Prentice Hall Professional, 2003

5 Kunhare, Nilesh, Ritu Tiwari, and Joydip Dhar. "Network packet analysis in real time traffic and study of snort IDS during the variants of DoS attacks." Hybrid Intelligent Systems: 19th International Conference on Hybrid Intelligent Systems (HIS 2019) held in Bhopal, India, December 10-12, 2019 19. Springer International Publishing, 2021

6 Zhang, Dongyan, and Shuo Wang. "Optimization of traditional Snort intrusion detection system." IOP Conference Series: Materials Science and Engineering. Vol. 569. No. 4. IOP Publishing, 2019

7 Kurundkar, G. D., N. A. Naik, and S. D. Khamitkar. "Network intrusion detection using Snort." International Journal of Engineering Research and Applications 2.2 (2012): 1288-1296

8 Khamphakdee, Nattawat, Nunnapus Benjamas, and Saiyan Saiyod. "Improving intrusion detection system based on snort rules for network probe attack detection." 2014 2nd International Conference on Information and Communication Technology (ICoICT). IEEE, 2014

9 Kumar, Vinod, and Om Prakash Sangwan. "Signature based intrusion detection system using SNORT." International Journal of Computer Applications & Information Technology 1.3 (2012): 35-41

10 Preethi, T., et al. "A Novel Approach for Anomaly Detection using Snort Integrated with Machine Learning." 2024 11th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, 2024

11 Jin, Shangjie, Meijian Li, and Zhentao Wang. "Research and Design of Preprocessor plugin based on PCRE under snort platform." 2011 International Conference on Control, Automation and Systems Engineering (CASE). IEEE, 2011

12 Bezborodov, Sergey. "Intrusion Detection Systems and Intrusion Prevention System with Snort provided by Security Onion." (2016)

13 Tripathy, Sudhanshu Sekhar, and Bichitrananda Behera. "EVALUATION OF FUTURE PERSPECTIVES ON SNORT AND WIRESHARK AS TOOLS AND TECHNIQUES FOR INTRUSION DETECTION SYSTEM." Available at SSRN 5048278 (2024)

14 Snort logo. © Cisco Systems, Inc. Used for educational purposes

15 Stock/vector icons. Icons used in figures sourced from The Noun Project, Freepik, Vecteezy, and other open icon libraries (individual authors not Retained)

16 Microsoft Azure icons. Icons sourced from official Microsoft Azure icons set, © Microsoft Corporation, used in accordance with their guidelines

Contents

Example 1 – Detecting Port Scans

- Purpose : Port scans are often used by attackers to identify open ports on a network, which can then be exploited for further attacks. Detecting such scans is crucial for network defense.
- Flags SF (SYN-FIN) :
  - The SYN-FIN flag combination is rare in legitimate traffic but common in port scans, especially stealth scans. The attacker sends packets with both SYN (synchronize) and FIN (finish) flags set, a technique often used to evade detection
  - SYN: Initiates a connection request
  - FIN: Signals the end of a connection
- Reconnaissance Detection : Port scanning is typically the first step in a network attack, helping attackers to identify vulnerable services or hosts. This rule helps to spot these reconnaissance attempts early on.
- Usage: Firewalls & IDS/IPS : This rule can be integrated into firewalls or intrusion detection/prevention systems to automatically detect and block port scan activities.
- Considerations : While this rule can detect basic scans, attackers may try to bypass detection with more advanced techniques, like fragmented packets or scanning at a low frequency.
- Additional Insights : Port scanning is a precursor to more serious attacks, such as exploiting vulnerable services (e.g., open SSH or HTTP ports). Detecting these scans helps in early threat mitigation.

```
alert tcp any any -> any any (flags: SF; msg:"Possible Port Scan"; sid:1000002;)
```

Example 2 – Blocking Malicious IPs

- Purpose : Blocking traffic from known malicious IP addresses is a fundamental defensive measure to mitigate direct attacks, such as DDoS, botnets, and unauthorized access attempts.
- Drop Action :
  - The drop action tells Snort to actively prevent packets from the specified IP address, blocking any further communication from that malicious source
  - Traffic Blocked: All traffic from IP 192.168.100.10 to any destination is dropped
- Use Case : Blacklisting attackers or sources of spam, malware, or other malicious activity. Once an attacker's IP is identified, blocking their IP helps prevent further damage.
- Integration with Other Tools : This rule can be automatically triggered by other detection systems (e.g., IPS or SIEM systems) when they identify a malicious activity originating from a specific IP.
- Additional Insights :
  - Blocking IPs manually can be a quick response, but it must be done carefully. Legitimate traffic may also be inadvertently blocked, especially with dynamic IPs.
  - Dynamic Blacklisting: For better automation, consider integrating Snort with threat intelligence feeds to dynamically blacklist IPs involved in active attacks.

```
drop ip 192.168.100.10 any -> any any (msg:"Blocked malicious IP"; sid:1000003;)
```

