SNORTSHIELD NETWORK

PROJECT REPORT

Done by

Mr. UNNIKRISHNAN AP

Mr. DEEPAK C

Mr. MUHAMMED HASHIR B

Mr. DEVENDRA KUMAR SAHOO

Under the guidance of

Mrs. SINI NAIR

(Scientist D)

In partial fulfilment of the requirement for

CERTIFICATION COURSE IN
CYBER SECURITY AND ETHICAL HACKING

NATIONAL INSTITUTE OF ELECTRONICS &
INFORMATION TECHNOLOGY, CALICUT

(An Autonomous Scientific Society of Ministry of Electronics & Information Technology)

Government of India

Post box no.5, NIT Campus P.O, Calicut, Kerala, India

DECLARATION

We hereby declare that the project work entitled

SNORTSHIELD NETWORK

Done at

NATIONAL INSTITUTE OF ELECTRONICS &

INFORMATION TECHNOLOGY, CALICUT

(An Autonomous Scientific Society of Ministry of Electronics & Information Technology)

Government of India

Post box no.5, NIT Campus P.O., Calicut 673601, Kerala, India.

Under the guidance of

Mrs. SINI S NAIR

Name of the Students Signature

Mr. Unni Krishnan A P

Mr. Deepak c

Mr. Muhammed Hashir B

Mr. Devendra Kumar Sahoo

Place: **CALICUT**

Date: 06/12/2023

ACKNOWLEDGEMENT

First of all, we express our sincere gratitude to the almighty whose blessings helped us to materialise this project.

We would like to thank **Dr Pratap Kumar S**, Director, NIELIT Calicut for providing all the facilities required.

Our sincere thanks to our project guide **Mrs. Sini S Nair**, Scientist D, NIELIT Calicut, for spending her precious time for us and giving her valuable ideas and suggestions, which helped completing our project successfully.

We express our deep gratitude to **Mr. Hari K** and **Mr. Arun Mohan** for providing us with necessary technical support.

Finally, we would like to express our sincere thanks to all of our friends, colleagues, parents and all those who have directly assisted during this project.

ABSTRACT

In response to the dynamic challenges posed by the evolving cyber landscape, the SnortShield Network project emerges as a sophisticated initiative designed to fortify network security. At its core, this project integrates a powerful suite of technologies, including Windows Server Active Directory Domain Services (ADDS), IIS Server, DHCP Server, pfSense Firewall, and the renowned Snort intrusion detection and prevention system.

SnortShield Network is strategically crafted to establish a comprehensive defence framework, ensuring real-time identification and mitigation of potential security threats. By leveraging the strengths of each component, the project aims to create a cohesive and resilient network infrastructure. Windows Server ADDS centralizes user and resource management, IIS Server facilitates seamless web hosting, DHCP Server optimizes network address allocation, and pfSense Firewall enhances overall network security.

The hallmark of this project is the proactive role played by Snort, continuously monitoring network traffic to detect and prevent intrusions effectively. SnortShield Network aspires to provide organizations with a secure digital environment where data integrity is paramount, user access is managed judiciously, and the network remains impervious to the ever-changing threat landscape.

Embark on the journey with SnortShield Network, where security is not just a requirement, but a dynamic and proactive force ensuring the integrity of digital interactions.

CONTENTS

INTRODUCTION

In an era defined by interconnected digital landscapes and persistent cybersecurity threats, the need for a resilient and proactive defence strategy is paramount. The SnortShield Network project represents a pioneering approach to fortify network security, combining cutting-edge technologies to create a robust defence ecosystem.

At its foundation, SnortShield Network integrates Windows Server Active Directory Domain Services (ADDS), IIS Server, DHCP Server, and pfSense Firewall, with a focal point on the renowned Snort intrusion detection and prevention system. This collaborative ensemble is meticulously orchestrated to establish a comprehensive defence mechanism against a myriad of cyber threats.

This project is not just about mitigating risks; it's about redefining network security paradigms. Windows Server ADDS streamlines user and resource management, IIS Server enables seamless web hosting, DHCP Server optimizes address allocation, and pfSense Firewall bolsters overall network security. However, the linchpin of this initiative lies in the proactive capabilities of Snort, diligently safeguarding the network by identifying and thwarting potential intrusions in real-time.

Join us on this transformative journey with SnortShield Network, where security is not a mere feature but an integral component of every digital interaction. Together, let's fortify the digital landscape and ensure that networks remain resilient, secure, and ready to face the challenges of an ever-evolving cyber terrain.

Motivation

In the ever-expanding digital realm, where innovation and connectivity thrive, so too do the challenges presented by evolving cybersecurity threats. The motivation behind the SnortShield Network project is rooted in the recognition that mere security measures are no longer sufficient; a proactive and adaptive defence strategy is imperative.

We are motivated by the vision of creating a digital landscape where organizations operate with confidence, knowing that their networks are fortified against the relentless tide of cyber threats. SnortShield Network is not merely a response to challenges; it's a commitment to redefine the standards of network security.

Objective

"SnortShield Network aims to create a fortified and responsive network security infrastructure. The objective is to seamlessly integrate Snort intrusion detection, Windows Server ADDS, IIS Server, DHCP Server, and pfSense Firewall to proactively identify and mitigate security threats. Through this project, we strive to establish a secure digital environment, empowering organizations to navigate the digital landscape with confidence and resilience."

Problem Definition

In the face of an ever-evolving digital threat landscape, organizations encounter challenges in establishing a proactive defence against cyber intrusions. The problem at hand is the need for a comprehensive network security infrastructure that not only detects but actively prevents potential threats. Existing solutions often lack the integration required to fortify networks seamlessly. The SnortShield Network project addresses this issue by focusing on the seamless integration of Snort intrusion detection alongside Windows Server ADDS, IIS Server, DHCP Server, and pfSense Firewall. The overarching problem is the absence of a unified defence mechanism, hindering organizations from navigating the digital realm securely and confidently. SnortShield Network seeks to provide a solution that combines these crucial components, offering a cohesive defence strategy against the dynamic threat landscape.

Project Overview

The SnortShield Network project represents a pioneering endeavour to fortify network security in the face of ever-evolving cyber threats. Rooted in the recognition of existing challenges in establishing a unified defence strategy, this project strategically integrates powerful components to create a resilient and proactive security infrastructure.

Key Components:

1. **Snort Intrusion Detection:** Positioned as the cornerstone, Snort brings real-time threat detection capabilities, actively monitoring and mitigating potential security risks.
2. **Windows Server ADDS:** Centralizing user and resource management, ADDS ensures efficient and secure network operations.
3. **IIS Server:** Facilitating seamless web hosting, IIS provides a secure platform for digital interactions.
4. **DHCP Server:** Optimizing address allocation, DHCP enhances network efficiency and scalability.
5. **pfSense Firewall:** Offering comprehensive firewall protection, pfSense strengthens the overall security posture against external threats.

Objective: The primary goal of SnortShield Network is to redefine network security paradigms by seamlessly integrating these components. This initiative aims to empower organizations with a cohesive defence mechanism, enabling them to navigate the digital landscape securely.

Problem Addressed: The project addresses the challenge of fragmented defence mechanisms, providing a solution to the absence of a unified and proactive security infrastructure.

Expected Outcomes:

- Real-time threat detection and mitigation.
- Centralized network management for streamlined operations.
- Seamless and secure web hosting.
- Optimized address allocation for enhanced efficiency.
- Comprehensive defence against cyber threats through pfSense Firewall.

Impact: SnortShield Network anticipates making a substantial impact on network security, providing organizations with the tools needed to proactively defend against cyber threats and ensuring a resilient digital future.

This project overview sets the stage for an innovative and impactful initiative that redefines how network security is approached, emphasizing integration, proactive defence, and adaptability in the face of emerging threats

INSTALLATION and SETUP GUIDE

1. VirtualBox

2.Windows server 2022

3. Windows client 8

4.pFsense

5.SNORT

6.Kali Machine

7.Python 3.7 and later

Cisco Packet Tracer

Cisco Packet Tracer is a network simulation tool developed by Cisco Systems. It is used for educational purposes, particularly in the field of networking and telecommunications. Packet Tracer allows users to design, configure, and troubleshoot network scenarios in a virtual environment, simulating the behaviour of real networks.

It's important to note that while Packet Tracer is a valuable learning tool for networking students and professionals, it may not include all the features and complexities found in a production network environment. For more advanced and real-world scenarios, network engineers often use other simulation tools or physical equipment

Hierarchical Network Design

Hierarchical network design is an approach to designing computer networks that emphasizes organizing the network into distinct layers or tiers. This design philosophy helps in creating scalable, modular, and easily manageable networks. The primary goal of hierarchical network design is to improve performance, manageability, and flexibility

A typical hierarchical network design consists of three main layers:

Access Layer

- The access layer is the lowest layer in the hierarchy, responsible for connecting end-user devices such as computers, printers, and IP phones to the network
- This layer focuses on providing network access, user authentication, and controlling user traffic
- Devices commonly found at the access layer include switches and access points

Distribution Layer

- The distribution layer serves as an aggregation point for the access layer. It consolidates and manages the traffic from multiple access layer devices
- This layer is responsible for routing between different VLANs (Virtual Local Area Networks) and implementing policies, filtering, and access control lists (ACLs
- Distribution layer devices often include layer 3 switches and routers

Core Layer

- The core layer is the backbone of the network and is responsible for high-speed, efficient transportation of data between distribution layer devices
- It is designed for high-speed, low-latency communication and should be kept as simple as possible to ensure fast data transfer
- Redundancy and fault tolerance are crucial considerations in the core layer to prevent network outages
- Devices at the core layer are typically high-performance routers and switches

Network using Cisco Packet Tracer

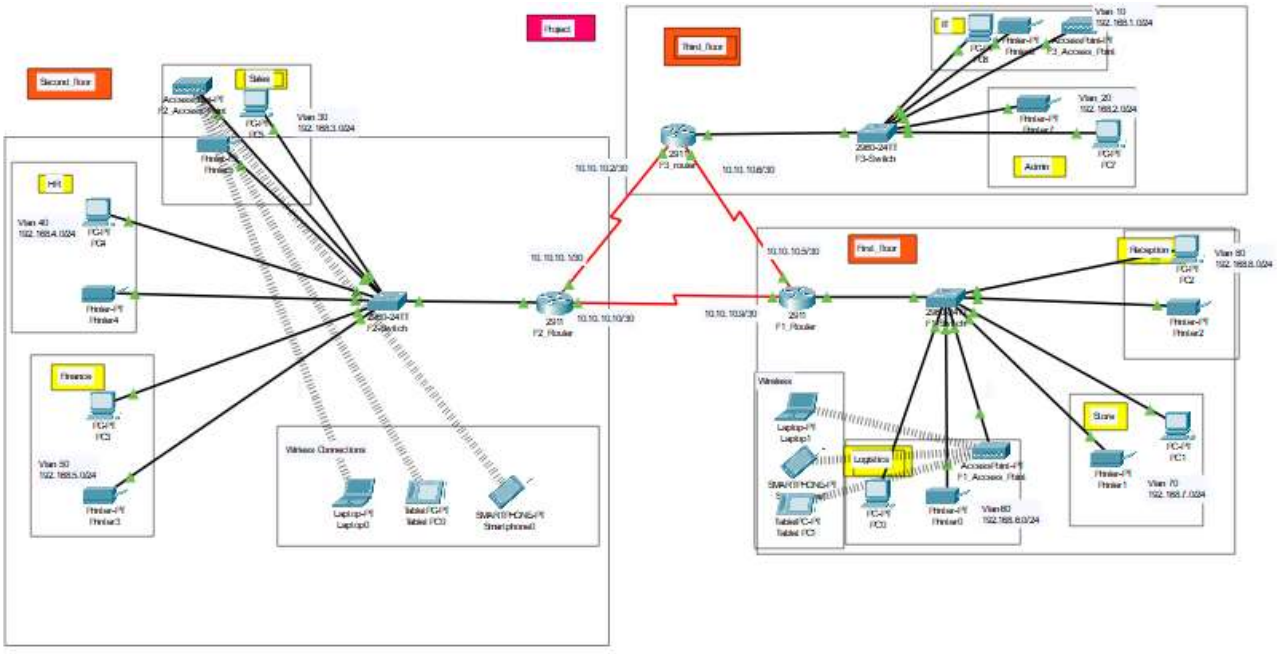


Fig1.0

Source code of the Hotel Management system.

