PROJECT REPORT

ON

Honeypot Analysis System

Carried Out at

CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING

ELECTRONIC CITY, BANGALORE.

UNDER THE SUPERVISION OF

Mr. Dattaraj Sadguru Shetye
Project Engineer

C-DAC Bangalore

Submitted By

Kushare Vishal Nivrutti (240850123012)
Makarand Shrirang Chandanshive (240850123013)
Sanjay Kumar Das (240850123021)
Ved Prakash Verma (240850123033)
Vipul Kumar Jaltare (240850123036)

PG DIPLOMA IN ADVANCED COMPUTING/PG DIPLOMA IN EMBEDDED SYSTEMS AND DESIGN

C-DAC, BANGALORE

Candidate's Declaration

We hereby certify that the work being presented in the report entitled Honeypot Analysis System, in the partial fulfillment of the requirements for the award of Post-Graduate Diploma and submitted in the department of **IT Infrastrure System And Security** of the C-DAC Bangalore, is an authentic record of our work carried out during the period 1st January 2025– 10th February 2025 under the supervision of Mr. Dattaraj Sadguru Shetye (Project Engineer), C-DAC Bangalore.

The matter presented in the report has not been submitted by us for the award of any degree of this or any other Institute/University.

Name- Vishal Nivrutti Kushare

Signature

Name- Makarand Shrirang Chandanshive

Signature-

Name- Sanjay Kumar Das

Signature-

Name- Ved Prakash Verma

Signature-

Name- Vipul Kumar Jaltare

Signature-

Counter Signed by

Mr. Dattaraj Sadguru Shetye

CERTIFICATE

This is to certify that project report entitled **"Honeypot Analysis System"** which is submitted by **Kushare Vishal Nivrutti, Makarand Shrirang Chandanshive, Sanjay Kumar Das, Ved Prakash Verma, Vipul Kumar Jaltare** of Centre for Development of Advanced Computing-Bengaluru, is a record of the candidates own work carried out by them under our supervision. The matter embodied in the thesis is original and has not been submitted for the award of any other Degree.

Date:

Mr. Dattaraj Sadguru Shetye

Signature:

ACKNOWLEDGEMENT

We take this opportunity to express my gratitude to all those people who have been directly and indirectly with me during the competition of this project

We pay thank to **Mr. Dattaraj Sadguru Shetye** who has given guidance and a light to me during this major project. His versatile knowledge about "**Honeypot Analysis System**" has eased me in the critical times during the span of this Final Project.

We acknowledge here out debt to those who contributed significantly to one or more steps. We take full responsibility for any remaining sins of omission and commission.

Student Name:

Kushare Vishal Nivrutti

Makarand Shrirang Chandanshive

Sanjay Kumar Das

Ved Prakash Verma

Vipul Kumar Jaltare

ABSTRACT

Honeypot analysis is a critical cybersecurity technique used to detect, deflect, and analyze unauthorized access attempts on a network. By deploying a decoy system designed to lure attackers, honeypots provide valuable insights into cyber threats, attack methodologies, and malicious actors' behaviors.

This project focuses on the design and implementation of a honeypot system, tailored to capture and log unauthorized access attempts. It provides an in-depth study of cyber-attack strategies, enabling security teams to enhance network defenses proactively. The system incorporates advanced architecture, efficient data management, API integration, and robust security mechanisms to ensure its effectiveness. Additionally, the project explores various honeypot types, logging techniques, and real-world applications, offering a comprehensive framework for organizations to bolster their cybersecurity posture.

Through detailed analysis and continuous monitoring, this honeypot system aims to contribute significantly to threat intelligence, incident response, and the development of more resilient security protocols.

TABLE OF CONTENTS

ABBREVIATIONS AND ACRONYMS

1. SSH -Secure Shell
2. HTTP - Hypertext Transfer Protocol
3. SMB - Server Message Block
4. FTP - File Transfer Protocol
5. SNORT - Network Intrusion Detection and Prevention System
6. IDS - Intrusion Detection System
7. IPS - Intrusion Prevention System
8. IP - Internet Protocol,
9. NMAP - Network Mapper
10. OS - Oprating System
11. HIDS - Host-Based Intrusion Detection System
12. HIPS - Host-Based Intrusion Prevention System
13. TCP - Transmission Control Protocol
14. HTTPS - Hypertext Transfer Protocol Secure
15. API - Application Programming Interface
16. DDoS - Distributed Denial of Service
17. SFTP - Secure File Transfer Protocol
18. XSS - Cross-Site Scripting
19. UFW - Uncomplicated Firewall
20. NetBIOS - Network Basic Input/Output System

Chapter 1

INTRODUCTION

1.1 Overview

Cybersecurity threats have evolved in complexity, making traditional security measures inadequate for mitigating advanced persistent threats (APTs) and zero-day attacks. Honeypots serve as strategic countermeasures by acting as decoy systems designed to attract, monitor, and analyze malicious activities. By luring attackers into an environment where their actions can be logged and studied, honeypots provide critical intelligence for strengthening overall network security. This project presents a comprehensive approach to implementing a honeypot system to track intrusion attempts, understand attack vectors, and develop better defense mechanisms.

1.2 Objectives

- Design and implement a honeypot system
- Analyze attack patterns and behaviors
- Enhance security measures based on insights gained
- Improve network defense mechanisms
- Provide real-time logging and alerting features
- Develop an API for retrieving and visualizing attack data

Scope of the project

This project focuses on:

The honeypot system will be designed to mimic real network services while logging all unauthorized access attempts. This project will explore different types of honeypots, including low-interaction, high-interaction, and hybrid solutions. The scope includes analyzing attack data, studying attacker behaviors, and leveraging insights to refine network security policies. Additionally, the system will integrate with Security Information and Event Management (SIEM) solutions to enhance threat visibility and incident response capabilities.

Chapter 2

LITERATURE SURVEY

Honeypots are deceptive security mechanisms designed to attract and monitor cyber attackers. By simulating vulnerable systems, they allow cybersecurity researchers to analyze attack techniques, gather intelligence, and improve security defenses. This literature survey explores previous studies, methodologies, and advancements in honeypot-based threat detection and analysis.

Evolution of Honeypots

The concept of honeypots was first introduced in the 1990s to monitor and study hacker behavior. Early research focused on low-interaction honeypots that emulated services without providing full system access. Over time, more advanced high-interaction honeypots emerged, allowing deeper attacker engagement and analysis.

Key Studies:

- Cheswick (1992): Early research on deception-based security models.
- Lance Spitzner (1999): Developed the Honeynet Project, which improved attacker tracking techniques.
- Yuill et al. (2004): Explored honeypots for intrusion prevention and malware analysis.

Honeypots in Network Security:

Honeypots are widely used to detect network intrusions, ransomware, and brute-force attacks.

- Network-Based Honeypots: Deploy services such as SSH, SMB, HTTP, and FTP to capture attacker actions.
- IDS/IPS Integration: Honeypots work alongside Intrusion Detection Systems (Snort, Suricata) to improve detection rates.

Key Studies:

- Provos (2004): Introduced Honeyd for network monitoring.
- Zhang et al. (2015): Studied SSH honeypots for brute-force attack analysis.

Chapter 3

SOFTWARE REQUIREMENT SPECIFICATIONS

HARDWARE REQUIREMENTS:

- Kali Linux OS 4 Gb Ram 20 Gb Storage-

Requirement	**Minimum Specs**	**Recommended Specs**
CPU	1 GHz (x86/x64)	Multi-core processor
RAM	2 GB (minimum)	4 GB
Storage	20GB (minimum)	30 GB+ (for extra tools)

- Ubuntu OS 6 Gb Ram 40 Gb Storage-

Requirement	**Minimum Specs**	**Recommended Specs**
CPU	2 GHz (dual-core)	4+ core processor
RAM	4GB (minimum)	6 GB
Storage	25GB (minimum)	40GB

SOFTWARE REQUIREMENTS:

- **IDS/IPS Firewall-**

  In a **Honeypot Analysis Project**, **IDS/IPS and firewalls** play a crucial role in monitoring, detecting, and preventing cyber threats. While honeypots are designed to attract attackers, IDS/IPS systems analyze the attacks, and firewalls help isolate and protect the network from unauthorized access.

Network Design-

The typical honeypot security architecture consists of:

- Honeypot Server – Deployed in a controlled environment to capture attacks.
- Firewall – Restricts incoming and outgoing traffic to isolate the honeypot.
- IDS/IPS System – Monitors and detects threats targeting the honeypot.
- Logging & Analysis Module – Stores and processes attack logs.
- Visualization Dashboard – Provides real-time attack insights.

Traffic Flow-

1. Attacker scans network and finds the honeypot.
2. Firewall allows selective traffic to reach the honeypot.
3. Honeypot records attacker activity and forwards logs to IDS.
4. IDS detects attack patterns and alerts security teams.
5. IPS blocks malicious activity if configured in prevention mode.

- **NMAP-**

  Nmap (Network Mapper) is a powerful tool used for network scanning and security auditing. In a Honeypot Analysis Project, Nmap is crucial for:

- Detecting Honeypots: Identifying if a system is a honeypot using fingerprinting techniques.
- Testing Honeypot Effectiveness: Simulating attacks to evaluate how the honeypot responds.
- Analyzing Attacker Behavior: Capturing and studying how attackers use Nmap on the honeypot.

Services-

- **SMB-**

SMB (Server Message Block) is a network protocol primarily used for file sharing, printer sharing, and remote access in Windows-based networks. It allows users and applications to access files and services over a network.

- Common Ports:
    - TCP 445 (modern SMB versions)
    - TCP 139 (older NetBIOS-based SMB)
- SMB Versions:
    - SMBv1 (Insecure, vulnerable to exploits like EternalBlue)
    - SMBv2 & SMBv3 (More secure, used in modern Windows systems)

- **HTTP-**

HTTP (Hypertext Transfer Protocol) is the foundation of web communication, enabling browsers and servers to exchange data. It operates on port 80 (HTTP) and port 443 (HTTPS).

- Common HTTP Services:
    - Web pages (HTML, CSS, JavaScript)
    - API requests and responses
    - File downloads and uploads

Use HTTP in a Honeypot Analysis Project?

- Since web servers are highly targeted, deploying an HTTP honeypot helps in:
- Identifying web-based attacks (SQL Injection, XSS,)
- Capturing hacker behavior (IP addresses, attack patterns)
- Testing web security defenses (IDS/IPS)

- **Apache2-**

Apache2 is one of the most widely used open-source web servers, primarily used to host websites and web applications. It runs on port 80 (HTTP) and port 443 (HTTPS) and supports various modules to enhance functionality, such as PHP, Python, and CGI scripts.

- Common Uses:
    - Hosting web pages and applications
    - Handling HTTP requests from users
    - Running backend scripts (PHP, CGI, etc.)

Use Apache2 in a Honeypot Analysis Project?

- Since real web servers are frequent targets for cyberattacks, setting up an Apache2 honeypot helps to:
  Capture web-based attack attempts (SQLi, XSS, brute force, DDoS)
- Analyze hacker behavior (IP tracking, attack patterns)

- **FTP-**

FTP (File Transfer Protocol) is a standard network protocol used to transfer files between a client and a server over a TCP-based network. It operates on:

- Port 21 (Command Channel) – Handles authentication and commands
- Port 20 (Data Channel) – Transfers files between client and server

FTP can work in two modes:

- Active Mode – The client opens a port, and the server connects back
- Passive Mode – The server opens a port, and the client connects to it

Using an FTP Honeypot Helps to:

- Capture unauthorized login attempts
- Identify brute force and dictionary attacks
- Detect attempted file exfiltration (data theft)
- Log hacker techniques and behavior

SSH-

SSH (Secure Shell) is a cryptographic protocol that allows secure remote access to servers and devices over a network. It is commonly used by system administrators for:

- Secure remote login
- File transfers (via SCP or SFTP)
- Executing remote commands

SSH Ports

- Port 22 (Default SSH Port) – Used for secure remote access
- Can be configured to use non-standard ports for security

Deploying an SSH Honeypot Helps to:

- Capture brute-force attempts and malicious login attempts
- Identify attacker IPs, usernames, and password patterns
- Detect automated bots and hacker tools scanning for SSH vulnerabilities
- Analyze hacker behavior and attack techniques

By setting up a fake (honeypot) SSH server, we can trick attackers into revealing their tactics without compromising real systems.

