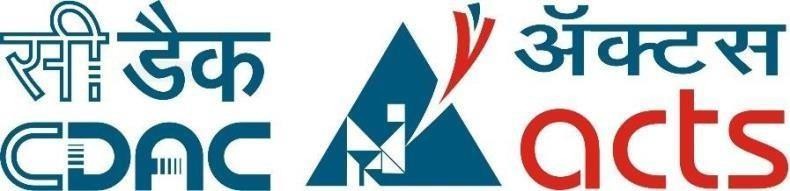
# Project Report On

Defense In Depth



Submitted in fulfilment for the award of

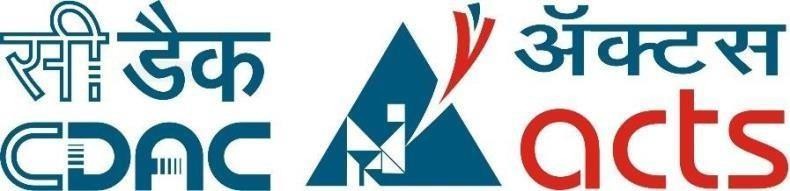
### Post Graduate Diploma in IT Infrastructure System & Security (PG-DITISS) from SUNBEAM INFOTECH (Hinjewadi Pune)

**Guided By:**

Mr. Sandeep Walvekar Sir

|  |  |  |
| --- | --- | --- |
|  | **Presented By:** |  |
|  |  |  |
| Kshitij Shrivastava |  | PRN: 240344223016 |
| Piyush Tiwari |  | PRN: 240344223027 |
| Samarth Tamrakar |  | PRN: 240344223032 |
| Saloni Junghare |  | PRN: 240344223031 |

**Centre of Development of Advanced Computing (C-DAC), Pune**



# CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

**Kshitij Shrivastava PRN: 240344223016**

**Piyush Tiwari PRN: 240344223027**

**Samarth Tamrakar PRN: 240344223032**

**Saloni Junghare PRN: 240344223031**

Have successfully completed their project on

“Defense In Depth”

Under the guidance of Mr. Sandeep Walvekar

### Project Guide Project Supervisor

Mr. Sandeep Walvelar Mr. Vishal Salunkhe

### HOD SUNBEAM

Mr. Nilesh Ghule

# ACKNOWLEDGEMENT

This project “Defense in Depth” was a great learning experience for us and we are submitting this work to Advanced Computing Training School (SUNBEAM INFOTECH HINJWADI).

We all are very glad to mention the name of *Mr. Sandeep Walvekar* for his valuable guidance to work on this project. His guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

We are highly grateful to *Mr. Nilesh Ghule.* (Manager, SUNBEAM training Centre), C- DAC, for her guidance and support whenever necessary while doing this course Post Graduate Diploma in *IT Infrastructure, Systems and Security (PG-DITISS)* through C- DAC ACTS, Pune.

Our most heartfelt thanks go to *Mr. Vishal Salunkhe* (Course Coordinator, PG- *DITISS*) who gave all the required support and kind coordination to provide all the necessities like required hardware, internet facility and extra Lab hours to complete the project and throughout the course up to the last day here in C- DAC ACTS, Pune.

Sincerely,

Kshitij Shrivastava Piyush Tiwari

Samarth Tamrakar

Saloni Junghare

# ABSTRACT

Applying the concept of Defence in Depth, we have demonstrated a vulnerable login website connected with a database in a network. We have tried to implement security concepts like DMZ, IDS-IPS, Proxy, Firewalls, Digital certificates at each layer inside our own network created with the help of a wireless-router.

To demonstrate layered security in our network, we have used various tools like Iptables, Nagios, Snort, Apache2, MySQL, Nginx, DNS, XCA and a wireless-router. We have tested our login website against some of the most common cyber-attacks like SQL Injection, Cross-Site Scripting, Brute Force, Dictionary Attacks and Session Hijacking with the help of Burp-suite and ZAP tool.

Thus, we have tried to implement and learn many different key concepts that we have studied in our course and bring them together to emulate an infrastructure. We have managed our project by deploying SCRUM FRAMEWORK and tried to showcase how Information Security can be achieved in a network.

# TABLE OF CONTENTS

### Introduction and Overview of Project

* 1. Purpose
  2. Aims and Objective

### Overall Description

* 1. Introduction
  2. Architecture

### System Requirements

* 1. Software Requirement
  2. Hardware Requirement
  3. Platform Used

### Installation & Screenshots

* 1. Installation of Strong Swan.
  2. Installation of Snort and Nagios.
  3. Installation of IP Tables.
  4. Configuration of IPSEC Protocol.

1. **Conclusion & Future Scope**
2. **References**

# INTRODUCTION AND OVERVIEW OF PROJECT

### Purpose

The Purpose of defense in depth is to create a multi-layered security framework that increases the complexity and difficulty for attackers to breach a system.

### Aims and Objective

The overall goal of defense in depth is to create a robust security posture that protects an organization’s assets through a combination of preventive, detective, and responsive measures This project also aims to set the best-practice of cybersecurity standards for Defense in Depth.

Objectives:

* + Implement multiple layers of security to prevent unauthorized users from accessing
  + Sensitive data and systems.
  + Establish mechanism to quickly detect suspicious activities or breaches and respond

Effectively.

* + Minimize the area within the system that are vulnerable to attack by hardening systems, applying patches, and removing unnecessary services and applications.

# OVERALL DESCRIPTION

### Introduction:

Defense in depth is a comprehensive cybersecurity strategy that uses multiple layers of protection to secure a system or network. This approach is based on the idea that no single security measure is sufficient to protect against all potential threats. By deploying multiple defensive measures at different levels, organizations can improve their overall security posture and reduce the risk of successful attacks. Here's an overview of the key components and principles of defense in depth:

** Layered Security**: The core principle of defense in depth is to use multiple layers of security controls. These layers are designed to protect different aspects of a system and provide redundancy in case one layer fails. The layers can include network security, endpoint security, application security, data security, and physical security.

** Preventive Controls**: These controls aim to prevent security incidents from occurring in the first place. Examples include firewalls, antivirus software, intrusion prevention systems (IPS), and access controls.

 **Detective Controls**: These controls are designed to identify and detect security incidents as they occur. They include intrusion detection systems (IDS), security information and event management (SIEM) systems, and monitoring and logging tools.

 **Corrective Controls**: Once a security incident is detected, corrective controls help mitigate the impact and restore systems to normal operation. These controls include incident response plans, backup and recovery solutions, and patch management.

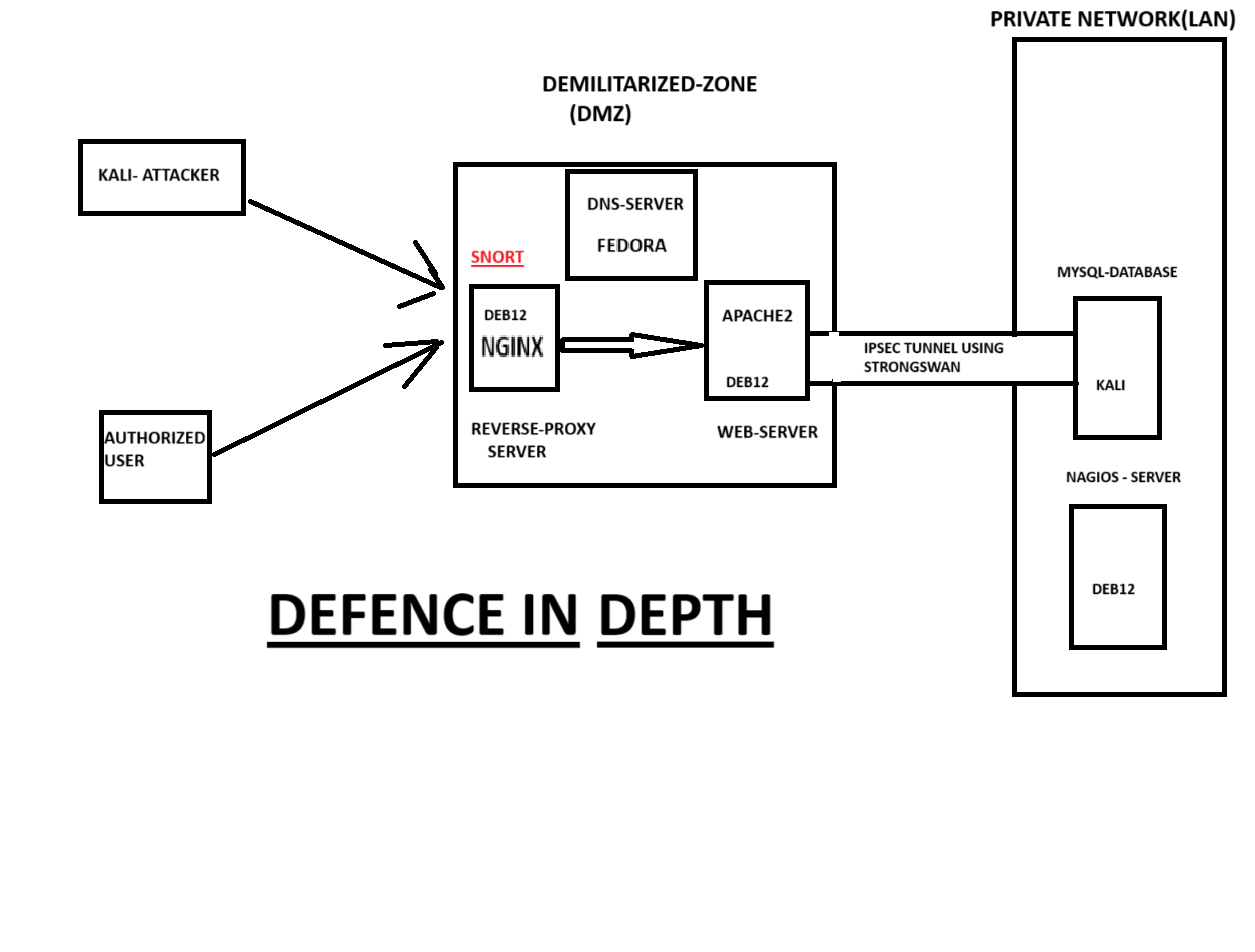
 **Diversity and Redundancy**: Defense in depth emphasizes using diverse security technologies and approaches to avoid single points of failure. Redundancy ensures that if one control fails, others are in place to provide protection.

** People and Processes**: Defense in depth is not just about technology. It also involves educating and training employees on security best practices and establishing robust security policies and procedures.

 **Physical Security**: Protecting the physical components of a system, such as servers and data centers, is an essential part of defense in depth. This includes measures like access controls, surveillance, and environmental controls.

 **Continuous Improvement**: Défense in depth is an ongoing process that requires regular assessment and updates. Organizations must continuously evaluate their security posture, adapt to new threats, and improve their defenses.

 **Risk Management**: Understanding and managing risks is a critical aspect of defense in depth. Organizations need to identify their most valuable assets and prioritize security measures based on risk assessments.



### Difference between Forward Proxy and Reverse Proxy:

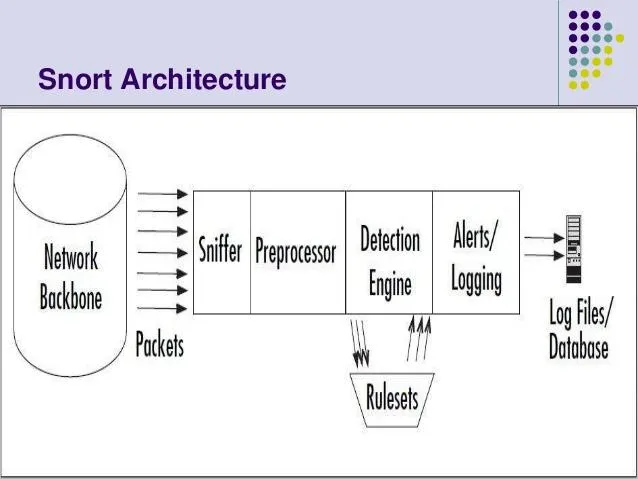
|  |  |  |
| --- | --- | --- |
| **Features** | **Forward Proxy** | **Reverse Proxy** |
| **Purpose** | Provides anonymity and caching to client | Improves Server Performance and Load Balancing and Security. |
| **Location** | Between the client and the Internet | Between the Internet and Server. |
| **Visibility** | The Client is aware of the proxy. | The Server is not aware of the Proxy. |
| **Configuration** | The client must be configured to use proxy. | Server must be configured to use proxy use. |
| **Use cases** | By Passing content filters, accessing restricted content | Load Balancing, caching, SSl, TLS off-loading web application Firewall |
| **Examples** | Squid, Proxy , Tor | Nginx, Apache, HAproxy. |
| **Directionality** | It Handles the Outgoing Request. | It handles the incoming request. |

**2.SNORT-**

Snort is an open-source intrusion detection and prevention system (IDS/IPS), to monitor network traffic and protect against potential threats. Snort was configured in two primary modes: alert and drop. The alert mode allows Snort to detect malicious activities by analyzing network packets against predefined rules and then generate alerts when suspicious behavior is identified. This is crucial for real-time monitoring and threat detection, as it provides insights into potential attacks such as port scans, SQL injections, or unauthorized access attempts.

In addition to alerting, I utilized Snort’s drop mode to actively prevent attacks. When a packet matches a rule configured with a drop action, Snort not only generates an alert but also discards the packet, effectively stopping the malicious traffic from reaching its destination. This adds a layer of security by not only identifying but also mitigating threats in real-time.

This dual approach of alerting and dropping provides comprehensive protection for the network, enabling both visibility into potential security incidents and proactive defense mechanisms. Through this project, I demonstrated how Snort can be effectively employed to enhance network security by detecting and preventing common network attacks.



### SQL: -

T SQL (Structured Query Language) as the core tool for managing and interacting with the database. SQL played a crucial role in storing, retrieving, and manipulating data efficiently, which is essential for the smooth operation of the system. I used SQL to create and maintain the database schema, defining tables, relationships, and constraints that ensure data integrity and consistency.