VLAN

VLAN, which stands for Virtual Local Area Network, is a technology used in computer networking to create logically segmented networks within a physical local area network (LAN). VLANs allow network administrators to group devices together, even if they are not physically located on the same network switch.

VLANs are widely used in enterprise networks to improve network efficiency, simplify network management, and enhance security. They are also employed in service provider networks and data centre environments where segmentation and isolation of network traffic are essential

VLAN Configuration implement in our Project

SWITCH F1

```
Switch>enable
Switch# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int range fa0/2-3
Switch(config-if-range)#switchport access vlan 80
% Access VLAN does not exist. Creating vlan 80
Switch(config-if-range)#ex
Switch(config)#interface range fastEthernet 0/6-7
Switch(config-if-range)#switchport access vlan 70
% Access VLAN does not exist. Creating vlan 70
Switch(config-if-range)#exit
Switch(config)#interface range fastEthernet 0/4-5
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 60
% Access VLAN does not exist. Creating vlan 60
Switch(config-if-range)#int fa0/8
```

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 60

Switch(config-if)#do wr

Building configuration...

SWITCH F2

Switch>enable

Switch# conf t

Switch(config)#int range f0/6-7

Switch(config-if-range)#switchport mode access

Switch(config-if-range)#switchport access vlan 50

% Access VLAN does not exist. Creating vlan 50

Switch(config-if-range)#do wr

Building configuration...

[OK]

Switch(config-if-range)#exit

Switch(config)#int range f0/4-5

Switch(config-if-range)#switchport mode access

Switch(config-if-range)#switchport access vlan 40

% Access VLAN does not exist. Creating vlan 40

Switch(config-if-range)#do wr

Building configuration...

[OK]

Switch(config-if-range)#ex

Switch(config)#int range f0/2-3

Switch(config-if-range)#switchport mode access

Switch(config-if-range)#sw

Switch(config-if-range)#switchport access vlan 30

% Access VLAN does not exist. Creating vlan 30

Switch(config-if-range)#do wr

SWITCH F3

Switch>enable

Switch#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#int fa0/3

Switch(config-if)#switchport mode access

Switch(config-if)#switchport acces vlan 10

% Access VLAN does not exist. Creating vlan 10

Switch(config-if)#do wr

Building configuration...

[OK]

Switch(config-if)#exit

Switch(config)#int range f0/5-6

Switch(config-if-range)#switchport mode access

Switch(config-if-range)#switchport access vlan 10

Switch(config-if-range)#do wr

Building configuration...

[OK]

Switch(config-if-range)#exit

Switch(config)#interface fastEthernet 0/4

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 20

% Access VLAN does not exist. Creating vlan 20

Switch(config-if)#int fa0/2

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 20

Switch(config-if)#exit

Inter-VLAN Routing

Inter-VLAN routing can be defined as a way to forward traffic between different VLAN by implementing a router in the network. As we learnt previously, VLANs logically segment the switch into different subnets, when a router is connected to the switch, an administrator can configure the router to forward the traffic between the various VLANs configured on the switch. We are nodes in the VLANs forwards traffic to the router which then forwards the traffic to the destination network regardless of the VLAN configured on the switch.

Configuring Inter-VLAN Routing (Router on a stick)

INTERVLAN F1 CONFIGURATION

Router>enable

Router# conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#int g0/0

Router(config-if)#exit

Router(config)#int g0/0.80

Router(config-subif)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0.80, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.80, changed state to up

Router(config-subif)#encapsulation dot1Q 80

Router(config-subif)#ip address 192.168.8.1 255.255.255.0

Router(config-subif)#exit

Router(config)#int gigabitEthernet 0/0.70

Router(config-subif)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0.70, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.70, changed state to up

Router(config-subif)#encapsulation dot1Q 70

Router(config-subif)#ip address 192.168.7.1 255.255.255.0

Router(config-subif)#exit

Router(config)#int gigabitEthernet 0/0.60

Router(config-subif)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0.60, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.60, changed state to up

Router(config-subif)#encapsulation dot1Q

% Incomplete command.

Router(config-subif)#ip ad

Router(config-subif)#ip address 192.168.6.1 255.255.255.0

% Configuring IP routing on a LAN subinterface is only allowed if that

subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,

or ISL vLAN.

Router(config-subif)#exit

INTERVLAN F2 CONFIGURATION

Router>enable

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#int g0/0.30

Router(config-subif)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0.30, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.30, changed state to up

Router(config-subif)#encapsulation dot1Q 30

Router(config-subif)#ip address 192.168.3.1 255.255.255.0

Router(config-subif)#exit

Router(config)#int g0/0.40

Router(config-subif)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0.40, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.40, changed state to up

Router(config-subif)#ip address 192.168.4.1 255.255.255.0

% Configuring IP routing on a LAN subinterface is only allowed if that

subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,

or ISL vLAN.

Router(config-subif)#encapsulation dot1Q 40

Router(config-subif)#ip address 192.168.4.1 255.255.255.0

Router(config-subif)#exit

Router(config)#int g0/0.50

%LINK-5-CHANGED: Interface GigabitEthernet0/0.50, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.50, changed state to up

Router(config-subif)#ip address 192.168.5.1 255.255.255.0

% Configuring IP routing on a LAN subinterface is only allowed if that

subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,

or ISL vLAN.

Router(config-subif)#encapsulation dot1Q 50

Router(config-subif)#ip address 192.168.5.1 255.255.255.0

Router(config-subif)#exit

Router(config)#do wr

Building configuration...

[OK]

INTERVLAN F3 CONFIGURATION

Router>enable

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#int g0/0.30

%LINK-5-CHANGED: Interface GigabitEthernet0/0.30, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.30, changed state to up

Router(config-subif)#encapsulation dot1Q 30

Router(config-subif)#ip address 192.168.3.1 255.255.255.0

```
Router(config-subif)#exit
Router(config)#int g0/0.40
%LINK-5-CHANGED: Interface GigabitEthernet0/0.40, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.40, changed state to up
Router(config-subif)#ip add
Router(config-subif)#ip address 192.168.4.1 255.255.255.0
% Configuring IP routing on a LAN subinterface is only allowed if that
subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,
or ISL vLAN.
Router(config-subif)#encapsulation dot1Q 40
Router(config-subif)#ip address 192.168.4.1 255.255.255.0
Router(config-subif)#exit
Router(config)#int g0/0.50
Router(config-subif)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0.50, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.50, changed state to up
Router(config-subif)#ip address 192.168.5.1 255.255.255.0
% Configuring IP routing on a LAN subinterface is only allowed if that
subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,
or ISL vLAN.
Router(config-subif)#encapsulation dot1Q 50
Router(config-subif)#ip address 192.168.5.1 255.255.255.0
Router(config-subif)#exit
Router(config)#do wr
Building configuration...
[OK]
```

DHCP server configuration in CPT

Network nodes require an IP address configuration that usually comprises the IP address, subnet mask, default gateway IP for the router, name servers and other values.

Administrators can manually set this information, resulting in a static configuration. A Dynamic Host Configuration Protocol (DHCP) server can also dynamically provide the information.

Generally, servers, routers, network printers and other such devices have a static configuration. Workstations, laptops, phones, tablets and other end-user devices receive their configuration via DHCP.

Create a scope

Before building the first pool of available IP addresses, it's critical to plan the deployment. Devices, such as servers, routers and even printers, may have static IP address configuration. Make sure you have identified these addresses and that allow for them in scope. Many administrators place all statically assigned IP addresses at the front of the scope

Configuring DHCP Server (Router as the DHCP Server) implement in our Project

DHCP and INTER VLAN Configuration Router F1

```
Router(config)#service dhcp
Router(config)#ip dhcp pool reception
Router(dhcp-config)#network 192.168.8.0 255.255.255.0
Router(dhcp-config)#default-router 192.168.8.1
Router(dhcp-config)#dns-server 192.168.8.1
Router(dhcp-config)#exit
Router(config)#ip dhcp pool store
Router(dhcp-config)#network 192.168.7.0 255.255.255.0
Router(dhcp-config)#default-router 192.168.7.1
Router(dhcp-config)#dns-server 192.168.7.1
```

Router(dhcp-config)#exit

Router(config)#ip dhcp pool logistics

Router(dhcp-config)#network 192.168.6.0 255.255.255.0

Router(dhcp-config)#dns-server 192.168.6.1

Router(dhcp-config)#default-router 192.168.6.1

Router(dhcp-config)#exit

Router(config)#int g0/0.60

Router(config-subif)#encapsulation dot1Q 60

Router(config-subif)#ip address 192.168.6.1 255.255.255.0

Router(config-subif)#exit

Router(config)#ip dhcp pool Logistics

Router(dhcp-config)#network 192.168.6.0 255.255.255.0

Router(dhcp-config)#default-router 192.168.6.1

Router(dhcp-config)#dns-server 192.168.6.1

DHCP and INTER VLAN Configuration Router 2

Router(config-subif)#encapsulation dot1Q 40

Router(config-subif)#ip address 192.168.4.1 255.255.255.0

Router(config-subif)#ex

Router(config)#int g0/0.50

Router(config-subif)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0.50, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.50, changed state to up

Router(config-subif)#ip address 192.168.5.1 255.255.255.0

% Configuring IP routing on a LAN subinterface is only allowed if that

subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,

or ISL vLAN.

Router(config-subif)#encapsulation dot1Q 50

Router(config-subif)#ip address 192.168.5.1 255.255.255.0

Router(config-subif)#exit

```
Router(config)#do wr
Building configuration...
[OK]
Router(config)#ip dhcp pool Finance
Router(dhcp-config)#network 192.168.5.0 255.255.255.0
Router(dhcp-config)#default-router 192.168.5.1
Router(dhcp-config)#dns-server 192.168.5.1
Router(dhcp-config)#exit
Router(config)#ip dhcp pool HR
Router(dhcp-config)#network 192.168.4.0 255.255.255.255
Router(dhcp-config)#no network 192.168.4.0 255.255.255.255
Router(dhcp-config)#network 192.168.4.0 255.255.255.0
Router(dhcp-config)#default-router 192.168.4.1
Router(dhcp-config)#dns-server 192.168.4.1
Router(dhcp-config)#exit
Router(config)#ip dhcp pool Sales
Router(dhcp-config)#network 192.168.3.0 255.255.255.0
Router(dhcp-config)#default-router 192.168.3.1
Router(dhcp-config)#dns-server 192.168.3.1
Router(dhcp-config)#exit
Router(config-if)#int g0/0.40
%LINK-5-CHANGED: Interface GigabitEthernet0/0.40, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.40, changed state to up
Router(config-subif)#encapsulation dot1Q 40
Router(config-subif)#ip address 192.168.5.1 255.255.255.0
% 192.168.5.0 overlaps with GigabitEthernet0/0.50
Router(config-subif)#no ip address 192.168.5.1 255.255.255.0
Router(config-subif)#ip address 192.168.4.1 255.255.255.0
Router(config-subif)#do wr
Building configuration...
```

[OK]

Router(config)#ip dhcp pool HR

Router(dhcp-config)#network 192.168.4.0 255.255.255.0

Router(dhcp-config)#default-router 192.168.4.1

Router(dhcp-config)#dns-server 192.168.4.1

Router(dhcp-config)#exit

Router(config)#interface g0/0.40

Router(config-subif)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0.40, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.40, changed state to up

Router(config-subif)#encapsulation dot1Q 40

Router(config-subif)#ip address 192.168.4.1 255.255.255.0

Router(config-subif)#exit

Router(config)#do wr

Building configuration...

