Title : Forensic Analysis of Cyber Attacks Using Cowrie Honeypot and Python

-Gitika Khira

Abstract

This project deploys a Cowrie honeypot on a local machine on port 2222 to capture attacker behavior. A custom Python analyzer parses logs, generates visualizations, detects anomalies, and produces forensic reports. Additionally, a fake attack simulator populates the honeypot logs with realistic attacker IPs, credentials, and commands for testing. The system provides a practical forensic lab for simulating, monitoring, and analyzing cyberattacks, which is useful for security analysts and researchers to study attack patterns.

---

1. Introduction

- **What honeypots are:**
  A honeypot is a system intentionally made vulnerable to attract attackers in order to study their methods. **Cowrie** is a widely-used honeypot that emulates SSH/Telnet services and logs attacker activity.
- **Why analyzing logs is important:**
  Honeypot logs help detect malicious activity such as brute-force attacks, malware delivery attempts, and unusual commands. Log analysis provides insights into attacker behavior, potential threats, and cybersecurity trends.
- **Objective of the project:**
  To build a forensic analyzer that parses honeypot logs on a **local machine**, visualizes attack data, detects anomalies, and simulates attacks for testing.

---

2. Objectives

- Deploy a honeypot on a **local machine**.
- Collect and log attacker IPs, credentials, and commands.
- Visualize attack patterns using charts and geolocation maps.
- Simulate attacks to test the forensic analyzer.

3. Tools & Technologies Used

- **Cowrie Honeypot** – for SSH/Telnet simulation.
- **Python Libraries:**
  - pandas, matplotlib, watchdog, scikit-learn, folium, reportlab
- **GeoLite2** – for geolocation of attacker IPs.
- **JSON logs** – generated by Cowrie.

---

4. Methodology / Steps Involved

1. **Honeypot Setup:** Install and configure Cowrie on port 2222.
2. **Log Parser:** Build log_parser.py to extract attacker IPs, credentials, and commands.
3. **Visualization:** Use graphs.py to generate charts of top attackers, commands, and credential attempts.
4. **GeoIP Mapping:** Plot attack locations on maps using geoip_map.py and GeoLite2.
5. **Real-Time Monitoring:** Use real_time_monitor.py to track attacks as they occur.
6. **Anomaly Detection:** Apply ml_anomaly.py for detecting unusual attack patterns.
7. **Fake Attack Simulator:** Use fake_cowrie_attacks.py to generate realistic attack logs.
8. **PDF Report Generation:** Summarize findings using report_generator.py.

---

5. Data Collection & Analysis

- **Collected logs:** cowrie.log, cowrie.json.
- **Analyzed using Python scripts:** analyze_logs.py.
- **Key analyses:**
  - Top attacking IPs.
  - Most attempted username/password combinations.
  - Commands executed by attackers.
  - Geolocation mapping using GeoIP2 and Folium.

6. Discussion

- Insights into attacker behavior.
- Importance of honeypots for cybersecurity research and threat intelligence.
- Limitations:
    - Local deployment may not capture all real-world attacks.
    - Geolocation can be inaccurate in some cases.

---

7. Conclusion

- The system provides a practical lab for simulating, monitoring, and analyzing cyberattacks.
- Useful for security analysts and researchers to study attack patterns and improve defenses.
- Deploy the honeypot on a VPS for capturing real-world attacks.
- Add more anomaly detection models using machine learning.
- Include automated alerting via email or dashboard for detected attacks

---

8. References

- Cowrie Honeypot documentation.
- Python libraries: pandas, matplotlib, folium, scikit-learn, reportlab.
- GeoLite2 documentation.
- Research papers/articles on honeypots and cyber forensics.