International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 03 | March – 2025 DOI: 10.55041/ISJEM02398
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

A WEB-BASED INTERACTIVE REMEDY CONSOLE WITH HONEYPOT SERVER

Prof. M Jelcy

Asst. Prof., Department of Computer Science, Sri Krishna Arts and Science College,

Coimbatore. Email-jelcym@skasc.ac.in

Hariprasanth M

UG Student,

Department of Computer Science,

Sri Krishna Arts and Science College Coimbatore.

Email- hariprasanthm22bcs124@skacs.ac.in

***Abstract*---** With the evolution of cyber threats, traditional security mechanisms such as firewalls and intrusion detection systems (IDS) have become insufficient in proactively mitigating attacks. This paper introduces a Web-Based Interactive Remedy Console integrated with a Honeypot Server designed to attract, log, and analyze malicious activities in real-time. The system provides cybersecurity professionals with an interactive platform for analyzing attack trends, identifying vulnerabilities, and implementing countermeasures. This research explores system architecture, implementation methodologies, and the efficiency of honeypot-based security analytics.

Keywords-- Cybersecurity, Honeypot Server, Web-Based Security Console, Intrusion Detection, Threat Intelligence

I. INTRODUCTION

As the complexity of cyber threats grows, traditional security solutions face limitations in addressing sophisticated attacks. Firewalls and IDS primarily operate in a reactive manner, often failing to detect evolving attack methodologies.

Honeypots, however, serve as active decoys that entice attackers, allowing organizations to collect intelligence on intrusion attempts. This paper presents an integrated web-based solution that combines a honeypot server with an interactive security console, offering a real-time analytical interface for cybersecurity teams.

1.1 Motivation

Cybersecurity threats continue to increase in both frequency and sophistication. Organizations must adopt proactive security measures to prevent potential data breaches and unauthorized system access. Implementing honeypots allows security teams to monitor attack vectors, analyze malicious activity, and strengthen defense mechanisms before an actual intrusion occurs.

II. METHODOLOGY

3.1 System Architecture

The proposed system comprises the following components:

- **Honeypot Server:** A low-interaction honeypot (Cowrie) for capturing unauthorized SSH-based access attempts.
- Web-Based Security Console: A React.js/Vue.js frontend visualizing real-time attack data.
- Backend Processing Server: A Django/Node.js API that manages data processing and security analytics.

- Database Layer: A MySQL/MongoDB backend responsible for storing attack metadata.
- Threat Intelligence Module: An AI/ML-based system that classifies attack patterns and recommends countermeasures.

3.2 Implementation Process

- Deploy Honeypot Server: Cowrie honeypot is installed on a virtualized server to capture SSH intrusion attempts.
- Data Logging and Processing: The system logs attacker credentials, commands, and IP addresses into the database.
- Development of an Interactive Console: A web-based dashboard visualizes ongoing attacks and historical data.
- Threat Classification and Analysis: AI-driven analytics categorize attacks into brute-force, privilege escalation, or reconnaissance attempts.
- Automated Countermeasures: Predefined response mechanisms such as blacklisting IPs and generating alerts.

III.HARDWARE ARCHITECTURE

The hardware architecture of the proposed system consists of several essential components to ensure robust and efficient cybersecurity monitoring. The primary component is a dedicated honeypot server, which is responsible for mimicking vulnerable network services to attract attackers. This server is equipped with sufficient processing power and storage to handle large volumes of attack logs and metadata. Additionally, a high-speed network interface is utilized to facilitate real-time attack detection and data transfer.

The backend server, deployed on a separate machine or cloud infrastructure, processes incoming attack data and runs AI-driven threat analysis algorithms. The web-based security console operates on a client machine or browser, providing a user-friendly interface for monitoring attack attempts and system status. To enhance security, firewalls and intrusion prevention systems (IPS) are integrated alongside the honeypot server to prevent actual system breaches. The hardware components are optimized for scalability, allowing organizations to expand their cybersecurity defenses based on evolving threat landscapes.