[OK]

Router(config)#ip dhcp pool hr

Router(dhcp-config)#network 192.168.4.0 255.255.255.0

Router(dhcp-config)#dns-server 192.168.4.1

Router(dhcp-config)#default-router 192.168.4.1

Router(dhcp-config)#exit

DHCP Configuration Router 3

ip dhcp pool IT
network 192.168.1.0 255.255.255.0
default-router 192.168.1.1
dns-server 192.168.1.1

ip dhcp pool admin
network 192.168.2.0 255.255.255.0
default-router 192.168.2.1
dns-server 192.168.2.1

Trunk Port

A trunk port is a specific type of port on a network switch that allows data to flow across a network node for multiple virtual local area networks or VLANs. Think of the trunk port as a “bundle” of individual branches or capillaries in a telecom network connection.

The typical VLAN network is made up of virtualized network nodes. By contrast, the traditional network was a series of pieces of hardware connected together, where each one was its own network node.

Now, by the principle of virtualization, these pieces of hardware can be endowed with virtual partitions through “extra logic,” to change how they handle data

Trunk Port Configuration implement in our Project

```
Switch(config)#int f0/1

Switch(config-if) #switchport mode trunk

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up

Switch(config-if) #do wr

Building configuration...

[OK]
```

SSH (Secure Shell)

SSH, which stands for Secure Shell, is a cryptographic network protocol used for secure communication over an unsecured network. It is widely used for securely accessing and managing network devices, servers, and other computing resources. SSH provides a secure alternative to traditional methods such as Telnet, which transmit data in plain text, making them susceptible to interception and unauthorized access.

Configuring SSH for secure Remote access.

Configuring SSH (Secure Shell) for secure remote access involves setting up the SSH service on a device (such as a router or server) to allow encrypted and secure communication between the device and remote clients. Here's a basic guide on how to configure SSH

SSH Configuration in F1 Router

Router>enable

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#hostname F1_router

F1_router(config)#ip domain-name nielit

F1_router(config)#username admin password admin

F1_router(config)#crypto key gen

F1_router(config)#crypto key generate rsa

The name for the keys will be: F1_router.nielit

Choose the size of the key modulus in the range of 360 to 4096 for your

General Purpose Keys. Choosing a key modulus greater than 512 may take

a few minutes.

How many bits in the modulus [512]: 1024

% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

F1_router(config)#line vty 0 15

F1_router(config-line)#login local

F1_router(config-line)#transport input ssh

F1_router(config-line)#do wr

Building configuration...

[OK]

SSH Configuration in F2 Router

Router>enable

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#hostname F2_router

F2_router(config)#ip domain-name nielit

F2_router(config)#user

F2_router(config)#username admin password admin

F2_router(config)#crypto key generate rsa

The name for the keys will be: F2_router.nielit

Choose the size of the key modulus in the range of 360 to 4096 for your

General Purpose Keys. Choosing a key modulus greater than 512 may take

a few minutes.

How many bits in the modulus [512]: 1024

% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

F2_router(config)#line vty 0 15

F2_router(config-line)#login local

F2_router(config-line)#transport in

F2_router(config-line)#transport input ssh

F2_router(config-line)#do wr

Building configuration...

[OK]

SSH Configuration in F3 Router

Router>enable

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

.Router(config)#hostname F2_router

F2_router(config)#ip domain-name nielit

F2_router(config)#username admin password admin

F2_router(config)#crypto key generate rsa

The name for the keys will be: F2_router.nielit

Choose the size of the key modulus in the range of 360 to 4096 for your

General Purpose Keys. Choosing a key modulus greater than 512 may take

a few minutes.

How many bits in the modulus [512]: 1024

% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

F2_router(config)#line vty 0 15

F2_router(config-line)#login local

F2_router(config-line)#transport input ssh

F2_router(config-line)#do wr

Building configuration...

[OK]

OSPF

OSPF, which stands for Open Shortest Path First, is a link-state routing protocol used in computer networks. It is one of the Interior Gateway Protocols (IGPs) designed to help routers dynamically learn and exchange routing information within an autonomous system (AS). OSPF is particularly well-suited for large and complex networks due to its efficient and scalable design

Open Shortest Path First (OSPF) protocol States

Open Shortest Path First (OSPF) is a link-state routing protocol that is used to find the best path between the source and the destination router using its own Shortest Path First). OSPF is developed by Internet Engineering Task Force (IETF) as one of the Interior Gateway Protocol (IGP), i.e, the protocol which aims at moving the packet within a large autonomous system or routing domain. It is a network layer protocol which works on protocol number 89 and uses AD value 110. OSPF uses multicast address 224.0.0.5 for normal communication and 224.0.0.6 for update to designated router (DR)/Backup Designated Router (BDR).

OSPF configuration in F1 Router

Router>enable

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#

Router(config)#

Router(config)#router ospf 10

Router(config-router)#network 10.10.10.4 255.255.255.252 area 0

Router(config-router)#no network 10.10.10.4 255.255.255.252 area 0

Router(config-router)#network 10.10.10.5 255.255.255.252 area 0

Router(config-router)#network 10.10.10.9 255.255.255.252 area 0

Router(config-router)#network 192.168.8.0 255.255.255.0 area 0

Router(config-router)#network 192.168.7.0 255.255.255.0 area 0

Router(config-router)#network 192.168.6.0 255.255.255.0 area 0

Router(config-router)#do wr

Building configuration...

[OK]

OSPF configuration in F2 Router

Router>enable

Router# conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#router ospf 10

Router(config-router)#network 10.10.10.10 255.255.255.252 area 0

Router(config-router)#network 10.10.10.1 255.255.255.252 area 0

07:37:29: %OSPF-5-ADJCHG: Process 10, Nbr 192.168.8.1 on Serial0/1/1 from LOADING to FULL, Loading Done

Router(config-router)#network 10.10.10.1 255.255.255.252 area 0

Router(config-router)#network 192.168.3.0 255.255.255.0 area 0

Router(config-router)#network 192.168.4.0 255.255.255.0 area 0

Router(config-router)#network 192.168.5.0 255.255.255.0 area 0

Router(config-router)#do wr

Building configuration...

[OK]

OSPF configuration in F3 Router

Router>enable

Router# conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#router ospf 10

Router(config-router)#network 10.10.10.10 255.255.255.252 area 0

Router(config-router)#network 10.10.10.1 255.255.255.252 area 0

07:37:29: %OSPF-5-ADJCHG: Process 10, Nbr 192.168.8.1 on Serial0/1/1 from LOADING to FULL, Loading Done

Router(config-router)#network 10.10.10.1 255.255.255.252 area 0

Router(config-router)#network 192.168.3.0 255.255.255.0 area 0

Router(config-router)#network 192.168.4.0 255.255.255.0 area 0

Router(config-router)#network 192.168.5.0 255.255.255.0 area 0

Router(config-router)#do wr

Building configuration...

[OK]

Switchport Security

Switchport security is a feature available on network switches that enhances security by controlling which devices are allowed to connect to individual switch ports. This feature is particularly useful in preventing unauthorized devices from gaining access to the network and helps mitigate security risks associated with unauthorized access or network attacks.

Configuring switchport security on the switches

Switchport security, also known as port security, is a feature on network switches that allows you to control which devices are allowed to connect to individual switch ports. This feature helps enhance network security by restricting unauthorized devices from gaining network access. Port security is commonly used to prevent unauthorized devices, such as rogue computers or unauthorized access points, from connecting to the network.

Here are the basic steps to configure switchport security on a Cisco switch. Keep in mind that specific commands might vary depending on the switch model and the version of the operating system it is running (e.g., Cisco IOS).

Port - Security Configuration implement in our Project

Switch>enable

Switch#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#int f0/3

Switch(config-if)#switchport port-security maximum 1

Switch(config-if)#switchport port-security mac-address sticky

Switch(config-if)#switchport port-security violation shutdown

Switch(config-if)#do wr

Building configuration...

[OK]

SNORT SHIELD NETWORK

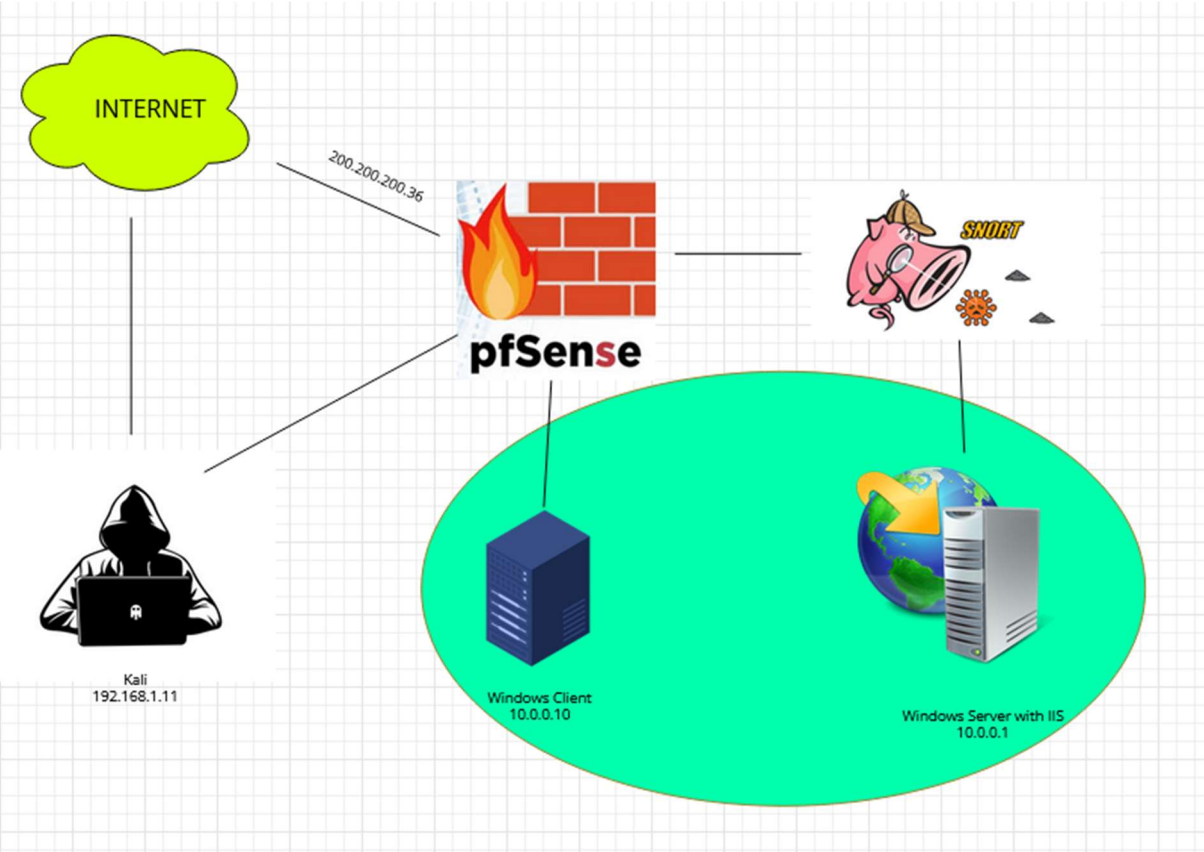


Fig 2.0

Windows Server

Windows Server is a line of Microsoft operating systems (OSes) comprised of extremely powerful machines. Windows Server was first launched in April 2003. It's typically installed on heavy-use servers serving as a backbone for most IT companies, applications, and services. The server handles the administrative group-related activities on a network. It organizes, stores, sends, and receives files from devices connected to a network.