Chapter 4

SYSTEM ARCHITECTURE

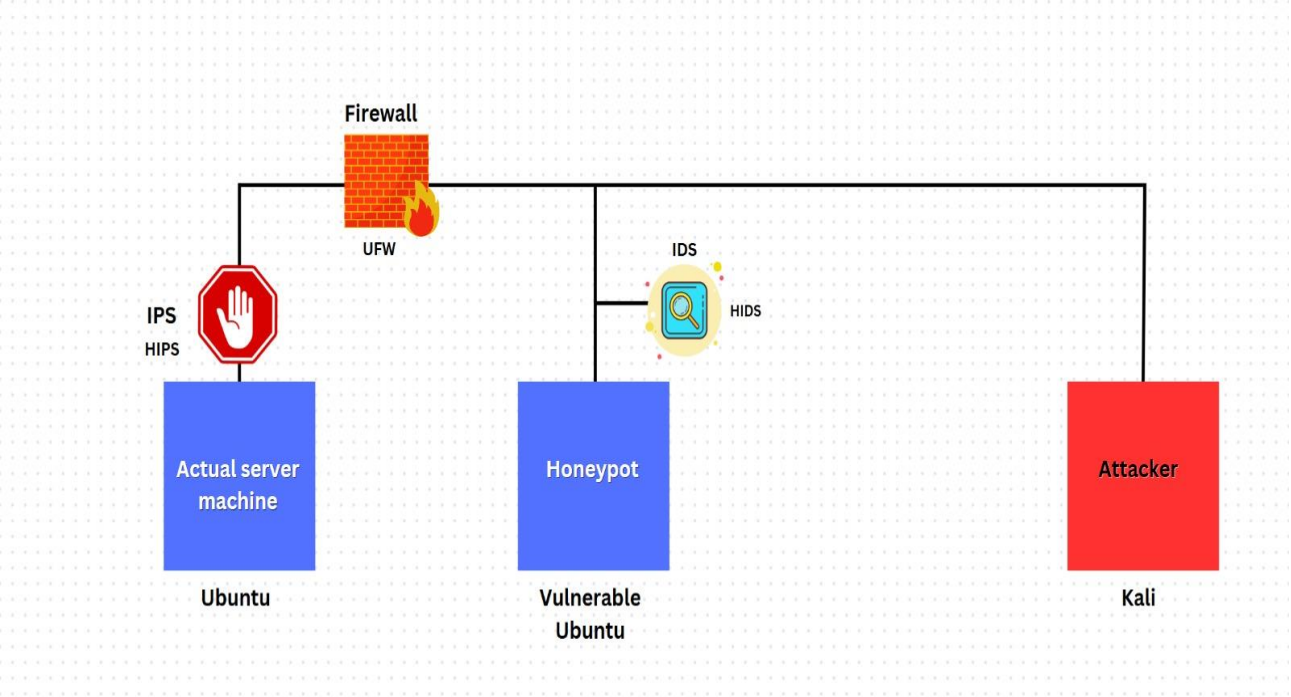


Fig 1

1. Components of the Diagram

◈ Attacker (Kali Linux)

- The red box labeled "Attacker" represents a potential cyber threat.
- Uses Kali Linux, a penetration testing OS that includes tools like Nmap, Metasploit, Hydra, etc.
- The attacker attempts to scan, exploit, and access services running on the honeypot.

◈ Honeypot (Vulnerable Ubuntu)

- The blue box labeled "Honeypot" acts as a decoy system.
- It runs intentionally vulnerable services like SSH, SMB, FTP, HTTP (Apache2) to attract attackers.
- Captures attacker behavior and logs activities for analysis.

◈ IDS (Intrusion Detection System)

- IDS monitors the honeypot's activity to detect attacks.
- Host-based IDS (HIDS) is deployed on the honeypot to log malicious activities.
- Examples: Snort, OSSEC, Suricata.

◈ Firewall (UFW - Uncomplicated Firewall)

- UFW (Uncomplicated Firewall) controls network traffic.
- Ensures that only specific traffic reaches the honeypot while protecting the real server.

◈ IPS/HIPS (Intrusion Prevention System / Host-based IPS)

- IPS (Network-level) and HIPS (Host-based) protect the actual server machine.
- Automatically blocks detected malicious traffic to prevent unauthorized access.

◈ Actual Server Machine (Ubuntu)

- A protected server that hosts critical data and services.
- IPS/HIPS & Firewall (UFW) protect it from unauthorized access.

2. How the System Works-

1.Attacker Scans the Network

- The attacker uses Nmap or other tools to discover open ports and services on the network.

2.Attacker Targets the Honeypot

- Since the honeypot runs vulnerable services, the attacker tries to exploit them (e.g., brute-force SSH, exploit Apache2, or access SMB shares).

3.Honeypot Captures the Attack

- The honeypot logs all attacker activities.
- Example: If an attacker uses an SSH brute-force attack, the honeypot records the IP, usernames, and passwords used.

4.IDS Monitors and Reports Attack Activity

- The IDS detects suspicious behavior and logs it for further analysis.

5.IPS Protect the Actual Server

- If an attacker tries to reach the real server, the firewall (UFW) and IPS/HIPS block the traffic.

6.Security Analysts Analyze the Attack

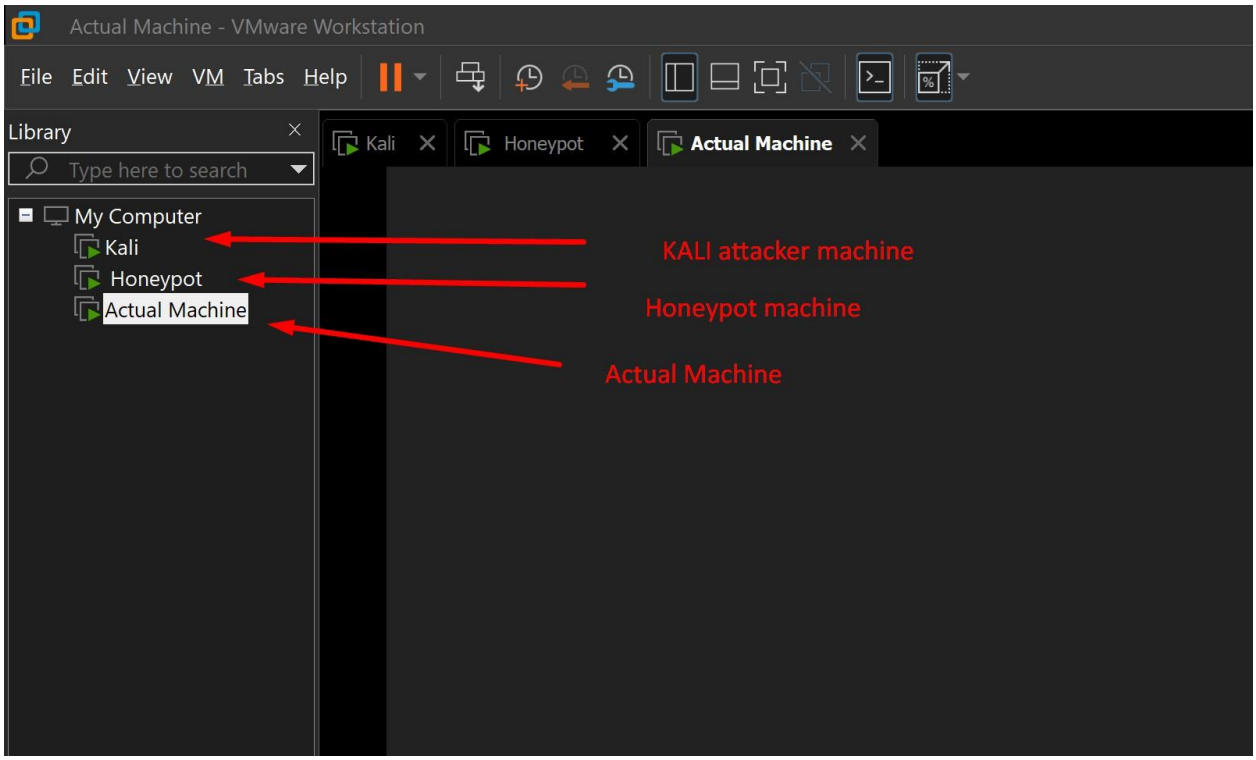
- Logs from the honeypot and IDS are analyzed using tools like Wireshark, ELK Stack, and Splunk.
- The security team improves defenses based on attack patterns

IMPLEMENTATION

For this Project we need 3 VMs - Kali as attacker machine

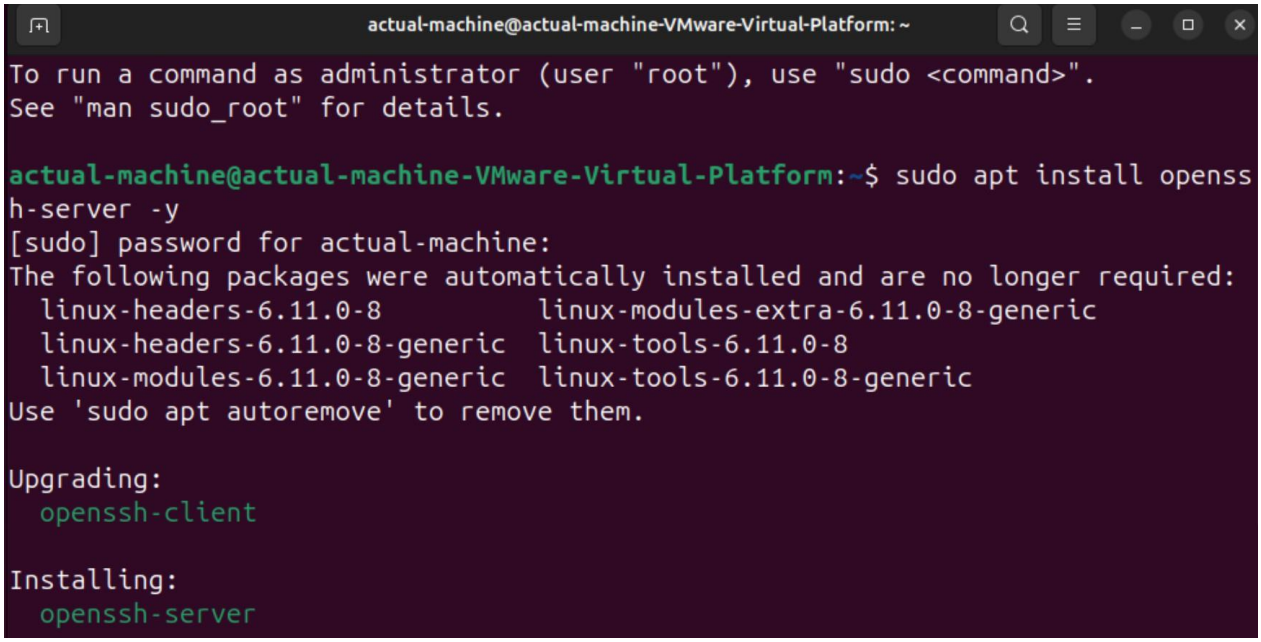
2 Ubuntu machines - 1 as a Honeypot

1 as an Actual server machine

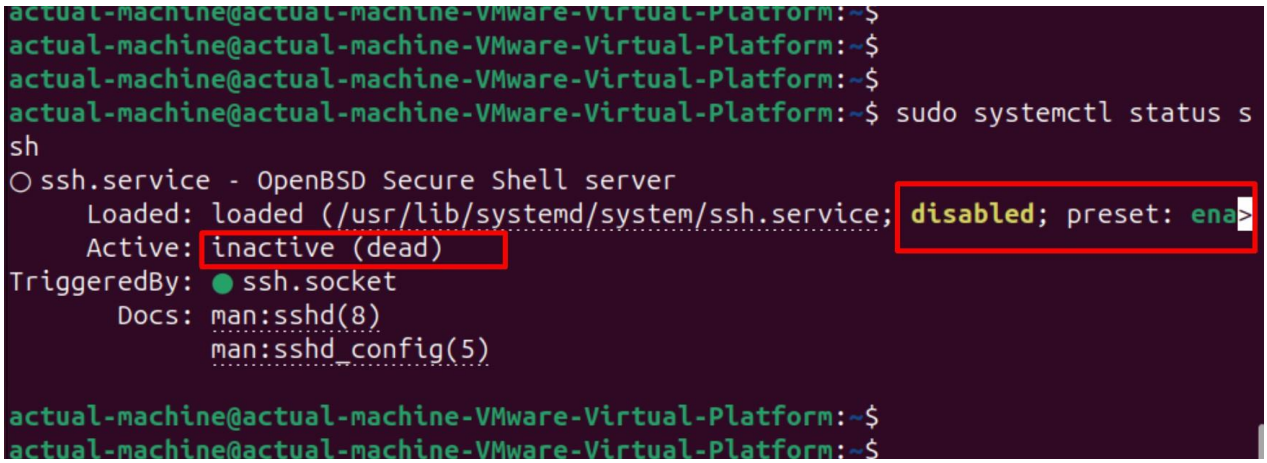


First we need to **Install OpenSSH in both the machines - Honeypot and Actual machine**

sudo apt install openssh-server -y



Now we have to check the status of the ssh - **sudo systemctl status ssh**



Here we need to start the ssh service - **sudo systemctl start ssh**

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo systemctl start ss
h
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

