SNORT INTRUSION DETECTION SYSTEM

Syed Muneeb

CODEALPHA

Introduction

With an increasing number of IoT devices connecting to both private and public networks, securing sensitive data has become critical. Snort and other intrusion detection systems (IDS) are important in network security because they monitor traffic, detect suspicious activity, and generate alerts based on predetermined patterns or aberrant behavior. This lab shows how to install, configure, and test Snort's capability, allowing us to build a tailored network defense layer to protect sensitive data delivered and other endpoints.

Objective

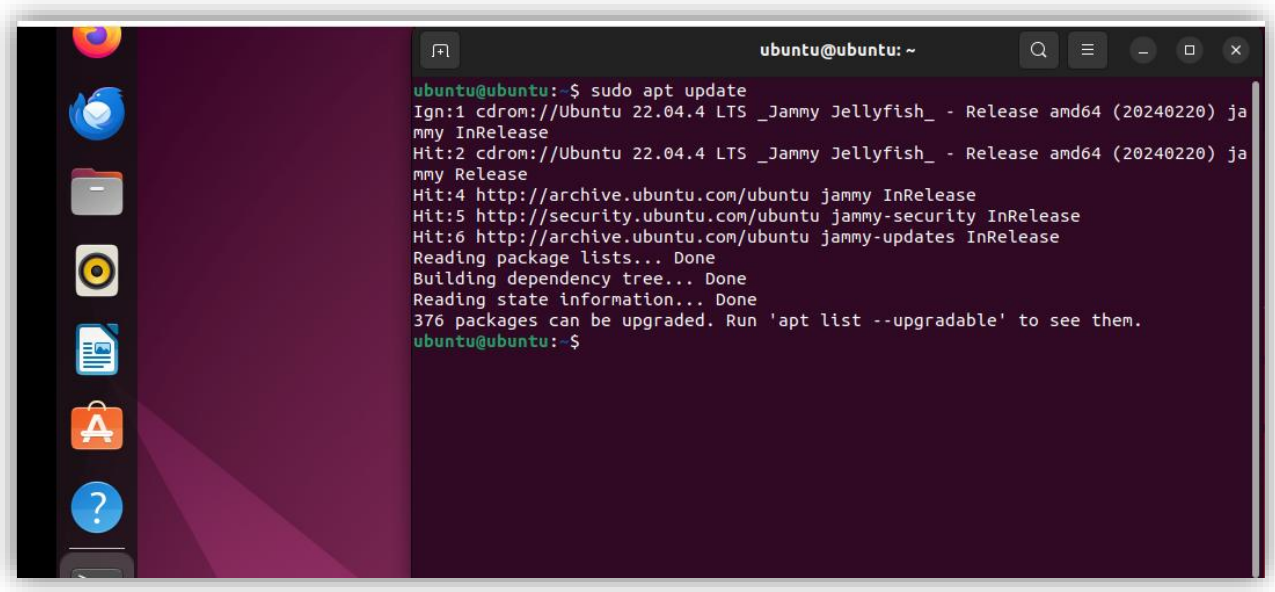
This lab is intended to provide an in-depth overview of Snort's configuration and rule-writing capabilities, covering.

Snort installation on Ubuntu Linux.

Basic configuration and tweaking of Snort's main configuration file. Custom Snort rules can be created to detect specific network traffic, such as ICMP (ping) queries and HTTPS requests to specific URLs. Testing these configurations to watch and analyze Snort's alert creation. By the end, users will have hands-on experience using Snort to actively monitor and safeguard network traffic.

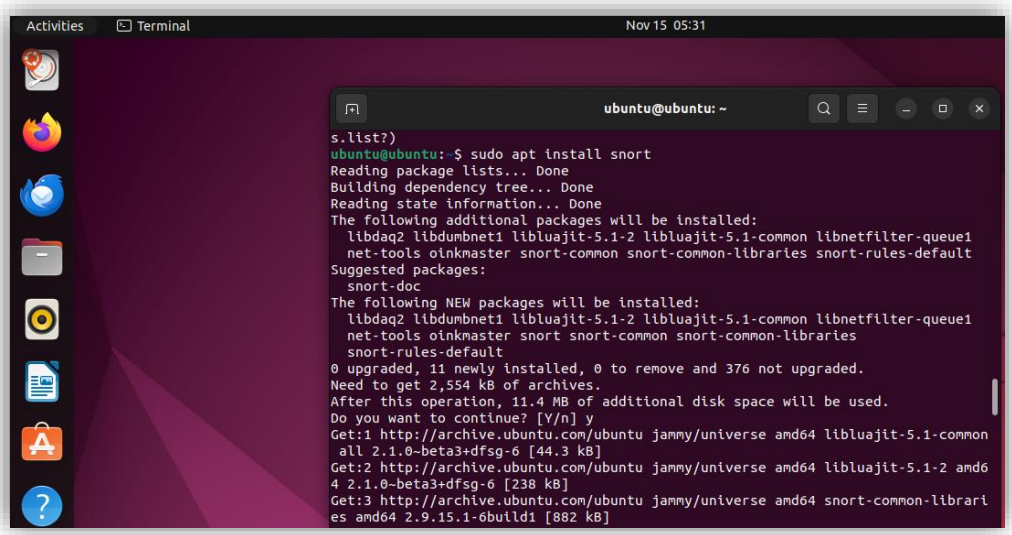
Installing Snort on Linux (Ubuntu)

To ensure a smooth installation, first update the system package list:



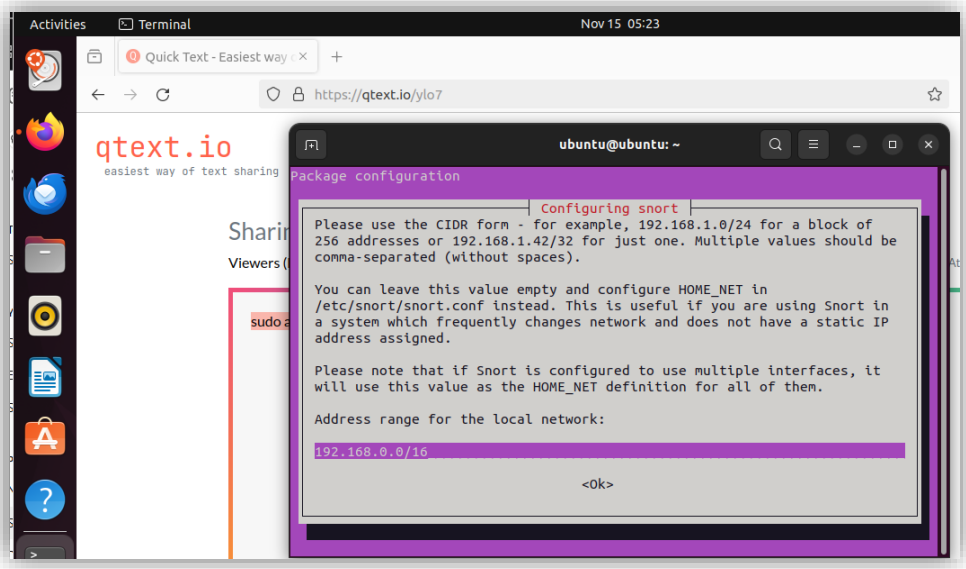
Explanation:

Regularly upgrading packages guarantees that all installed software, including Snort's dependencies, is up to date and free of known security issues. The -y option automatically validates the upgrade, which speeds up the process.



