EXPT NO: 08 ROLLNO:220701240

DEMONSTRATE INTRUSION DETECTION SYSTEM

AIM:

he aim of this experiment is to understand the working of Snort as an Intrusion Detection and Prevention System (IDS/IPS) by exploring its various operation modes including Sniffer, Logger, NIDS/NIPS, and PCAP analysis. It involves learning the rule structure used in Snort for traffic filtering and alert generation, investigating traffic logs

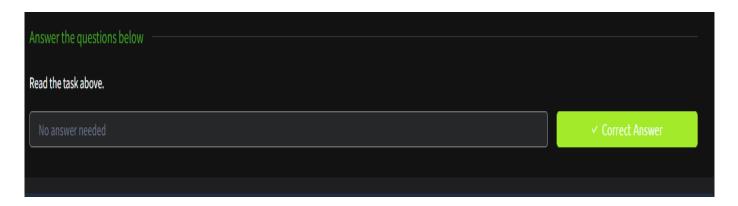
PROCEDURE:

- 1. Study the theory behind IDS/IPS systems and their types.
- 2. Launch Snort in different modes using CLI parameters (Sniffer, Logger, IDS).
- 3. Use sample PCAPs and logs to analyze network traffic.
- 4. Write detection rules to filter specific traffic based on headers, flags, and content.
- 5. Investigate alerts generated by Snort and understand their components.
- 6. Test configuration files and custom rule sets for rule accuracy and performance.

TASK 1 – INTRODUCTION

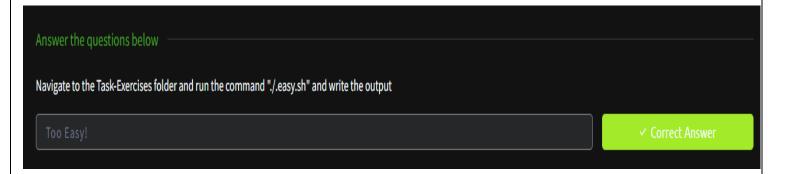
- Snort is an open-source NIDS/NIPS maintained by Cisco Talos.
- It detects malicious traffic using rules and generates alerts.
- Offers live traffic inspection, packet logging, and protocol analysis.
- Can operate in Sniffer, Logger, and IPS modes.
- Cross-platform compatibility with modular architecture.

- Widely used in blue-team and enterprise defense setups.



TASK 2 – INTERACTIVE MATERIAL AND VM

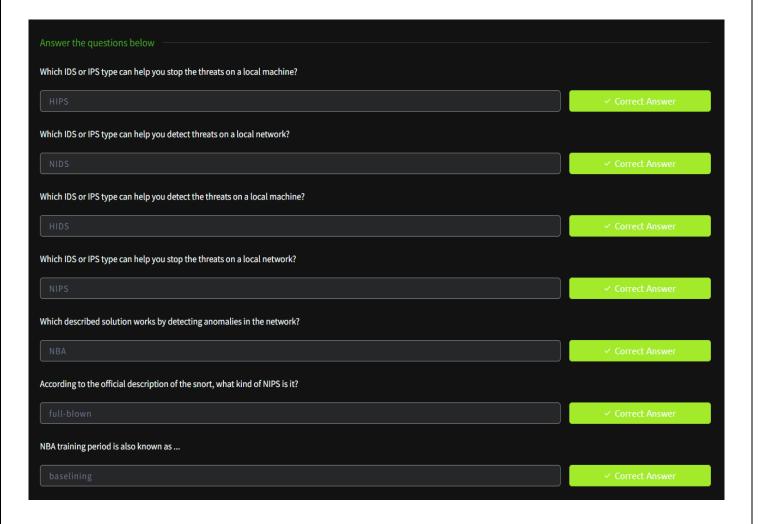
- Run the command `./.easy.sh` in Task-Exercises folder.
- Validates VM setup and script execution permissions.
- Output message verifies readiness: "Too Easy!"
- Ensures user environment is configured to start Snort labs.
- No packet analysis in this task just interaction validation.
- Sets the base for upcoming hands-on tasks.



TASK 3 – INTRODUCTION TO IDS/IPS

- Covers the distinction between NIDS/HIDS and NIPS/HIPS.

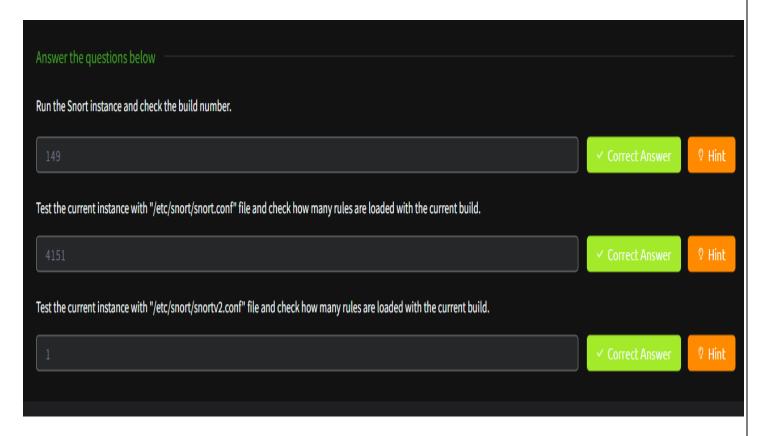
- Behavior-based IPS systems (NBA) require a training period (baselining).
- IPS systems can actively drop or block malicious packets.
- Explains signature-based, behavior-based, and policy-based detection.
- Matches Snort modes to appropriate protection scopes (HIDS, NIDS, HIPS, NIPS).
- Clarifies that Snort is a full-blown IPS with multi-mode functionality.5yyy