One of the primary functions of SQL in my project was to handle user data, including registration, login authentication, and user profile management. SQL queries were used to insert new user records, update existing information, and retrieve user data based on specific criteria. This allowed for dynamic interaction with the database, enabling the application to respond to user inputs and provide relevant information.

Additionally, SQL was employed to perform complex queries that involved multiple tables and conditions. This was particularly useful for generating reports, analyzing trends, and extracting meaningful insights from the data. By leveraging SQL’s capabilities, I was able to ensure that the database operations were both efficient and secure, with optimized queries and appropriate indexing to enhance performance.

Overall, SQL was an integral part of the project, enabling robust data management and seamless interaction between the application and the database. Its versatility and power made it the ideal choice for handling the data-driven aspects of the project.

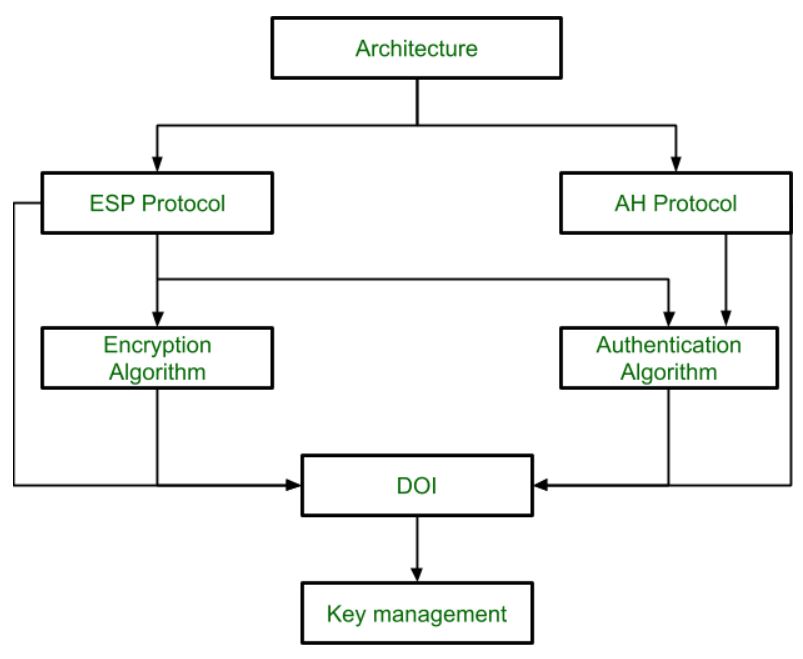
**IPSEC: -**

IPsec (Internet Protocol Security) to establish a secure connection between the web server and the MySQL database. IPsec was chosen for its robust security features, which include data encryption, integrity, and authentication, ensuring that sensitive data transmitted between the web server and the database remains protected from unauthorized access and tampering.

The primary goal of using IPsec was to safeguard the communication channel between the two servers, especially since they are hosted on separate devices. By configuring an IPsec tunnel, all data exchanged between the web server and the MySQL database is encrypted, making it unreadable to potential eavesdroppers. This encryption is critical in preventing attacks such as man-in-the-middle (MitM) and data interception, which could compromise the integrity and confidentiality of the information.

Moreover, IPsec ensures that only authenticated and trusted devices can communicate, providing an additional layer of security through mutual authentication. This prevents unauthorized systems from accessing the database, thereby protecting the web application’s backend from potential breaches.

The implementation of IPsec in this project not only enhances the security of the data transmission but also aligns with best practices for protecting sensitive information in a distributed architecture. By leveraging IPsec, I was able to create a secure, reliable, and efficient connection between the web server and the MySQL database, contributing to the overall resilience of the system.



### NAGIOS: -

A Nagios is an open-source IT infrastructure monitoring tool that provides comprehensive monitoring and alerting for servers, networks, applications, and services. It is widely used by IT administrators to ensure that critical systems are running smoothly and to detect potential issues before they affect users. Here’s a detailed description of Nagios and its key features:

Nagios is designed to monitor and manage the availability and performance of IT infrastructure components. It helps organizations maintain high uptime by providing real-time alerts and detailed insights into system health. With Nagios, administrators can proactively address issues, perform root cause analysis, and plan for capacity needs.

### Key Features

1. **Monitoring Capabilities**
   * **Server Monitoring**: Nagios can monitor Windows, Linux, and Unix servers to track CPU load, memory usage, disk space, and more.
   * **Network Monitoring**: It supports monitoring of network devices such as routers, switches, and firewalls, checking for connectivity, latency, packet loss, and bandwidth utilization.
   * **Application Monitoring**: Nagios can monitor applications and services, including web servers, databases, and custom applications, ensuring they are running and performing optimally.
   * **Log Monitoring**: It can analyze log files for specific patterns and alert administrators to potential security threats or operational issues.
2. **Alerting and Notifications**
   * Nagios provides customizable alerting options to notify administrators of issues via email, SMS, or other methods. This ensures that critical issues are addressed promptly.
3. **Visualization and Reporting**
   * **Dashboards**: Nagios offers web-based dashboards that provide a comprehensive view of the IT infrastructure, with status information, performance data, and visual graphs.
   * **Reports**: Generate reports on availability, historical performance, and trends to help with capacity planning and decision-making.
4. **Extensibility**
   * **Plugins**: Nagios uses a flexible plugin architecture that allows administrators to extend its capabilities. There are thousands of community-contributed plugins available for monitoring various systems and applications.
   * **Custom Scripts**: Users can write custom scripts to monitor specific parameters unique to their environment.
5. **Scalability**
   * Nagios can scale to monitor large and complex environments. It supports distributed monitoring setups that allow multiple Nagios instances to monitor different network segments.

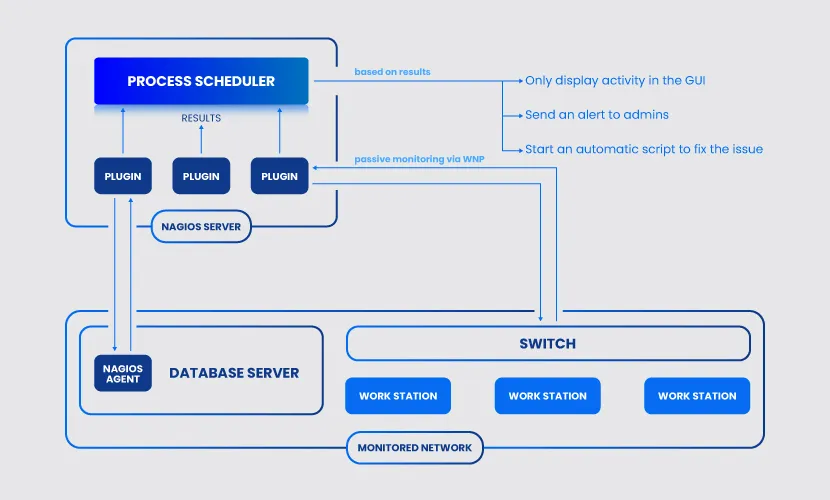
**Integration**

* + Nagios can integrate with other tools and platforms, such as ticketing systems and configuration management tools, to streamline operations and automate responses to alerts.

1. **Community and Support**
   * Being open source, Nagios has a strong community that contributes to its development and provides support through forums and documentation. There are also commercial versions available (Nagios XI) that offer additional features and support.

### Use Cases

* **Proactive Monitoring**: Identify and address issues before they affect end users by setting up alerts for performance thresholds and anomalies.
* **Root Cause Analysis**: Use historical data and logs to identify the root cause of issues and prevent them from recurring.
* **Capacity Planning**: Analyze trends and usage patterns to plan for future infrastructure needs.
* **Compliance**: Ensure systems meet compliance requirements by monitoring configurations and changes.



**Management Framework Scrum**

Scrum is a popular framework used in project management, particularly for software development. It is part of the Agile methodology, which emphasizes flexibility, collaboration, and customer feedback. Here are some key aspects of what Scrum is used for:

1. **\*\*Managing Complex Projects\*\***: Scrum is designed to handle complex and adaptive problems by breaking projects into smaller, manageable parts called "sprints," which typically last 1-4 weeks. This allows teams to deliver working software or project increments regularly.

2. **\*\*Enhancing Collaboration\*\***: Scrum encourages collaboration through regular meetings such as daily stand-ups, sprint planning, sprint reviews, and retrospectives. These meetings foster communication among team members and stakeholders, ensuring everyone is aligned and any issues are addressed promptly.

3**. \*\*Improving Product Quality\*\*:** By focusing on iterative development and continuous feedback, Scrum helps teams refine their products over time, leading to higher quality outcomes that meet customer needs more effectively.

4. **\*\*Increasing Flexibility and Adaptability\*\*:** Scrum allows teams to respond quickly to changes in project requirements, market conditions, or customer feedback. This adaptability is crucial in fast-paced environments where requirements can evolve rapidly.

5**. \*\*Fostering a Culture of Continuous Improvement\*\*:** The retrospective meetings at the end of each sprint encourage teams to reflect on their processes and identify areas for improvement, leading to continuous enhancement of both the product and team performance.

6. **\*\*Enhancing Transparency\*\*:** Scrum's emphasis on transparency helps stakeholders stay informed about the project's progress and any challenges faced, enabling more informed decision-making.

Overall, Scrum is a versatile framework that helps teams manage projects effectively, improve collaboration, and deliver value incrementally while adapting to changes and continuous feedback.

**Device Used in Our Project Is Router**

A router is a networking device that directs data packets between computers and networks, helping to establish an efficient and secure communication pathway within a network. Here's how a router works:

**1. \*\*Receiving Data Packets\*\*:** Routers receive incoming data packets from various devices on a network, such as computers, smartphones, and other connected devices. These packets contain information like the source and destination IP addresses.

**2. \*\*Reading Headers\*\*:** The router examines the headers of data packets to determine their intended destination. The header contains information that helps the router make decisions about where to send the packets.

**3. \*\*Routing Tables\*\*:** Routers use routing tables to decide the best path for forwarding packets to their destination. These tables contain information about network topology, which helps routers determine the most efficient route for each packet.

**4. \*\*Forwarding Packets\*\*:** Based on the routing table and the information in the packet header, the router forwards the packets to the next appropriate router or the final destination within a network. This process continues until the packet reaches its intended endpoint.

**5. \*\*Network Address Translation (NAT)\*\*:** Many routers perform NAT, which allows multiple devices on a local network to share a single public IP address. NAT modifies the IP address information in the packet headers to ensure that data is correctly routed between devices on a private network and the wider internet.

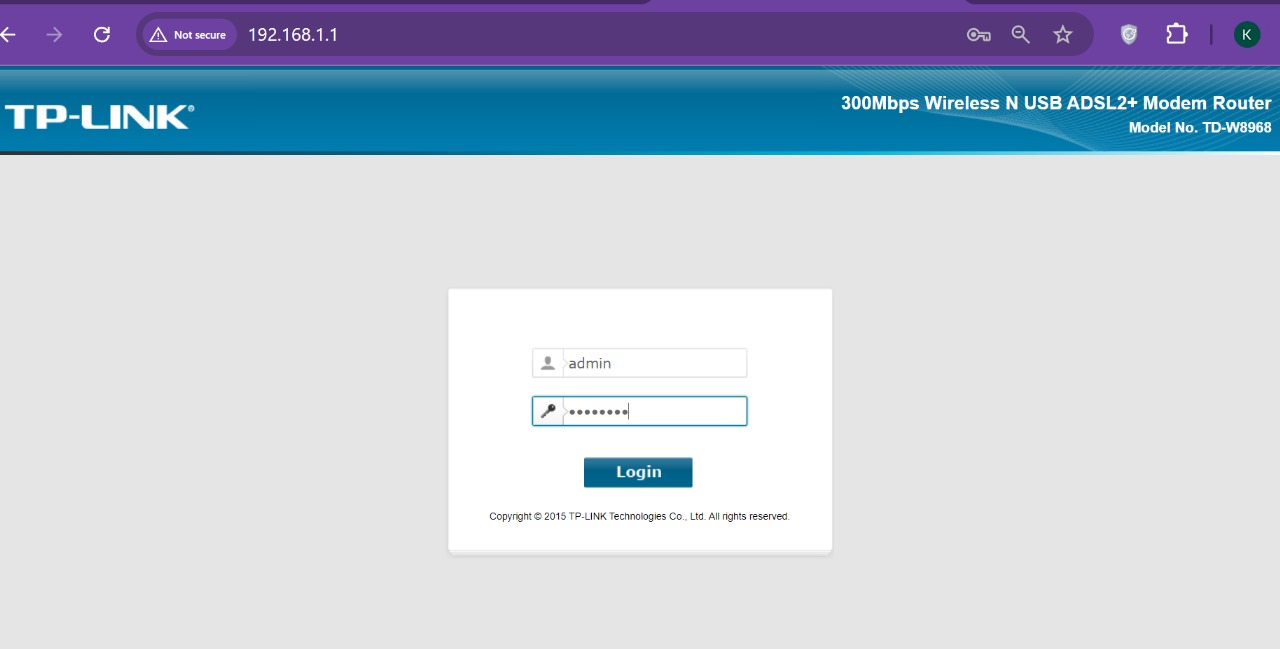
**6. \*\*Ensuring Security\*\*:** Routers often include firewalls and other security features to protect the network from unauthorized access and potential threats. They can filter incoming and outgoing traffic based on predefined security rules.