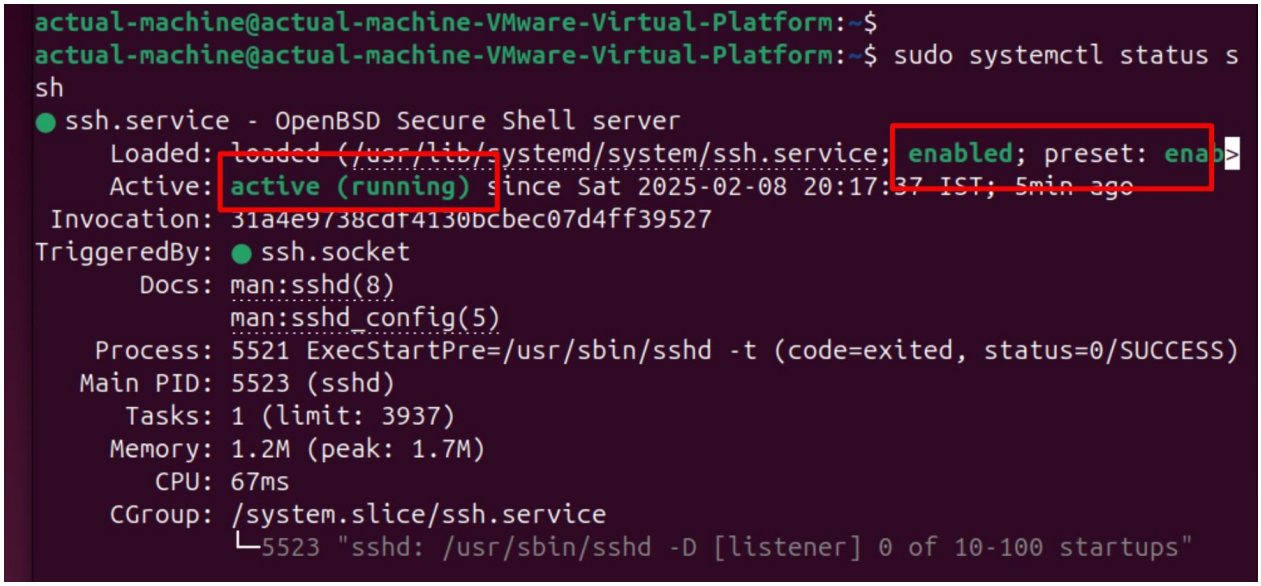
Now we need to enable the ssh service using command - **sudo systemctl enable ssh**

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo systemctl enable s
sh
Synchronizing state of ssh.service with SysV service script with /usr/lib/system
d/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
Created symlink '/etc/systemd/system/sshd.service' → '/usr/lib/systemd/system/ss
h.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/ssh.service' → '/us
r/lib/systemd/system/ssh.service'.
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

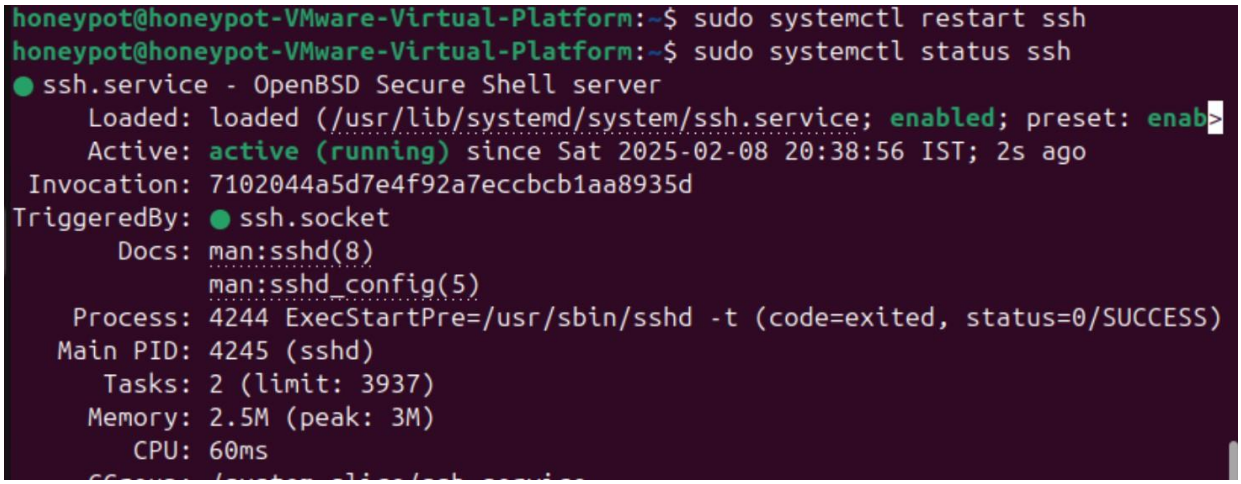
Now we will have to restart the service to refresh service's state - **sudo systemctl restart ssh**

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo systemctl restart
ssh
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Now again we have to check the status of the ssh - **sudo systemctl status ssh**



We can see the status is **active** and **running.** We will perform the same operations in our



Now we need to start the **HTTP** service on both the machines **Honeypot** as well as Actual server machine

For running HTTP service first we need to install HTTPweb application service For that we need to run the command - **sudo apt install apache2 -y**

```
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo apt install apache2 -y
The following packages were automatically installed and are no longer required:
  linux-headers-6.11.0-8          linux-modules-extra-6.11.0-8-generic
  linux-headers-6.11.0-8-generic  linux-tools-6.11.0-8
  linux-modules-6.11.0-8-generic  linux-tools-6.11.0-8-generic
Use 'sudo apt autoremove' to remove them.

Installing:
  apache2
```

Now we need to check the status of the apache2 service - **sudo systemctl status apache2**

```
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: >
     Active: active (running) since Sat 2025-02-08 20:46:02 IST; 1min 28s ago
 Invocation: 93c6c6e68b844066a81312d57bd609fb
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 5026 (apache2)
      Tasks: 55 (limit: 3937)
     Memory: 5.6M (peak: 6.1M)
        CPU: 189ms
     CGroup: /system.slice/apache2.service
             ├─5026 /usr/sbin/apache2 -k start
             ├─5028 /usr/sbin/apache2 -k start
             └─5030 /usr/sbin/apache2 -k start
```

Here we see the status is **active** and **running**

Now we will try to install SMB service in both the VMs
First we need to use this command - **sudo apt install -y samba** to install samba server in our Ubuntu VM

```
actual-machine@actual-machine-VMware-Virtual-Platform: ~
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo apt install -y sam
ba
[sudo] password for actual-machine:
The following packages were automatically installed and are no longer required:
  linux-headers-6.11.0-8          linux-modules-extra-6.11.0-8-generic
  linux-headers-6.11.0-8-generic  linux-tools-6.11.0-8
  linux-modules-6.11.0-8-generic  linux-tools-6.11.0-8-generic
Use 'sudo apt autoremove' to remove them.

Installing:
  samba
```

Creating secure SMB share

sudo mkdir -p /srv/samba/secure sudo
chmod 770 /srv/samba/secure
sudo chown nobody:nogroup /srv/samba/secure

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo mkdir -p /srv/samb
a/secure
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo chmod 770 /srv/sam
ba/secure
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo chown nobody:nogro
up /srv/samba/secure
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Configuring secure SMB in Actual Ubuntu server
sudo vim /etc/samba/smb.conf

```
Processing triggers for libc-bin (2.40-1ubuntu3.1) ...
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo vim /etc/samba/smb
.conf
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Now we need to add few configurations at the bottom

[SecureShare]
path = /srv/samba/secure browsable =
yes
read only = no
valid users = @smbusers force
user = nobody create mask =

```
[SecureShare]
path = /srv/samba/secure
browsable = yes
read only = no
valid users = @smbusers
force user = nobody
create mask = 0640
directory mask = 0750
guest ok = no
```

0640 directory mask = 0750
guest ok = no

Now we need to create a SMB user
sudo groupadd smbusers
sudo useradd -M -s /sbin/nologin smbuser sudo
usermod -aG smbusers smbuser sudo smbpasswd -
a smbuser

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo groupadd smbusers
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo useradd -m -s /sbi
n/nologin smbuser
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo usermod -aG sbmuse
rs smbuser
usermod: group 'sbmusers' does not exist
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo usermod -aG smbuse
rs smbuser
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sduo smbpasswd -a smbus
er
Command 'sduo' not found, did you mean:
  command 'sudo' from deb sudo (1.9.15p5-3ubuntu5)
  command 'sudo' from deb sudo-ldap (1.9.15p5-3ubuntu5)
Try: sudo apt install <deb name>
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo smbpasswd -a smbus
er
New SMB password:
Retype new SMB password:
Added user smbuser.
```

Harden SMB configuration - **/etc/samba/smb.conf** disable **SMBv1** and do the following changes
server min protocol = SMB2
server max protocol = SMB3 map
to guest = never

```
32     server string = %h server (Samba, Ubuntu)
33
34  server min protocol = SMB2
35  server max protocol = SMB3
36  map to guest = never
37
38 #### Networking ####
```

Then save and exit after changing the configuration

Then restart the SMB service - **sudo systemctl restart smbd**

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo systemctl restart
smbd
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Now we need to secure this with the firewall -
sudo ufw allow 139,445/tcp sudo
systemctl restart ufw

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw allow 139,445/
tcp
Skipping adding existing rule
Skipping adding existing rule (v6)
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo systemctl restart
ufw
```

Now let's do the SMB configuration in the **Honeypot machine** First

we need to install samba - **sudo apt install -y samba** Then creating

the vulnerable SMB share -
sudo mkdir -p /srv/samba/public sudo
chmod 777 /srv/samba/public

```
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo mkdir -p /srv/samba/public
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo chmod 777 /srv/samba/public
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Now we need to configure the SMB for vulnerability - **sudo vim /etc/samba/smb.conf**
Then we need to add this configuration at the bottom :

[Public]
path = /srv/samba/public browsable =
yes
read only = no guest ok
= yes force user =
nobody create mask =
0777
directory mask = 0777

```
[Public]
  path = /srv/samba/public
  browsable = yes
  read only = no
  guest ok = yes
  force user = nobody
  create mask = 0777
  directory mask = 0777

"/etc/samba/smb.conf" 252L, 9074B
```

Enable SMBv1 deliberately insecure - add at the end of the Global settings
server min protocol = NT1 guest
ok = yes

```
33
34 server min protocol = NT1
35 guest ok = yes
36
37 #### Networking ####
```