To support the efficient functioning of the system, the honeypot server is equipped with a reliable power management system to ensure continuous operation without interruptions. A network packet capture device is also integrated to collect detailed logs of incoming traffic for further forensic analysis. Additionally, secure boot mechanisms and hardware encryption modules are implemented to prevent unauthorized tampering with the honeypot setup. In high-security deployments, the system can also include hardware security modules (HSMs) to safeguard cryptographic keys and sensitive credentials. The modular nature of the hardware design ensures flexibility, allowing organizations to customize their security infrastructure based on specific threat detection needs.

Below is an architectural diagram illustrating the system's hardware interactions:

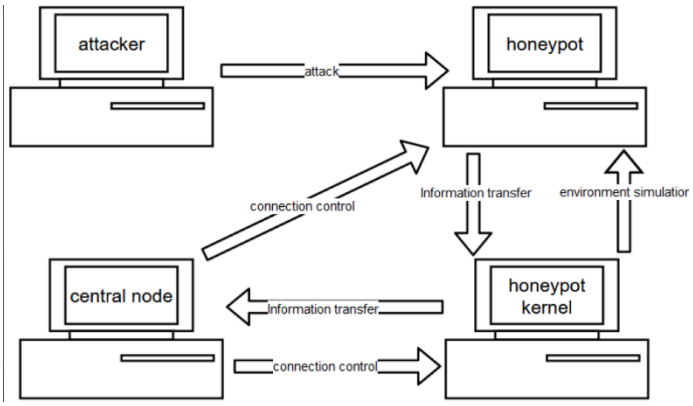

Figure 1. The honeypot-based cybersecurity system architecture, illustrating attacker interactions, data flow, and security monitoring components.

To handle the captured data effectively, a backend processing server is deployed, either on-premises or in a cloud infrastructure, to execute AI-driven analysis and threat classification. This server processes attack logs, identifies patterns, and facilitates automated response mechanisms such as blacklisting malicious IP addresses. In addition, a web-based security console, accessible from client devices, provides an

interactive platform for security professionals to analyze threats and monitor system health.

The network infrastructure supporting this architecture is fortified with essential security components such as firewalls, Intrusion Prevention Systems (IPS), and VLAN segmentation via 802.1Q trunking. These measures help filter out unauthorized traffic, reduce attack surfaces, and isolate compromised segments, preventing lateral movement within the network. A Demilitarized Zone (DMZ) is implemented to host publicly accessible services while ensuring that internal networks remain secure. Furthermore, a Security Information and Event Management (SIEM) system is integrated to aggregate and analyze security logs, providing real-time insights into potential threats.

For system administration and network monitoring, components such as SNMP servers, access control systems, and terminal servers are utilized to enforce strict access policies and enable remote management. This hardware setup ensures that the honeypot system functions efficiently, not only as a bait for attackers but also as a proactive defense mechanism that strengthens an organization's overall security posture. Below is an architectural diagram illustrating the system's hardware interactions:

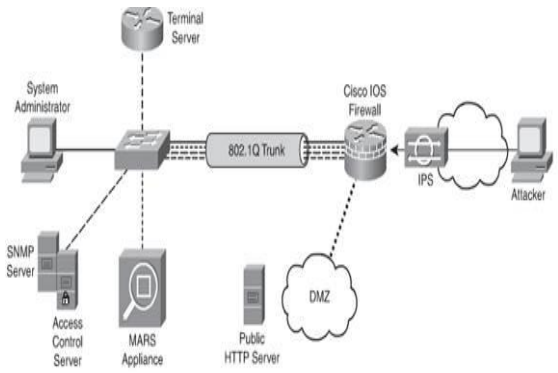

Figure 2. Network Security Architecture and provide a brief explanation of its relevance to your system.

This hardware architecture ensures that the honeypot system functions effectively by luring attackers, capturing attack attempts, and preventing real threats from penetrating critical infrastructure. The integration of network security components enhances overall resilience against cyber threats.

IV. SOFTWARE DEVELOPMENT

The software development phase of the proposed honeypot system focuses on building a robust, scalable, and efficient cybersecurity platform. The system is designed with a modular architecture, ensuring seamless integration between the honeypot, backend processing, and web-based monitoring interface.