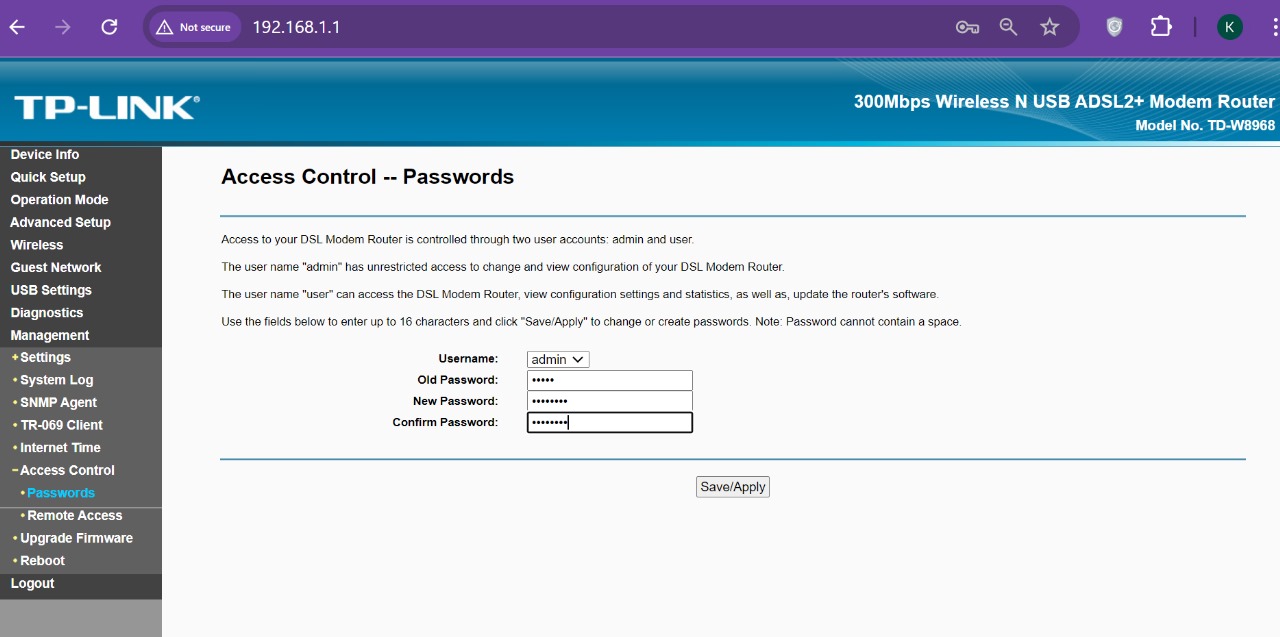
**7. \*\*Managing Traffic\*\*:** Routers can prioritize network traffic, ensuring that critical data, like video calls or gaming, receives priority over less time-sensitive data, such as email downloads. This helps maintain a smooth and efficient network performance.

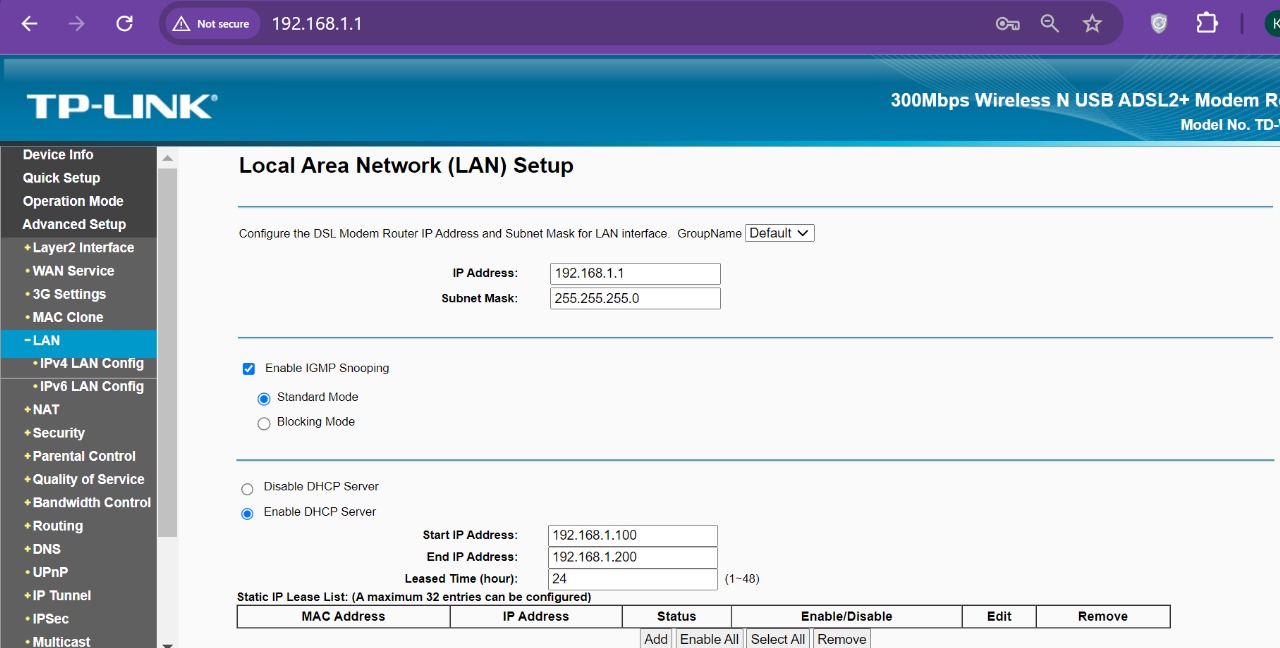
**8. \*\*Wi-Fi Access (in wireless routers) \*\*:** Wireless routers also serve as access points for Wi-Fi networks, allowing devices to connect wirelessly and share the internet connection and local network resources.

Routers are essential for both small home networks and large enterprise networks, facilitating the flow of information and maintaining connectivity between devices across different network segments.

**Router Login Details**







**Demilitarised Zone Concept-**

A DMZ or demilitarized zone is a perimeter network that protects and adds an extra layer of security to an organization’s internal local-area network from untrusted traffic.

The end goal of a demilitarized zone network is to allow an organization to access untrusted networks, such as the internet, while ensuring its private network or LAN remains secure. Organizations typically store external-facing services and resources, as well as servers for the [**Domain Name System (DNS)**](https://www.fortinet.com/resources/cyberglossary/what-is-dns), [**File Transfer Protocol (FTP)**](https://www.fortinet.com/resources/cyberglossary/file-transfer-protocol-ftp-meaning), mail, proxy, Voice over Internet Protocol (VoIP), and web servers, in the DMZ.

**How Does A DMZ Network Work**

Businesses with a public website that customers use must make their web server accessible to the internet. To protect the corporate local area network, the web server is installed on a separate computer from internal resources. The DMZ enables communication between protected business resources, like internal databases, and qualified traffic from the Internet.

A DMZ network provides a buffer between the internet and an organization’s private network. The DMZ is isolated by a security gateway, such as a firewall, that filters traffic between the DMZ and a LAN. The default DMZ server is protected by another security gateway that filters traffic coming in from external networks.

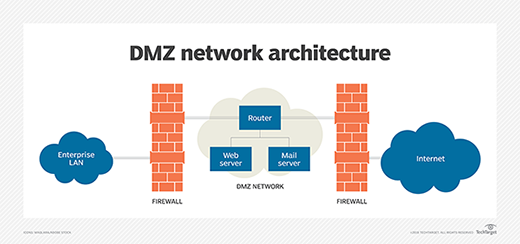
It is ideally located between two firewalls, and the DMZ firewall setup ensures incoming network packets are observed by a firewall—or other security tools—before they make it through to the servers hosted in the DMZ. This means that even if a sophisticated attacker is able to get past the first firewall, they must also access the hardened services in the DMZ before they can do damage to a business.

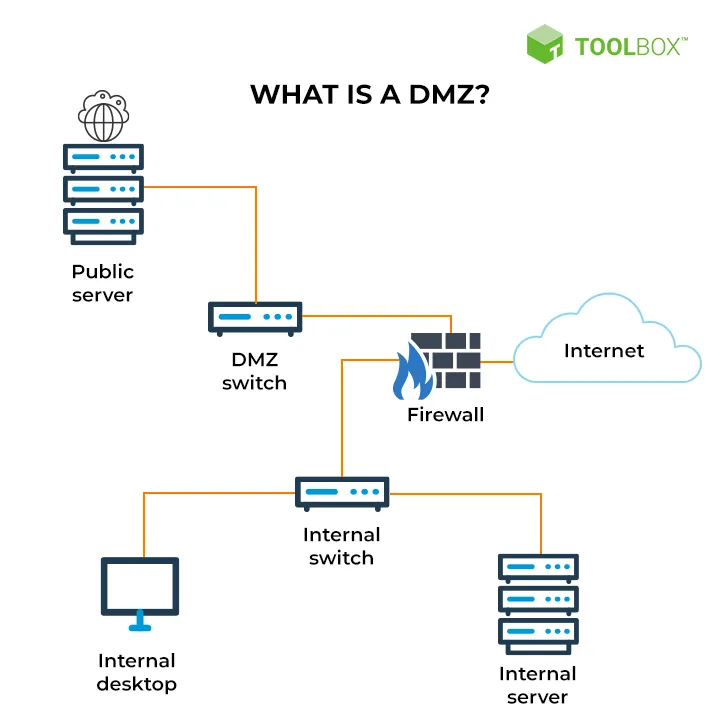
If an attacker is able to penetrate the external firewall and compromise a system in the DMZ, they then also have to get past an internal firewall before gaining access to sensitive corporate data. A highly skilled bad actor may well be able to breach a secure DMZ, but the resources within it should sound alarms that provide plenty of warning that a breach is in progress.

Services of a DMZ include:

1. DNS servers
2. FTP servers
3. Mail servers
4. Proxy servers
5. Web servers

**Architecture Of DMZ Network—**



****

**Web Server Used in Our Project Are Nginx And Apache2**

**1.Nginx--**Nginx (pronounced "engine-x") is a high-performance web server and reverse proxy server known for its speed, reliability, and low resource consumption. It is widely used to handle HTTP and HTTPS requests and offers a range of features that make it popular for modern web applications. Here are some key aspects of Nginx:

**1. \*\*Web Server\*\*:** Nginx is primarily used as a web server, serving static content like HTML, CSS, JavaScript, and images. It is optimized for handling many concurrent connections, making it ideal for high-traffic websites.

**2. \*\*Reverse Proxy\*\*:** Nginx can act as a reverse proxy server, forwarding client requests to one or more backend servers (such as application servers or database servers). This helps distribute the load and improve the performance and scalability of web applications.

**3. \*\*Load Balancer\*\*:** As a load balancer, Nginx can distribute incoming traffic across multiple servers, ensuring no single server becomes a bottleneck. This improves the availability and reliability of applications by distributing the workload evenly.

**4. \*\*SSL/TLS Termination\*\*:** Nginx can handle SSL/TLS termination, decrypting secure HTTPS requests before forwarding them to backend servers. This offloads the SSL processing from application servers, improving performance and simplifying certificate management.

**5. \*\*Caching\*\*:** Nginx can cache static and dynamic content, reducing the load on backend servers and speeding up response times for frequently requested resources. This caching capability enhances the overall performance and efficiency of web applications.

**6. \*\*HTTP/2 Support\*\*:** Nginx supports the HTTP/2 protocol, which offers improved performance over HTTP/1.1 by enabling multiplexing, header compression, and prioritization of requests, resulting in faster loading times for web pages.

**7. \*\*Security Features\*\*:** Nginx includes security features such as request rate limiting, IP address whitelisting/blacklisting, and support for Web Application Firewalls (WAFs), helping protect applications from common web threats.

**8. \*\*Highly Configurable\*\*:** Nginx is known for its flexible configuration options, allowing administrators to customize settings for various use cases, from simple static sites to complex, distributed architectures.

**9. \*\*Open Source and Community Support\*\*:** Nginx is open-source software with a large and active community, providing extensive documentation, support forums, and a rich ecosystem of third-party modules and extensions.

Overall, Nginx is a versatile and efficient tool for managing web traffic, enhancing application performance, and ensuring the reliability and security of web services.

**2.Apache2**

Apache HTTP Server, commonly referred to as Apache, is one of the most widely used web servers in the world. It is open-source software developed and maintained by the Apache Software Foundation. Here are the core concepts and features of the Apache 2 web server:

**Core Concepts**

**1. \*\*Serving Web Content\*\*:** Apache is designed to serve web pages over the Internet. It listens for requests from clients (such as web browsers) and responds by serving the requested content, which can be static files like HTML, CSS, images, or dynamic content generated by server-side scripts.

**2. \*\*Modular Architecture\*\*:** Apache is highly modular, meaning that it can be extended with a variety of modules to add features and functionality. Some common modules include:

- \*\*mod\_ssl\*\*: Adds support for SSL/TLS to secure connections.

- \*\*mod\_rewrite\*\*: Provides URL rewriting capabilities.

- \*\*mod\_proxy\*\*: Allows Apache to function as a reverse proxy.

- \*\*mod\_php\*\*: Enables PHP scripting for dynamic content generation.

**3. \*\*Configuration Files\*\*:** Apache's behavior is controlled by configuration files, typically named `httpd.conf` or `apache2.conf`. These files define server settings, such as ports to listen on, document root locations, virtual hosts, and security settings.

**4. \*\*Virtual Hosts\*\*:** Apache supports virtual hosting, allowing multiple websites to be hosted on a single server. This can be done using:

- \*\*Name-based Virtual Hosting\*\*: Different websites are served based on the domain name requested by the client.

- \*\*IP-based Virtual Hosting\*\*: Different websites are served based on the IP address of the request.

**5. \*\*Security Features\*\*:** Apache includes several security features to protect web applications, such as access control, authentication, and support for Secure Socket Layer (SSL) to encrypt data in transit.

**6. \*\*Logging and Monitoring\*\*:** Apache provides extensive logging capabilities, allowing administrators to monitor access and error logs to diagnose issues, track usage, and analyze traffic patterns.

**7. \*\*Performance Optimization\*\*:** Apache can be configured to improve performance through techniques such as caching, load balancing, and connection handling optimizations. It supports multi-processing modules (MPMs) like `prefork`, `worker`, and `event`, each with different approaches to handling concurrent connections.

**8. \*\*Cross-Platform Support\*\*:** Apache is cross-platform and can run on various operating systems, including Linux, Windows, and macOS, making it a versatile choice for web hosting.

***--How Apache Works***

**1. \*\*Request Handling\*\*:** When a client sends a request to the server, Apache receives the request and processes it based on its configuration settings. It determines the appropriate response by checking the requested URL, associated handlers, and any rewrite rules.

**2. \*\*Static vs. Dynamic Content\*\*:** For static content, Apache directly serves files from the filesystem. For dynamic content, it can interface with scripts and applications using CGI, FastCGI, or modules like `mod\_php` to generate content dynamically.