Now restart SMB - **sudo systemctl restart smbd**

```
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo systemctl restart smbd
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Configure firewall rule to expose the Honeypot and allow the traffic
sudo ufw allow 139,445/tcp sudo
systemctl restart ufw

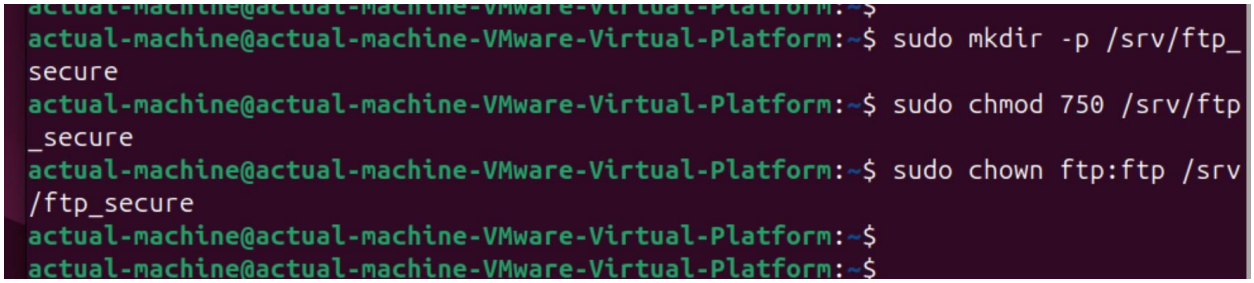
```
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo ufw allow 139,445/tcp
Rules updated
Rules updated (v6)
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo systemctl restart ufw
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Now we will again configure another service FTP, now need to install **vsftpd** service First we will configure the actual server machine
sudo apt install -y vsftpd

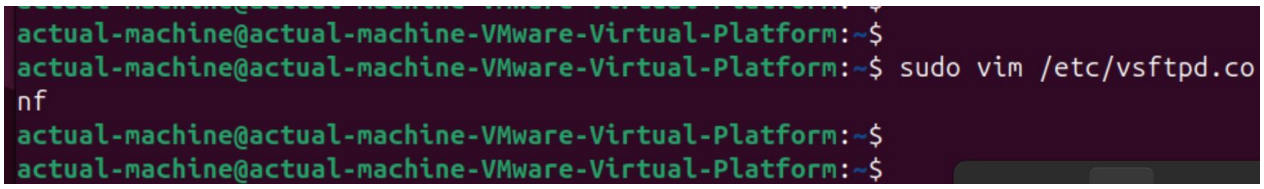
```
actual-machine@actual-machine-VMware-Virtual-Platform: ~
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo apt install -y vsf
tpd
[sudo] password for actual-machine:
The following packages were automatically installed and are no longer required:
  linux-headers-6.11.0-8          linux-modules-extra-6.11.0-8-generic
  linux-headers-6.11.0-8-generic  linux-tools-6.11.0-8
  linux-modules-6.11.0-8-generic  linux-tools-6.11.0-8-generic
Use 'sudo apt autoremove' to remove them.

Installing:
  vsftpd
```

Create a Secure FTP Directory - **sudo**
mkdir -p /srv/ftp_secure sudo chmod

750 /srv/ftp_secure sudo chown ftp:ftp /srv/ftp_secure

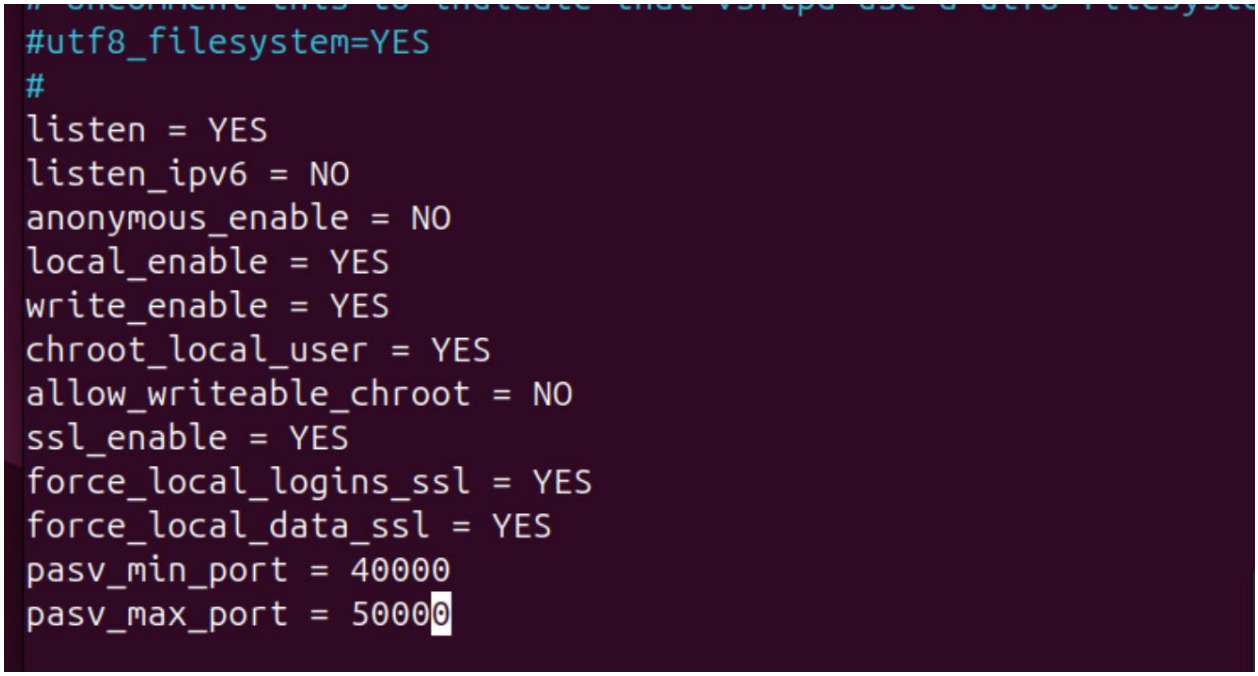


Configure Secure FTP - **sudo vim /etc/vsftpd.conf**



Then add these lines at the end
listen=YES listen_ipv6=NO
anonymous_enable=NO
local_enable=YES
write_enable=YES
chroot_local_user=YES
allow_writeable_chroot=NO
ssl_enable=YES

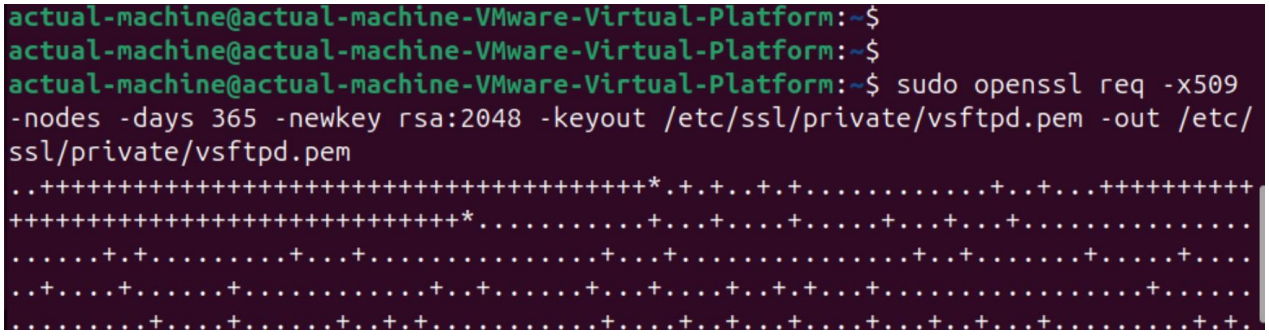
force_local_logins_ssl=YES
force_local_data_ssl=YES
pasv_min_port=40000
pasv_max_port=50000



Secure FTP with User and TLS

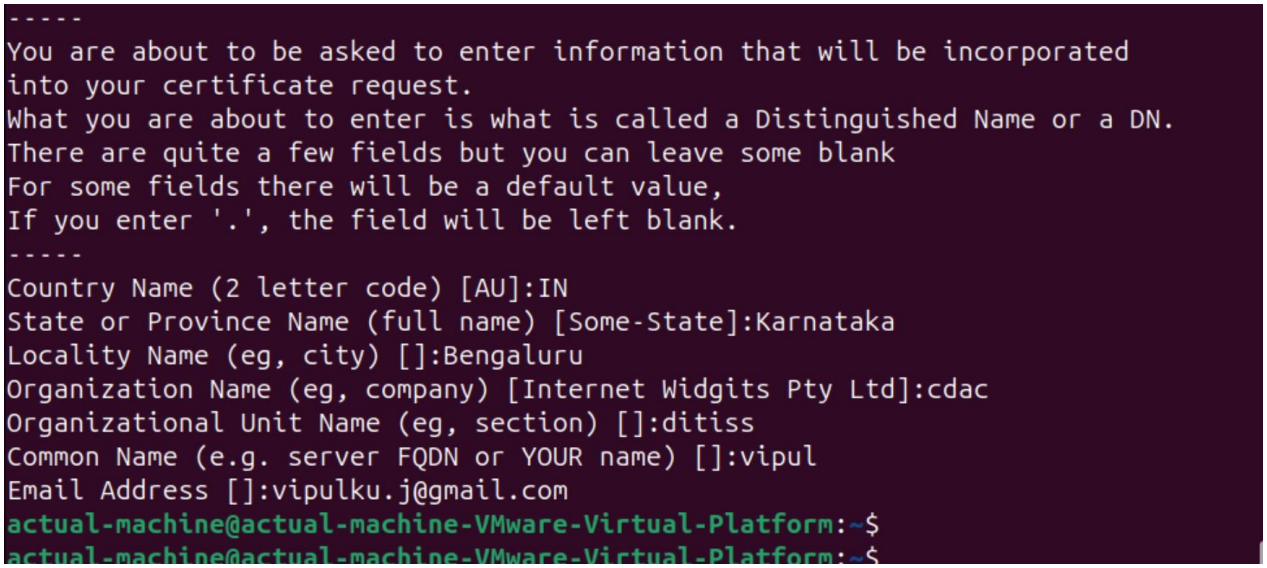
Create an FTP user:

sudo useradd -m -d /srv/ftp_secure -s /sbin/nologin ftpuser sudo
passwd ftpuser
sudo chown ftpuser:ftpuser /srv/ftp_secure



Now we need to generate certificate -

sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/vsftpd.pem -out /etc/ssl/private/vsftpd.pem



Edit /etc/vsftpd.conf and add: **rsa_cert_file=/etc/ssl/private/vsftpd.pem rsa_private_key_file=/etc/ssl/private/vsftpd.pem**

Now restart vsftpd service - **sudo systemctl restart vsftpd**

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo systemctl restart
vsftpd
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Allow FTP Through Firewall
sudo ufw allow 21/tcp
sudo ufw allow 40000:50000/tcp sudo
systemctl restart ufw

Honeypot Machine (Vulnerable FTP Server)

Install vsftpd - **sudo apt install -y vsftpd**

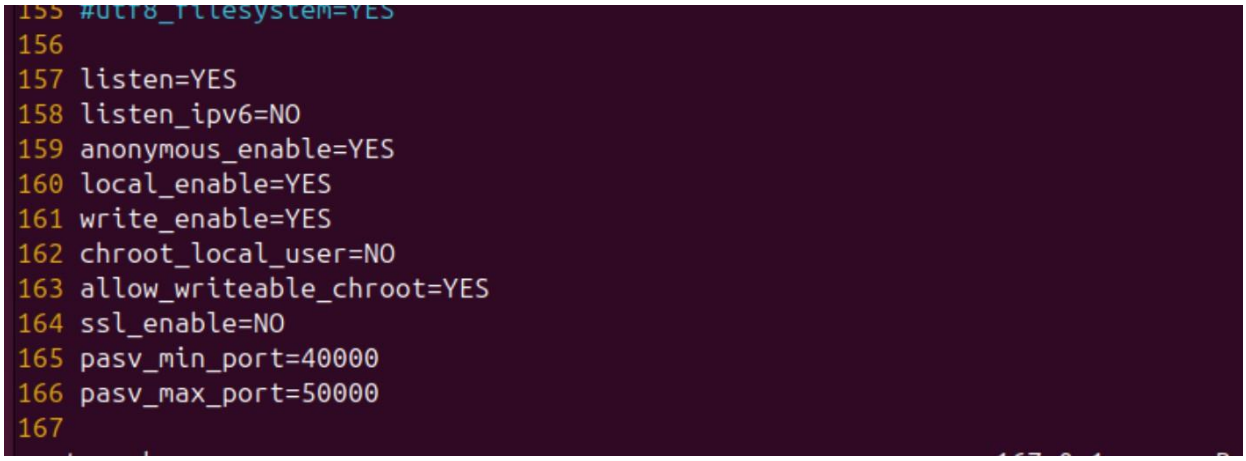
Then create a vulnerable directory -
sudo mkdir -p /srv/ftp_vulnerable
sudo chmod 777 /srv/ftp_vulnerable
sudo chown nobody:nogroup /srv/ftp_vulnerable

```
Processing triggers for man-db (2.12.1-3) ...
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo mkdir -p /srv/ftp_vulnerable
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo chmod 777 /srv/ftp_vulnerable
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo chown nobody:nogroup /srv/ftp_
vulnerable
honeypot@honeypot-VMware-Virtual-Platform:~$
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Now we need to configure vulnerable FTP - **sudo vim /etc/vsftpd.conf**
Then do the following changes in the configuration files

listen=YES
listen_ipv6=NO

anonymous_enable=YES
local_enable=YES
write_enable=YES
chroot_local_user=NO
allow_writeable_chroot=YES
ssl_enable=NO
pasv_min_port=40000
pasv_max_port=50000



Now we need to restart the FTP service - **sudo systemctl restart vsftpd**

Allow FTP traffic -
sudo ufw allow 21/tcp
sudo ufw allow 40000:50000/tcp sudo
systemctl restart ufw

