Web Honeypot for Detecting Unauthorized Login Attempts Project Report

KONDURU RAKESH

1. Introduction

Cybersecurity threats are constantly evolving, and attackers frequently attempt unauthorized access to web applications. This project focuses on developing a web-based honeypot that mimics a login page and an admin panel to log, analyze, and prevent brute-force attacks. The honeypot captures malicious login attempts, logs the attacker's details, and takes countermeasures like blacklisting IPs and sending email alerts.

2. Objectives

The main objectives of this project are:

1. Simulate a vulnerable login page to attract and capture unauthorized login attempts.

2. Log attacker details including IP, username, password, and user-agent.

3. Implement rate limiting to slow down brute-force attacks.

4. Blacklist IPs after repeated failed attempts.

5. Send email alerts to notify administrators of suspicious activities.

6. Track geolocation of attackers using IP-based lookup.

3. Technologies Used

Component	Technology
Backend Framework	Flask (Python)
Rate Limiting	Flask-Limiter
Logging	File-based logging (honeypot.log)
Email Alerts	SMTP (Gmail)
IP Tracking	ipinfo.io API
Virtualization	Kali Linux in VirtualBox
Browser Testing	Mozilla Firefox

4. System Architecture

The honeypot consists of the following key components:

4.1 Fake Login Page

A normal-looking login page (/login) that captures credentials without granting access.

Any username and password entered are logged.

4.2 Fake Admin Panel

A decoy admin login (/admin/login) designed to attract bots.

Attackers trying to access /admin are tricked into entering credentials.

4.3 Logging System

Stores attack details in honeypot.log:

IP address

Username & Password entered

Browser (User-Agent)

Country (if detected)

4.4 IP Blacklisting Mechanism

Tracks failed attempts per IP in failed_attempts.txt.

After 5 failed login attempts, the IP is blacklisted in blacklist.txt.

Blacklisted IPs are denied access to the honeypot.

4.5 Email Alert System

Sends alerts to the administrator when an admin login attempt is detected.

Uses Gmail SMTP to send notifications.

5. Implementation Details

5.1 Code Structure

honeypot.py – Main Python script that runs the honeypot.

honeypot.log – Stores all attack logs.

blacklist.txt – Stores blacklisted IPs.

failed_attempts.txt – Tracks failed login attempts per IP.

5.2 Running the Honeypot

Step 1: Install Dependencies

pip install flask flask-limiter requests

Step 2: Run the Honeypot

python3 honeypot.py

Step 3: Access the Honeypot

Open a web browser and visit:

Login Page: http://127.0.0.1:8080/

Fake Admin Panel: http://127.0.0.1:8080/admin

Step 4: Check Logs

cat honeypot.log

6. Results & Observations

During testing, the honeypot successfully logged multiple unauthorized login attempts with different usernames and passwords. Below are sample log entries:

[LOGIN] IP: 127.0.0.1, Username: admin, Password: 1234, User-Agent: Mozilla/5.0

[ADMIN] IP: 192.168.1.100, Username: root, Password: password123, User-Agent: Chrome/99.0

Key Findings

1. Multiple login attempts were detected, simulating brute-force attacks.

2. Common weak passwords like 1234, password123, and admin were used frequently.

3. Bots attempted to access /admin multiple times.

4. IP blacklisting successfully prevented further attempts after 5 failures.

5. Email alerts were received for admin login attempts.

7. Security Enhancements & Future Work

7.1 Improvements

✅ Database Integration – Use SQLite/MySQL instead of text files for tracking attacks.

✅ Webhook Alerts – Implement Discord or Telegram notifications for real-time monitoring.

✅ Decoy File Download – Create fake download links to track bot activity.

7.2 Future Features

🚀 AI-based Attack Analysis – Use machine learning to detect patterns in attacks.

🚀 Captcha Implementation – Introduce captchas after multiple failed attempts.

🚀 Deception Techniques – Generate fake error messages to mislead attackers.

8. Conclusion

This project successfully implemented a web-based honeypot to detect and analyze unauthorized login attempts. The system effectively logs attack details, blacklists suspicious IPs, and alerts administrators in real time. This approach can be extended to monitor and defend real-world web applications against cyber threats.