**3. \*\*Response Delivery\*\*:** After processing the request, Apache sends the generated response back to the client. This includes HTTP headers and the requested content.

**Use Cases**

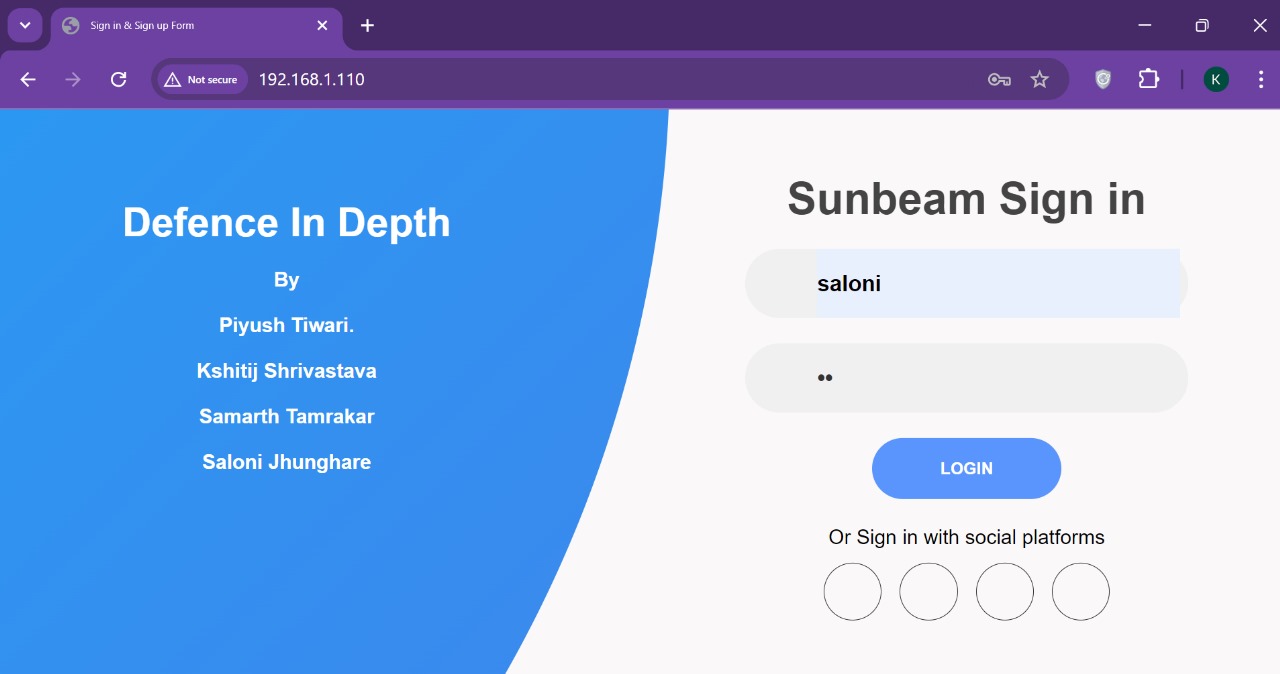
**- \*\*Hosting Static Websites\*\*: Serving** static HTML, CSS, and JavaScript files.

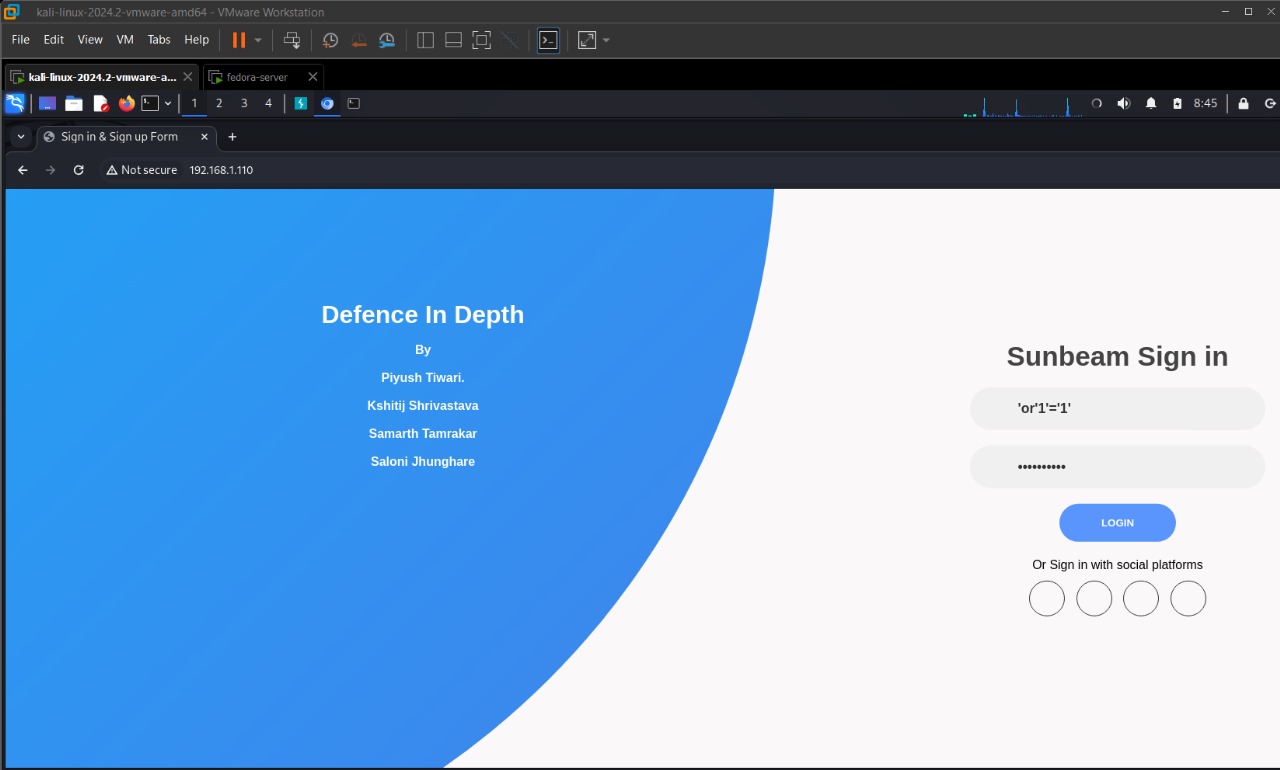
**- \*\*Dynamic Web Applications\*\*:** Running applications using languages like PHP, Python, or Ruby.

**- \*\*Reverse Proxy\*\*:** Forwarding requests to backend servers for load balancing or security.

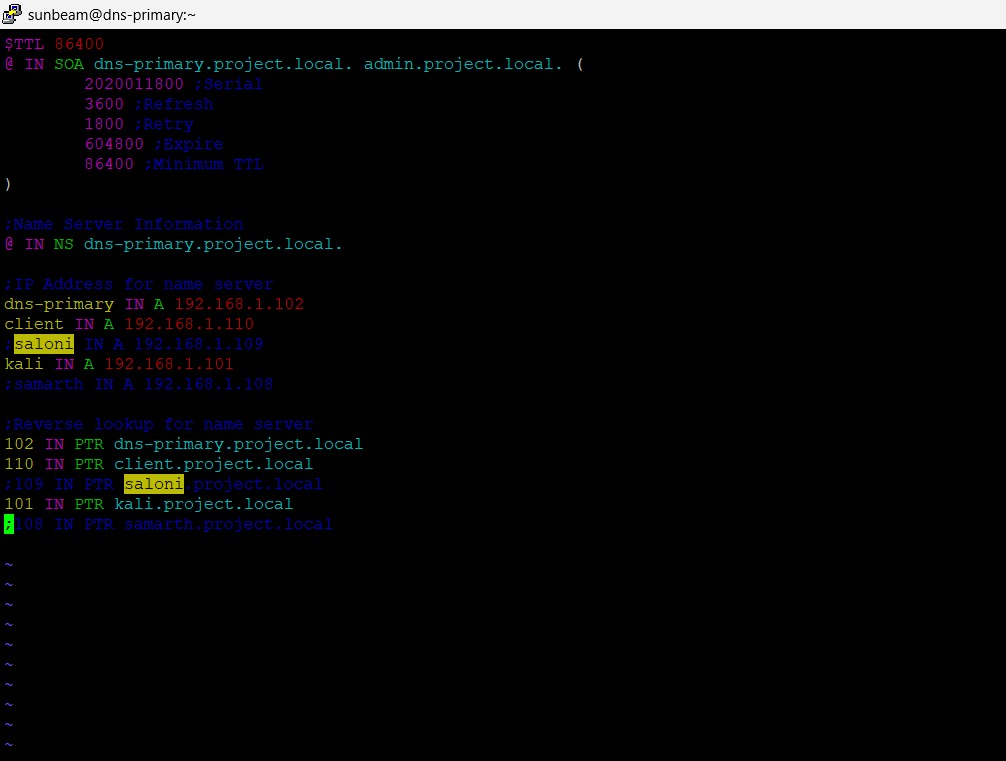
Overall, Apache HTTP Server is a robust and flexible web server solution that can be tailored to meet the needs of a wide range of web hosting scenarios, from small personal websites to large enterprise applications.

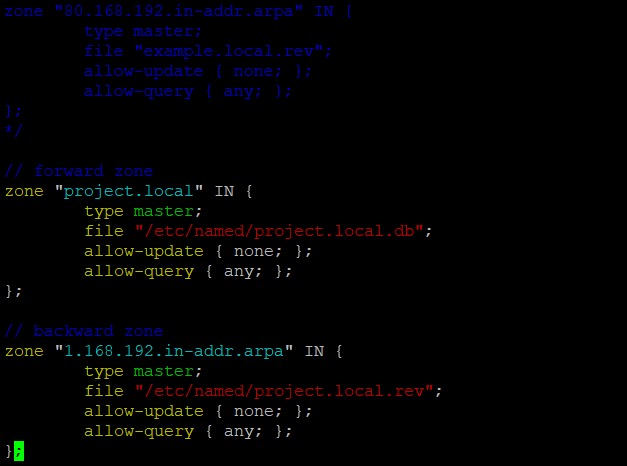
**Hosting Our Website Using Web Servers**





**DNS SERVER CONFIGURATION**

****



**IP Tables Concept Used: --**

**Iptables** is a command-line utility used to configure and manage the Linux kernel firewall, providing packet filtering, network address translation (NAT), and port forwarding capabilities. It is part of the net filter framework within the Linux kernel and is used to set up, maintain, and inspect the tables of IP packet filter rules. Here are the key aspects of iptables:

1. **Packet Filtering**: iptables can filter network packets based on various criteria, such as source and destination IP addresses, ports, protocols, and interfaces. This allows administrators to define rules for what traffic is allowed or blocked.
2. **Rule Specification**: Each rule in iptables specifies conditions that must be met for the rule to apply, and an action to be taken if the conditions are met. Common actions include ACCEPT (allow the packet), DROP (silently discard the packet), and REJECT (discard the packet and send an error back to the source).
3. **Scriptable and Extensible**: iptables rules can be saved and restored using scripts, making it easier to manage complex configurations. It also supports modules that extend its functionality for specific protocols or use cases.

### SYSTEM REQUIREMENTS



Base Machine Software

**Requirements**

* VM Ware Workstation Ver-17 Windows 11
* Virtual Box Version-7.0

**Hardware Requirements**

* RAM 16 GB
* Processor i7/ Ryzen 5 or above
* Hard disk 50 GB

**Devices Used**

* Debian 12—As Web Server
* Kali Linux—Database Server
* Fedora—DNS Server
* Debian 12—Proxy Server
* Debian 12—Nagios Server
* Router—TP-Link.

### INSTALLATION AND SCREENSHOTS

**NAGIOS---**

Nagios is a popular open-source monitoring system that helps you monitor the health and performance of various IT infrastructure components such as servers, services, applications, and network devices.

Here's a step-by-step guide on how to install and configure Nagios on a Linux-based system

1.Update your System.

Sudo apt update

Sudo apt update-y

2.Install Required Updates. —Nagios Required some development tools and libraries to be installed on your System.

sudo apt install -y build-essential apache2 php openssl perl make \ gcc libapache2-mod-php libgd-dev libssl-dev daemon wget \ libmcrypt-dev bc gawk dc build-essential snmp libnet-snmp-perl gettext

3.Install Nagios Core

a)-Create a user and Group for Nagios:

sudo useradd nagios

sudo groupadd nagcmd

sudo usermod -a -G nagcmd nagios

sudo usermod -a -G nagcmd www-data

4. Download and Extract Nagios Core:

cd /tmp

wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.12.tar.gz

tar xzf nagios-4.4.12.tar.gz

cd nagios-4.4.12

5.Compile and Install Nagios:

./configure --with-command-group=nagcmd

make all

sudo make install

sudo make install-init

sudo make install-commandmode

sudo make install-config

sudo make install-webconf

6.Configure Apache for Nagios:

a)-Enable the Apache rewrite and CGI Modules.

sudo a2enmod rewrite

sudo a2enmod cgi

1. Restart Apache:

sudo systemctl restart apache2

8.- Create a Nagios Web User:

sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

9.Install Nagios Plugins. —Download and Extract Nagios Plugins.

cd /tmp

wget https://nagios-plugins.org/download/nagios-plugins-2.4.3.tar.gz

tar xzf nagios-plugins-2.4.3.tar.gz

cd nagios-plugins-2.4.3

10.Compile and install Nagios Plugins:

./configure --with-nagios-user=nagios --with-nagios-group=nagios

make

sudo make install

11.Start Nagios—Verify the Nagios Configuration and Enable Nagios to start the Service:

sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Enable—

sudo systemctl enable nagios

sudo systemctl start nagios

12.Axcess Nagios web Interface

a)-Open a Web browser and go to http://<server-ip>/nagios.

b)-Log in using the ”nagiosadmin” username and the password you set earlier.

13.Basic Configuration

a) --- **Hosts and Services:** You can define what to monitor by editing configuration files located in /usr/local/nagios/etc/objects/. The main files to edit are:

* hosts.cfg: Define hosts you want to monitor.
* services.cfg: Define services to monitor on the hosts.

b) --- **Nagios Commands:** You can define custom commands in the commands.cfg file located in /usr/local/nagios/etc/objects/.