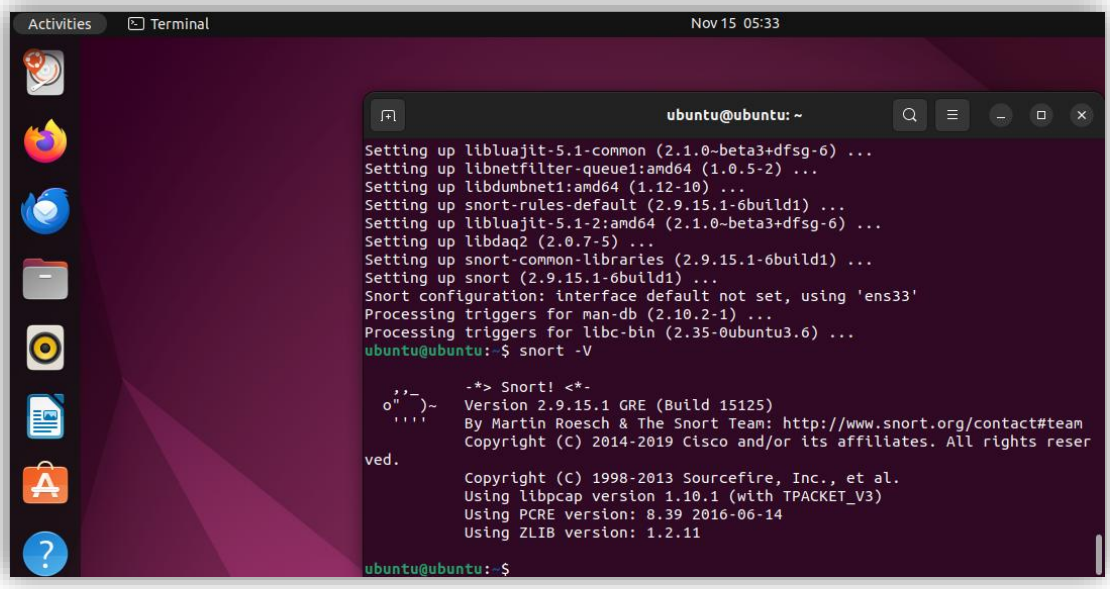
Explanation:

This script leverages Ubuntu's Advanced Package Tool (APT) to locate and install Snort from official repositories, resulting in a clean installation with few compatibility difficulties. The -y parameter automatically accepts prompts, making the installation non-interactive and beneficial when configuring many devices.



Verify the Installation:

Make sure Snort installed successfully by looking at the version.



Justification:

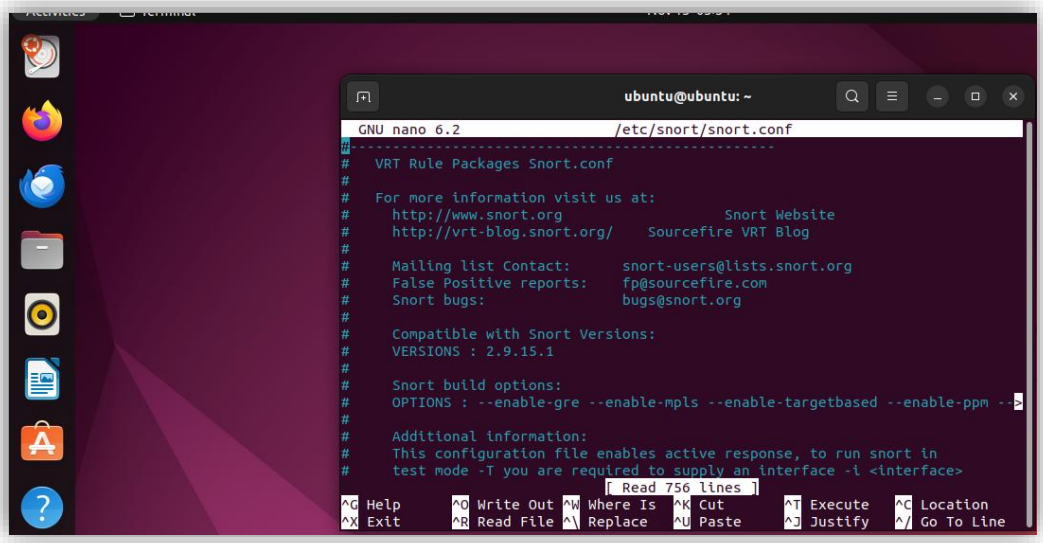
It's important to make sure the right Snort version is installed because different versions may have different features or rule syntax. Verifying the version enables us to correctly refer to Snort's documentation and steer clear of compatibility issues when setting up and creating rules.

Configuring and Starting Snort

Examining the Configuration File for Snort

Snort's primary configuration file, snort.conf, which is found at /etc/snort/snort.conf, is the foundation of its functionality. From creating IP variables to configuring paths for rules, log files, and plug-ins, this file manages Snort's operations.

To open snort.conf, use this command.

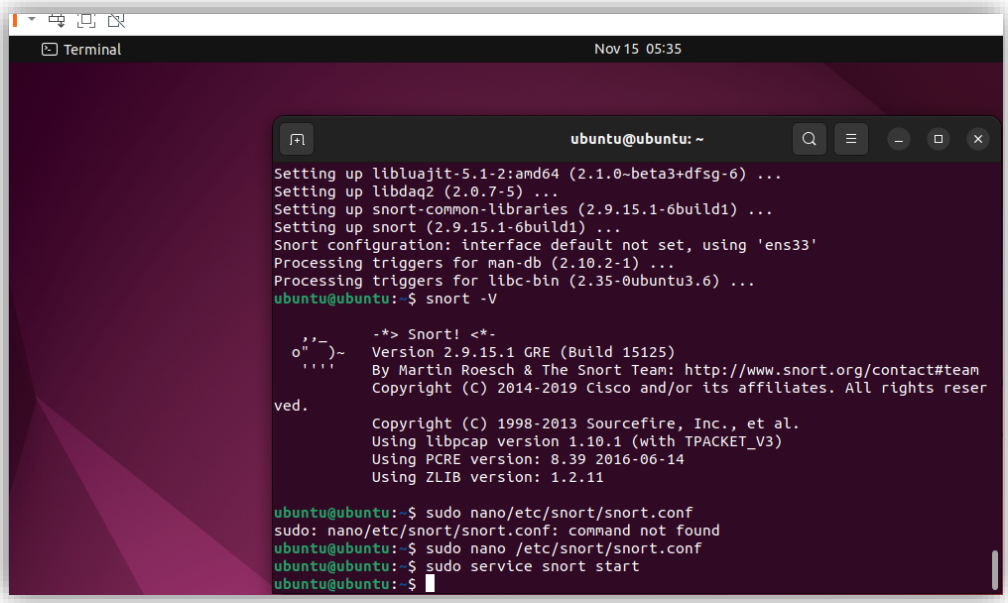


Explanation:

Editing the configuration file directly in the terminal is simple when using nano as a text editor. Sections specifying IP addresses, ports, and the locations of extra rule files can be found in snort.conf. By altering these sections, Snort's focus can be tailored to exclusively monitor particular network segments or types of traffic.

Launching the Snort Program

After configuration, launch the service to activate Snort:



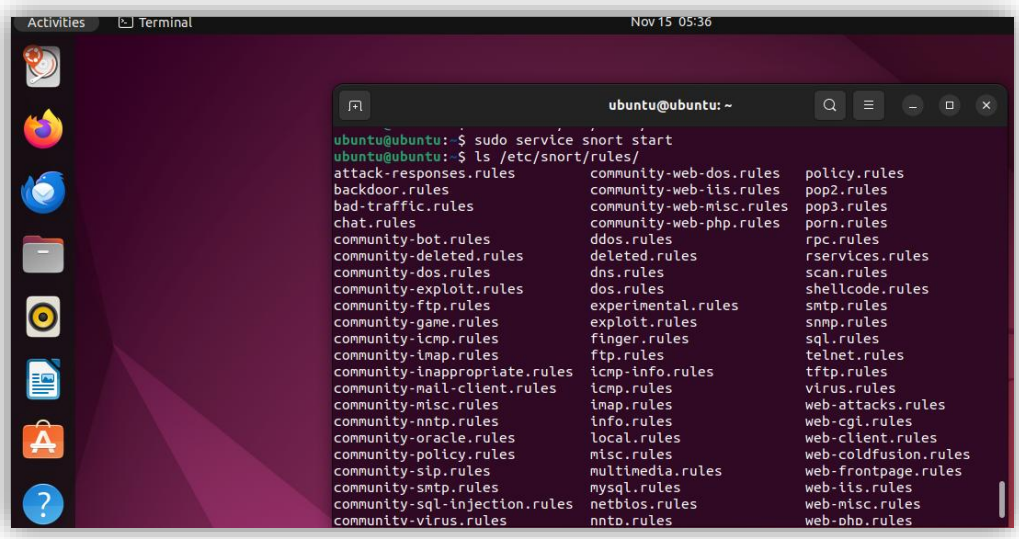
Explanation:

Snort may continually monitor traffic in accordance with the parameters in snort.conf when it is run as a background service. Snort is perfect for continuous intrusion detection in this mode, which is necessary to detect threats in real time across all connected devices.

Finding Snort Rules

Examining the Predefined Rules of Snort

/etc/snort/rules/ contains Snort's rules. Enumerate every rule that is available using:



Justification:

Snort's pre-established rule sets target different kinds of network attacks, such as trojans, viruses, and denial-of-service attacks. These rules serve as Snort's signature-based detection system, which sends out alerts in response to recognized threat indicators or particular packet patterns. Before adding new rules, we can understand Snort's baseline capabilities by reviewing these rules, which provide insight into the types of threats Snort can detect.

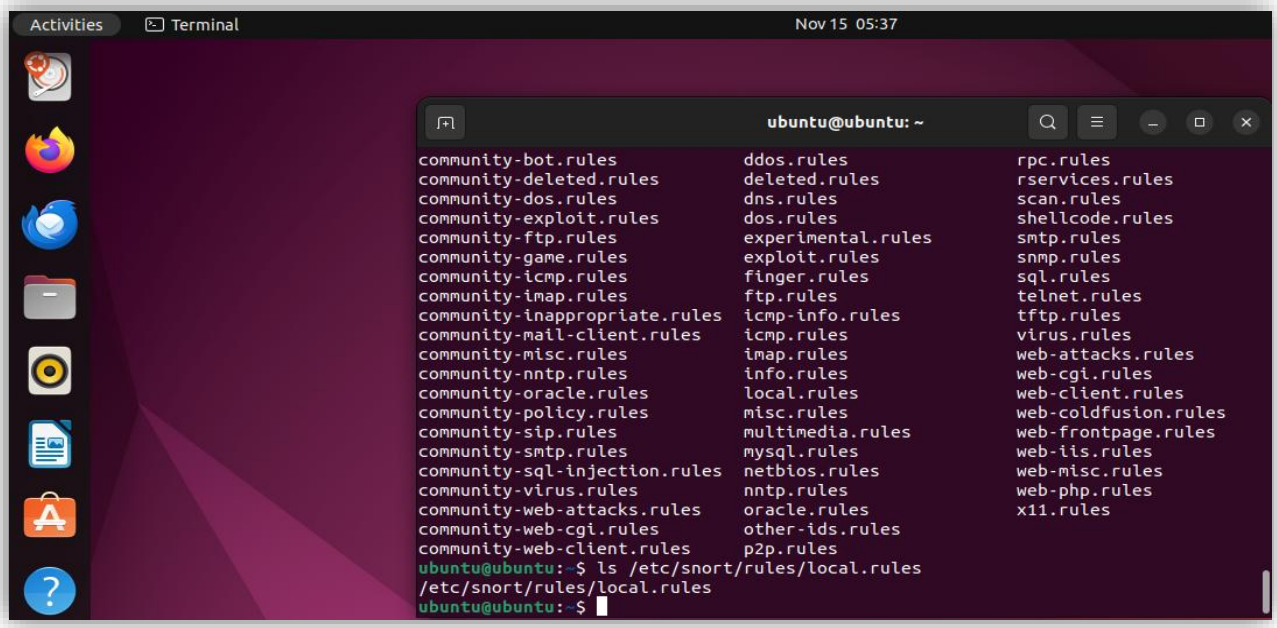
Types of Snort Rules Available

Several rule categories are included with Snort, including:
Malware Rules: Look for patterns in known malware communication. Web Attack Rules: Determine which HTTP requests are malicious and directed at web servers. DoS Attack Guidelines: Identify possible signs of a denial-of-service attack. We can choose extra rules that meet our unique network security requirements by being aware of the default rule kinds.