```
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo ufw allow 21/tcp
sudo ufw allow 40000:50000/tcp
sudo systemctl restart ufw
Skipping adding existing rule
Skipping adding existing rule (v6)
Skipping adding existing rule
Skipping adding existing rule (v6)
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Then monitor FTP logs when the attacker attacks - **tail -f /var/log/vsftpd.log**

Now we need to configure firewall rules first we need to configure the **Actual server machine**

Enable UFW and Firewall rules -
sudo ufw default deny incoming
sudo ufw default allow outgoing

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw default deny i
ncoming
sudo ufw default allow outgoing
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Allow Essential Services like - SSH, HTTP, SMB and
FTP **sudo ufw allow 22/tcpSSH (for remote access)**
sudo ufw allow 80/tcp HTTP (if needed)

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw allow 22/tcp
Rules updated
Rules updated (v6)
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw allow 80/tcp
Rules updated
Rules updated (v6)
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Allow Specific Services (SMB, FTP) -

sudo ufw allow from 192.168.41.0/24 to any port 139,445 proto tcp # SMB sudo ufw allow from 192.168.41.0/24 to any port 21 proto tcp # FTP

sudo ufw allow from 192.168.41.0/24 to any port 40000:50000 proto tcp # Passive FTP

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw allow from 192
.168.41.0/24 to any port 139,445 proto tcp
[sudo] password for actual-machine:
Rules updated
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw allow from 192
.168.41.0/24 to any port 21 proto tcp
Rules updated
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw allow from 192
.168.41.0/24 to any port 40000:50000 proto tcp
Rules updated
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Enable Logging & Restart the Firewall

sudo ufw logging on

sudo systemctl restart ufw

Then reboot the system

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw logging on
sudo systemctl restart ufw
Logging enabled
actual-machine@actual-machine-VMware-Virtual-Platform:~$
```

Verify Rules - **sudo ufw status verbose**

```
actual-machine@actual-machine-VMware-Virtual-Platform:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
139,445/tcp                ALLOW IN    Anywhere
21/tcp                     ALLOW IN    Anywhere
40000:50000/tcp            ALLOW IN    Anywhere
22/tcp                     ALLOW IN    Anywhere
80/tcp                     ALLOW IN    Anywhere
139,445/tcp                ALLOW IN    192.168.41.0/24
21/tcp                     ALLOW IN    192.168.41.0/24
40000:50000/tcp            ALLOW IN    192.168.41.0/24
139,445/tcp (v6)           ALLOW IN    Anywhere (v6)
21/tcp (v6)                ALLOW IN    Anywhere (v6)
40000:50000/tcp (v6)       ALLOW IN    Anywhere (v6)
22/tcp (v6)                ALLOW IN    Anywhere (v6)
80/tcp (v6)                ALLOW IN    Anywhere (v6)
```

Now we will set the rules on **Honeypot Machine (Vulnerable Firewall)**

Allow All Incoming Traffic (Deliberately to make it Insecure)
sudo ufw default allow incoming
sudo ufw default allow outgoing

```
honeypot@honeypot-VMware-Virtual-Platform: ~
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo ufw default allow incoming
sudo ufw default allow outgoing
[sudo] password for honeypot:
Default incoming policy changed to 'allow'
(be sure to update your rules accordingly)
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Open Ports for SMB and FTP with No Restrictions
sudo ufw allow 21/tcp sudo
ufw allow 139,445/tcp
sudo ufw allow 40000:50000/tcp #passive FTP ports

```
(be sure to update your rules accordingly)
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo ufw allow 21/tcp
sudo ufw allow 139,445/tcp
sudo ufw allow 40000:50000/tcp
Skipping adding existing rule
Skipping adding existing rule (v6)
Skipping adding existing rule
Skipping adding existing rule (v6)
Skipping adding existing rule
Skipping adding existing rule (v6)
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Disable Logging (Avoid Detection by Attackers)

sudo ufw logging off