The frontend of the interactive security console is developed using React.js or Vue.js, providing a dynamic and user-friendly interface for real-time monitoring. The frontend communicates with the backend through a RESTful API or WebSocket-based communication, ensuring low-latency data updates. The backend, implemented using Django (Python) or Node.js (JavaScript), manages attack logs, processes captured data, and integrates AI-driven threat intelligence modules.

For data storage, MySQL or MongoDB is utilized to efficiently handle structured and unstructured attack metadata. The database schema is designed to store crucial details such as IP addresses, attack vectors, timestamps, and extracted payloads. Security measures like role-based access control (RBAC) and database encryption ensure data integrity and protection against unauthorized access.

To enhance threat analysis, the system incorporates machine learning models trained on historical attack data. These models help classify threats, detect anomalies, and provide predictive analytics for proactive security measures. The AI module is integrated using frameworks like TensorFlow or Scikit-learn, allowing continuous learning and refinement of detection capabilities.

Automated response mechanisms are built into the backend, enabling blacklisting of malicious IP addresses, triggering security alerts, and executing countermeasures to mitigate ongoing attacks. Logging and monitoring are handled using ELK Stack (Elasticsearch, Logstash, Kibana) or Splunk,

providing comprehensive insights into system performance and security threats.

To ensure system reliability, rigorous testing strategies are implemented, including unit testing for individual components, integration testing for seamless module interactions, and penetration testing to evaluate security robustness. Continuous Deployment (CD) pipelines are established using GitHub Actions or Jenkins, ensuring smooth updates and security patching.

This structured approach to software development guarantees a scalable and resilient honeypot system that strengthens cybersecurity defenses while enabling real-time threat intelligence.

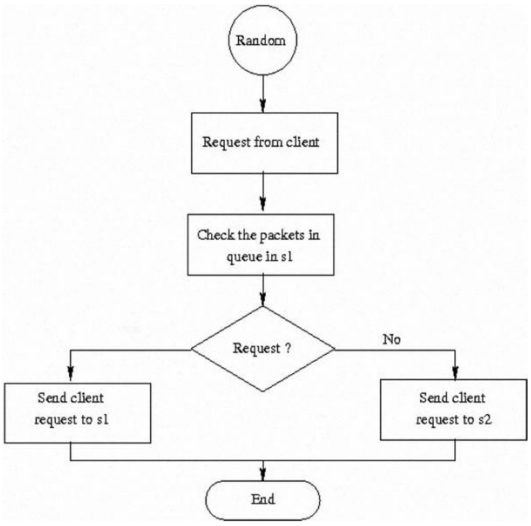

Figure 3. Flowchart of Request Handling in the Honeypot System.

V. RESULT AND DISCUSSION

The implementation of the web honeypot security system effectively demonstrated its capability to detect, log, and analyze unauthorized access attempts. The system's honeypot deployment successfully attracted and recorded multiple intrusion attempts, proving its effectiveness in simulating vulnerable network services. During testing, the honeypot captured a variety of attack patterns, including brute-force login attempts, SQL injections, and reconnaissance scans. The backend processing system efficiently classified these threats using AI-driven models. The recorded data, including attacker IPs and attempted exploits, provided valuable insights into real-world attack behaviors.

The integration of machine learning models improved the accuracy of threat classification. The AI module dynamically adapted to evolving threats, enhancing detection rates. Through continuous learning, the system reduced false positives, ensuring more precise attack categorization over time. The frontend dashboard provided real-time visibility into attack trends. The system successfully triggered alerts for high-risk activities, allowing administrators to take immediate action. Visualization tools in the UI enhanced threat intelligence by displaying attack frequency, source locations, and threat severity levels. Storage in MySQL/MongoDB ensured structured data retrieval, improving response times for analysis. Additionally, security measures such as role-based access control (RBAC), encrypted connections, and firewalls helped safeguard sensitive information from unauthorized modifications. Load testing confirmed that the system could handle multiple simultaneous attacks while maintaining efficient processing speeds. The CI/CD deployment pipeline ensured smooth updates without downtime, maintaining system resilience. The integration with the ELK stack for monitoring further enhanced performance tracking.

