Net – Hunt Web Based Honeypot Network Intrusion Detection System

Prof. Suruchi Deshmukh, Pranav Chaudhari, Soham Chaukaskar, Prathamesh Kadam
School Of Computing, MIT ADT University, Pune, India
School Of Computing, MIT ADT University, Pune, India
School Of Computing, MIT ADT University, Pune, India
School Of Computing, MIT ADT University, Pune, India

---***---

Abstract - Net-Hunt is lightweight, multi-service honeypot based Network Intrusion Detection System (NIDS) designed to capture high-fidelity attacker telemetry. It emulates commonly targeted network service SSH, Telnet, HTTP, and FTP on an isolated virtual machine, records every interaction in time zone aware JSON Lines (JSONL) format, and maintains per-service, per-day log files to simplify downstream ingestion and archival. Each event record includes a UTC timestamp, source address and port, service identifier, event type (for example: connection, username tried, password tried, request, command), and a truncated raw payload for safe analysis. This project documents safe deployment practices, a repeatable testing methodology, and an end-to-end pipeline from raw capture to analyst consumption. Deliverable include the Python honeypot implementation, a system d service for reliable operation, demo scripts for attack simulation, log shipping examples, and a final report with measured test results and example dashboards.

Key Words: Honeypot, Network Intrusion Detection System, Cybersecurity, Threat Intelligence, OpenSearch.

1.INTRODUCTION

Cybersecurity incidents are growing rapidly as organizations increasingly depend on interconnected systems. Traditional defense mechanisms such as firewalls, signatures based Intrusion Detection Systems (IDS), and antivirus solutions often struggle to detect novel or zero-day attacks. They frequently generate excessive alerts, causing alert fatigue among analysts and delaying genuine threat responses. A honeypot is a purposely vulnerable or simulated system designed to lure attackers. Because no legitimate user should interact with a honeypot, every event it records is a high confidence indicator of malicious intent. By analyzing these interactions, security teams can study attacker behavior, gather Indicators of Compromise (IOCs), and strengthen their defensive posture. Modern deployments integrate honeypots into centralized analytic pipelines, using tools such as File beat and Open Search to visualize attack patterns in real time. Such systems form the basis of threat intelligence platforms, providing insight into attacker tactics, techniques, and procedures (TTPs). The present study presents a web-based honeypot NIDS, Net hunt that combines various decoy services, centralized logging and web dashboard to visualize real-time alerts. The purpose is to show how the controlled honeypot settings can give high fidelity threat information and be completely isolated out of operational networks.

II. LITERATURE SURVEY

Over the past two decades, researchers have extensively explored the application of honeypots in network defense. 1) Studiawan, H., Djanali, S., & Pratomo, B. A. (2016). "Graph-Based Forensic Analysis of Web Honeypot." Journal of Telecommunications and Information Technology, 64(2), 60–65.This study developed a web honeypot that collects attacker interactions and performs forensic log analysis using graph-based clustering. The approach enables visualization of attacker behavior and categorization of intrusion patterns, demonstrating that honeypots can provide valuable behavioral insights for digital forensics. 2) Franco, J., Aris, A., Canberk, B., & Uluagac, A. S. (2021). "A Survey of Honeypots and Honey nets for IoT, IIoT, and CPS." arXiv:2108.02287.The authors presented an extensive survey of honeypot and honey net research in Internet-of Things (IoT) and Cyber-Physical Systems (CPS). The paper introduced a taxonomy for classifying honeypots based on purpose, interaction level, and architecture, and highlighted the growing role of honeypots in protecting distributed, resource constrained environments. 3) Gajjar, H., & Malek, Z. (2024)."Network Intrusion Detection System using Honeypot in Cloud Environment." International Journal of Intelligent Systems and Applications in Engineering, 12(3), 3859–3863.This paper proposed the integration of honeypots into cloud-based intrusion detection systems. It implemented a Linux-based honeypot on cloud servers to detect unauthorized access attempts and discussed security and legal issues associated with honeypot deployment on third-party infrastructure, aligning closely with the goals of Net-Hunt's AWS-ready design. 4) Fan, W., Du, Z., Smith-Creasey, M., & Fernández, D. (2024). "Honey-doc: An Efficient Honeypot Architecture Enabling All-Round Design." arXiv:2402.06516.The researchers presented Honey-doc, a modular honeypot architecture comprising Decoy, Captor, and Orchestrate components. Using software-defined networking (SDN), the system dynamically deploys decoys across a network to capture diverse attack traffic efficiently. This work demonstrates advanced automation techniques that inspire the future scalability of Net hunt.

I. PROPOSED SYSTEM ARCHITECTURE

Net-hunt consists of three primary layers:

1) Sensor Layer:
Emulates decoy services such as SSH, Telnet, FTP, and HTTP. Each service records all incoming connections, commands, and payloads in JSON format.

2) Backend Processing Layer:

Handles log ingestion, enrichment (GeoIP, ASN), and correlation. It processes logs into a searchable database using OpenSearch or Elasticsearch.

3) Web Dashboard Layer:
Provides a real-time interface for security analysts to monitor alerts, attacker geolocation, and attack patterns.