```
Skipping adding existing rule (v6)
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo ufw logging off
Logging disabled
honeypot@honeypot-VMware-Virtual-Platform:~$
```

Restart Firewall - **sudo systemctl restart ufw**

& Check the firewall rules (we need to reboot the system sometime)- **sudo ufw status verbose**

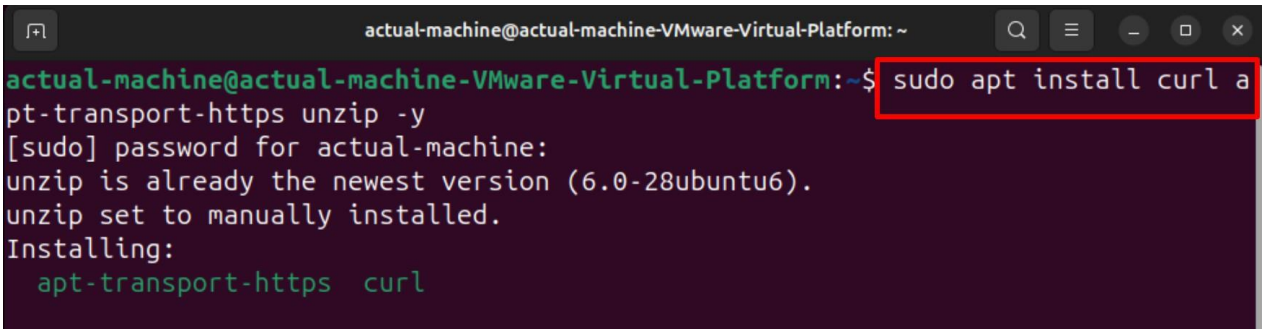
```
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo ufw enable
Firewall is active and enabled on system startup
honeypot@honeypot-VMware-Virtual-Platform:~$ sudo ufw status verbose
Status: active
Logging: off
Default: allow (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
139,445/tcp                ALLOW IN    Anywhere
21/tcp                     ALLOW IN    Anywhere
40000:50000/tcp            ALLOW IN    Anywhere
139,445/tcp (v6)           ALLOW IN    Anywhere (v6)
21/tcp (v6)                ALLOW IN    Anywhere (v6)
40000:50000/tcp (v6)       ALLOW IN    Anywhere (v6)
```

Now we will setup Wazuh server on Actual VM

We'll refer the documentation for the quick installation -
https://documentation.wazuh.com/current/quickstart.html

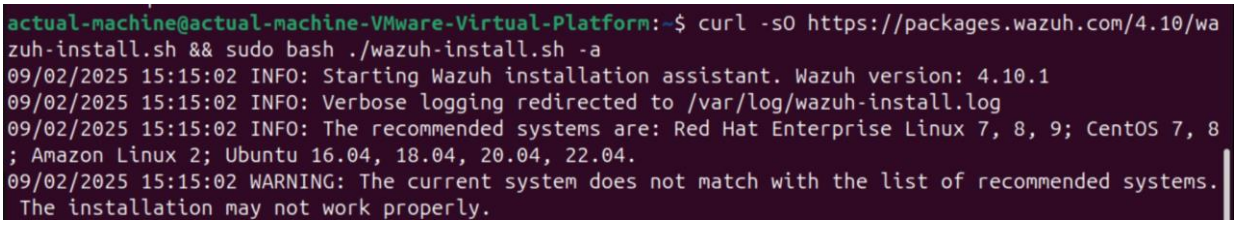
sudo apt install curl apt-transport-https unzip -y

curl - A tool for transferring data from URLs.
apt-transport-https– Allows APT to use HTTPS repositories.
unzip– Extracts .zipfile

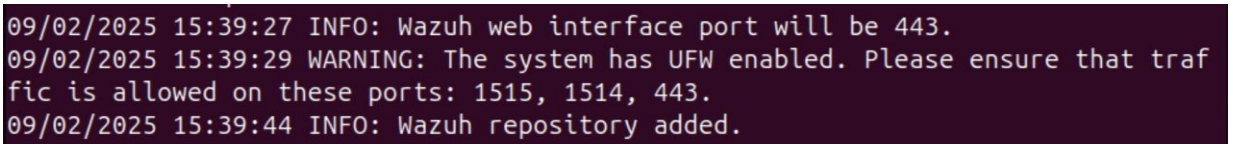


Install Wazuh Manager (Server) -
curl -sO https://packages.wazuh.com/4.10/wazuh-install.sh sudo bash wazuh-install.sh --wazuh-manager

@@@@ **curl -sO https://packages.wazuh.com/4.10/wazuh-install.sh && sudo bash ./wazuh-install.sh -a**



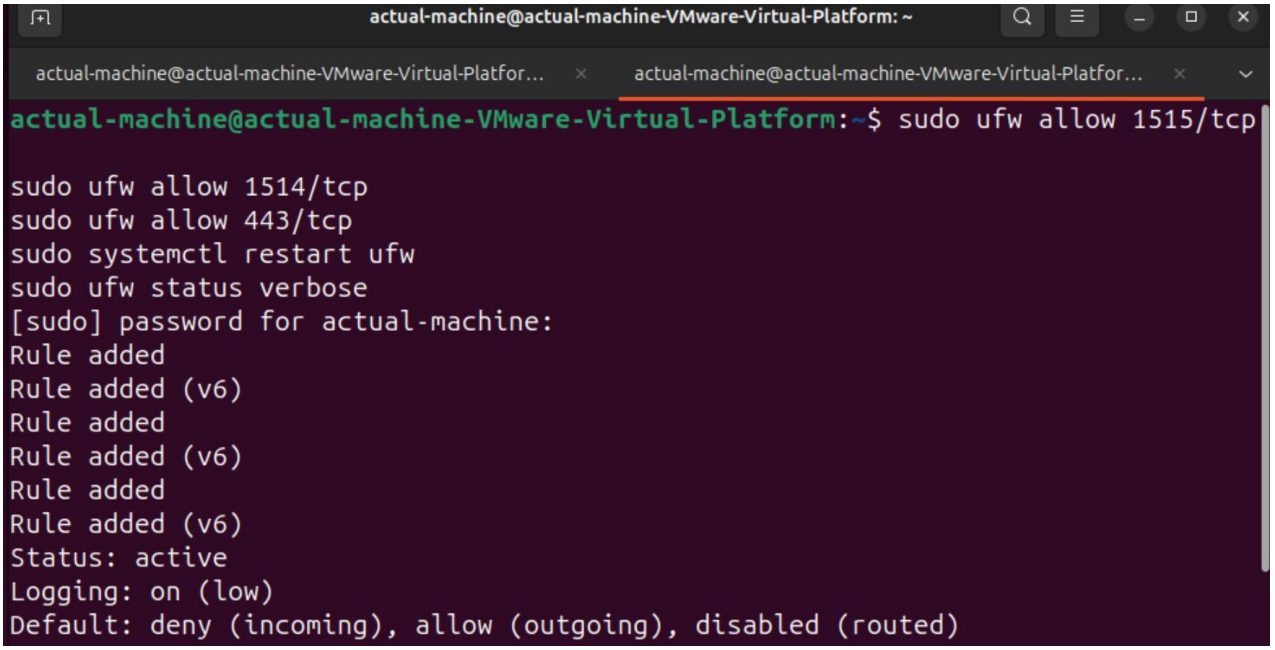
You need to do the following changes to Wazuh



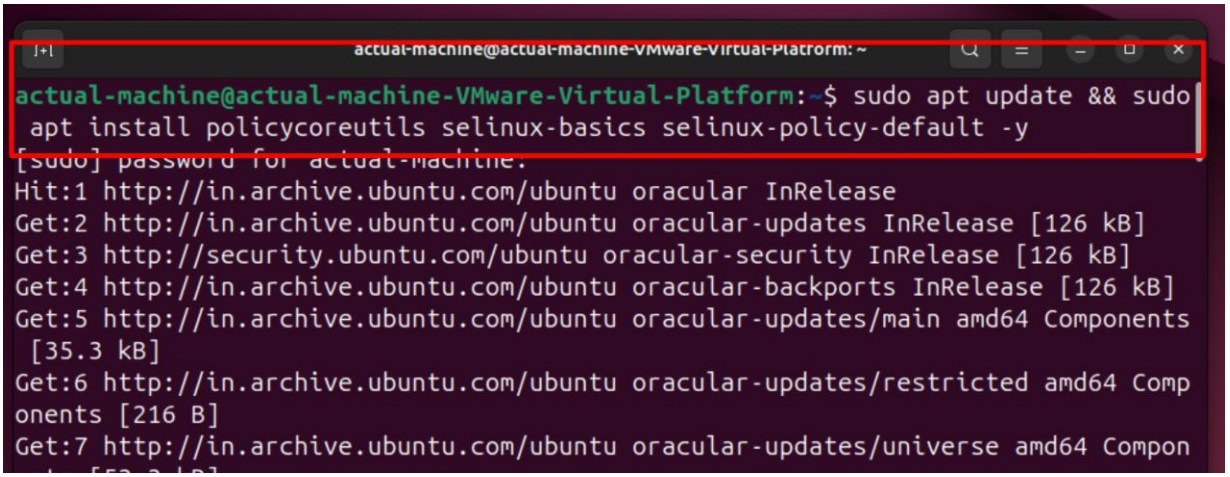
Because while installing this shows the error message

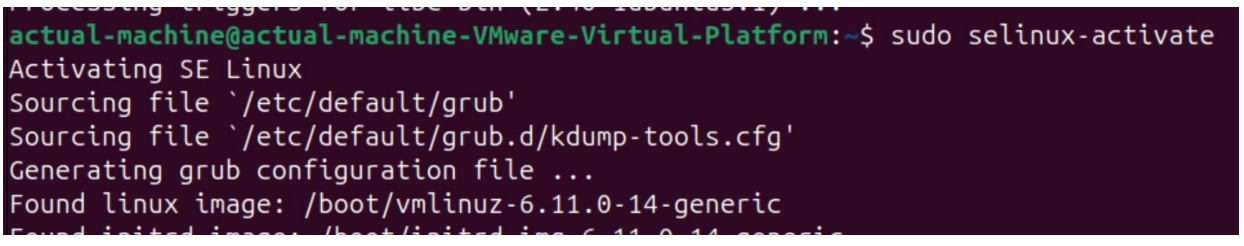
sudo ufw allow 1515/tcp

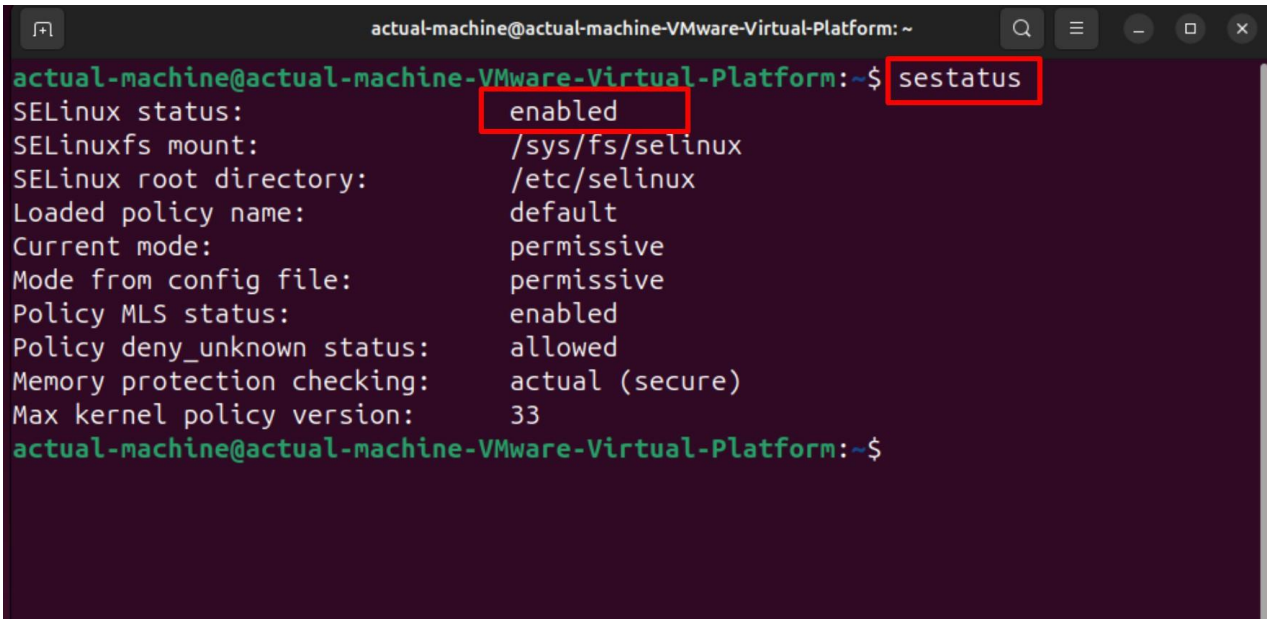
sudo ufw allow 1514/tcp
sudo ufw allow 443/tcp
sudo systemctl restart sudo
ufw status verbose

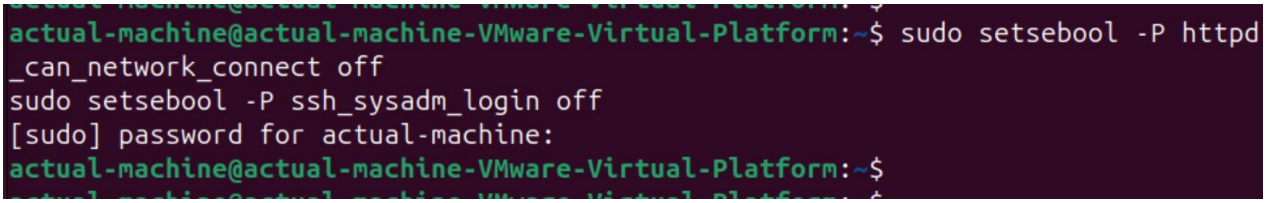


SElinix as IPS

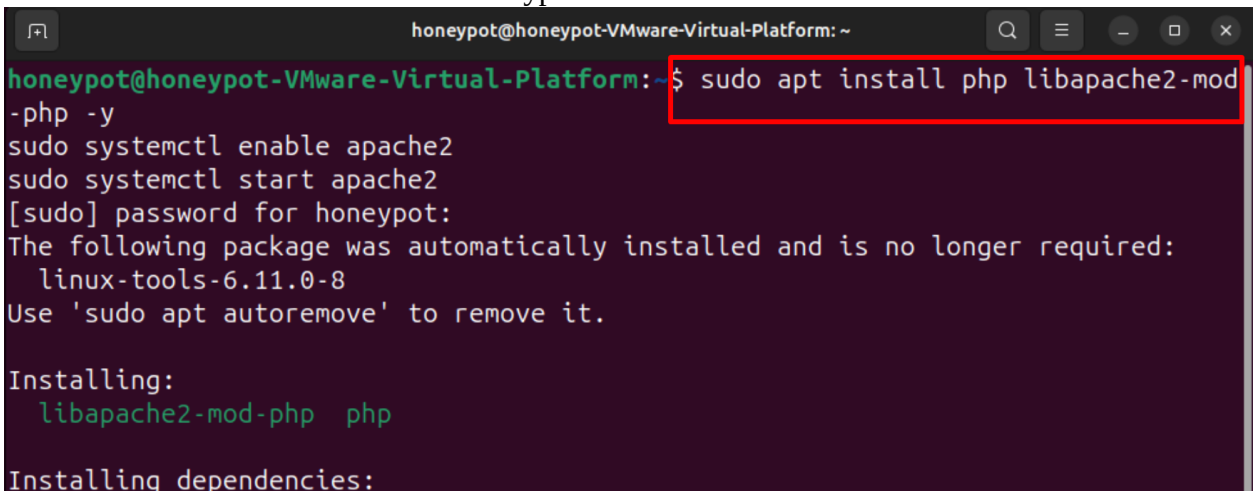








Install the website in both the VMs honeypot and actual machine



Vulnerable website

```
honeypot@honeypot-VMware-Virtual-Platform:/var/www/html$
honeypot@honeypot-VMware-Virtual-Platform:/var/www/html$
honeypot@honeypot-VMware-Virtual-Platform:/var/www/html$ sudo chmod 777 /var/www
/html/index.php
honeypot@honeypot-VMware-Virtual-Platform:/var/www/html$
honeypot@honeypot-VMware-Virtual-Platform:/var/www/html$
```

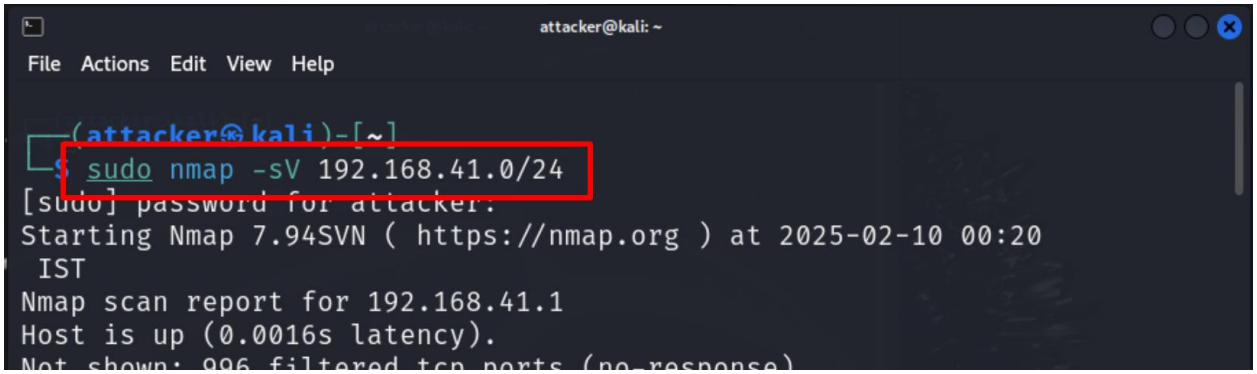
Now we will create a secure php website on healthy VM
And give the permission - **sudo chmod 644 /var/www/html/index.php**

Now KALI attacker point of view

Let's do the ping scan and see how many hosts are up - **sduo nmap -sP 192.168.41.0/24**