ADDS (Active Directory Domain Services)

Active Directory Domain Services (AD DS) is a server role in Active Directory that allows admins to manage and store information about resources from a network, as well as application data, in a distributed database.

AD DS helps admins manage network elements -- both computing devices , users and reorder them into a custom hierarchical structure. AD DS also integrates security by authenticating logons and controlling access to directory resources

Active Directory is a directory service that runs on Microsoft Windows Server. It is used for identity and access management. AD DS stores and organizes information about the people, devices and services connected to a network. AD DS serves as a locator service for those objects and as a way for organizations to have a central point of administration for all activity on the corporate network.

DHCP with Web Server

A DHCP (Dynamic Host Configuration Protocol) server is a network server that automatically provides and assigns IP addresses and other related network configuration information to devices on a TCP/IP network. This information typically includes subnet mask, default gateway, DNS (Domain Name System) servers, and other parameters required for devices to communicate effectively on the network

Combining DHCP (Dynamic Host Configuration Protocol) with a web server can be useful in scenarios where you want to dynamically assign IP addresses to devices on our network and provide additional configuration information via a web interface.

IIS Server

Internet Information Services, also known as IIS, is a Microsoft web server that runs on Windows operating system and is used to exchange static and dynamic web content with internet users.

IIS uses various protocols for communication and data exchange with remote clients or computers, such as HTTP, SMTP, and FTP. As a core Windows product, IIS comes integrated with Windows Server and runs on Windows OS.

IIS comes with built-in authentication, authorization, and access control features to strengthen our web application security. We can create system and application administrator accounts individually for granular-level access. Other security features include request filtering to whitelist/blacklist traffic, dynamic IP blocking, SSL and TLS encryption, webpage compression, and FTP-specific security controls.

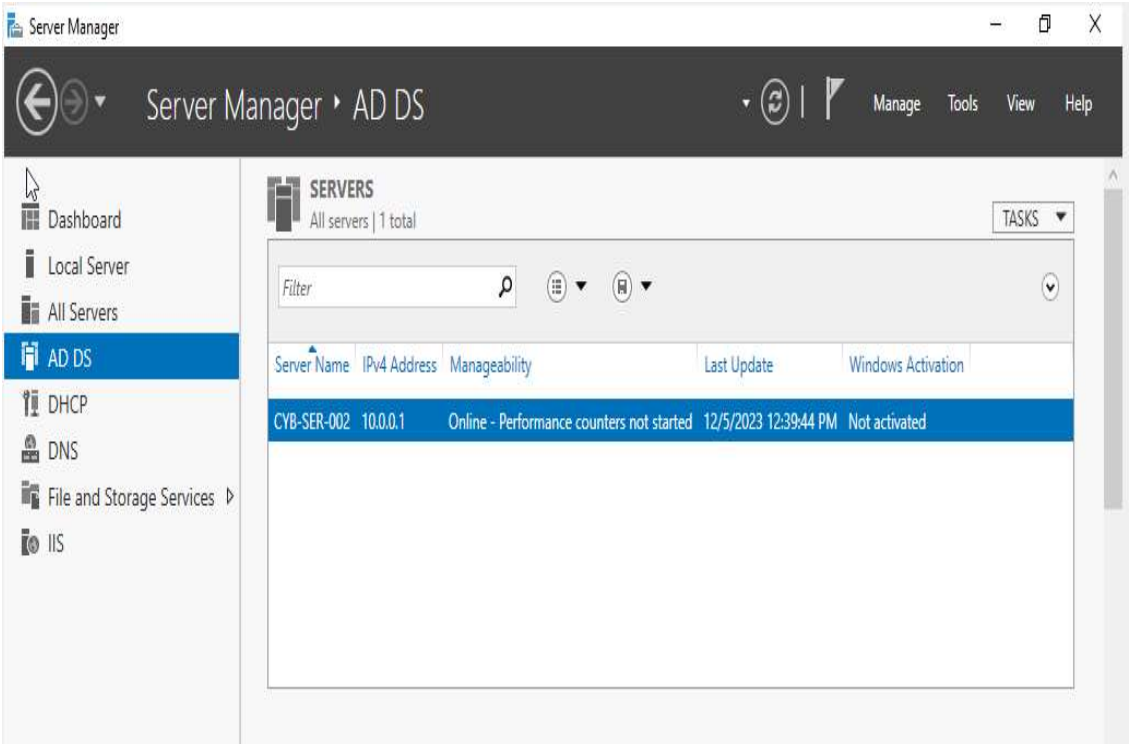


Fig 3.1

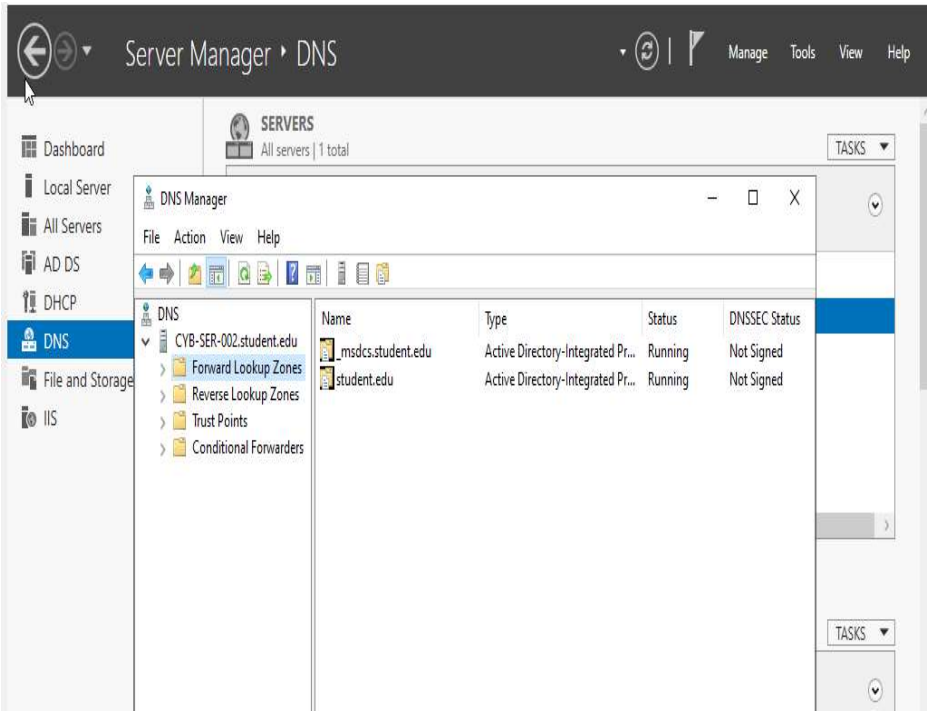

Server Manager • DNS
Manage
Tools
View
Help
Dashboard
Local Server
All Servers
AD DS
DHCP
DNS
File and Storage
IIS
SERVERS
All servers | 1 total
TASKS
DNS Manager
File Action View Help
DNS
CYB-SER-002.student.edu
Forward Lookup Zones
Reverse Lookup Zones
Trust Points
Conditional Forwarders
Name
Type
Status
DNSSEC Status
_msdcs.student.edu
Active Directory-Integrated Pr...
Running
Not Signed
student.edu
Active Directory-Integrated Pr...
Running
Not Signed

Fig 3.2

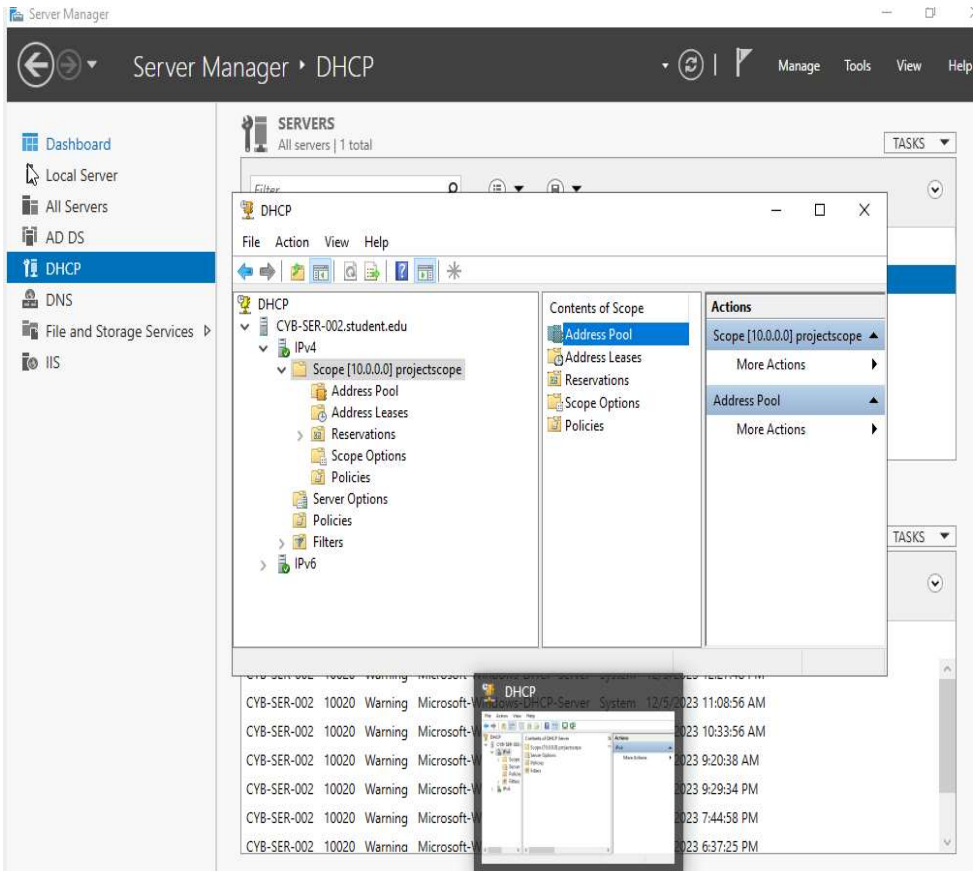

Server Manager
Server Manager • DHCP
Manage
Tools
View
Help
Dashboard
Local Server
All Servers
AD DS
DHCP
DNS
File and Storage Services
IIS
SERVERS
All servers | 1 total
TASKS
DHCP
File Action View Help
DHCP
CYB-SER-002.student.edu
IPv4
Scope [10.0.0.0] projectscope
Address Pool
Address Leases
Reservations
Scope Options
Policies
Server Options
Policies
Filters
IPv6
Contents of Scope
Address Pool
Address Leases
Reservations
Scope Options
Policies
Actions
Scope [10.0.0.0] projectscope
More Actions
Address Pool
More Actions
CYB-SER-002 10020 Warning
12/5/2023 11:08:56 AM
10:33:56 AM
9:20:38 AM
9:29:34 PM
7:44:58 PM
6:37:25 PM

Fig 3.3

This helps you separate your web applications in IIS for better security and availability. An application pool has single or multiple applications managed by one or more worker processes in IIS. A worker process handles the client requests specific to an application pool. Isolation ensures the crashing or failure of an application in a particular pool doesn't affect the applications in other pools.