TASK 4 – FIRST INTERACTION WITH SNORT

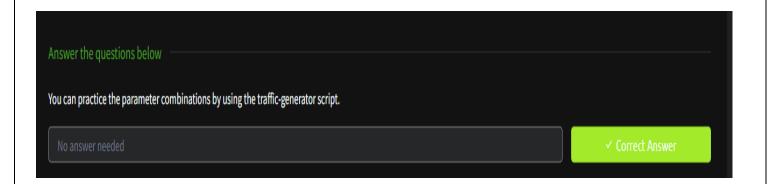
- Use `snort -V` to check Snort version and build.
- Run self-test using `snort -T -c <config>` to verify config validity.
- Load default and alternative configs to compare rule counts.

- `-T` tests configuration files for syntax and rule loading.
- Answers: Build number = 149, Rules loaded (default) = 4151, (v2) = 1.
- Validates setup before real traffic analysis begins.



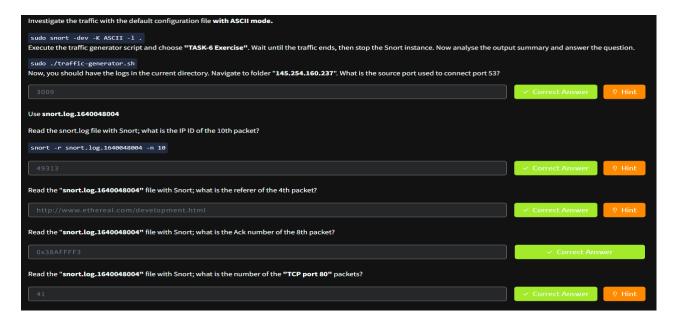
TASK 5 – OPERATION MODE 1: SNIFFER MODE

- Enables real-time packet inspection (like tcpdump).
- Use flags like `-v`, `-d`, `-e`, and `-X` for verbosity and headers.
- `-i` specifies the interface to sniff.
- Allows combining flags for detailed analysis: `-v -d -e`.
- Useful for viewing live traffic payloads and headers.
- No alerting or logging just packet visibility.



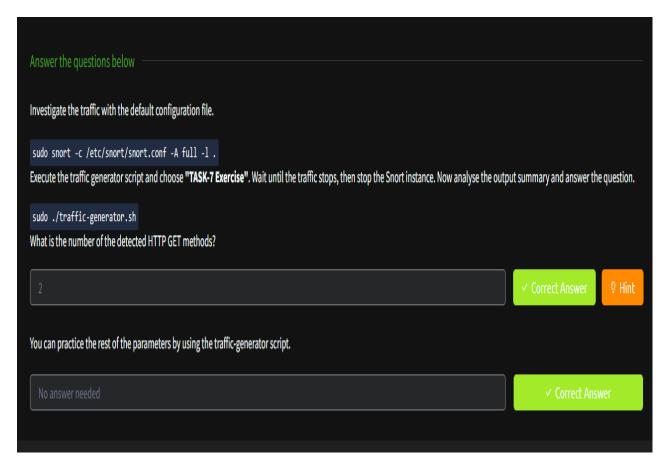
TASK 6 – OPERATION MODE 2: PACKET LOGGER MODE

- Logs packets in ASCII or tcpdump format to disk.
- `-l` specifies the log directory, default is `/var/log/snort`.
- Use `-r` to read logged files and `-n` to limit packets.
- Analyze logs for source ports, IP IDs, ACK numbers, and referers.
- Use filters like BPF to isolate packets (e.g., 'tcp port 80').
- Enables offline packet analysis from previously captured sessions.