```
┌──(attacker㉿kali)-[~]
└─$ sudo nmap -sP 192.168.41.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-10 00:23 IST
Nmap scan report for 192.168.41.1
Host is up (0.00097s latency).
MAC Address: 00:50:56:C0:00:08 (VMware)
Nmap scan report for 192.168.41.2
Host is up (0.00059s latency).
MAC Address: 00:50:56:E9:2E:E1 (VMware)
Nmap scan report for 192.168.41.129
Host is up (0.0026s latency).
MAC Address: 00:0C:29:64:72:3B (VMware)
Nmap scan report for 192.168.41.130
Host is up (0.0028s latency).
MAC Address: 00:0C:29:1B:1C:0C (VMware)
Nmap scan report for 192.168.41.254
Host is up (0.00045s latency).
MAC Address: 00:50:56:FD:C1:3B (VMware)
Nmap scan report for 192.168.41.128
Host is up.
Nmap done: 256 IP addresses (6 hosts up) scanned in 2.01 seconds
```

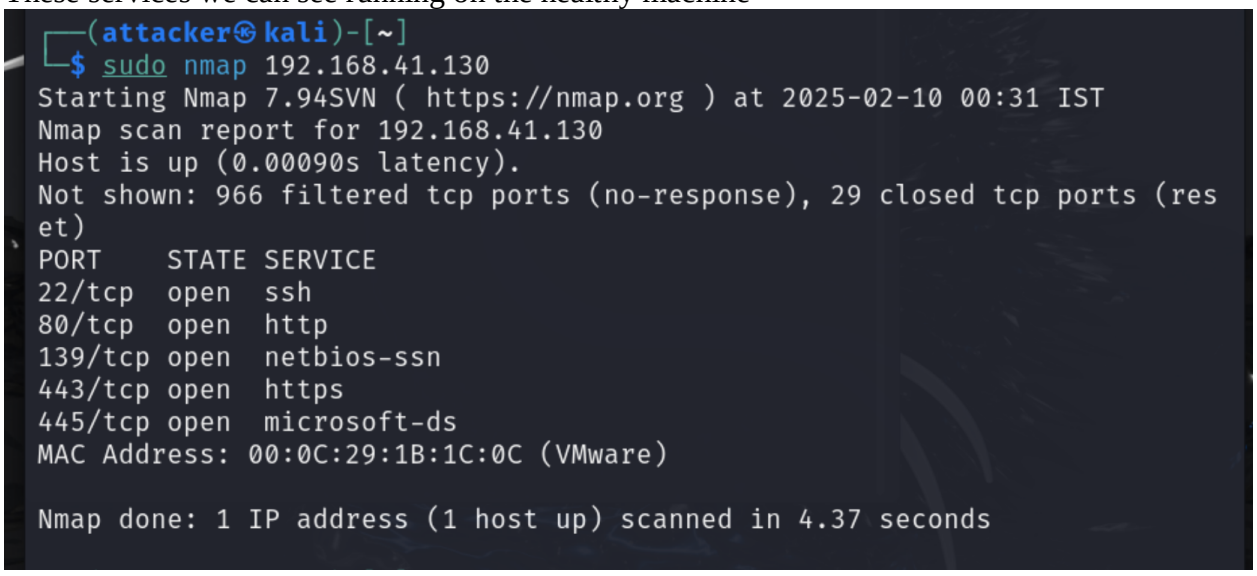
Now we now which hosts are up we will do the version info services that are running - **sudo nmap -sV 192.168.41.0/24**



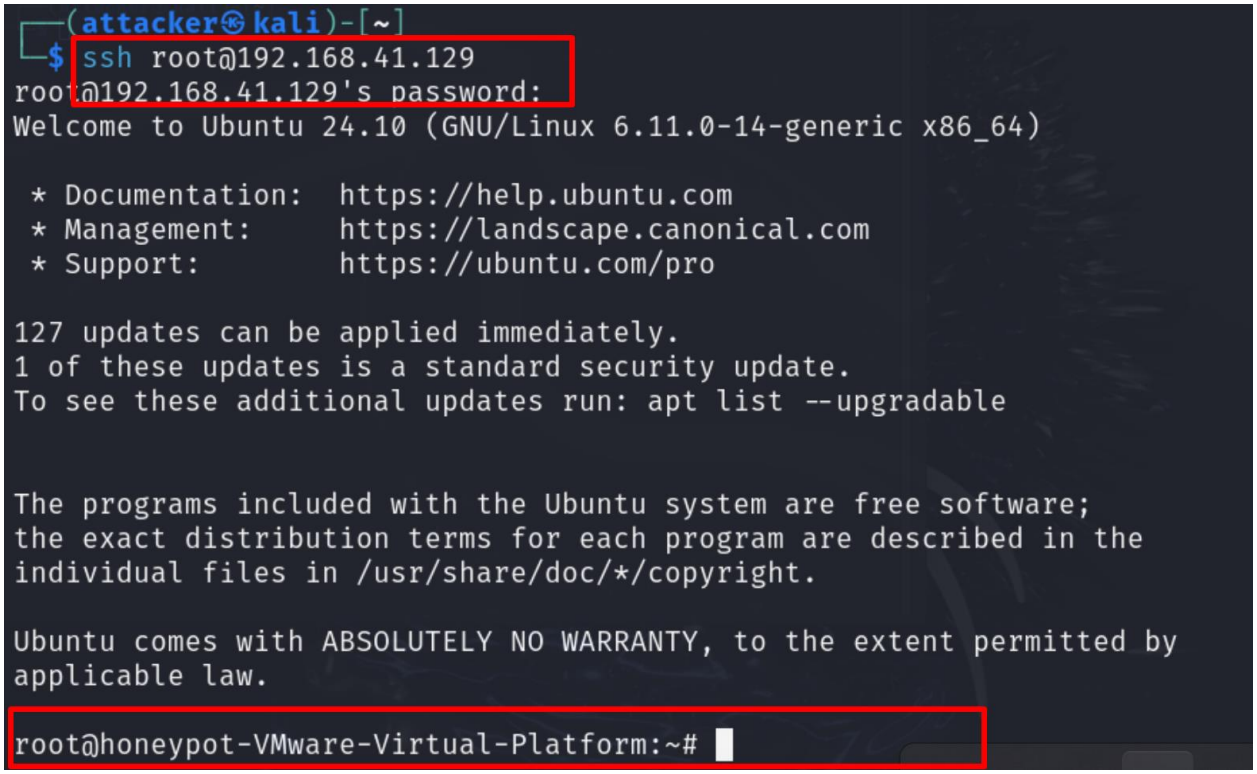
We can see 2 hosts having ports open is **192.168.41.129(honeypot)** and **192.168.41.130** actual healthy machine

These are the services that you can see running on the honeypot machine

These services we can see running on the healthy machine



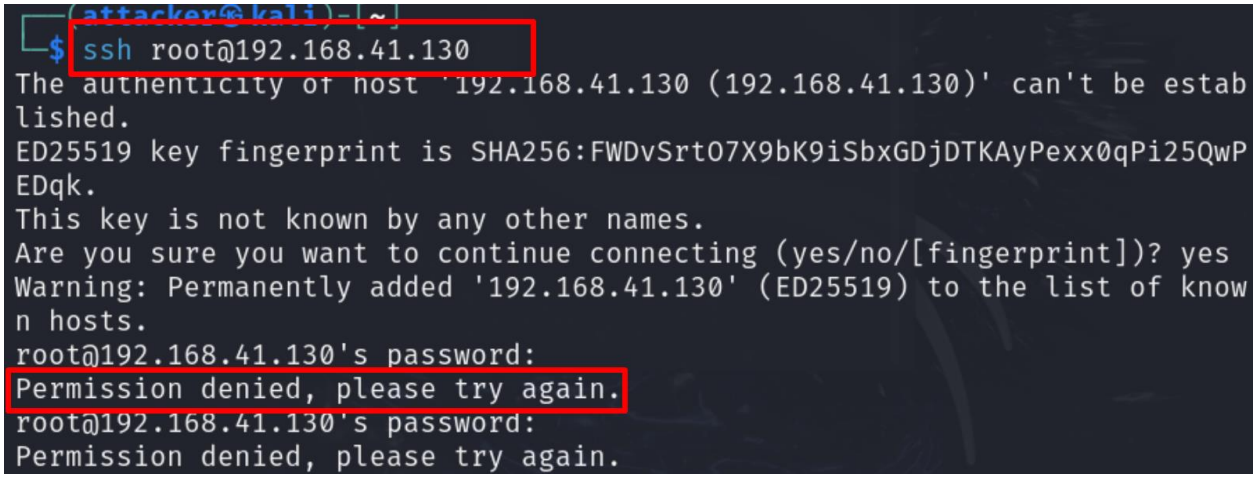
Here the attacker tries to get the ssh password - the most default password if **123**



Hence the SSH login is **successful**

Now on the healthy VM -

The permission is **denied** due to the firewall and configuration file



Now the attacker tries to attack the other services of the IP - **192.168.41.129** since that's a vulnerable one. Let’s perform the dictionary attack on the HTTP port 80

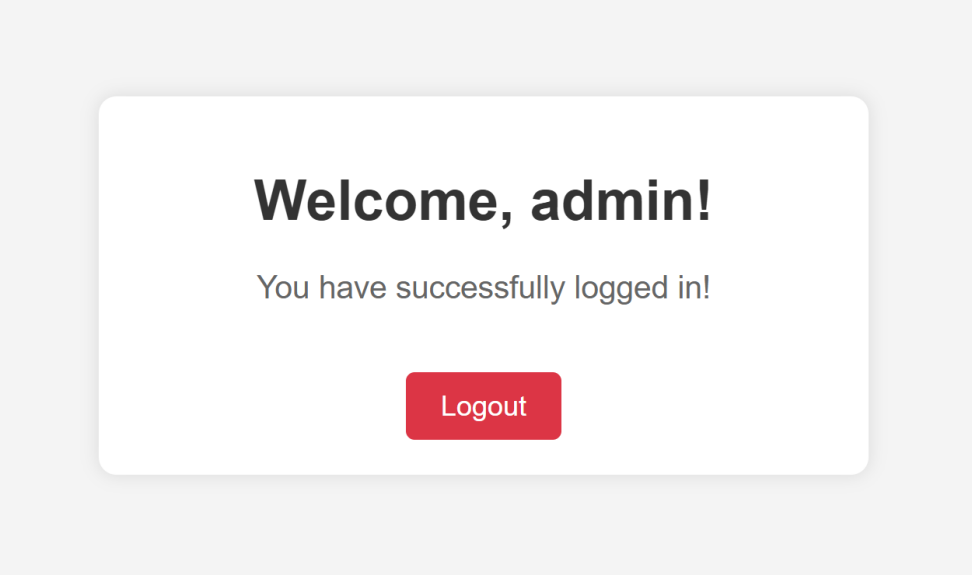
```
┌──(attacker㉿kali)-[~]
└─$ sudo hydra -L /usr/share/wordlists/usernames.txt -P /us
r/share/wordlists/passwords.txt http-get://192.168.41.129
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Pl
ease do not use in military or secret service organizations
, or for illegal purposes (this is non-binding, these *** i
gnore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting
 at 2025-02-10 01:28:49
[WARNING] You must supply the web page as an additional opt
```

By doing this dictionary attack the attacker got username and passwords **256** valid passwords