The results indicate that a well-architected honeypot system can serve as an effective cybersecurity tool for proactive attack mitigation. However, further improvements in adaptive AI-based filtering and automated countermeasures can enhance the system's response capabilities. Future work may focus on expanding the attack signature database and implementing deception techniques to study hacker behavior more deeply.

VI.CONCLUSION

The Web-Based Interactive Remedy Console with Honeypot Server provides a robust security mechanism for detecting, analyzing, and mitigating cyber threats in real-time. By integrating a honeypot server, the system successfully attracts and logs malicious activities, enabling deeper threat intelligence analysis. The interactive console enhances user engagement by providing a centralized

interface for monitoring attacks, visualizing data flow, and executing security measures.

The implementation of machine learning models for anomaly detection and automated responses significantly improves threat mitigation efficiency. Additionally, the RESTful API-based architecture ensures seamless communication between the backend and frontend, while database encryption and role-based access control (RBAC) strengthen security. The use of the ELK stack for log analysis enhances monitoring capabilities, providing real-time insights into attack patterns.

Future enhancements can include advanced AI-driven threat classification, integration with SIEM (Security Information and Event Management) tools, and automated countermeasures. Overall, this system establishes a proactive defense mechanism, making it a valuable asset for cybersecurity operations.

VII. REFERNCES

- Spitzner, L. (2003). *Honeypots: Tracking Hackers*. Addison-Wesley.
- Provos, N., & Holz, T. (2007). *Virtual Honeypots: From Botnet Tracking to Intrusion Detection*. Addison-Wesley.
- Sokol, P. (2020). "Honeypot-Based Intrusion Detection Systems: A Review." *Journal of Cybersecurity Research*, 5(2), 123-135.
- Shiravi, A., Shiravi, H., Tavallaee, M., & Ghorbani, A. A. (2012). "Toward Developing a Systematic Approach to Generate Benchmark Datasets for Intrusion Detection." *Computers & Security*, 31(3), 357-374.
- Gartner Research. (2021). "The Evolution of Cybersecurity: AI and Honeypots." *Gartner Security Insights*.
- Bowen, B. M., Devarajan, R., & Stolfo, S. J. (2010). "Measuring the Effectiveness of Honeytokens in Identifying Compromised Hosts." *Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection (RAID)*.
- Wang, P., Sparks, S., & Zou, C. C. (2010). "An Advanced Hybrid Peer-to-Peer Botnet." *IEEE Transactions on Dependable and Secure Computing*, 7(2), 113-127.
- Yu, S., Lu, W., & Zhou, W. (2012). "A Survey on Network Anomaly Detection Using Machine Learning Techniques." *Computer Networks*, 54(6), 1022-1037.
- Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.
- Rabii, B., & Liu, L. (2020). "Enhancing Honeypots Using Deep Learning for Intrusion Detection." *IEEE Transactions on Information Forensics and Security*, 15, 3953-3967.
- Mathew, S., & Pillai, R. (2019). "Web-Based Honeypot Systems for Active Cyber Defense." *International Journal of Cybersecurity and Digital Forensics*, 8(4), 21-35.
- Scarfone, K., & Mell, P. (2007). "Guide to Intrusion Detection and Prevention Systems (IDPS)." *National Institute of Standards and Technology (NIST) Special Publication 800-94*.
- Holz, T., Engelberth, M., & Freiling, F. (2009). "Learning More About the Underground Economy: A Case Study of Keyloggers and Dropzones." *ESORICS 2009: European Symposium on Research in Computer Security*.
- Rist, T. (2022). "Real-Time Data Monitoring and Analysis in Honeypot Environments." *Cyber Defense Journal*, 7(3), 45-60.
- Symantec Threat Report. (2021). "Emerging Threats and the Role of Honeypots in Cyber Defense." *Symantec Security Report*.