Writing and Adding a Snort Rule

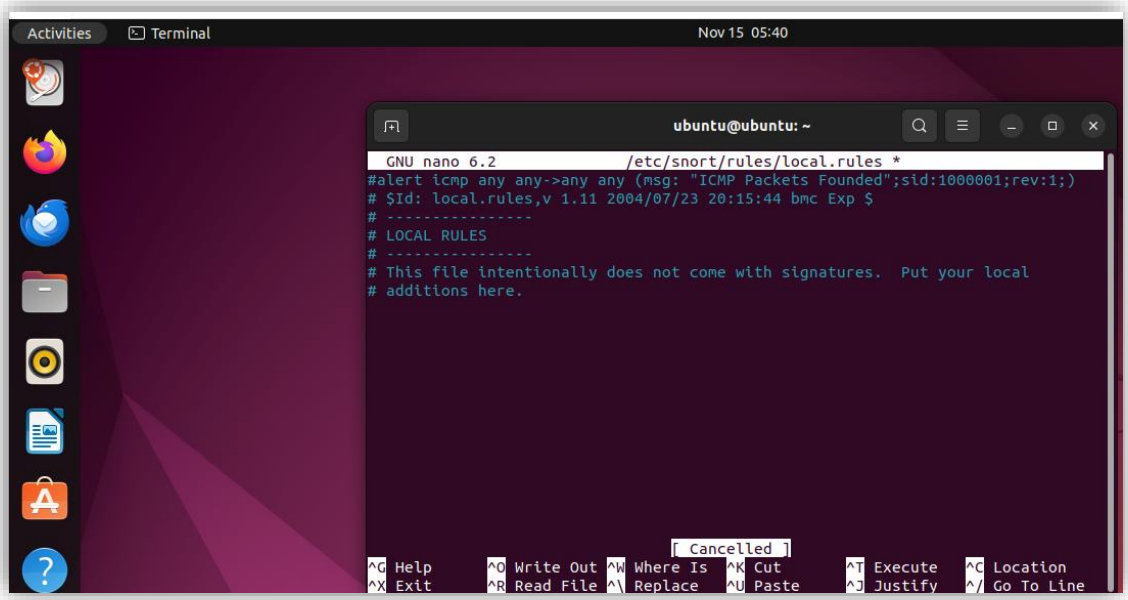
Opening the file local.rules

User-defined rules are stored in the local.rules file. To access it, use.
Justification: Snort's modular rule structure enables the inclusion of new rules in local.rules that are distinct from the default rules. Isolating user-specific rules with local.rules facilitates maintenance and upgrades, particularly when importing new default rules from Snort updates.



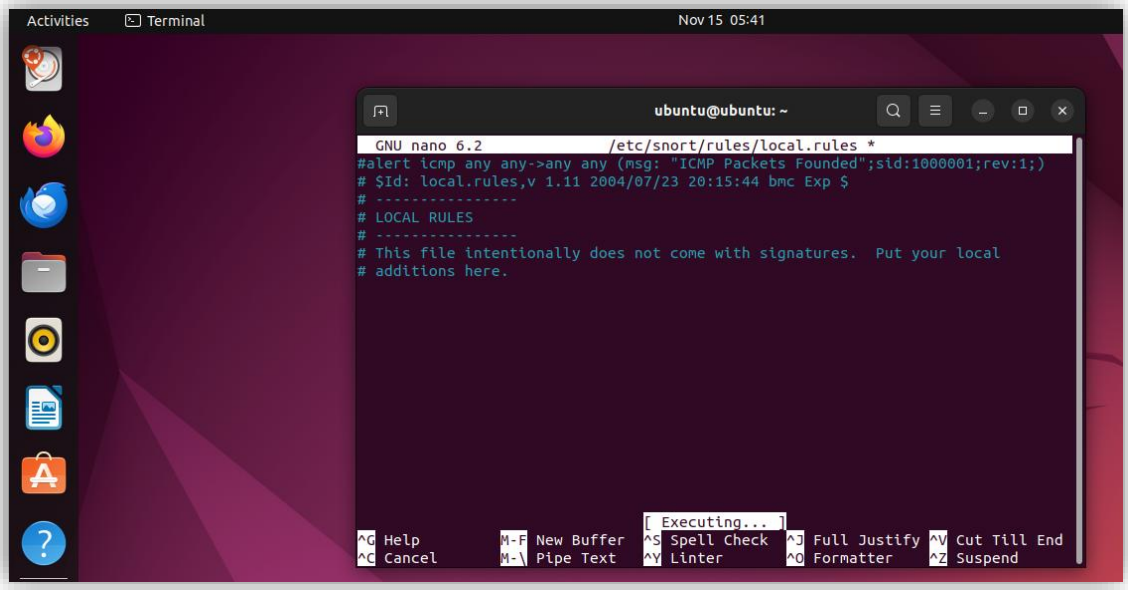
Developing an ICMP Detection Rule

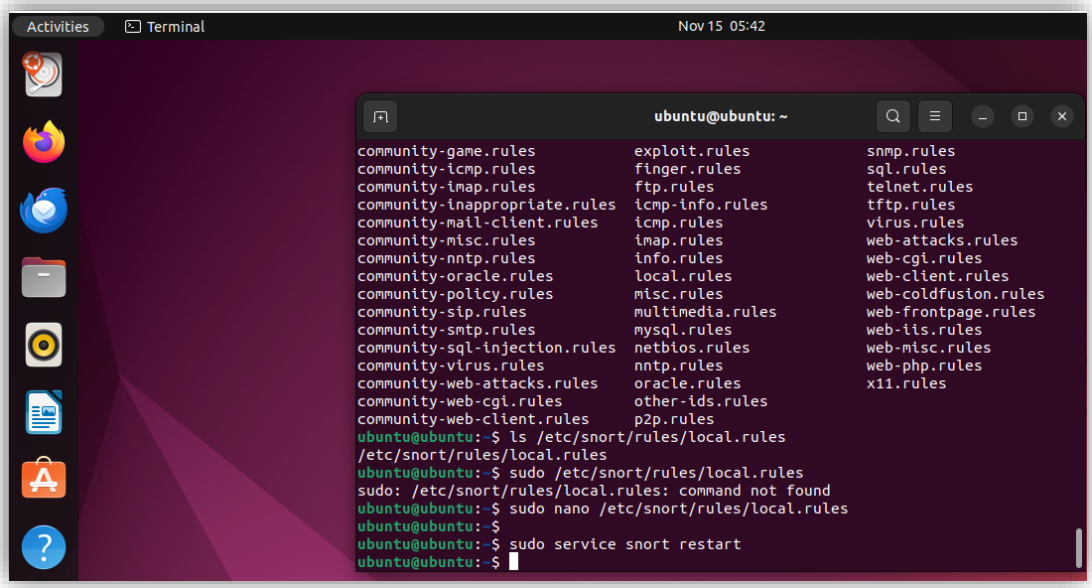
ICMP traffic, which is frequently used for ping testing, is detected by the following rule.