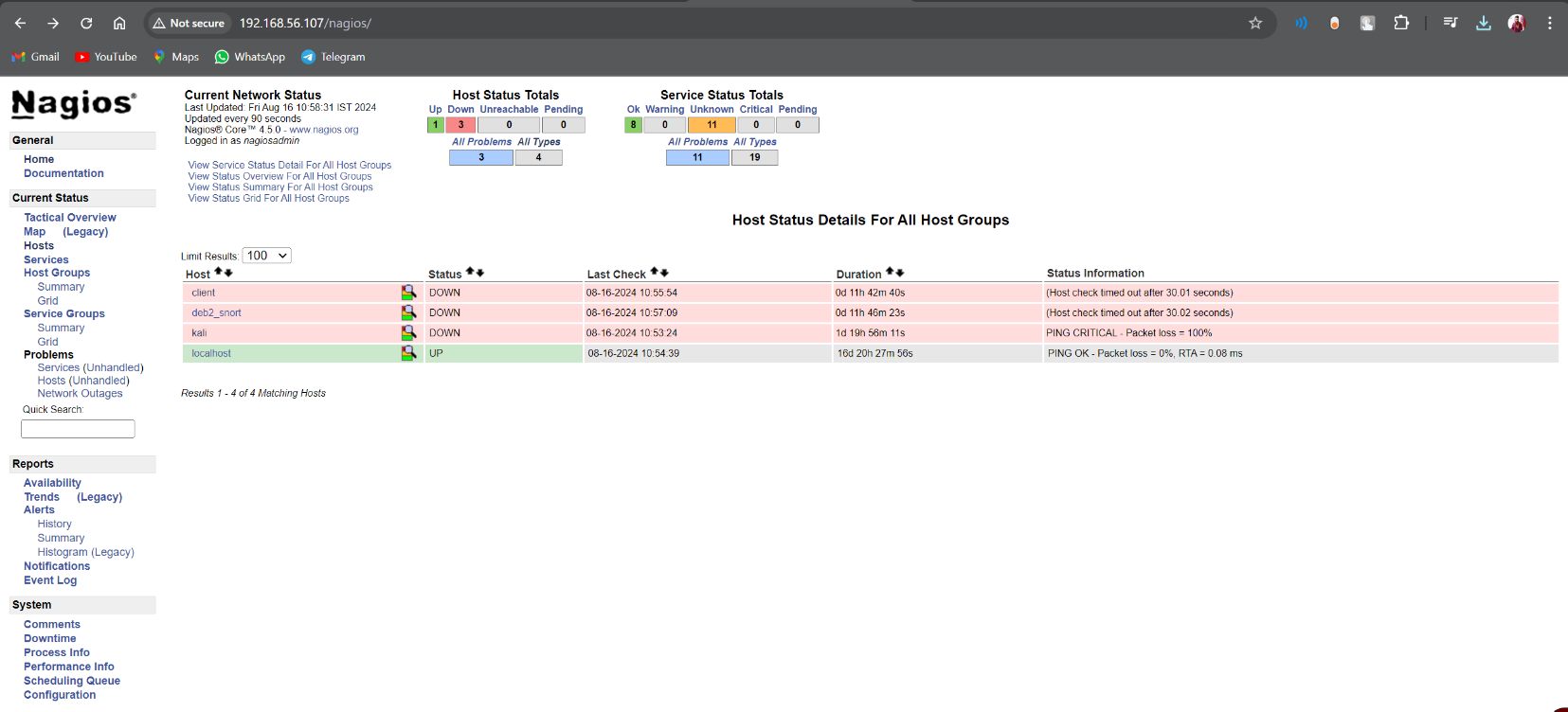
c)—Apply Configuration Changes: --After making changes to configuration files verify and restart Nagios:

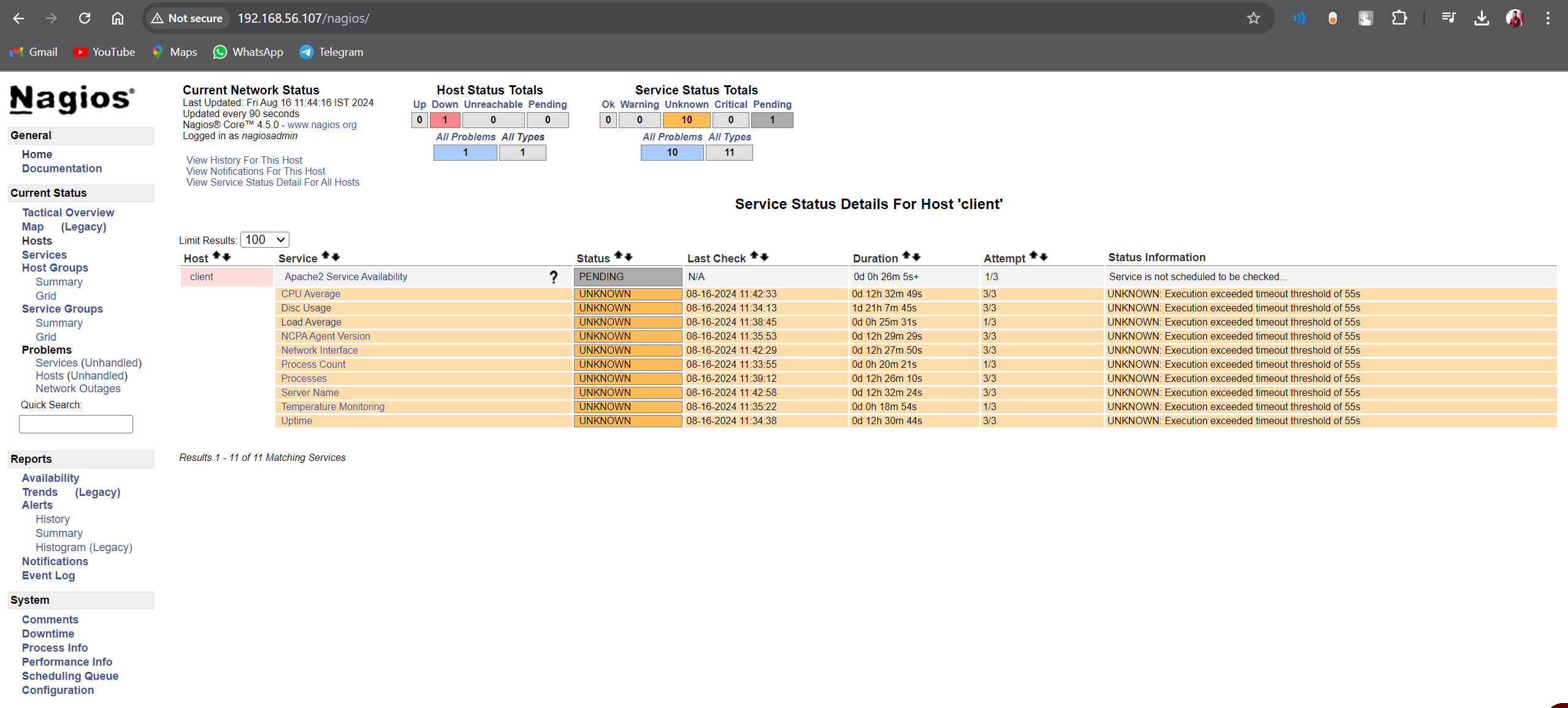
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

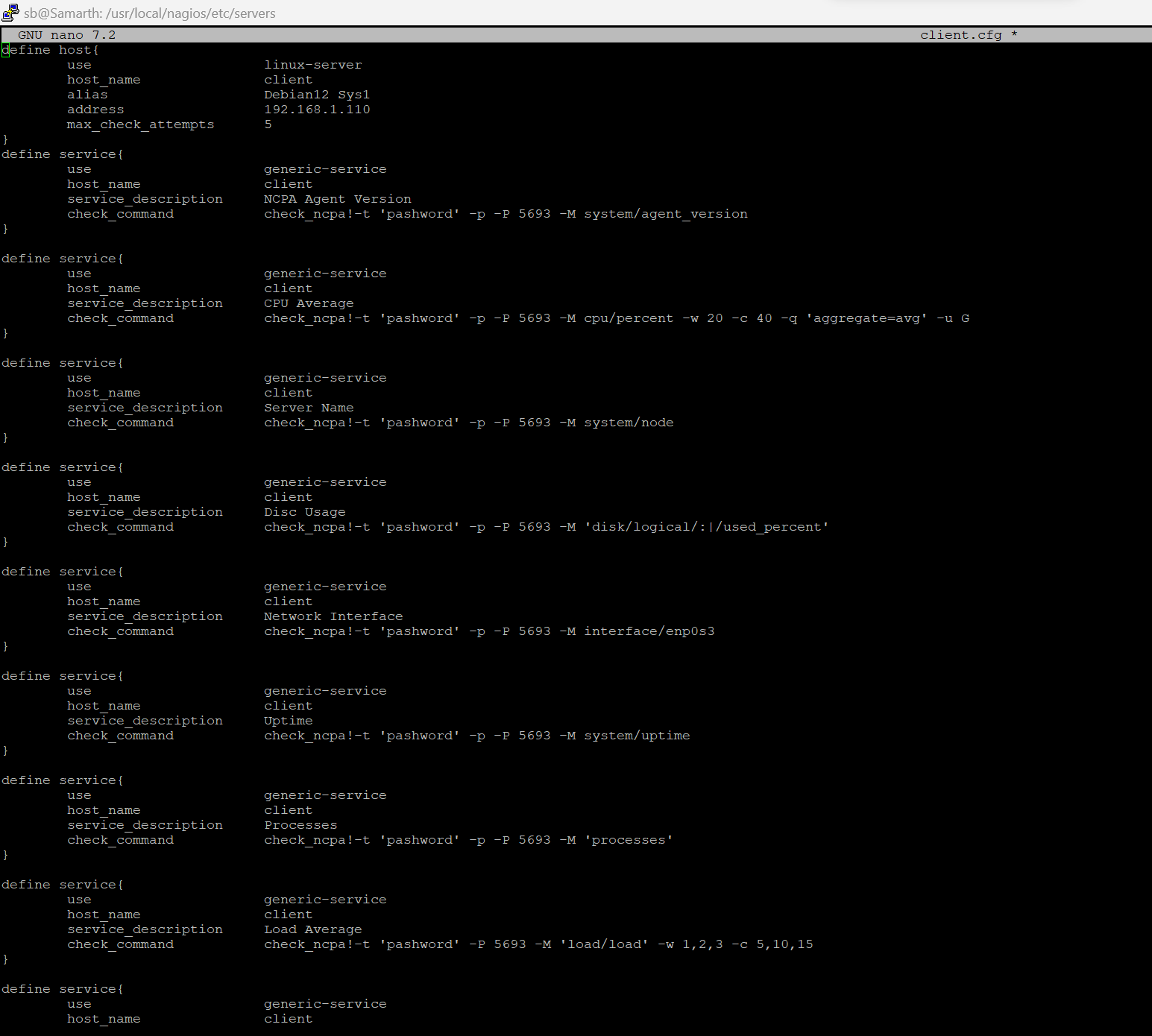
sudo systemctl restart nagios

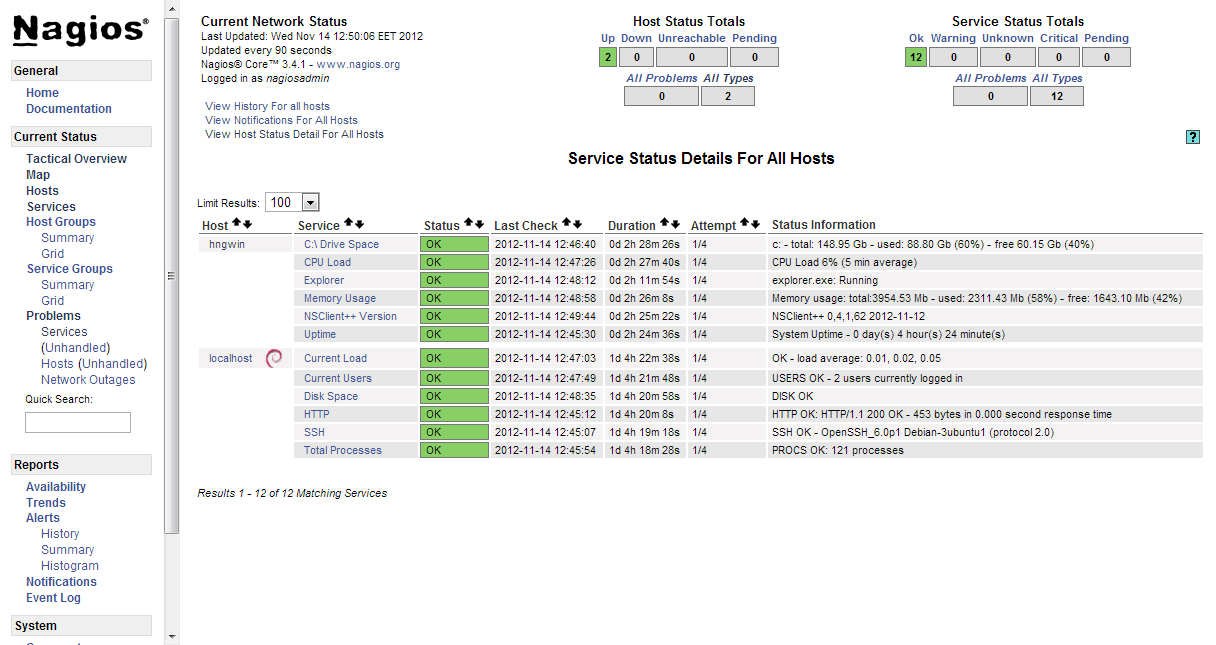
This setup provides a basic Nagios installation. For more advanced configurations, you may want to explore additional Nagios plugins, third-party addons, and best practices for managing large numbers of hosts and services.