Example 3 – Detecting SQL Injection

- Purpose : SQL injection (SQLi) is a common web application attack where attackers manipulate SQL queries to gain unauthorized access to databases, steal sensitive information, or execute arbitrary commands.
- Content Filtering :
  - The rule looks for the content "UNION SELECT" in HTTP traffic. This is a classic pattern used in SQL injection attacks to retrieve data from multiple tables in a database.
  - nocase: Ensures that the rule is case-insensitive, as SQL injection payloads can be written in various cases to evade detection.
- Web Application Protection : This rule is useful for detecting basic SQL injection attacks targeting web applications that use SQL queries. If the query contains SQL commands like UNION SELECT, it may indicate an attempt to manipulate the database.
- Usage : Web Application Firewalls (WAF): This rule is useful in conjunction with WAFs for blocking malicious traffic trying to exploit vulnerabilities in web applications.
- Limitations : The rule only detects basic SQL injection patterns. Advanced attackers might use encoded or obfuscated payloads that don't match this rule exactly. Additional rules or more advanced analysis techniques may be required for comprehensive detection.
- Additional Insights : SQL injection attacks remain one of the most common and damaging types of cyberattacks. This Snort rule is a starting point for detecting these attacks, but web applications should also use parameterized queries and prepared statements to defend against SQLi.

```
alert tcp any any -> any 80 (content:"UNION SELECT"; nocase; msg:"SQL Injection Attempt"; sid:1000004;)
```

Contents

Question 1

What is Snort primarily used for?

i. Encrypting network traffic

ii. Detecting network intrusions

iii. Replacing firewalls

iv. Performing penetration testing

Question 1

What is Snort primarily used for?

Explanation: Snort is a Network Intrusion Detection System (NIDS) that monitors traffic in real time and compares packets against rules to detect malicious activity.

i. Encrypting network trafic (× Encryption is handled by VPNs and TLS, not Snort)

ii. Detecting network intrusions (✓Correct: Snort analyzes packets and raises alerts on attacks)

iii. Replacing firewalls (× Snort complements firewalls, it does not replace them)

iv. Performing penetration testing (× Snort is defensive, not an offensive testing tool)

Question 2

Which of the following is NOT a mode in Snort?

i. Sniffer mode
ii. Logger mode
iii. Exploiter mode
iv. Detection mode

Which of the following is NOT a mode in Snort?

Explanation: Snort supports three main operating modes: sniffer, packet logger, and IDS (detection) mode. There is no such mode called “exploiter”.

i. Sniffer mode (✓Correct: Captures and displays packets)

ii. Logger mode (✓Correct: Stores packets for later analysis)

iii. Exploiter mode (× Not a valid Snort mode)

iv. Detection mode (✓Correct: Full IDS mode using rules and signatures)

Question 3

What does a Snort rule use to identify attack patterns?

i. AI learning
ii. Signatures and rules
iii. Random sampling
iv. Firewall policies

Question 3

What does a Snort rule use to identify attack patterns?

Explanation: Snort detects attacks by comparing packets against predefined signatures and anomaly rules.

i. AI learning (× Not native in classical Snort)

ii. Signatures and rules (✓Correct: Core detection mechanism of Snort)

iii. Random sampling (× Detection is deterministic, not random)

iv. Firewall policies (× Firewalls enforce access control, not IDS logic)

Question 4

Which of the following is a limitation of Snort?

i. It prevents all cyber attacks

ii. It has a high rate of false positives

iii. It replaces antivirus software

iv. It requires no configuration

Question 4

Which of the following is a limitation of Snort?

Explanation: Snort requires careful tuning. Poor configuration may generate many false positives, detecting legitimate traffic as malicious.

i. It prevents all cyber attacks (× Snort is IDS, not IPS by default)

ii. It has a high rate of false positives (✓Correct: misconfiguration leads to alert noise)

iii. It replaces antivirus software (× It complements, not replaces, AV)

iv. It requires no configuration (× Snort actually requires precise configuration)

Question 5

Which Snort rule would detect SQL injection attempts?

i. alert tcp any any -> any 80 (content:"UNION SELECT"; nocase; msg:"SQL Injection"; sid:1000004;)

ii. drop ip any any -> any any (msg:"Drop all traffic"; sid:1000005;)

iii. alert udp any any -> any 53 (msg:"DNS query detected"; sid:1000006;)

iv. pass tcp any any -> any any (msg:"Ignore traffic"; sid:1000007;)

Question 5

Which Snort rule would detect SQL injection attempts?

Explanation: SQL injection often includes patterns like UNION SELECT. This rule inspects HTTP traffic on port 80 and alerts when that payload appears.

i. alert tcp any any -> any 80 (content:"UNION SELECT"; nocase; msg:"SQL Injection"; sid:1000004;) (✓Correct: Detects SQL injection content in HTTP traffic)

ii. drop ip any any -> any any (msg:"Drop all traffic"; sid:1000005;) (× Drops all traffic, not SQL-specific)

iii. alert udp any any -> any 53 (msg:"DNS query detected"; sid:1000006;) (× Monitors DNS, unrelated to SQL)

iv. pass tcp any any -> any any (msg:"Ignore traffic"; sid:1000007;) (× Ignores traffic entirely)