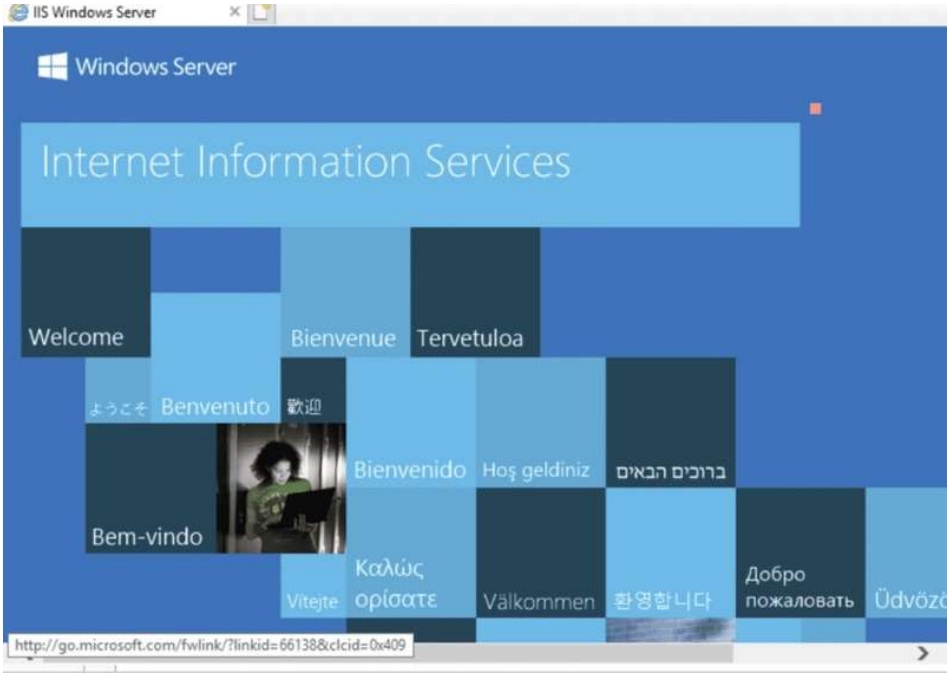


Fig 3.4

HTML Code of the TODO page

```
/* Variables */

:root {

        --primary: #EA40A4;

        --business: #3a82ee;

        --personal: var(--primary);

        --light: #EEE;

        --grey: #888;

        --dark: #313154;

        --danger: #ff5b57;

--shadow: 0 1px 3px rgba(0, 0, 0, 0.1);
```

```
--business-glow: 0px 0px 4px rgba(58, 130, 238, 0.75);
--personal-glow: 0px 0px 4px rgba(234, 64, 164, 0.75);
}
/* End of Variables */
/* Resets */
* {
	margin: 0;
	padding: 0;
	box-sizing: border-box;
	font-family: 'montserrat', sans-serif;
}
input:not([type="radio"]):not([type="checkbox"]), button {
	appearance: none;
	border: none;
	outline: none;
	background: none;
	cursor: initial;
}
/* End of Resets */
body {
	background: var(--light);
	color: var(--dark);
}
section {
	margin-top: 2rem;
	margin-bottom: 2rem;
	padding-left: 1.5rem;
	padding-right: 1.5rem;
```

```
}
h3 {
	color: var(--dark);
	font-size: 1rem;
	font-weight: 400;
	margin-bottom: 0.5rem;
}
h4 {
	color: var(--grey);
	font-size: 0.875rem;
	font-weight: 700;
	margin-bottom: 0.5rem;
}
.greeting .title {
	display: flex;
}
.greeting .title input {
	margin-left: 0.5rem;
	flex: 1 1 0%;
	min-width: 0;
}
.greeting .title,
.greeting .title input {
	color: var(--dark);
	font-size: 1.5rem;
	font-weight: 700;
}
.create-todo input[type="text"] {
```

```
        display: block;
        width: 100%;
        font-size: 1.125rem;
        padding: 1rem 1.5rem;
        color: var(--dark);
        background-color: #FFF;
        border-radius: 0.5rem;
        box-shadow: var(--shadow);
        margin-bottom: 1.5rem;
}
.create-todo .options {
        display: grid;
        grid-template-columns: repeat(2, 1fr);
        grid-gap: 1rem;
        margin-bottom: 1.5rem;
}
.create-todo .options label {
        display: flex;
        flex-direction: column;
        align-items: center;
        justify-content: center;
        background-color: #FFF;
        padding: 1.5rem;
        box-shadow: var(--shadow);
        border-radius: 0.5rem;
        cursor: pointer;
}
input[type="radio"],
```

```
input[type="checkbox"] {
	display: none;
}
.bubble {
	display: flex;
	align-items: center;
	justify-content: center;
	width: 20px;
	height: 20px;
	border-radius: 999px;
	border: 2px solid var(--business);
	box-shadow: var(--business-glow);
}
.bubble.personal {
	border-color: var(--personal);
	box-shadow: var(--personal-glow);
}
.bubble::after {
	content: '';
	display: block;
	opacity: 0;
	width: 0px;
	height: 0px;
	background-color: var(--business);
	box-shadow: var(--business-glow);
	border-radius: 999px;
	transition: 0.2s ease-in-out;
}
```

```
.bubble.personal::after {
	background-color: var(--personal);
	box-shadow: var(--personal-glow);
}
input:checked ~ .bubble::after {
	width: 10px;
	height: 10px;
	opacity: 1;
}
.create-todo .options label div {
	color: var(--dark);
	font-size: 1.125rem;
	margin-top: 1rem;
}
.create-todo input[type="submit"] {
	display: block;
	width: 100%;
	font-size: 1.125rem;
	padding: 1rem 1.5rem;
	color: #FFF;
	font-weight: 700;
	text-transform: uppercase;
	background-color: var(--primary);
	box-shadow: var(--personal-glow);
	border-radius: 0.5rem;
	cursor: pointer;
	transition: 0.2s ease-out;
}
```

```
.create-todo input[type="submit"]:hover {
        opacity: 0.75;
}
.todo-list .list {
        margin: 1rem 0;
}
.todo-list .todo-item {
        display: flex;
        align-items: center;
        background-color: #FFF;
        padding: 1rem;
        border-radius: 0.5rem;
        box-shadow: var(--shadow);
        margin-bottom: 1rem;
}
.todo-item label {
        display: block;
        margin-right: 1rem;
        cursor: pointer;
}
.todo-item .todo-content {
        flex: 1 1 0%;
}
.todo-item .todo-content input {
        color: var(--dark);
        font-size: 1.125rem;
}
.todo-item .actions {
```

```
        display: flex;
        align-items: center;
}
.todo-item .actions button {
        display: block;
        padding: 0.5rem;
        border-radius: 0.25rem;
        color: #FFF;
        cursor: pointer;
        transition: 0.2s ease-in-out;
}
.todo-item .actions button:hover {
        opacity: 0.75;
}
.todo-item .actions .edit {
        margin-right: 0.5rem;
        background-color: var(--primary);
}
.todo-item .actions .delete {
        background-color: var(--danger);
}
.todo-item.done .todo-content input {
        text-decoration: line-through;
        color: var(--grey);
}
```

Screenshot of webpage from windows client

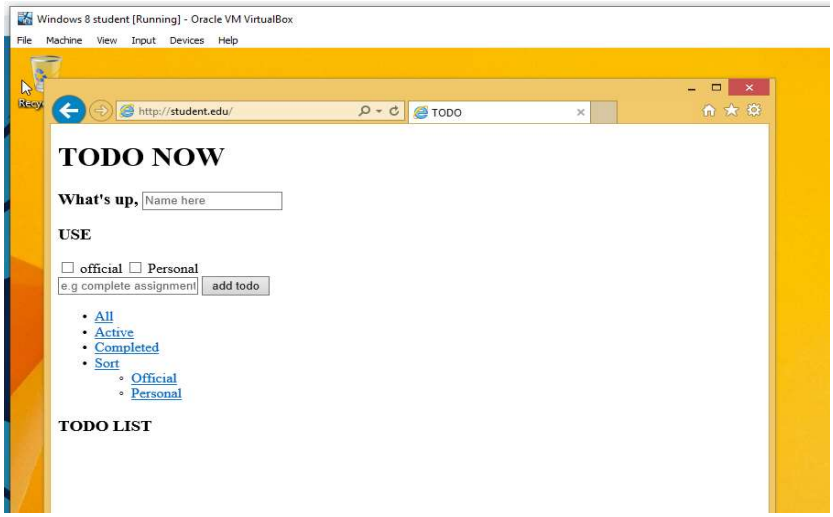


Fig 4.1

Screenshot of webpage from Kali client

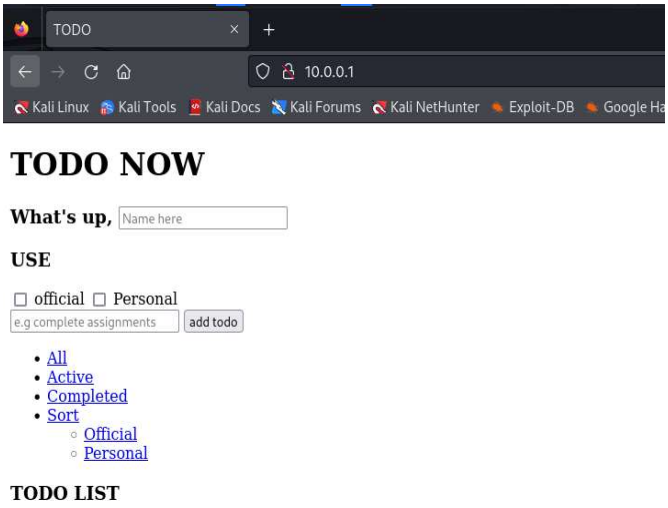


Fig 4.2

Port Scanning

The process of scanning a computer's port is called port scanning. It provides information on whether a device's ports are open, closed or filtered. It is mainly performed to identify if a port is sending or receiving any information.

Port scanning also involves the sending of data to specific ports and analysing the responses to identify vulnerabilities.

It is also one of the techniques used by attackers to discover devices/services they can break into.

Port Scanning Work

A port scanner inspects your IP address block for hosts and open ports using Transmission Control Protocol/Internet Protocol (TCP/IP) network protocols.

To learn how it exactly works, we need to deep dive into the basic components of port scanning – ports, port numbers and the techniques used to accomplish it.

Purpose of Port Scanning

The purpose of port scanning is to acquire information from the servers to which the ports are attached. Port scanning is carried out by system administrators to monitor endpoints as well as by attackers for malicious purposes

Port Scanning Important

Since port scanning identifies open ports and services available on a network, it is used by security professionals to identify any security vulnerabilities on that particular network. While it is highly essential for network management, it is unfortunately being used extensively by cybercriminals as well.

SOURCE CODE of PORTSCAN

#!/usr/bin/python

#(The line #!/usr/bin/python is known as a shebang or hash bang. It's used at the beginning of a script file in Unix-like operating systems to specify the interpreter that should be used to run the script. This indicates that the script should be interpreted and executed using the Python interpreter located at /usr/bin/python. When you run a script with a shebang from the command line, the operating system uses the specified interpreter to execute the script.)

import socket

#(The import socket statement in Python is used to include the functionality of the socket module in your code. The socket module provides a set of classes and functions that enable network communication. Sockets are fundamental building blocks for networking in software development.)

```
import sys
```

#(The import sys statement in Python is used to include the functionality of the sys module in our code. The sys module provides access to some variables used or maintained by the Python interpreter and functions that interact strongly with the interpreter. It's often used for command-line argument processing, interacting with the interpreter, and exiting the program.)

```
import time
```

#(The import time statement in Python is used to include the functionality of the time module in our code. The time module provides various time-related functions, including functions for working with timestamps, measuring time intervals in our code.)

```
import threading
```

#(The import threading statement in Python is used to include the functionality of the threading module in our code. The threading module provides a way to create and work with threads, which are smaller units of a process. Threads allow you to execute multiple tasks concurrently.)

```
usage = "python port_scan.py TARGET START_PORT END_POINT"
```

#(This statement is providing a usage statement for a Python script named port_scan.py. This usage statement suggests that the script expects three command-line arguments:

1. TARGET: The target IP address or hostname that we want to perform a port scan on.
2. START_PORT: The starting port number for the port scan range.
3. END_PORT: The ending port number for the port scan range.)

```
print("_" * 70)
```

#(The line print("_" * 70) is a Python code snippet that prints a line consisting of 70 underscores (_) to the console.)

```
print("python simple port scanner")
```