9. References

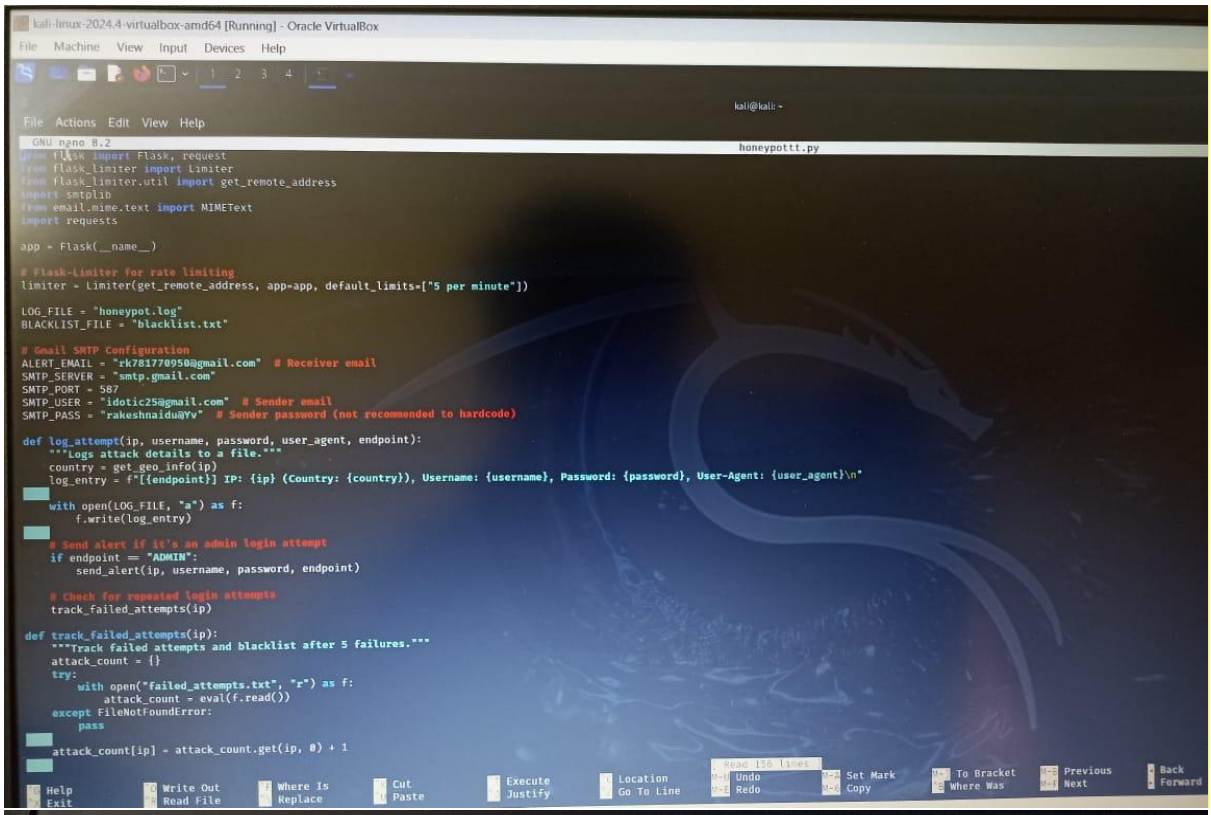
Flask Documentation: https://flask.palletsprojects.com/

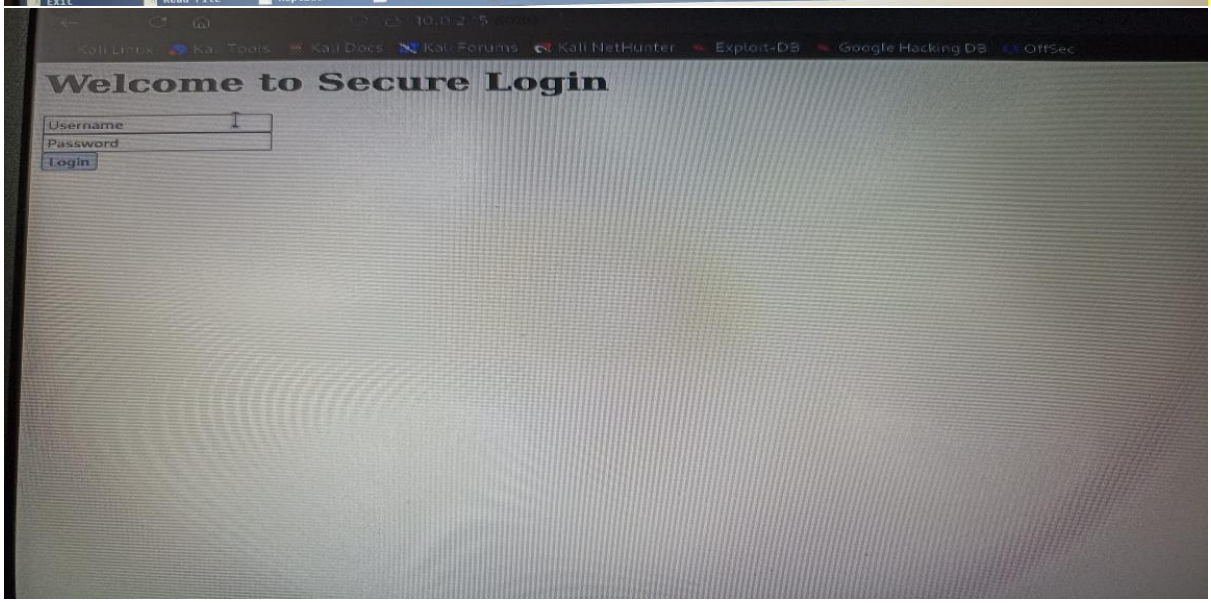
Flask-Limiter: https://flask-limiter.readthedocs.io/