```
ssword: baseball
[80][http-get] host: 192.168.41.129   login: azureuser   pa
ssword: abc123
1 of 1 target successfully completed, 256 valid passwords f
ound
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished
 at 2025-02-10 01:28:53
```

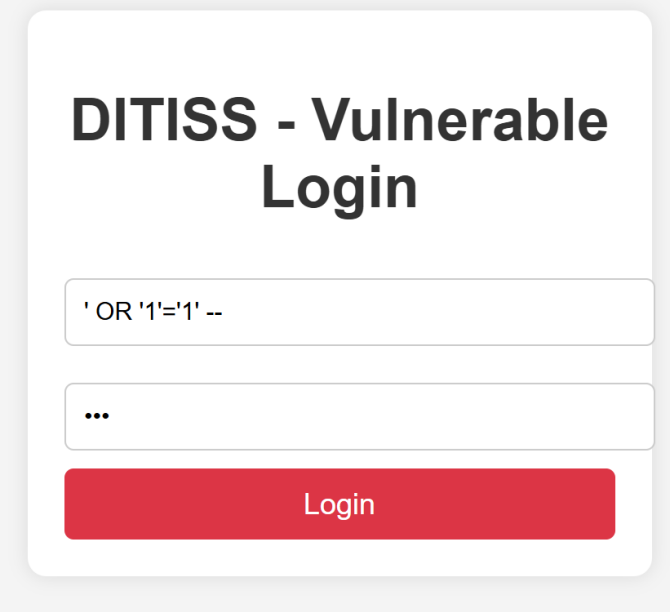
Now let's try one of the username and password

Username - admin
Passwords - password

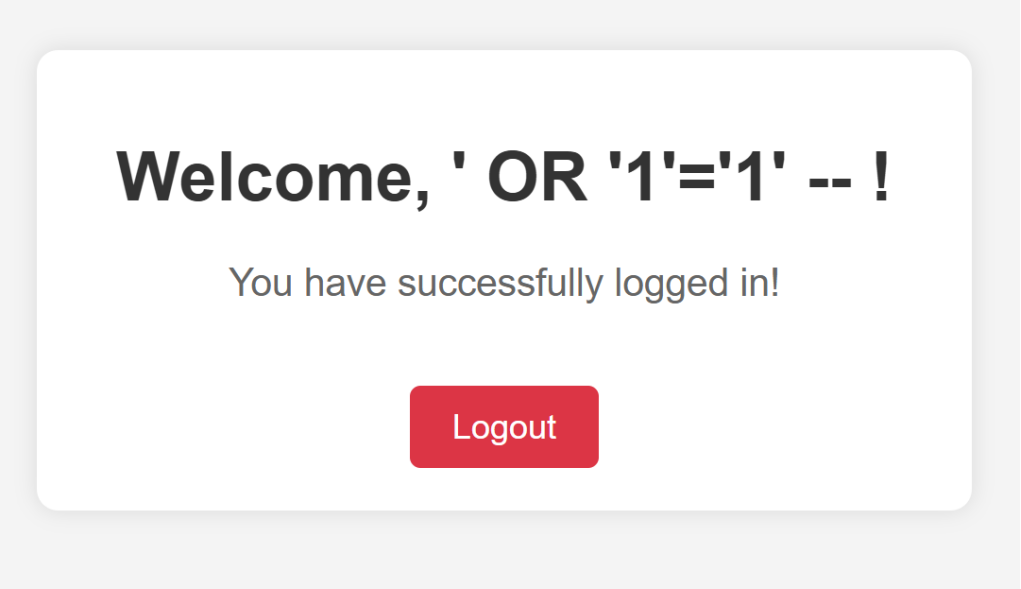


We get the message as login successful. Hence the dictionary attack is successful

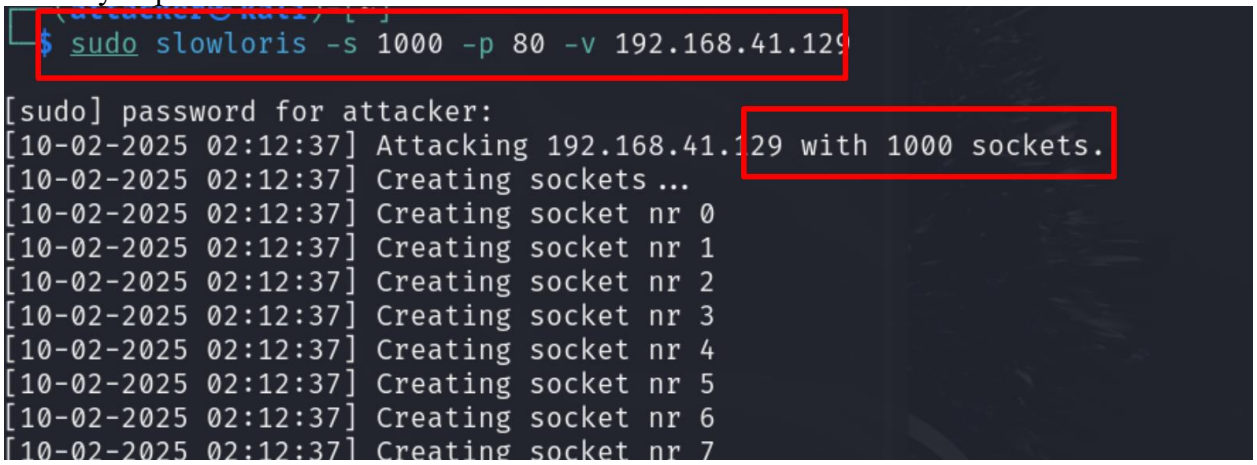
Now we will try to perform SQL injection attack putting **' OR '1'='1' --**



The login is successful leaving the password field blank



Let's try to perform the DoS attack



This attack will continue till 1000 counts keeping the website inaccessible for users

```
[10-02-2025 02:12:47] Creating socket nr 658
[10-02-2025 02:12:47] Creating socket nr 659
[10-02-2025 02:12:47] Creating socket nr 660
[10-02-2025 02:12:47] Creating socket nr 661
[10-02-2025 02:12:47] Creating socket nr 662
[10-02-2025 02:12:51] timed out
```

Attack on SMB open shares

```
┌──(attacker㉿kali)-[~]
└─$ smbclient -L //192.168.41.129 -U anonymous

Password for [WORKGROUP\anonymous]:

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        Public          Disk
        IPC$            IPC       IPC Service (honeypot-VMware-Virtual-Pl
atform server (Samba, Ubuntu))
Reconnecting with SMB1 for workgroup listing.

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            HONEYPOT-VMWARE-VIRTUAL-PLATFORM
```

We can see different shared resources

Now we will check the FTP anonymous login

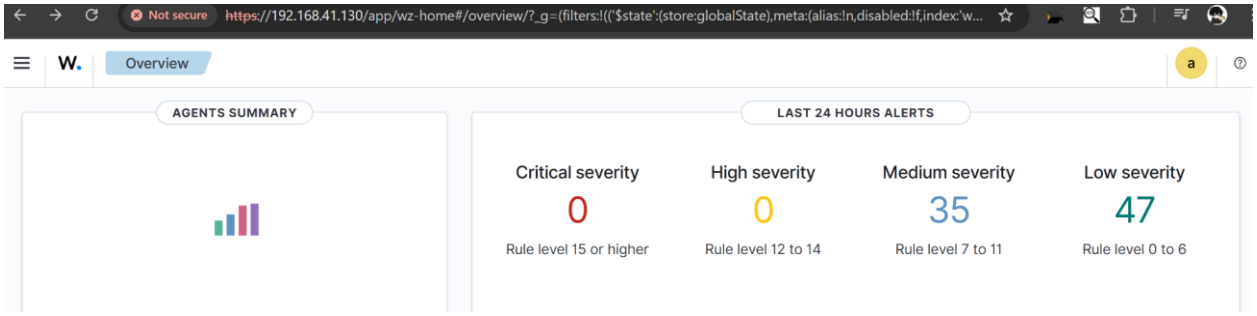
```
┌──(attacker㉿kali)-[~]
└─$ nmap --script ftp-anon -p 21 192.168.41.129
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-10 02:49 IST
Nmap scan report for 192.168.41.129
Host is up (0.0024s latency).

PORT   STATE SERVICE
21/tcp open  ftp
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
```

Let’s try to do it

```
┌──(attacker㉿kali)-[~]
└─$ ftp 192.168.41.129
Connected to 192.168.41.129.
220 (vsFTPd 3.0.5)
Name (192.168.41.129:attacker): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Here we can see it is **successful**

Now let's check the **wazuh dashboard**
For the alerts



This shows the severity of the attacks that has been happening which is okay as compared to the Honeypot that we made

CONCLUSION

The **Honeypot Analysis Project** aimed to deploy and analyze deceptive security systems designed to attract, detect, and analyze cyber threats. By simulating vulnerable services, the honeypot captured real-world attack patterns, allowing security researchers to study hacker techniques and improve network defenses.

- During the project, different types of honeypots were deployed, including:
- Low-interaction honeypots (e.g., Cowrie, Dionaea) to log basic attack attempts.
- High-interaction honeypots (e.g., OpenSSH, SMB, FTP, HTTP) to engage attackers.
- Network-based honeypots to monitor protocols like SMB, FTP, HTTP, NetBIOS, and SSH.

Key Objectives Achieved:

- Captured real-world attack data to understand hacker behavior.
- Logged unauthorized access attempts and exploitation attempts.
- Analyzed common attack vectors such as SQL Injection, Brute Force, and Malware Injection.
- Tested Intrusion Detection Systems (IDS) and Firewalls to improve network security.
- Enhanced security policies by identifying weaknesses in network configurations.

Most Common Attacker Techniques:

- Brute-force login attempts on SSH, FTP, and SMB.
- SQL Injection and Cross-Site Scripting (XSS) attacks on web-based honeypots.
- Port scanning and enumeration using Nmap, NetBIOS, and SMB tools.
- Exploitation of unpatched vulnerabilities (e.g., EternalBlue on SMB, Apache2 exploits).

Importance of Honeypots in Cybersecurity

The Honeypot Analysis Project demonstrated that honeypots are powerful tools for cybersecurity research and defense. Their key advantages include:

- Early Threat Detection – Identifies attack attempts before they reach real systems.
- Understanding Attack Tactics – Helps security teams analyze attacker behavior.
- Improving Incident Response – Provides insights to strengthen security policies.

- Deception as a Defense Mechanism – Wastes attacker time and resources

Conclusion & Future Scope

The project successfully demonstrated how honeypots can detect, analyze, and defend against cyber threats. By deploying different network and application honeypots, we gained valuable insights into real-world attack techniques.

- Future Improvements:
  - Deploying AI-based threat analysis for automated attack detection.
  - Expanding to cloud-based honeypots for monitoring attacks on cloud environments.
  - honeypots with SIEM tools for enterprise-level security monitoring.
  - Using deception technology like active defense mechanisms to counter attackers.

By continuously evolving honeypot strategies, organizations can stay ahead of cyber threats and strengthen their cyber defense mechanisms.

REFERENCES

1. Fortinet. (n.d.). What is a Honeypot? Retrieved from https://www.fortinet.com/resources/cyberglossary/what-is-honeypot

2. CrowdStrike. (n.d.). Honeypots. Retrieved from https://www.crowdstrike.com/en-us/cybersecurity-101/exposure-management/honeypots/#:~:text=A%20honeypot%20is%20a%20cybersecurity,methods%20and%20motivations%20of%20adversaries

3. IP With Ease. (n.d.). Honeypot vs Honeynet - Complete Guide. Retrieved from https://ipwithease.com/honeypot-vs-honeynet-complete-guide

4. Metallic. (n.d.). Honeypots: A Walk Down Memory Lane. Retrieved from https://metallic.io/blog/honeypots-a-walk-down-memory-lane

5. Adlumin. (n.d.). Honeypots 101: Origin, Services, and Types. Retrieved from https://adlumin.com/post/honeypots-101-origin-services-and-types

6. Proofpoint. (n.d.). History of Honeypots. Retrieved from https://www.proofpoint.com/us/threat-reference/honeypot#:~:text=and%20proactive%20manner.-,History%20of%20Honeypots,with%20Berferd%E2%80%9D%20by%20Bill%20Cheswick

7. IRJMETS. (2023). Evolution of Honeypots. Retrieved from [https://www.irjmets.com/uploadedfiles/paper/issue_5_may_2023/38648/final/fin_irjmets1683961611.pdf](https://www.irjmets.com/uploadedfiles/paper/issue_5_may_2023/38648

/final/fin_irjmets1683961611.pdf)

8. University of Arizona. (n.d.). Teaching Presentation on Honeypots. Retrieved from https://www2.cs.arizona.edu/~collberg/Teaching/466-566/2012/Resources/presentations/2012/topic12-final/report.pdf

9. Sapphire. (n.d.). What are Honeypots? Retrieved from https://www.sapphire.net/blogs-press-releases/what-are-honeypots

10. PSU. (n.d.). High Interaction Honeypot. Retrieved from https://mcn.cse.psu.edu/paper/guan-chongqi/wisec23-chongqi.pdf

11. ResearchGate. (n.d.). AI-Driven Adaptive Honeypots for Dynamic Cyber Threats. Retrieved from https://www.researchgate.net/publication/385483051_AI-Driven_Adaptive_Honeypots_for_Dynamic_Cyber_Threats

12. Geeks for Geeks. (n.d.). Honeypot vs Honeynet. Retrieved from https://www.geeksforgeeks.org/honeypot-vs-honeynet

13. Kaspersky. (n.d.). What is a Honeypot? Retrieved from https://www.kaspersky.com/resource-center/threats/what-is-a-honeypot

14. ResearchGate. (n.d.). Honeypot System Placement in an Organizational Network. Retrieved from https://www.researchgate.net/figure/Honeypot-system-placement-in-an-organizational-network_fig2_367413267

15. Comparitech. (n.d.). How to Establish a Honeypot on Your Network. Retrieved from [https://www.comparitech.com/net-admin/how-to-establish-a-honeypot-on-your-network](https://www.comparitech.com/net-admin/how-to-establish-a-honeypot-on-your-

network)

16. Sectigo Store. (n.d.). What is a Honeypot in Network Security? Retrieved from https://sectigostore.com/blog/what-is-a-honeypot-in-network-security-definition-types-uses