#(The line print("python simple port scanner") is a Python code snippet that prints the string "python simple port scanner" to the console. The print function is commonly used in Python to display information or messages.)

print("_" * 70)

start_time = time.time()

#(The line start_time = time.time() is a Python code snippet that captures the current time in seconds since the epoch using the time module and assigns it to the variable start_time. This is often used to measure the starting point of an operation or to calculate the elapsed time during the execution of a program.)

if len(sys.argv) != 4:

#(The if len(sys.argv)!=4 is a conditional statement in Python, and it is commonly used to check whether the number of command-line arguments provided to a script is not equal to 4.)

print(usage)

#(The line print(usage) would print the content of the usage variable to the console.)

sys.exit()

#(This is a function from the sys module in Python, and its primary purpose is to terminate the program.)

try:

#(This statement in Python is used to implement exception handling. It allows you to write a block of code in which exceptions may occur)

target = socket.gethostbyname(sys.argv[1])

#(The line target = socket.gethostbyname(sys.argv[1]) is a Python code snippet that uses the socket module to obtain the IP address associated with a given hostname.)

except socket.gaierror:

#(This statement is used to catch exceptions of type socket.gaierror)

print("Name resolution error")

#(The line print("Name resolution error") is a simple Python statement that prints the string "Name resolution error" to the console.)

sys.exit()

#(This is a function from the sys module in Python, and its primary purpose is to terminate the program)

start_port = int(sys.argv[2])

#(The line start_port = int(sys.argv[2]) is a Python code snippet that converts the third command-line argument (provided when running the script) into an integer and assigns it to the variable start_port)

end_port = int(sys.argv[3])

#(The line end_port = int(sys.argv[3]) is a Python code snippet that converts the fourth command-line argument (provided when running the script) into an integer and assigns it to the variable end_port.)

print("scanning target", target)

#(The line print("scanning target", target) is a Python code snippet that prints a message to the console)

def scan_port(port):

#(The line def scan_port(port): is the start of a function definition in Python. This line declares a function named scan_port that takes a single parameter port. In Python, functions are defined using the def keyword.

Here's a simple breakdown of the code:

- def: This keyword is used to declare the start of a function definition.
- scan_port: This is the name of the function.
- (port): This is the parameter list for the function. In this case, the function takes a single parameter named port.)

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

#(The line s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) is a Python code snippet that creates a new socket object using the socket module. This line establishes a TCP socket (SOCK_STREAM) for IPv4 addresses (AF_INET).

- socket.socket(): This is a constructor function in the socket module that creates a new socket object.
- socket.AF_INET: This constant represents the address (domain) family for IPv4.

- socket.SOCK_STREAM: This constant represents the socket type for a TCP socket. SOCK_STREAM indicates a reliable, connection-oriented socket that uses the TCP protocol.
- s = socket.socket(socket.AF_INET, socket.SOCK_STREAM): This line creates a new socket object (s) configured to use IPv4 addresses and the TCP protocol.

This socket object (s) can be used for various network operations, such as connecting to a server, sending and receiving data, and performing other socket-related tasks.)

s.settimeout(2)

#(The line s.settimeout(2) is a Python code snippet that sets a timeout on a socket. In this case, it sets a timeout of 2 seconds for the socket s. This means that if an operation on the socket takes longer than 2 seconds to complete, a socket.timeout exception will be raised.

- s: This is a socket object, presumably created using socket.socket().
- settimeout(): This is a method of the socket object that sets a timeout for blocking operations.
- 2: This is the timeout value in seconds.
- s.settimeout(2): This line sets a timeout of 2 seconds for the socket s.)

conn = s.connect_ex((target, port))

#(The line conn = s.connect_ex((target, port)) is a Python code snippet that attempts to establish a connection to a specified target (IP address or hostname) and port using a socket object (s). The connect_ex method is a non-blocking version of the connect method, and it returns an error code instead of raising an exception if the connection attempt fails.

- s: This is a socket object, presumably created using socket.socket(socket.AF_INET, socket.SOCK_STREAM).
- connect_ex((target, port)): This is a method of the socket object that attempts to establish a connection to the specified target and port. The connect_ex method takes a tuple containing the target address and port as its argument.
- conn: This variable stores the result of the connection attempt. If the connection is successful, conn will be equal to 0. If there's an error, conn will be an error code.)

```
if not conn:

print("Port {} is OPEN".format(port))

s.close()
```

#(The code snippet you provided checks whether the conn variable is zero (indicating a successful connection) and then prints a message indicating that the specified port is open. Additionally, it closes the socket (s) after printing the message.)

```
for port in range(start_port, end_port + 1):

thread = threading.Thread(target=scan_port, args=(port,))

thread.start()
```

#(The code snippet you've provided is a loop that iterates over a range of port numbers (start_port to end_port inclusive) and creates a new thread for each port using the threading module. Each thread is set to execute the scan_port function with the current port as an argument.

- for port in range (start_port, end_port + 1): This is a for loop that iterates over a range of port numbers, starting from start_port and ending at end_port + 1. The loop variable port takes on each value within this range in each iteration.
- thread = threading. Thread (target=scan_port, args=(port,)): Inside the loop, a new thread is created using the Thread class from the threading module. The target parameter specifies the function that the thread will execute (scan_port), and the args parameter provides the arguments to the function (in this case, the current port value).
- thread.start(): This line starts the newly created thread, causing it to execute the scan_port function with the specified port as an argument.)

```
thread.join()
```

#(The thread.join() method is a method in Python's threading module that is used to wait for a thread to complete its execution before moving on to the next part of the program.)

```
end_time = time.time()
```

#(The line end_time = time.time() is a Python code snippet that records the current time using the time module and assigns it to the variable end_time. This is often used to measure the elapsed time between two points in a script.)

print('Time elapsed:', end_time - start_time, 's')

#(The line print('Time elapsed:', end_time - start_time, 's') is a Python code snippet that prints the elapsed time between two points in our script. It calculates the difference between the recorded start_time and end_time using the variables you've assigned.

- 'Time elapsed:': This is a string literal that serves as part of the printed message.
- end_time - start_time: This expression calculates the time difference between end_time and start_time. The result is the elapsed time.
- ,'s': This is another string literal indicating that the unit of the elapsed time is seconds.
- print(...): This function prints the specified values to the console.)

pfSense

pfSense can be installed on most commodity hardware, including old computers and embedded systems. pfSense is typically configured and operated though a user-friendly web interface, making administration easy even for users with limited networking knowledge. Generally, one never needs to use terminal or edit config files to configure the router. Even software updates can be run from the web UI.

pfSense is mostly used as a router and firewall software, and typically configured as DHCP server, DNS server, WiFi access point, VPN server, all running on the same hardware device. pfSense also allows for installation of third-party open-source packages such as Snort or Squid through a built in Package Manager, making it the default choice of many network administrators.

Snort

Snort is an open-source intrusion detection and prevention system (IDS/IPS). It is capable of performing real-time traffic analysis and packet logging on IP networks.

Snort is highly customizable and can be configured to detect various types of malicious activities, such as port scans, denial-of-service attacks, and more. It uses rulesets to identify patterns in network traffic that may indicate a security threat.

pfsense snort

When used together, pfSense and Snort can enhance the security of a network by providing a powerful combination of firewall capabilities and intrusion detection/prevention. Users often deploy Snort as a package within pfSense to add an additional layer of security to their networks. The integration allows administrators to create rules to detect and block potentially malicious traffic based on Snort's analysis.

If we are considering implementing pfSense with Snort, it's recommended to refer to the official documentation for both pfSense and Snort for detailed installation, configuration, and usage instructions..