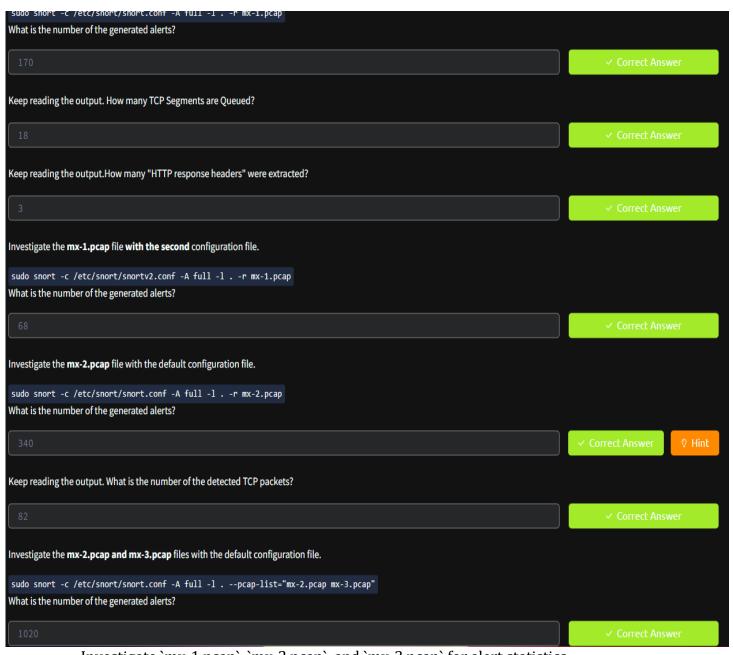
TASK 7 – OPERATION MODE 3: IDS/IPS

- Requires rule files and configuration ('-c <snort.conf>').
- Run with modes like `-A full`, `-A console`, `-A fast` for alert types.
- `-D` runs Snort in background, `-X` enables HEX output.
- Example rule: `alert icmp any any <> any any (msg: "ICMP Packet Found"; sid: 100001; rev:1;)`
- IPS mode: `-Q --daq afpacket -i eth0:eth1` enables inline prevention.
- Example: HTTP GET method count = 2 from generated traffic.



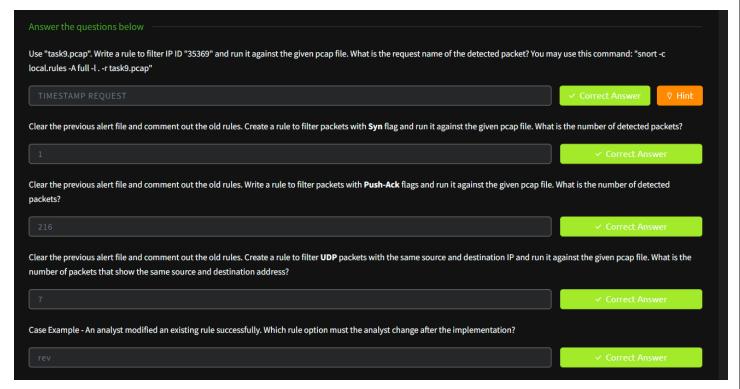
TASK 8 – OPERATION MODE 4: PCAP INVESTIGATION

- Use `-r <file.pcap>` to read PCAP files.
- Supports single and multiple PCAPs using `--pcap-list` and `--pcap-show`.



- Investigate 'mx-1.pcap', 'mx-2.pcap', and 'mx-3.pcap' for alert statistics.
- Analyze TCP segments, HTTP headers, and alert volumes.
- Snort detects alerts based on applied rulesets.
- Enables historical traffic analysis via packet replay.

TASK 9 – SNORT RULE STRUCTURE



- Rules include: action, protocol, source/destination IP & port, options.
- Use `msg`, `sid`, `rev`, `reference` in general rule options.
- Use `content`, `nocase`, `fast_pattern` in payload rules.
- Use `flags`, `id`, `sameip`, `dsize` in non-payload rules.
- Practice rule writing using `task9.pcap` and `local.rules`.
- Detect TCP flags, identical IPs, and payload patterns via custom rules.

TASK 10 - SNORT2 OPERATION LOGIC: POINTS TO REMEMBER

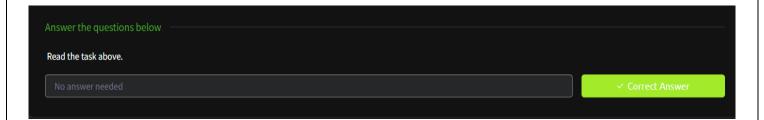
- Components: Packet Decoder, Pre-processors, Detection Engine, Logging, Plugins.
- DAQ modules (afpacket, pcap, nfq) control traffic acquisition.
- Configuration file: `snort.conf', custom rules: `local.rules`.
- Rulesets: Community, Registered, and Subscriber.

- Configuration involves enabling variables, output plugins, and custom rulesets.
- Avoid deleting working rules comment and test incrementally.



TASK 11 – CONCLUSION

- Snort provides multi-mode threat detection and prevention capabilities.
- Learning rule syntax is essential for creating custom detections.
- Test rules in lab before deploying in production.
- Incrementally enhance rules to avoid syntax or logic errors.
- Maintain backups of configuration and rule files.
- Refer to the Snort Challenge and official cheatsheet for continued practice.



RESULT:

Successfully understood the working of Snort in Sniffer, Logger, IDS, and PCAP modes. Gained hands-on experience in writing, applying, and testing detection rules using custom traffic and PCAP data. This equips learners with skills necessary for intrusion detection engineering in real-world environments.