PROJECT REPORT TITLE :

HONEYPOT SERVER TO DETECT ATTACT PATTERNS

Cyber Security Mini Project

77 Date: 15/07/2025

Submitted By: Saurabh Rajendra Chavanke

nstitute: Elevated LabS



18 Introduction

A Honeypot is a security mechanism that simulates vulnerable services to lure attaker and study their behavior.

The goal is to collect intelligence on attack patterns without compromising real system. Goal of high-interaction honeypot is to gain root—or administrator level— access to the server and then monitor the attacker'S activity.

2 Objective

Simulate fake SSH/FTP services

Log attacker attempt and commands

Analays repeated intrusion patterns

Note: Note:

Visualize attacker IP geolocation

3 Tools & Technologies Used



🐍 Python

Cowrie

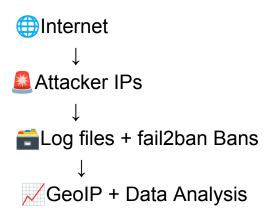
Nfail2ban

MaxMind GeoIIP

Purpose

Scripting & automation
SSH/FTP honeypot emulation
Auto IP blocking
IP Geolocation

4★ System Architecture



5 Implementation Steps

A) Poploy Honeypot on VM

- Installed Ubuntu on VirtualBox
- Set up Cowrie or custom python SSH server
- Enabled ports (22/21) for emulated services

B) 📩 Log Connection

- IP Address
- —♦Username tried
- __ Command attempted

C) Analyze Log Files

- Parsed logs with python script
- Detected brute-force patterns
- Counted top attacking IPs

D) N Block with fail2ban

- fail2ban setup to read logs
- Regex filters for Cowrie
- Auto ban via iptables

E) **OVISUALIZE IP Geolocation**

- Used GeoIP2 with IP logs
- created maps with folium

6 Sample Logs & Analysis

Top 5 Attacking IPs:

⊕IP Address	Attempts
102.22.34.55	48
185.234.123.10	33
182.75.65.20	25
196.52.20.18	19
203.0.113.77	17

7 IP Geolocation Map

- Using folium, attacker IPs were plotted
- Red markers = High threat
- Yellow = Medium
- Green = Low

- 🐮 Cowrie: https://github.com/cowrie/cowrie
- i fail2ban: https://www.fail2ban.org
- NaxMind: https://www.maxmind.com
- Python Docs: https://docs.python.org

Honeypot Project

This is a python-based honeypot to detect attack patterns.

Tools:

cowrie or custom python scripts, SSH/FTP emulation.

Mini Guide :

- A) Deploy honeypot on a VM.
- B) Log connections, ips, attempted command.
- C) Analize log file for repeated attempts.
- D) use fail2ban to block real threats.
- E) Visualize IP geolocation of attackers.

Deliverables :

Running honeypot + detailed logs + visual attack reports.

```
# Features :
- Real-time attack logging
- IP Address blocking
- Log visualization
  ## How to Run :
  1. clone the repo
  2. Run: python honeypot.py
  # Logs :
 Attack logs saved in log/attacks.db
  ## AUTHOR ##
```

SAURABH CHAVANKE

Codes:

1. Logger

```
import os
from datetime import datetime

log_dir = "logs"
log_file = os.path.join(log_dir, "honeypot.log")

os.makedirs(log_dir, exist_ok=True)

def log_attempt(ip, username, password):
    with open(log_file, "a") as f:
        log_entry = f"{datetime.now()} | IP: {ip} |
Username: {username} | Password: {password}\n"
        f.write(log_entry)
        print(f"[LOGGED] {log_entry.strip()}")
```

2. Blocker

```
from collections import defaultdict
MAX ATTEMPTS = 3
attempts = defaultdict(int)
blocked ips = "blocked ips.txt"
def is blocked(ip):
    try:
        with open(blocked_ips, "r") as f:
            return ip in f.read()
    except FileNotFoundError:
        return False
def register attempt(ip):
```

```
attempts[ip] += 1

if attempts[ip] >= MAX_ATTEMPTS:

    with open(blocked_ips, "a") as f:

       f.write(ip + "\n")

    return True

return False
```

3. Honeypot

```
import socket
import threading
from logger import log_attempt
from blocker import is blocked, register attempt
HOST = '0.0.0.0'
PORT = 2222
def handle client(client socket, addr):
    ip = addr[0]
    if is blocked(ip):
        print(f"[BLOCKED] Connection attempt from
blocked IP: {ip}")
        client socket.close()
        return
    client socket.send(b"Username: ")
    username =
client socket.recv(1024).decode().strip()
```

```
client socket.send(b"Password: ")
    password =
client socket.recv(1024).decode().strip()
    log attempt(ip, username, password)
    blocked = register attempt(ip)
    if blocked:
        client socket.send(b"You are blocked!\n")
    else:
        client socket.send(b"Access Denied.\n")
    client socket.close()
def start server():
    server = socket.socket(socket.AF INET,
socket.SOCK STREAM)
    server.bind((HOST, PORT))
    server.listen(5)
    print(f"[+] Honeypot running on port {PORT}")
```

```
while True:
        client socket, addr = server.accept()
        thread =
threading.Thread(target=handle client,
args=(client socket, addr))
        thread.start()
if name == " main ":
    start_server()
4.Geo visualizer
import matplotlib.pyplot as plt
from geopy.geocoders import Nominatim
import time
```

```
def get location(ip):
    geolocator = Nominatim(user agent="honeypot")
    try:
        location = geolocator.geocode(ip)
        return location.latitude,
location.longitude
    except:
        return None, None
def visualize blocked ips():
    ips = []
    with open("blocked ips.txt", "r") as f:
        ips = [line.strip() for line in
f.readlines()]
    latitudes, longitudes = [], []
    for ip in ips:
        lat, lon = get location(ip)
        if lat and lon:
```

```
latitudes.append(lat)
        longitudes.append(lon)
    time.sleep(1)
if latitudes:
   plt.scatter(longitudes, latitudes)
   plt.title("Blocked IPs - Geolocation Map")
   plt.xlabel("Longitude")
   plt.ylabel("Latitude")
   plt.grid(True)
   plt.show()
else:
   print("No valid IP locations found.")
```