Firewall Rules in LAN

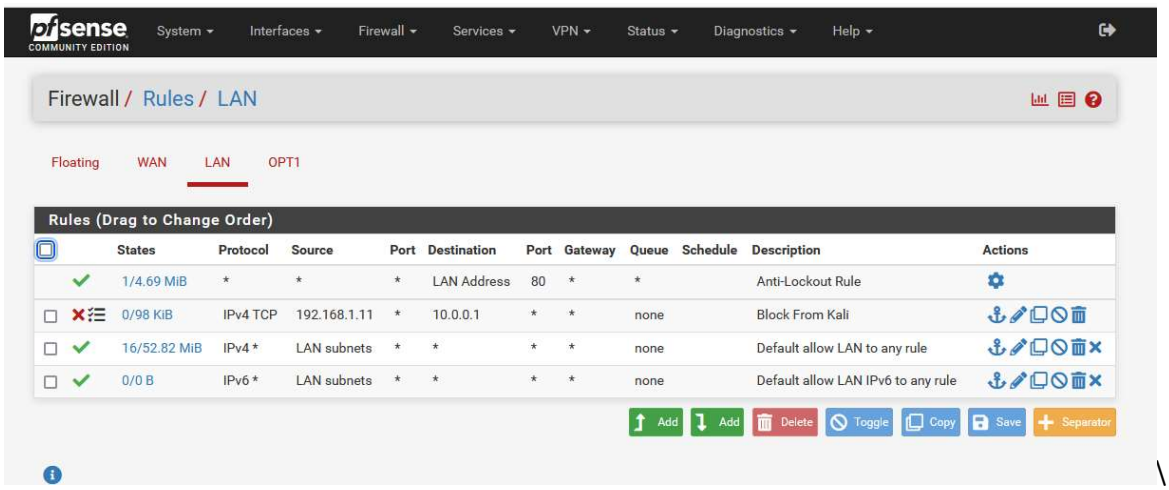


Fig 5.1

Firewall Rules in OPT1

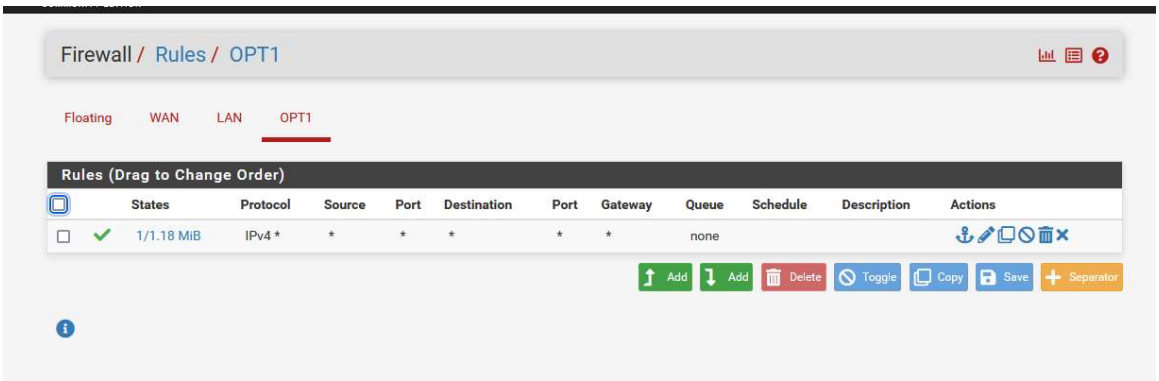


Fig 5.2

SNORT INTERFACES

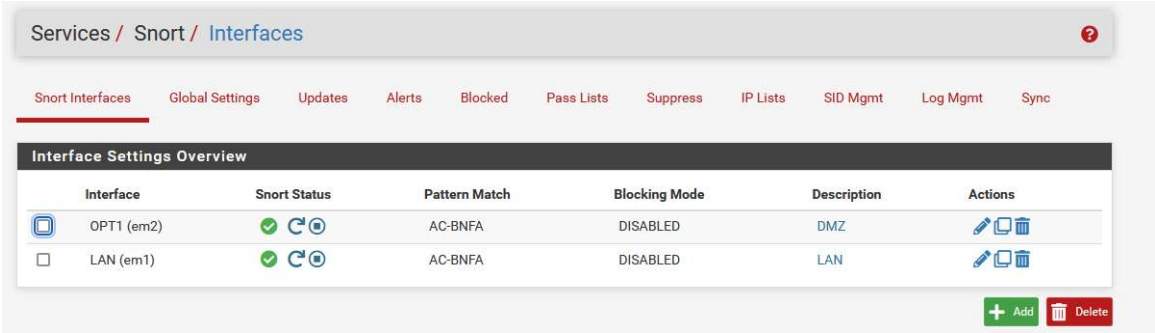


Fig 5.3

SNORT Configuration

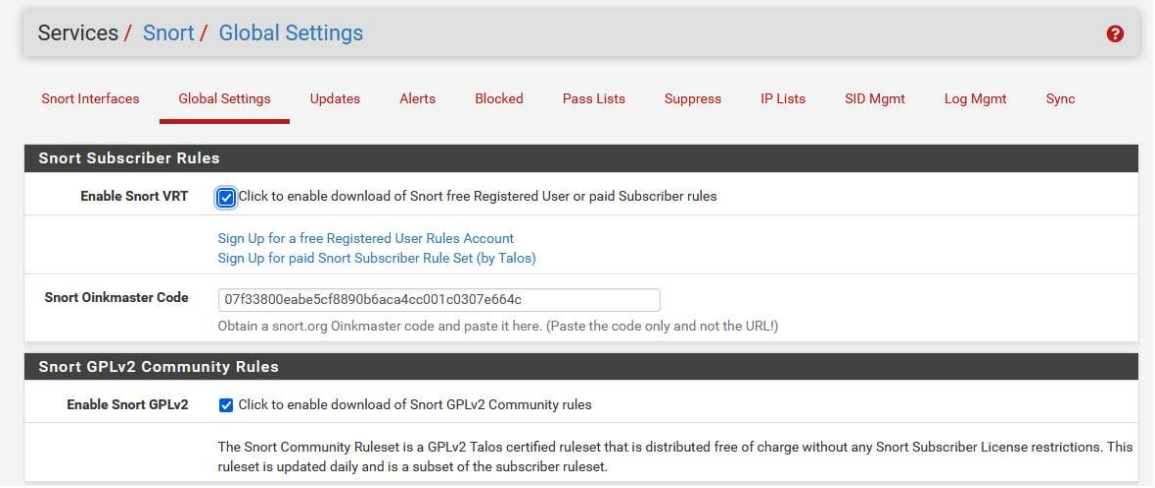


Fig 5.4

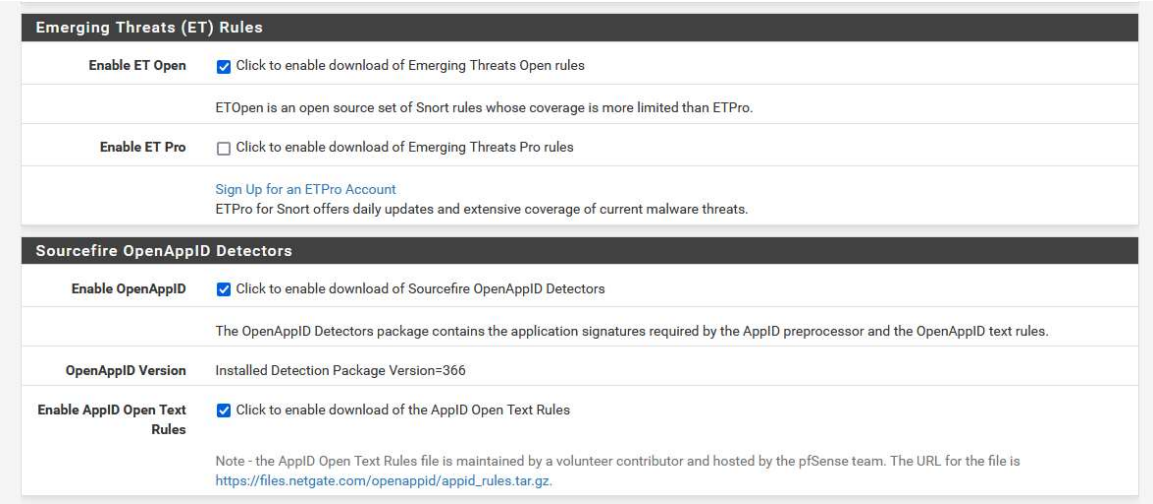


Fig 5.5

SNORT updates

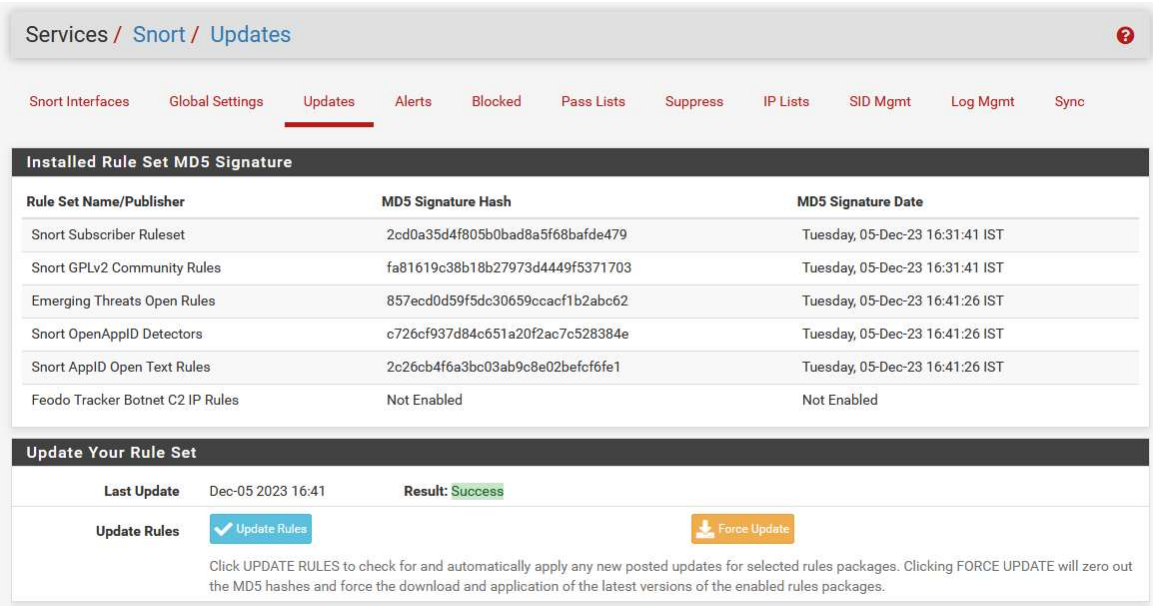


Fig 5.6

SNORT is providing Alerts

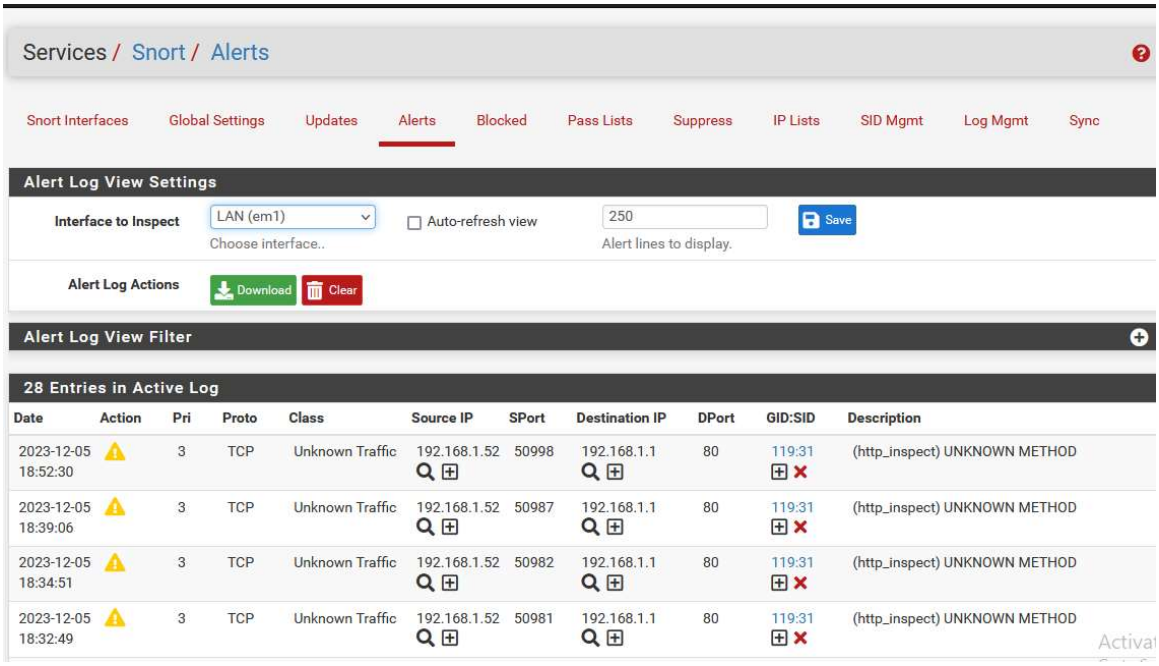


Fig 5.7

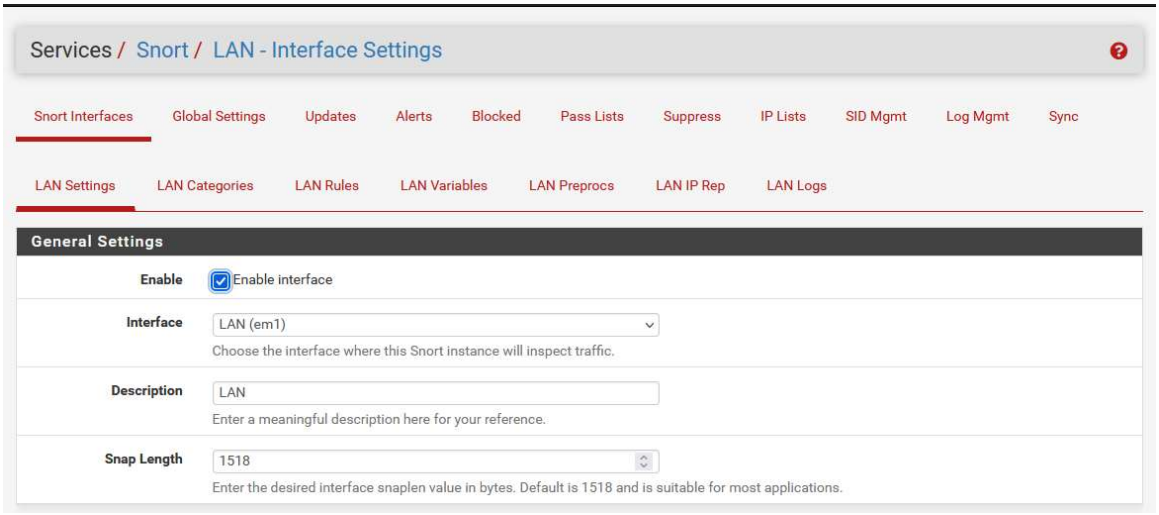


Fig 5.8

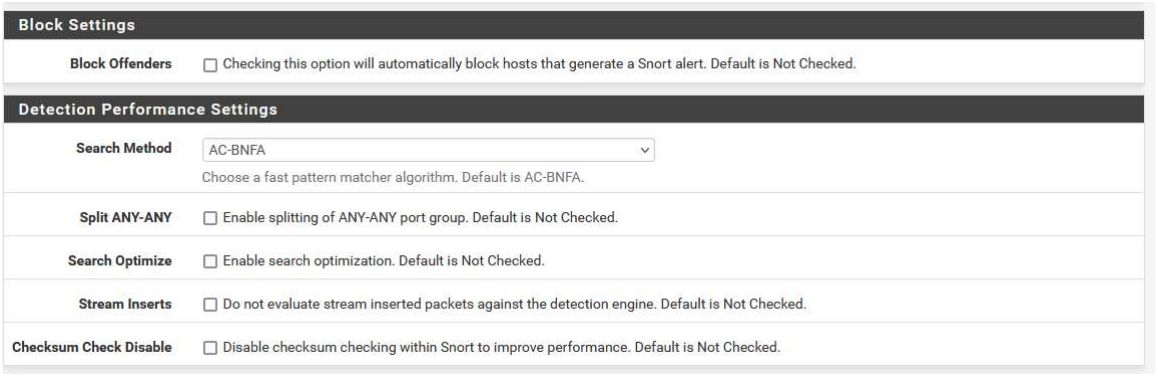


Fig 5.9

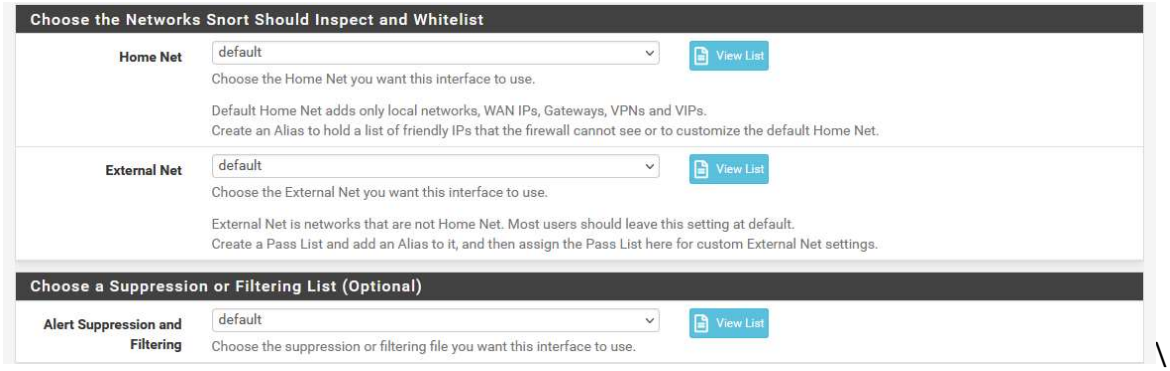


Fig 5.10

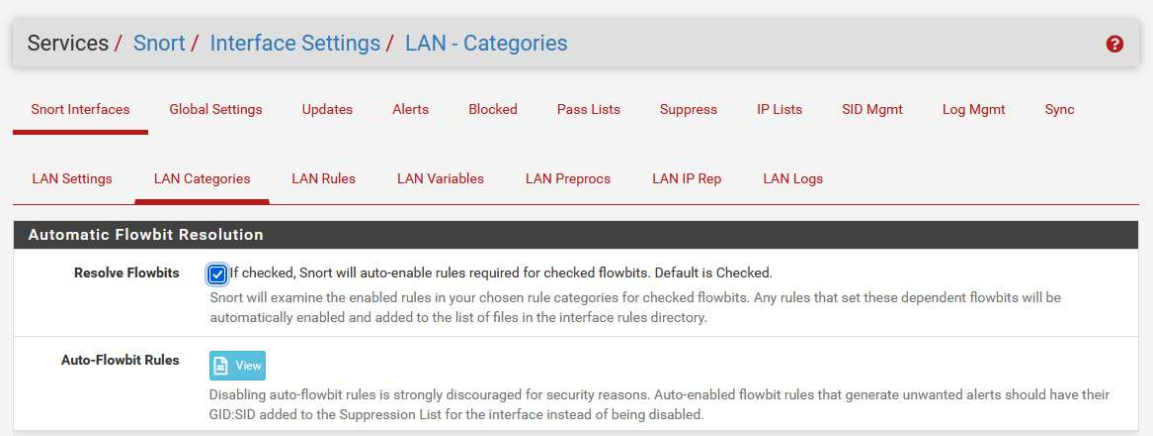


Fig 5.11

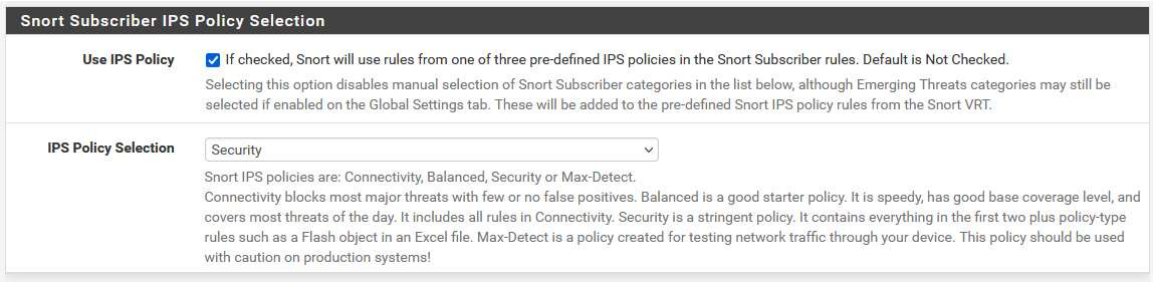


Fig 5.12

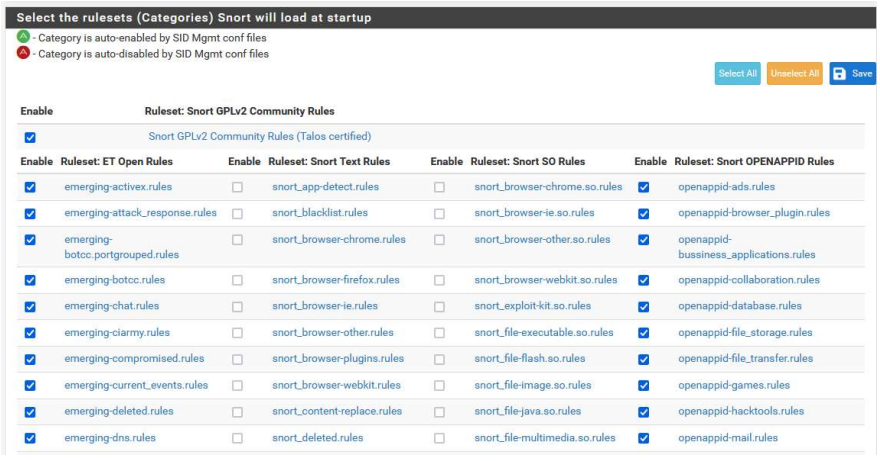


Fig 5.13

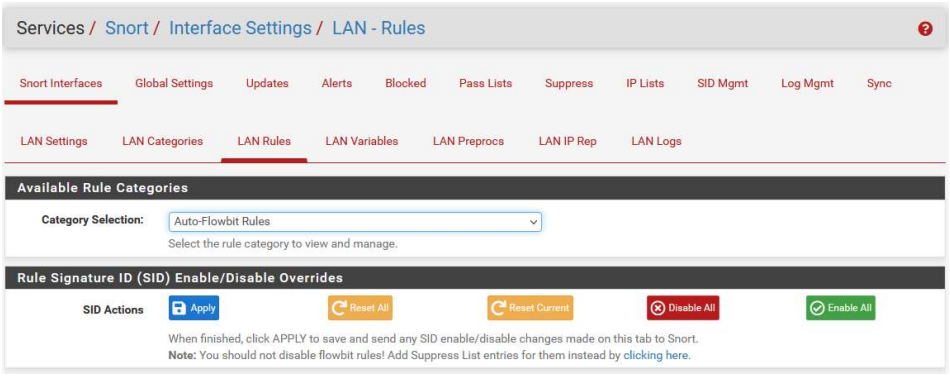
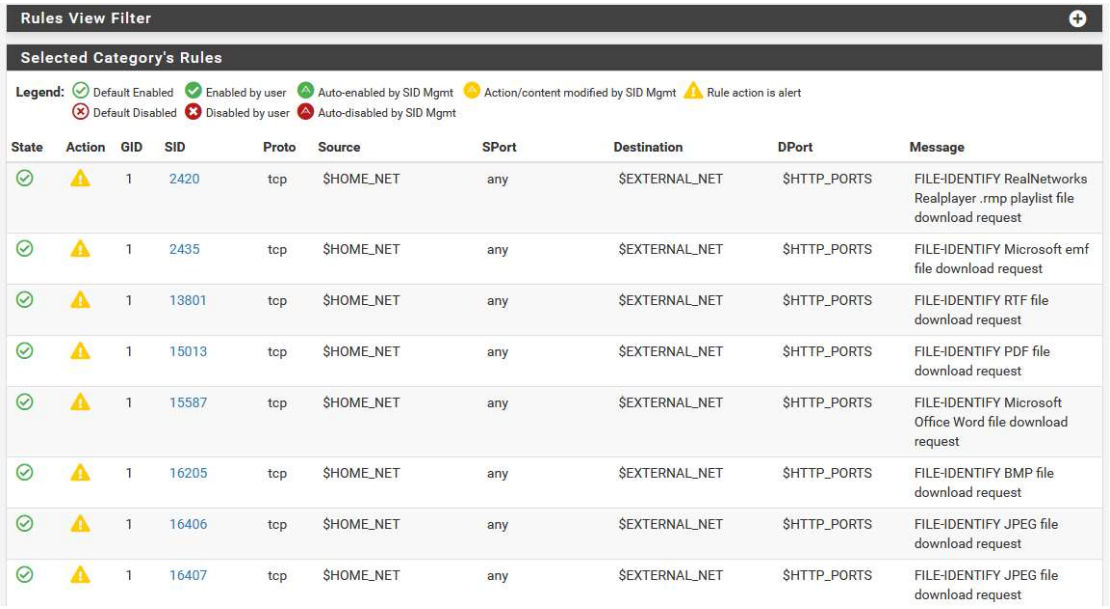


Fig 5.14



State	Action	GID	SID	Proto	Source	SPort	Destination	DPort	Message
		1	2420	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY RealNetworks Realplayer .rmp playlist file download request
		1	2435	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY Microsoft emf file download request
		1	13801	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY RTF file download request
		1	15013	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY PDF file download request
		1	15587	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY Microsoft Office Word file download request
		1	16205	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY BMP file download request
		1	16406	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY JPEG file download request
		1	16407	tcp	$HOME_NET	any	$EXTERNAL_NET	$HTTP_PORTS	FILE-IDENTIFY JPEG file download request

Fig 5.15

CONCLUSION