Justification:

ICMP Alerts: Although ICMP packets are helpful for confirming connectivity, they can also be used for reconnaissance or network mapping. When an ICMP packet (like a ping) travels across the network, this rule notifies us.
SID and REV: These IDs are crucial for versioning and tracking rules. Each rule has its own SID (Snort ID), and REV keeps note of changes to allow for future modifications without repeating rules.



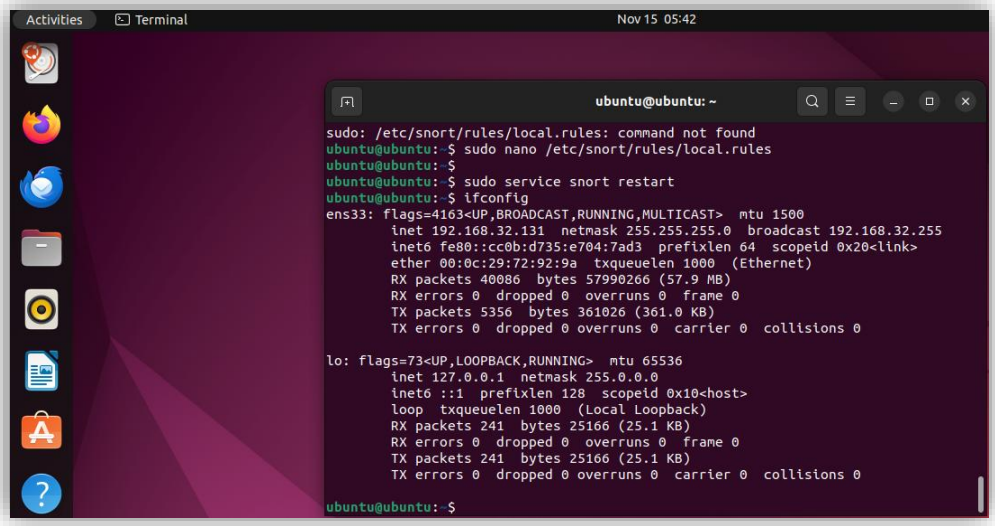


Reason:
By restarting, you may make that Snort reads and uses the most recent version of the local.rules file. Snort will continue to utilize the prior rule set without restarting, so any modifications made to local.rules will be lost.

Testing the ICMP Detection Rule

Discovering the Snort System's IP Address

The Snort system's IP address can be found using the ifconfig command.

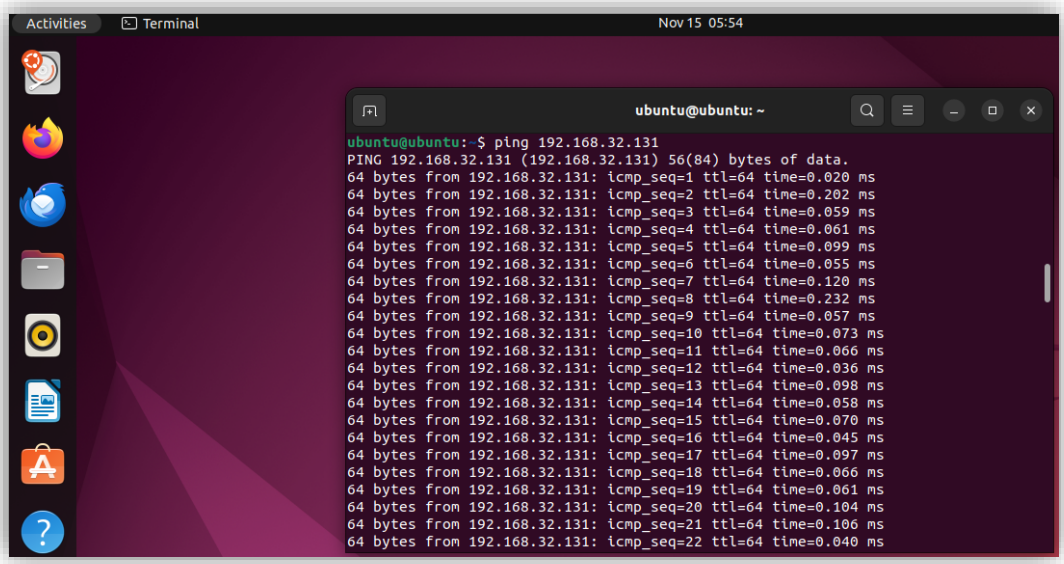


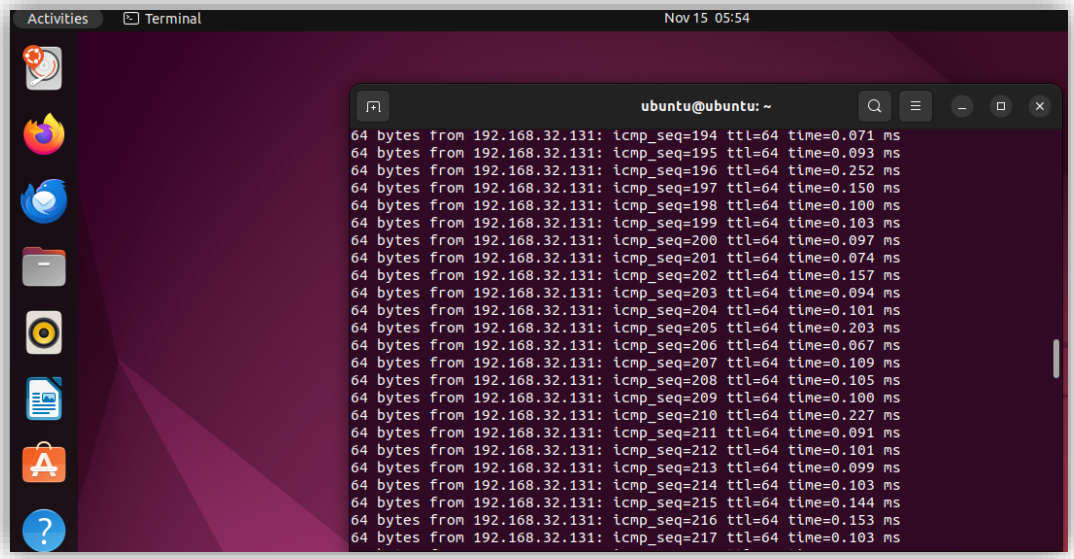
Explanation:

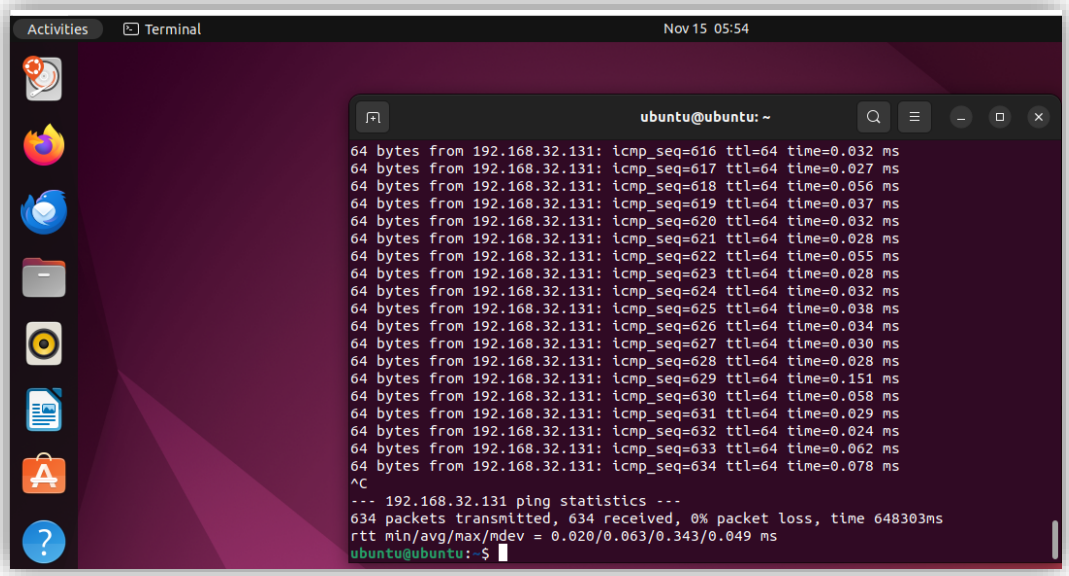
By providing traffic to a certain interface, we may test rules by knowing the Snort system's IP address. By restricting analysis to important traffic flows, this phase helps network administrators to choose precise segments for monitoring in production, which lowers resource consumption.

Transmitting ICMP Data

Ping Snort from a different computer to test the ICMP rule





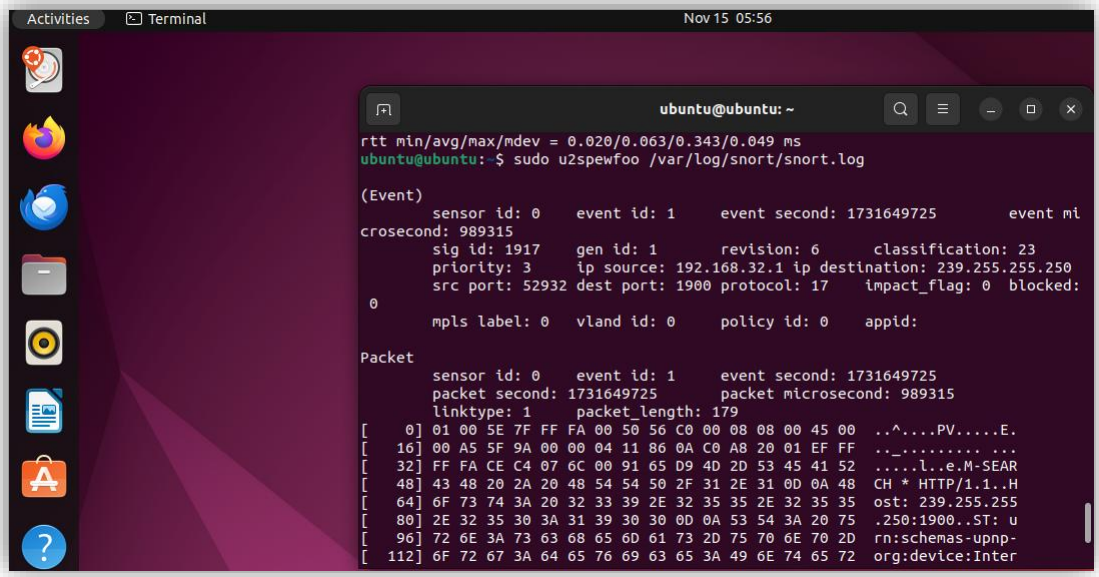


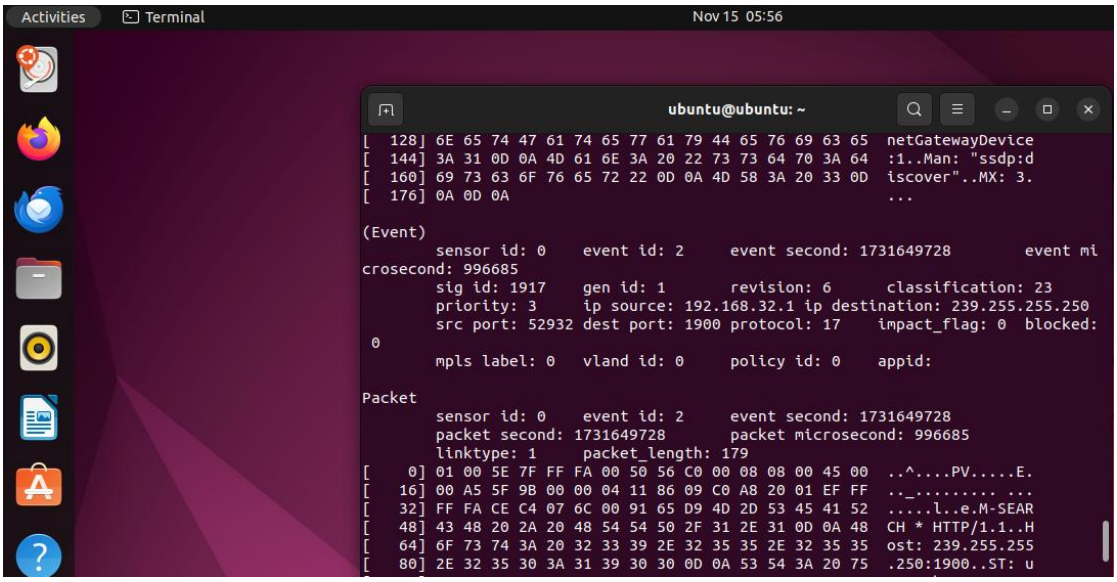
Justification:

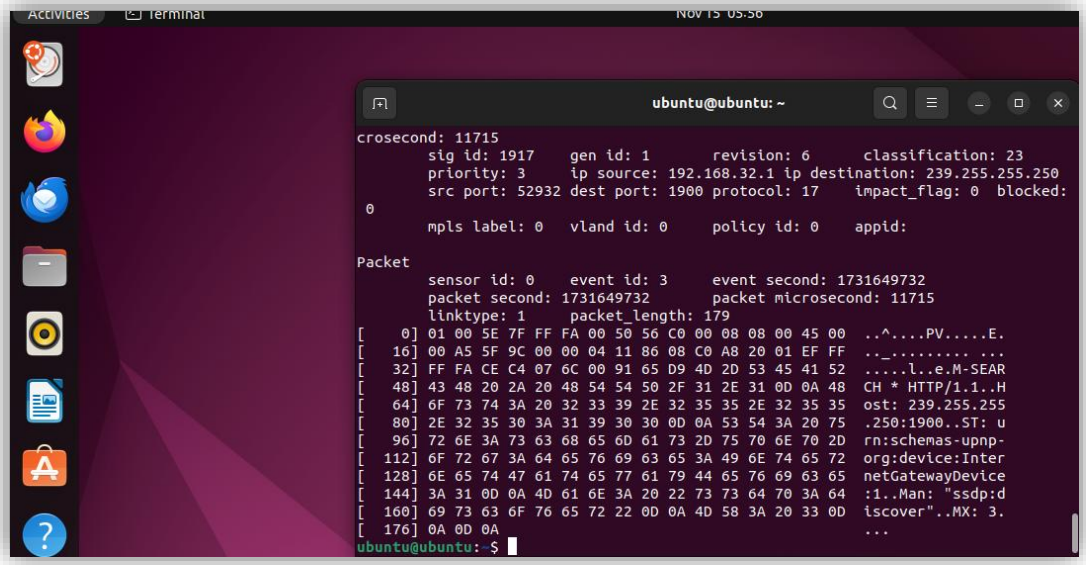
Snort should record an alert if this ICMP traffic matches our detection rule. Although ICMP requests are frequently innocuous, they can occasionally be used by attackers as reconnaissance tools.

Examining Alert Records

Use u2spewfoo to view Snort's alarm logs.







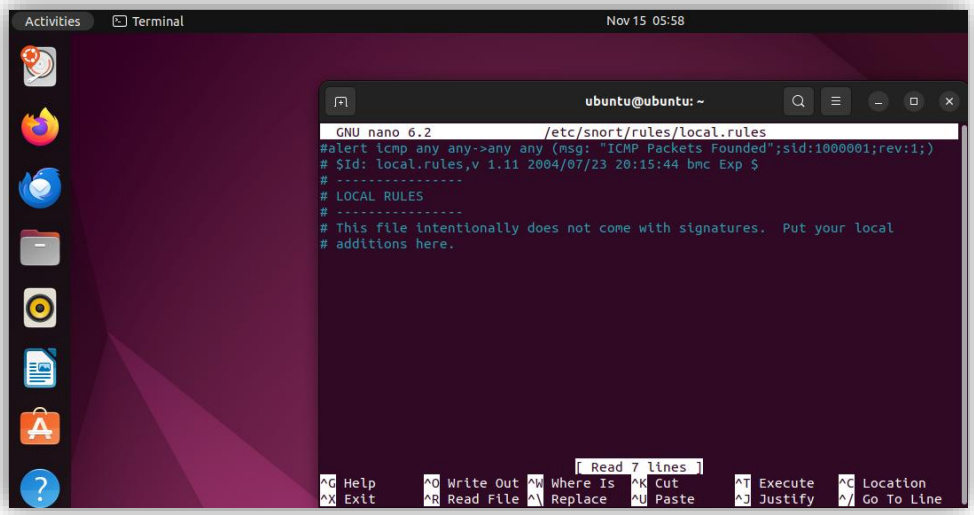
Explanation:

The tool u2spewfoo is used to decode the binary log format of Snort. By checking the alert log, you can verify whether our custom ICMP rule was activated and that Snort is actively keeping an eye out for unwanted traffic in accordance with our configurations.

Writing a Rule to Trigger an Alert for HTTPS Traffic to a Specific URL

Including a Rule for HTTPS

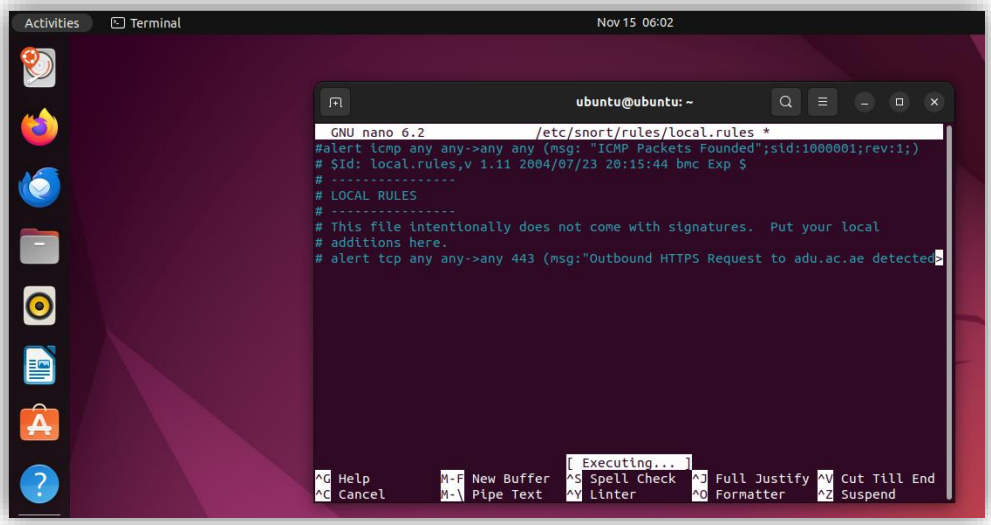
Add the following rule to local.rules to track HTTPS requests to a particular website.



Justification:

Content Correspondence: Within HTTPS traffic, content: "GET / HTTP/1.1"; content: "Host: www.adu.ac.ae"; targets HTTP headers. Snort can examine packet payloads thanks to content-based rules, which is helpful for identifying certain application-layer requests. Details of the Port (443): We concentrate on encrypted HTTPS communication by restricting detection to port 443. Security teams can spot possible exfiltration or unusual connections concealed by encryption by using HTTPS monitoring.

