HoneyPot Server to Detect Attack Patterns

Introduction

Modern networks face a wide range of cyberattacks targeting vulnerable services. Deploying honeypots enables defenders to gather intelligence on attacker behavior by simulating and monitoring such services. This project focuses on designing and implementing a honeypot using Cowrie, an SSH/Telnet emulation tool, to attract, log, and analyze malicious activities. The system is further enhanced with log analysis, automated threat blocking, and attacker geolocation visualization.

Abstract

This report presents the deployment and functionality of a Cowrie-based honeypot server designed to monitor and log unauthorized connection attempts. The key objectives include simulating vulnerable SSH/FTP services, capturing detailed logs, analyzing attack patterns, applying real-time threat defense with fail2ban, and visualizing attacker geolocations. This project demonstrates a complete workflow from VM setup to hands-on incident logging and analysis.

Tools Used

- **Cowrie Honeypot**: For simulating SSH/FTP services and recording attacker activity.
- **Virtualenv/Python**: For dependency isolation and environment management.
- **fail2ban**: For automated IP banning and defense against brute-force attacks.
- **Nmap**: For identifying active hosts and port scanning.
- **nano**: For configuration editing.
- **Log Analysis Tools**: For inspecting log files and extracting attack patterns.
- **Geolocation Visualization Tools**: For mapping attacker IP sources.

Steps Involved in Building the Project

1. **Setup and Cloning Cowrie**
   The Cowrie honeypot repository was cloned using Git, followed by environment preparation using Python's virtualenv to isolate dependencies.

2. **Environment Preparation**
   Python environment was initialized for Cowrie. Required dependencies were installed and a dedicated user (`cowrie`) was configured for secure operation.

3. **Configuration**
   The Cowrie configuration template was copied and edited to define honeypot parameters—SSH port, logging format, and other behavioral emulation settings.

4. **Honeypot Deployment**
   Cowrie was started within its Python virtual environment, simulating vulnerable SSH/Telnet endpoints and activating continuous service monitoring.

5. **Attack Simulation and Logging**
   Attack activity was simulated using Nmap (to discover targets) and SSH brute-force attempts from a separate VM. Cowrie captured commands like `pwd`, `whoami`, and file access attempts.

6. **Log File Analysis**
   Collected logs provided detailed records of incoming connections, IP addresses, attempted commands, and session activities. Repeated attacker behavior and suspicious commands were identified using log analysis.

7. **Real-Time Threat Blocking**
   fail2ban was configured to monitor Cowrie logs, automatically banning IP addresses associated with repeated attack attempts to enhance server security.

8. **Geolocation Visualization**
   Logged attacker IPs were processed using geolocation services to visualize the geographic distribution of threats, aiding in threat intelligence gathering.

Conclusion

This project successfully implemented a Cowrie-based honeypot solution that is capable of attracting and logging malicious SSH/FTP activity. The system supports detailed log collection, enables attacker pattern analysis, and actively defends itself using fail2ban. The geolocation visualization provides valuable insight into attacker distribution. Together, these steps offer an effective approach for threat detection, incident response, and cybersecurity research in a controlled environment.