**SCREEN SHOT NAGIOS ---SYSTEM**









**IPSEC CONFIGURATION ---**

To configure IPsec on two Debian machines, you'll need to set up a Virtual Private Network (VPN) using the strong Swan package. Here's a step-by-step guide to configure IPsec on two Debian machines:

### Prerequisites

1. **Two Debian machines** with static IP addresses.
2. **Root access** on both machines.

**Steps To Configure IPsec Using Strong Swan**

**Step 1: Install ‘strong Swan’ package on both machines:**

sudo apt update

sudo apt install strongswan strongswan-pki

**Step 2: Configure IPsec----** On both machines, you will need to edit the configuration files to set up the IPsec tunnel.

1. **Edit /etc/ipsec.conf**

Open the file and configure the connection. Here is an example configuration:

Machine A (192.168.1.1)

config setup

charondebug="ike 2, knl 2, cfg 2"

conn %default

ikelifetime=60m

keylife=20m

rekeymargin=3m

keyingtries=1

authby=secret

keyexchange=ikev2

conn myvpn

left=192.168.1.1

leftsubnet=192.168.1.0/24

right=192.168.2.1

rightsubnet=192.168.2.0/24

auto=start

Machine B— (192.168.2.1)

config setup

charondebug="ike 2, knl 2, cfg 2"

conn %default

ikelifetime=60m

keylife=20m

rekeymargin=3m

keyingtries=1

authby=secret

keyexchange=ikev2

conn myvpn

left=192.168.2.1

leftsubnet=192.168.2.0/24

right=192.168.1.1

rightsubnet=192.168.1.0/24

auto=start

2.**Edit /etc/ipsec.secrets**--- Add a shared secret key that both machines will use for authentication Choose a strong password or random key.

**Both machines—**

192.168.1.1 192.168.2.1 : PSK "your\_shared\_secret"

**3.Enable IP Forwarding--**On both machines, enable IP forwarding by editing the /etc/sysctl.conf file and uncommenting the following line:

net.ipv4.ip\_forward=1

**4.Apply the Changes with:**

sudo sysctl -p

**5.Start And Enable StrongSwan--**On both machines, start the strongSwan service and enable it to start on boot:

sudo systemctl start strongswan

sudo systemctl enable strongswan

**6.Verify the Connection:** On both machines, check the status of the IPsec connection:

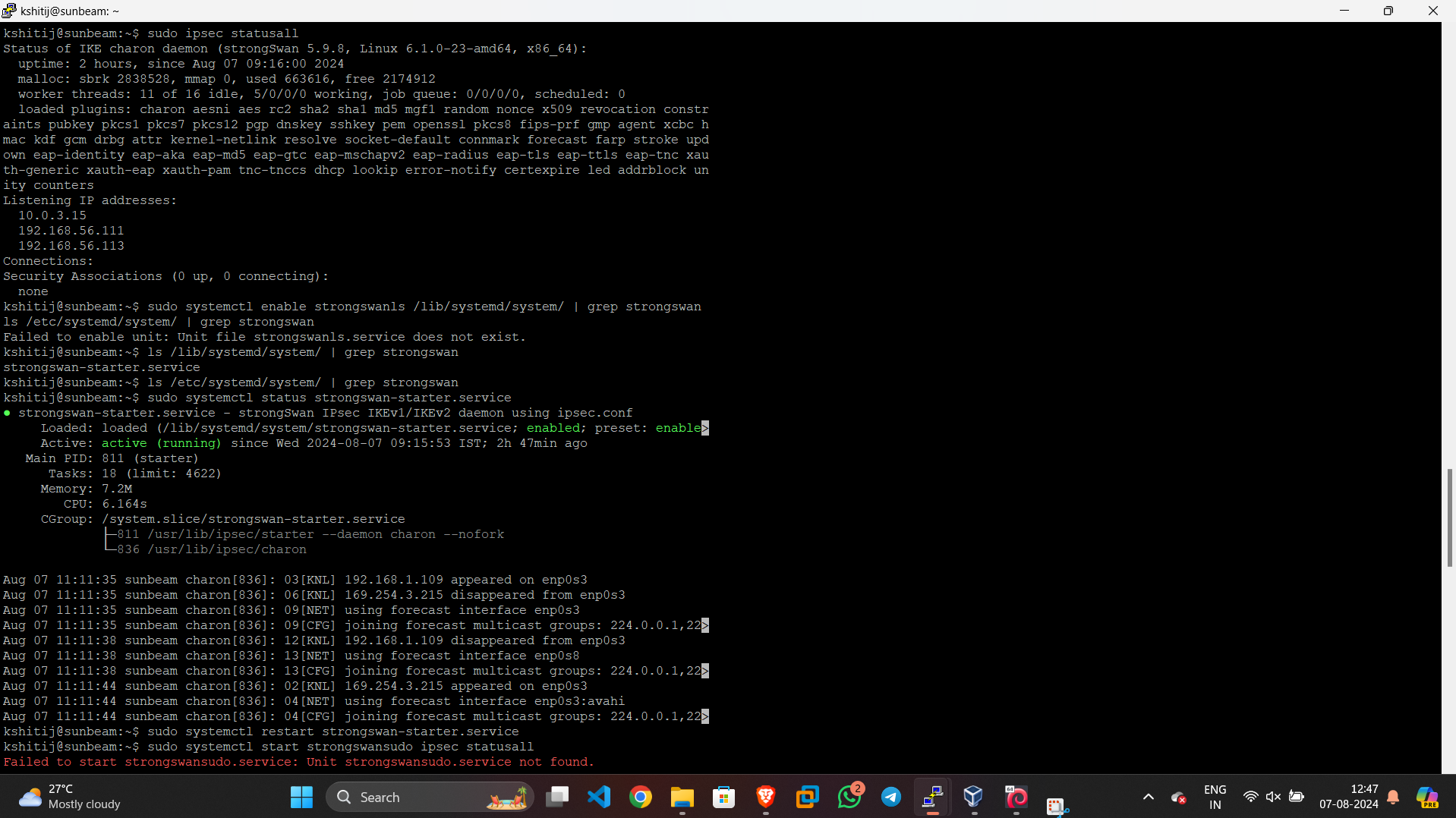
sudo ipsec statusall

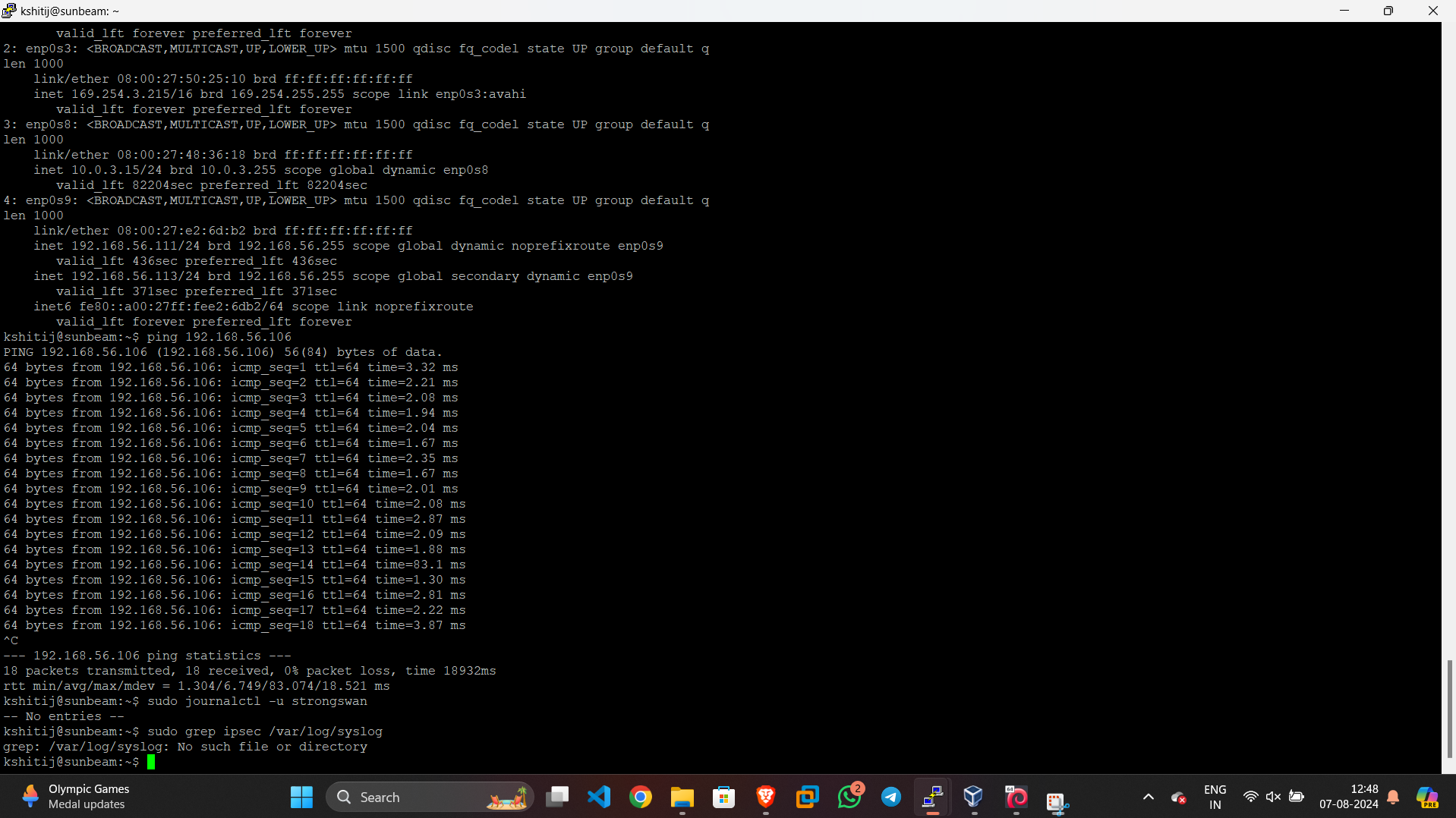
**7.Test the VPN: --**

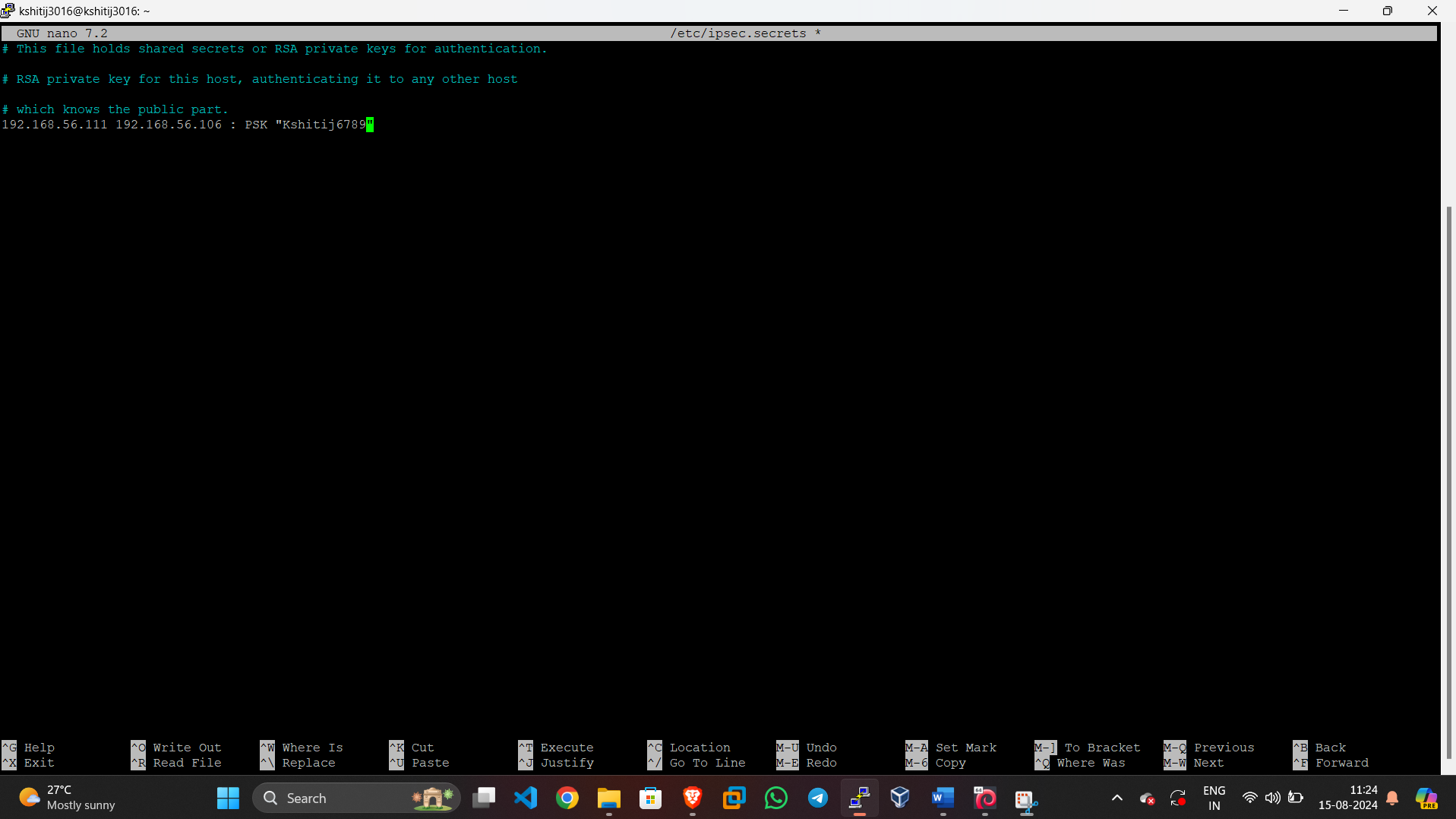
Try pinging from one subnet to the other to test the VPN tunnel:

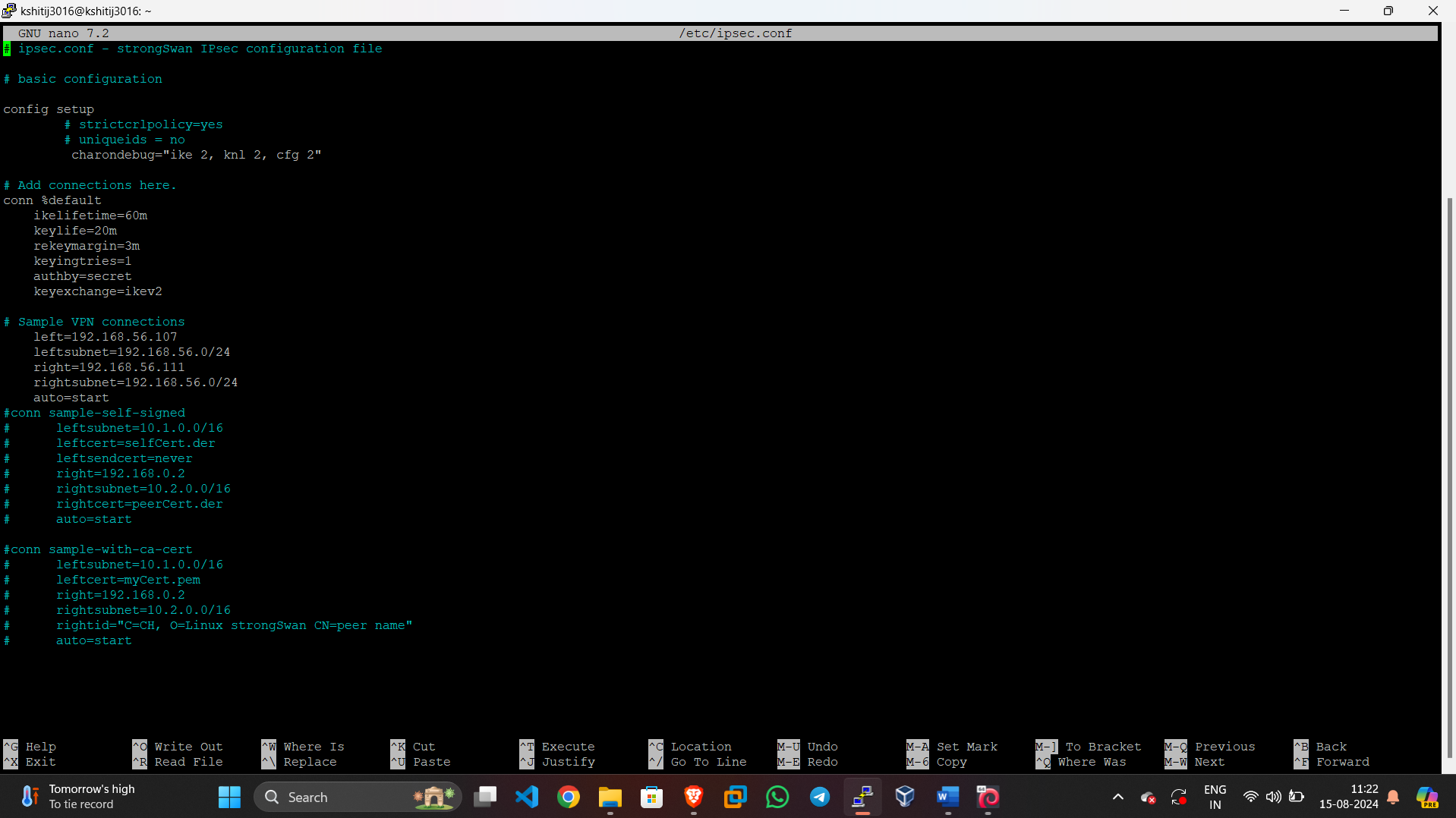
* From a host in 192.168.1.0/24, ping a host in 192.168.2.0/24.
* From a host in 192.168.2.0/24, ping a host in 192.168.1.0/24.

**SCREENSHOT IPsec CONFIGURATION**







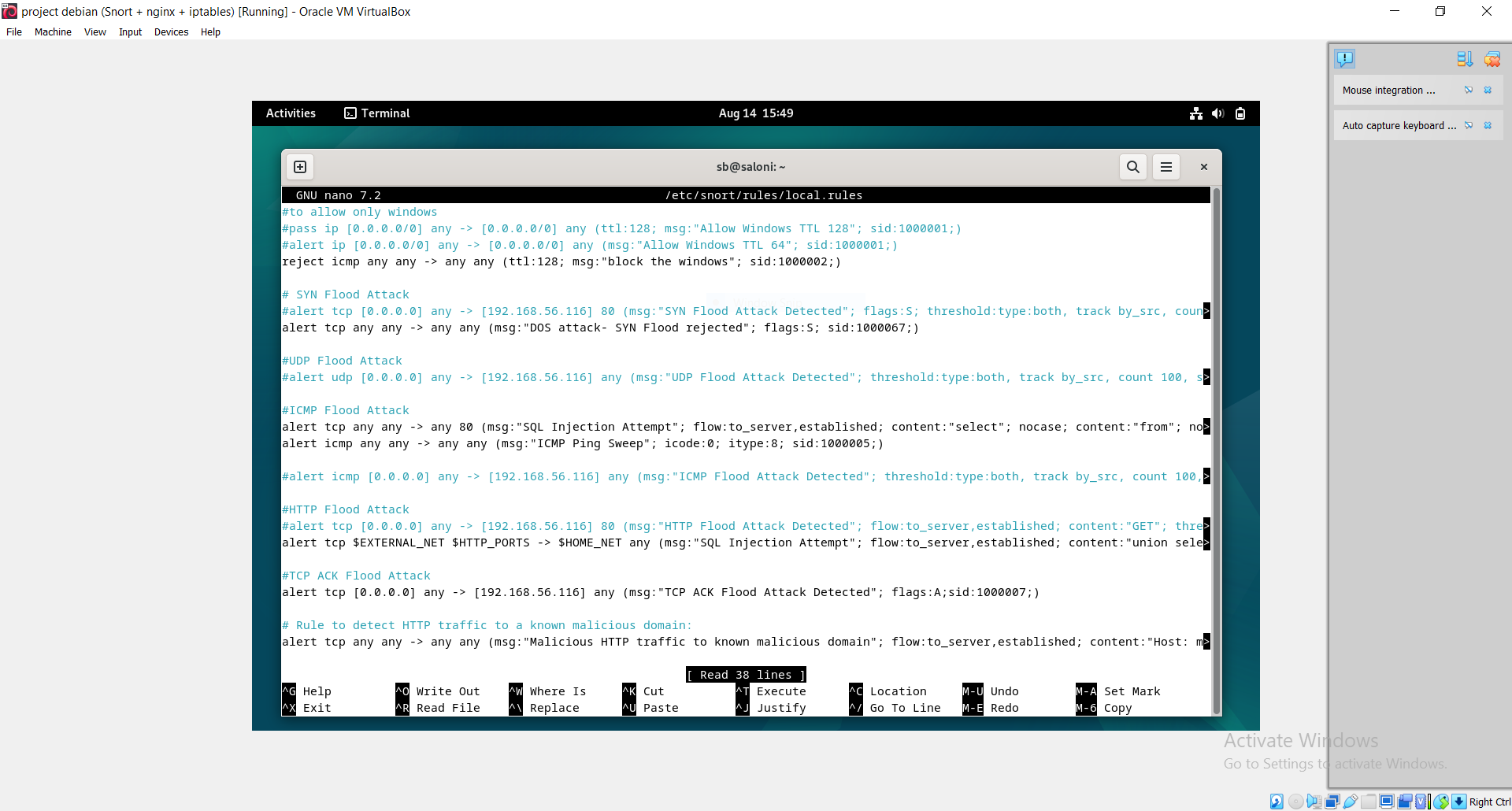


**SNORT CONFIGURATION ---**Snort is an open-source network intrusion detection system (NIDS) and intrusion prevention system (IPS) that is capable of performing real-time traffic analysis and packet logging on IP networks. Here's a step-by-step guide to installing and configuring Snort on a Linux-based system

1**.Update the System and Install the Required Packages**.

sudo apt install -y build-essential libpcap-dev libpcre3-dev libdumbnet-dev bison flex zlib1g-dev

2.**Configuring Snort ----**



**Snort Rules—**

#to allow only linux

#pass ip [0.0.0.0/0] any -> [0.0.0.0/0] any (ttl:128; msg:"Allow Windows TTL 128"; sid:1000001;)

#alert ip [0.0.0.0/0] any -> [0.0.0.0/0] any (msg:"Allow Windows TTL 64"; sid:1000001;)

reject icmp any any -> any any (ttl:128; msg:"block the windows"; sid:1000002;)

# SYN Flood Attack

#alert tcp [0.0.0.0] any -> [192.168.56.116] 80 (msg:"SYN Flood Attack Detected"; flags:S; threshold:type:both, track by\_src, count 100, seconds 10; sid:1000003;)

reject tcp any any -> any any (msg:"DOS attack- SYN Flood rejected"; flags:S; sid:1000067;)

#UDP Flood Attack

#alert udp [0.0.0.0] any -> [192.168.56.116] any (msg:"UDP Flood Attack Detected"; threshold:type:both, track by\_src, count 100, seconds 10; sid:1000004;)

#ICMP Flood Attack

alert tcp any any -> any 80 (msg:"SQL Injection Attempt"; flow:to\_server,established; content:"select"; nocase; content:"from"; nocase; pcre:"/select.+from/i"; classtype:web-application-at>

alert icmp any any -> any any (msg:"ICMP Ping Sweep"; icode:0; itype:8; sid:1000005;)

#alert icmp [0.0.0.0] any -> [192.168.56.116] any (msg:"ICMP Flood Attack Detected"; threshold:type:both, track by\_src, count 100, seconds 10; sid:1000005;)

#HTTP Flood Attack

#alert tcp [0.0.0.0] any -> [192.168.56.116] 80 (msg:"HTTP Flood Attack Detected"; flow:to\_server,established; content:"GET"; threshold:type:both, track by\_src, count 100, seconds 10;sid:1>

alert tcp $EXTERNAL\_NET $HTTP\_PORTS -> $HOME\_NET any (msg:"SQL Injection Attempt"; flow:to\_server,established; content:"union select"; http\_uri; nocase; sid:1000006;)

#TCP ACK Flood Attack

alert tcp [0.0.0.0] any -> [192.168.56.116] any (msg:"TCP ACK Flood Attack Detected"; flags:A;sid:1000007;)

# Rule to detect HTTP traffic to a known malicious domain:

alert tcp any any -> any any (msg:"Malicious HTTP traffic to known malicious domain"; flow:to\_server,established; content:"Host: maliciousdomain.com"; http\_header; sid:10001; rev:1;)

# Rule to detect brute force attacks:

alert tcp any any -> any 80 (msg:" Brute Force Attack";flow:to\_server,established;flags: PA; detection\_filter: track by\_dst,count 3, seconds 5; classtype:attempted-dos; sid:10002; rev:1;)

reject tcp any any -> any 22 (msg:"Brute Force SSH Login Attempt"; flow:to\_server,established; detection\_filter:track by\_src, count 5, seconds 60; classtype:attempted-admin; sid:1000019; r>

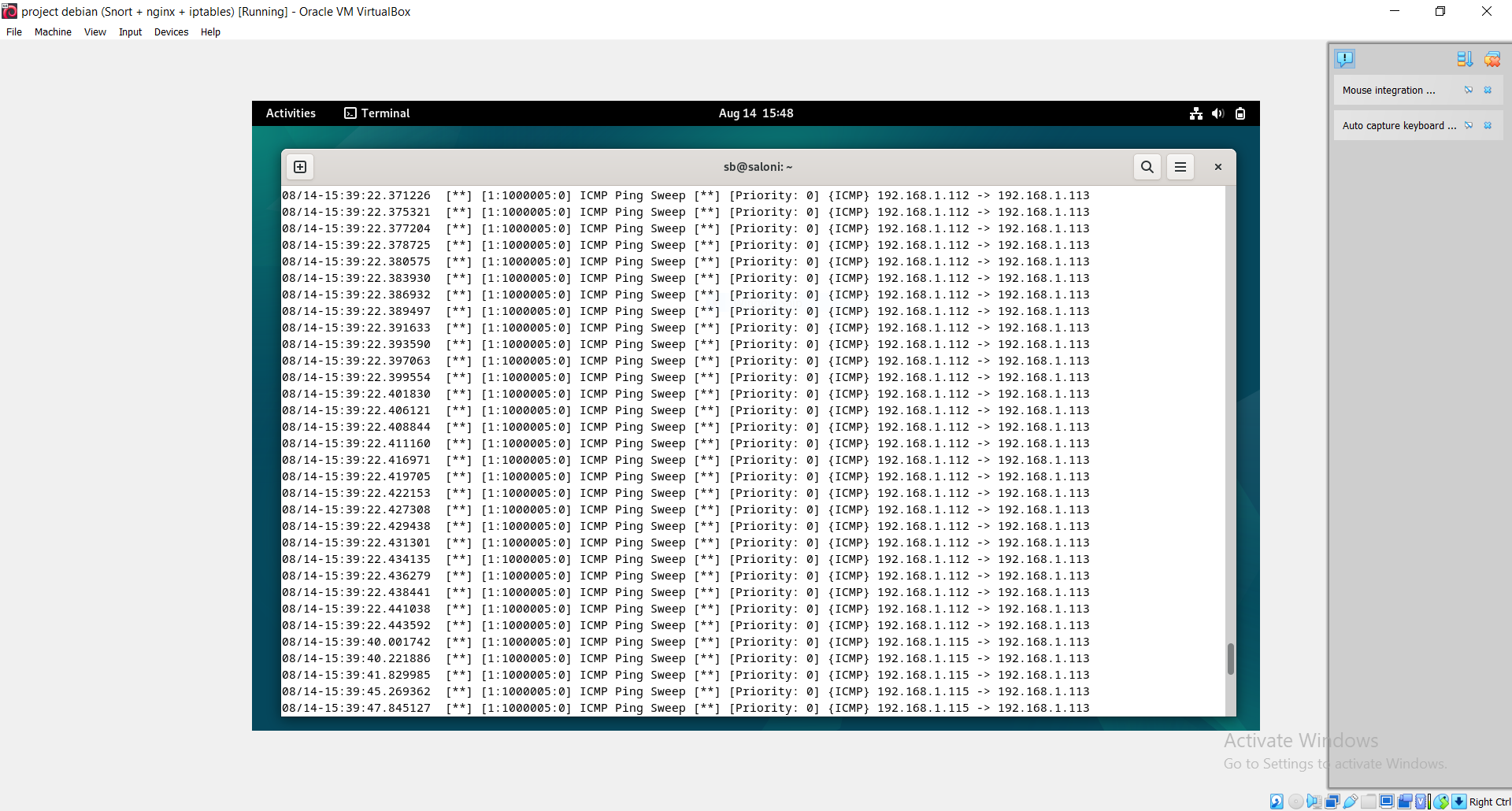
# Rule to detect SQL injection attacks:

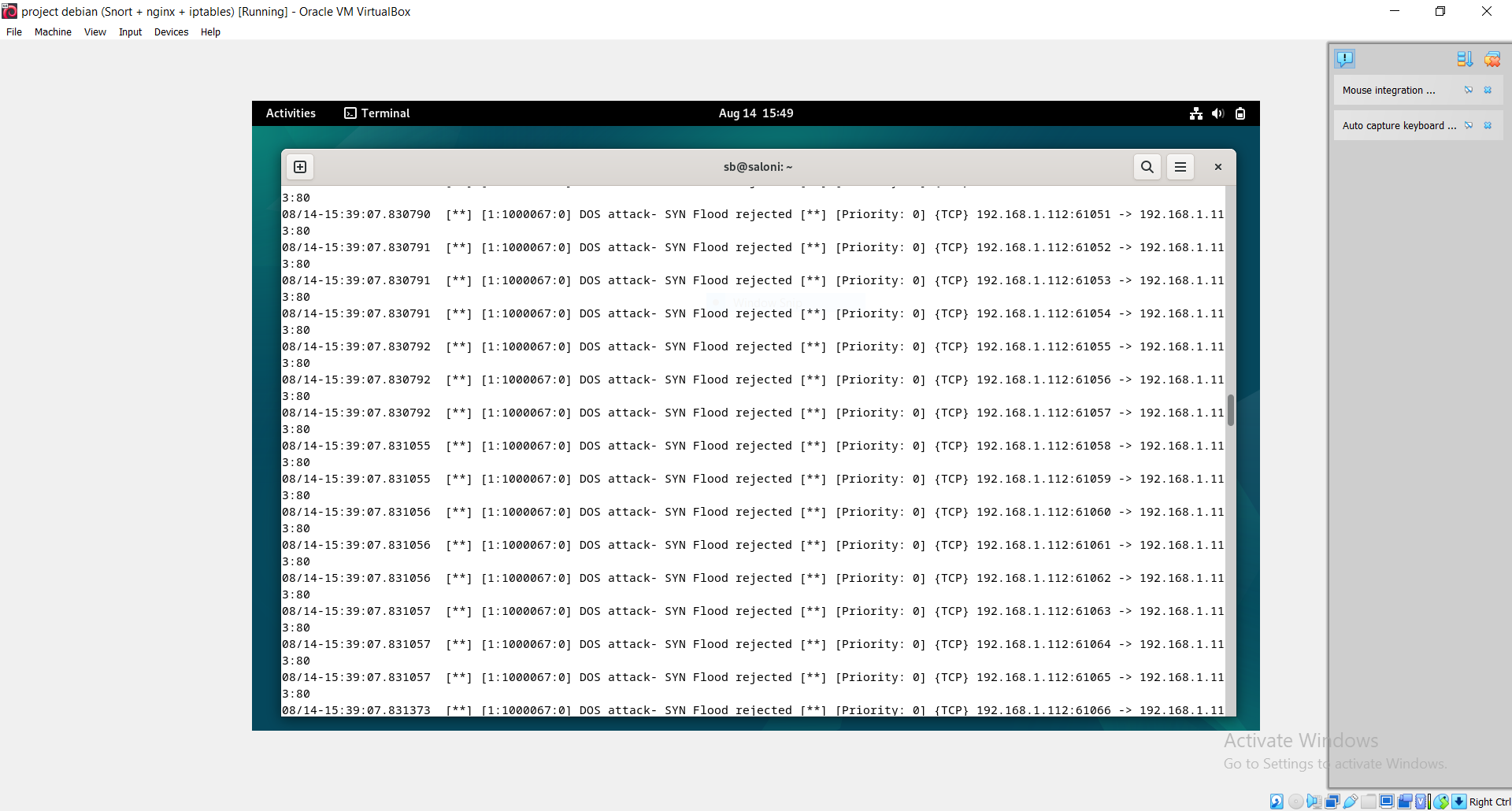
alert tcp any any -> any any (msg:"Possible SQL Injection Attack"; flow:to\_server,established; content:"' or 1=1--"; sid:10003; rev:1;)

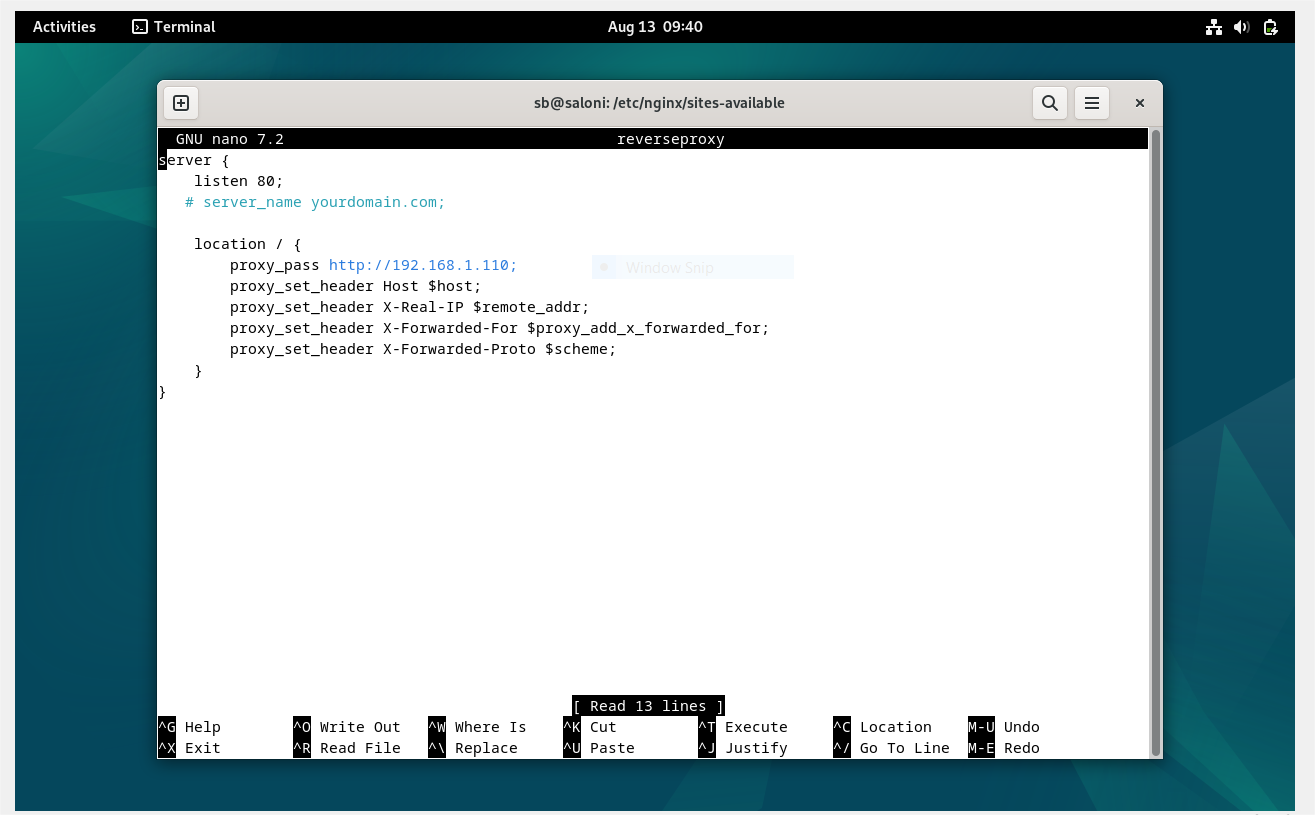
reject tcp any any -> any 80 (msg:"SQL Injection Attempt"; flow:to\_server,established; content:"select"; nocase; content:"from"; nocase; pcre:"/select.+from/i"; classtype:web-application-a>

#rule for dictionary attack

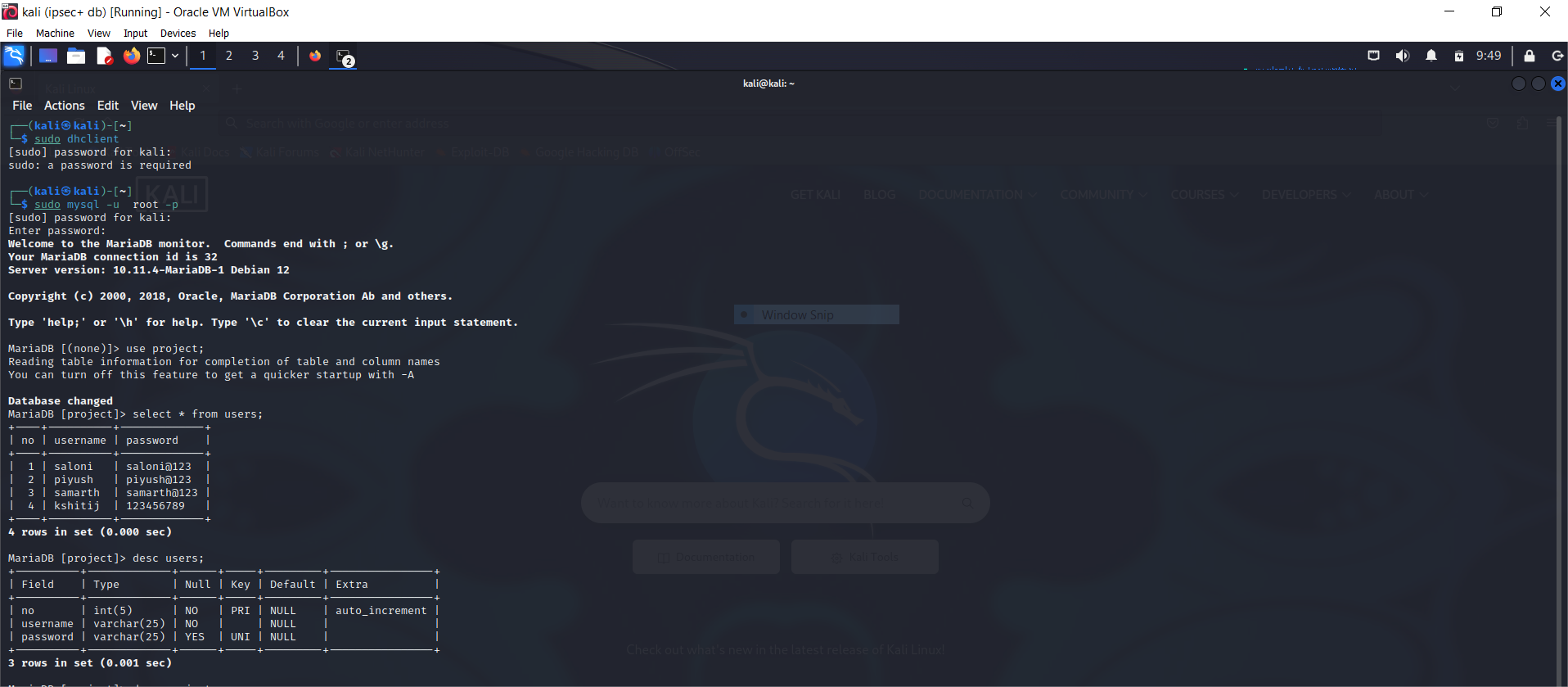
alert tcp any any -> $HOME\_NET 21 (msg:"FTP Dictionary Attack"; flow:to\_server,established; content:"USER"; nocase; detection\_filter:track by\_src, count 5, seconds 60; classtype:attempted->







**CREATING DATABASE WITH MYSQL**

****

****

### 5. CONCLUSION & FUTURE SCOPE

Defense in depth is a robust cybersecurity strategy that integrates multiple layers of security measures to protect an organization’s IT infrastructure and data. By deploying a comprehensive approach that includes network, endpoint, application, data, and physical security, organizations can significantly enhance their resilience against a variety of threats. The strategy relies on the principle of redundancy, ensuring that if one defense layer fails, others are in place to mitigate potential risks.

This approach is essential in today's complex threat landscape, where cyberattacks are increasingly sophisticated and varied. By adopting a defense-in-depth strategy, organizations can better manage security risks, ensure compliance with regulations, and maintain trust with customers and stakeholders. The effectiveness of defense in depth lies in its ability to provide continuous protection, detection, and response to security incidents, making it a critical component of any modern cybersecurity program.

### Future Scope:

 **Integration with Artificial Intelligence and Machine Learning**

* AI and machine learning are increasingly being used to enhance the capabilities of security tools. Future defence-in-depth strategies will likely leverage these technologies to improve threat detection, automate responses, and adapt defences in real time based on threat intelligence.

 **Zero Trust Architecture**

* The zero-trust model, which assumes that threats can come from both outside and inside the network, complements defence in depth by focusing on verifying every access request regardless of its origin. Future implementations will integrate zero trust principles to ensure more granular access control and continuous authentication.

 **Cloud Security**

* As organizations continue to migrate to cloud environments, defence in depth will evolve to include cloud-native security controls. This includes securing containers, microservices, and serverless architectures, as well as leveraging cloud providers' security features.

**6. REFERENCES**

1. <https://www.nagios.org/documentation/>

2. https://[www](https://www.snort.org/documents/snort-conf-examples).snort.org/documents/snort-conf-examples

3. <https://www.cisco.com/c/en/us/support/docs/routers/1700-series-modular-access-routers/71462-rtr-l2l-ipsec-split.html>