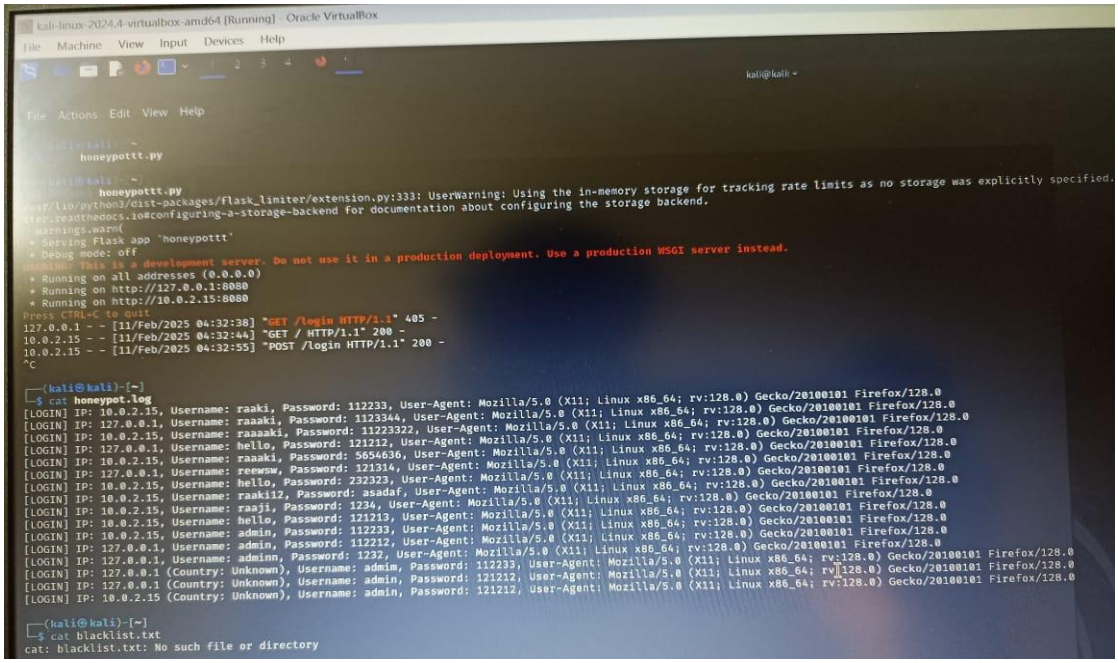
IP Info API: https://ipinfo.io/

Python SMTP Email: https://docs.python.org/3/library/smtplib.html

Screenshots :






kali-linux-2024.4-virtualbox-amd64 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
File Actions Edit View Help
honeypottt.py
honeypottt.py
* Serving Flask app 'honeypottt'
* Debug mode: off
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://10.0.2.15:8080
Press CTRL+C to quit
127.0.0.1 - - [11/Feb/2025 04:32:38] "GET /login HTTP/1.1" 405 -
10.0.2.15 - - [11/Feb/2025 04:32:44] "GET / HTTP/1.1" 200 -
10.0.2.15 - - [11/Feb/2025 04:32:55] "POST /login HTTP/1.1" 200 -
^C
(kali㉿kali)-[~]
$ cat honeypot.log
[LOGIN] IP: 10.0.2.15, Username: raaki, Password: 112233, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 127.0.0.1, Username: raaaki, Password: 1123344, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15, Username: raaaaki, Password: 11223322, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 127.0.0.1, Username: hello, Password: 121212, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15, Username: raaaki, Password: 5654636, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 127.0.0.1, Username: reewsw, Password: 121314, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15, Username: hello, Password: 232323, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15, Username: raaki12, Password: asadaf, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15, Username: raaji, Password: 1234, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15, Username: hello, Password: 121213, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15, Username: admin, Password: 112233, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 127.0.0.1, Username: admin, Password: 112212, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 127.0.0.1, Username: adminn, Password: 1232, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 127.0.0.1 (Country: Unknown), Username: admin, Password: 112233, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 127.0.0.1 (Country: Unknown), Username: admin, Password: 121212, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
[LOGIN] IP: 10.0.2.15 (Country: Unknown), Username: admin, Password: 121212, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
(kali㉿kali)-[~]
$ cat blacklist.txt
cat: blacklist.txt: No such file or directory