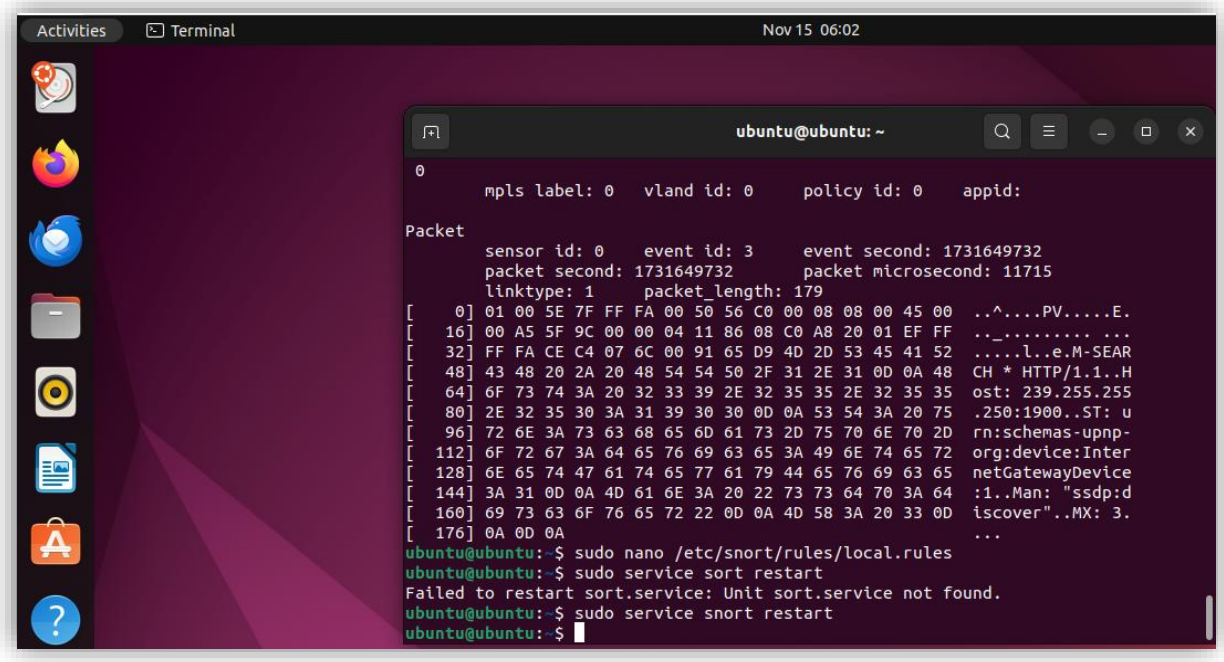
Restart Snort to Activate the New Rule:

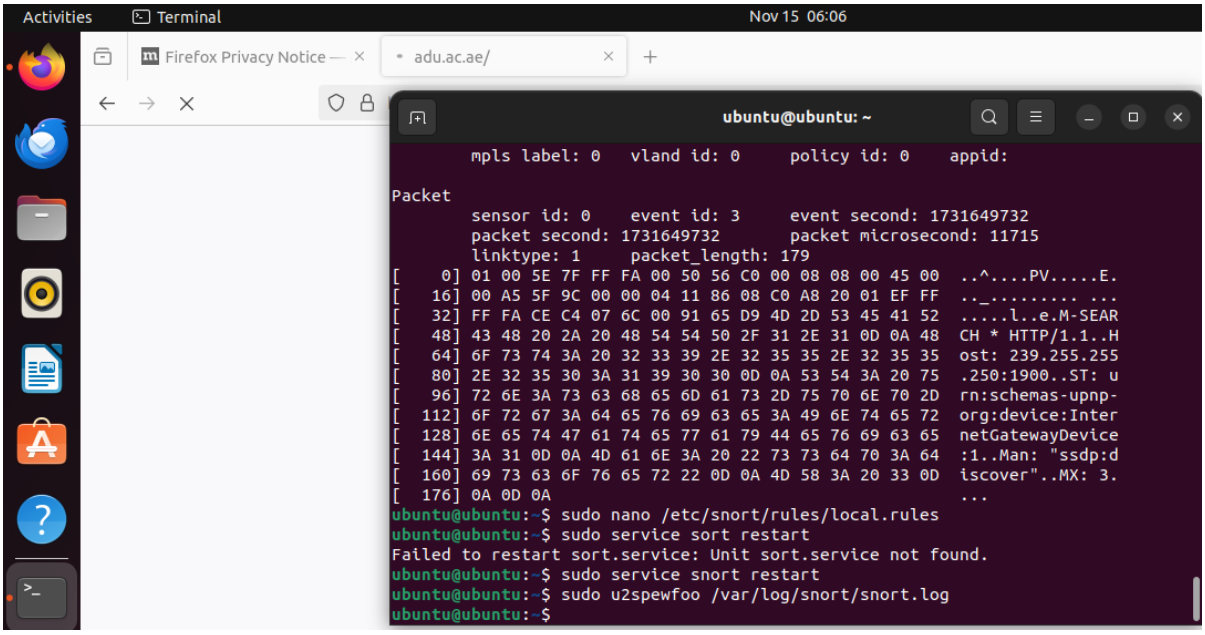


Turn on the HTTPS Rule and restart Snort

After restarting Snort, go to https://www.adu.ac.ae to activate the rule. Explanation: Network traffic with the appropriate headers is produced when the given URL is browsed. This rule can

serve as an early alert for hacked computers trying to visit dubious domains by detecting anomalous HTTPS traffic.





conclusion

In the quickly changing world of technology, the rise in the use of mobile devices has created both enormous security challenges and previously unheard-of convenience. Strong security measures like Intrusion Detection Systems (IDS) must be put in place because of the increased risk of bad

actors taking advantage of these devices as they link to both public and private networks. Strong open-source intrusion detection system (IDS) Snort has established itself as an essential tool for protecting networks against a wide range of attacks. This lab report demonstrated Snort's usefulness in improving network security by thoroughly examining its installation, configuration, and rule modification.

Snort, which provides real-time traffic analysis and threat detection capabilities, is an essential tool for contemporary network security. We investigated its installation, configuration, and customization in this lab, showcasing its capacity to identify and notify users of a range of network activity. We demonstrated Snort's adaptability in monitoring particular traffic patterns, such ICMP packets and HTTPS requests, by customizing its rules and parameters. These practical experiments demonstrated how Snort evolves from a general-purpose intrusion detection system to a specialized defense mechanism that can handle particular security threats.

The useful knowledge acquired from this lab highlights Snort's function in protecting a variety of settings, such as enterprise and mobile networks. We were able to identify particular dangers, like illegal HTTPS connections or reconnaissance efforts, thanks to custom rules, and recording methods gave us useful information for threat research. Snort continues to be an essential part of layered security solutions, guaranteeing strong defense against changing cyberthreats, despite issues like false positives and resource limitations. This exercise gives us useful abilities for practical security applications and emphasizes the significance of proactive network monitoring.