In the culmination of the SnortShield Network project, we've ushered in a new era of network security. By seamlessly integrating state-of-the-art technologies, we've not only tackled existing challenges but also laid the foundation for a robust and proactive defence paradigm.

The success of SnortShield lies in its unified defence mechanism, where each component contributes significantly to fortifying the network against dynamic cyber threats. Snort's real-time threat detection, combined with the centralized user management of Windows Server ADDS, seamless web hosting via IIS Server, optimized address allocation through DHCP Server, and the comprehensive firewall protection of pfSense, collectively redefine the benchmarks of network security.

More than a shield, SnortShield offers confidence in navigating the complexities of the digital landscape. The seamless integration empowers administrators to proactively detect, prevent, and respond to potential threats, fostering a secure and resilient digital environment.

Looking forward, SnortShield not only addresses current risks but stands ready to adapt to emerging challenges. Its scalability and flexibility ensure a future-proof solution, capable of evolving alongside the ever-changing cybersecurity landscape.

In conclusion, SnortShield is not just a project; it's an embodiment of innovation, collaboration, and an unwavering commitment to building a digital future where networks are fortified, secure, and resilient. As we reflect on this journey, the true measure of success is found in the enhanced security, efficiency, and confidence that SnortShield instils in every digital interaction.

REFERENCE

https://learningnetwork.cisco.com/s/packet-tracer-alternative-lab-solutions

https://networklessons.com/

https://docs.netgate.com/

https://www.w3schools.com

https://hackertarget.com/

https://hackertarget.com/

https://www.microsoft.com/en-in/windows-server