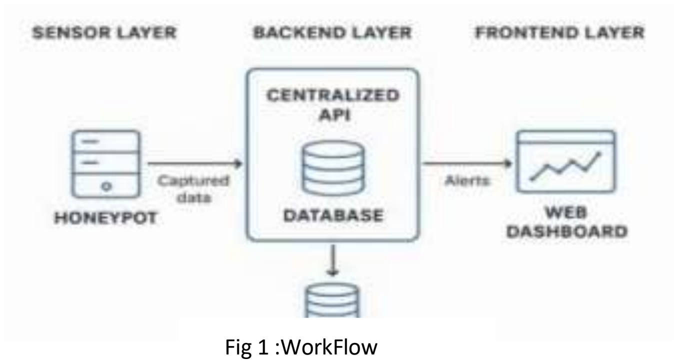

Fig 1 :WorkFlow

II. METHODOLOGY

The Net-hunt Web-Based Honeypot Network Intrusion Detection System is developed by the Waterfall Model that is a sequential process of software development whereby one step is not completed before another. This model will provide a well-structured and systematic approach which will suit the research and implementation needs of the project. All of the stages will be involved in the working process of building, testing, and validating the multi-service honeypot and its analysis environment.

A) Requirement Analysis:
The initial step entailed the collection of both non-functional and functional requirements necessary in the development of a sound and secure honeypot system.

The functional requirements covered the identification of the types of network services that would be emulated (SSH, Telnet, HTTP, and FTP), the way the logs were going to be collected, and the way the data would be processed and visualized.

The non-functional requirements were concerned with the isolation of the system, its scalability, low consumption of resources and safe deployment to a controlled virtual environment.

B) Design Phase : During the design stage the system architecture was planned in detail in three major layers:

1. Sensor Layer: emulates a variety of network services (SSH, Telnet, HTTP and FTP) that are available to attackers. All of the services operate on their own thread or asynchronous socket to capture all the inputs and store them in organized log files in the JavaScript log files in the format of JSONL.
2. Back-end Processing Layer: It deals with the parsing and enriching of raw logs. It adds other metadata including timestamps, attacker IP addresses and Geo-IP location information.
3. Web Dashboard Layer: This is a web layer developed in HTML, CSS, and JavaScript with builtin Open Search Dashboards to show analytic, numbers of attacks, and geolocation maps.

Data flow diagrams (DFDs) and architecture diagrams were designed to show the flow of data between these layers. The layer-to-layer communication was determined with the help of the secure protocols so as to provide isolation between the honeypot and dashboard components.

C) Implementation Phase:
Implementation stage entailed real development of the system. The honeypot was coded in Python 3, with the socket, asyncio and logging modules, in order to simulate actual network service interactions.

All services (e.g. SSH, HTTP) are started on a given port and

a) Connection attempts
b) Login credentials tried
c) HTTP requests
d) Commands or payloads sent

The logs are then created in the JSONL (JSON Lines) format to be easily parsing and subsequently enriching.

capture attacker traffic like:

D) Testing Phase:
A special attacker virtual machine (VM) was used to simulate the real-life intrusion scenarios and test it. The system of the honeypot and attacker was related by a host-only network, and no external contact with the Internet was provided.

E) Deployment Phase:
Once tested successfully, the honeypot was set up in a secure environment, which is not connected to the network, and is hosted on Kali Linux VM.

a) In order to be persistent and manageable:
b) The python honeypot script has been set up as an initial background service with systemd.
c) Logs were kept under /opt/net-hunt/logs/ and were automatic rotated on a daily basis.
d) File beat was set to ship these logs to an analysis VM Open Search backend.

III. RESULT

Net-hunt successfully logged all simulated attack attempts and generated structured telemetry for each event type.

a) Telnet and SSH received the highest number of probes.
b) Password guessing and HTTP reconnaissance were common patterns.
c) Enriched data revealed attack origins from multiple geographic regions.

Visual dashboards displayed real-time attack maps, event timelines, and top attacker IP.Results validate that a lightweight honeypot can effectively identify malicious activity with minimal false positives and operational cost.

```
[illegible]
```

Fig 2: Script Testing

```
[illegible]
```

Fig 3: User Log

```
/opt/nethunt/test_results_192_168_182_128_20251105_183813
01_nmap.txt  02_ssh_banner.txt  03_telnet_bruteforce.txt  04_http_scan.txt  05_ftp_test.txt  06_collect_logs.txt
```

Fig 4: Log File

3. CONCLUSIONS

The Net-hunt Honeypot Network Intrusion Detection System (NIDS) was developed to demonstrate how deception-based security mechanisms can proactively identify and analyze cyberattacks in a controlled environment. The system successfully emulated multiple common network services —

SSH, Telnet, HTTP, and FTP — and effectively recorded all malicious interactions from simulated attacker systems. The testing phase confirmed that the honeypot could accurately capture, store, and visualize intrusion attempts while maintaining full network isolation and operational stability.

The integration of File beat and Open-search allowed for real-time log forwarding, indexing, and visualization, turning raw network data into meaningful threat intelligence. The project validated that honeypots, when deployed safely, can serve as a low-cost, high- value intrusion detection mechanism capable of detecting unknown or zero-day threats that traditional systems often overlook. Overall, Net-hunt demonstrated the power of combining lightweight design, open-source tools, and virtualization to build a functional, saleable, and secure threat analysis environment.

1822, pp. 33–39, 2016.

4 .REFERENCES

[1] L. Spitzner, *Honeypots: Tracking Hackers*, Addison-Wesley, 2003.

[2] P. Kaur and H. Singh, "Hybrid Intrusion Detection System using Honeypots and Anomaly Detection,"*International Journal of Computer Applications*, vol. 165, no. 6, pp. 12–17, 2017.

[3] R. Tiwari, S. Kumar, and R. Jain, "Cloud-Based Honeypot Framework for IoT Threat Intelligence," *IEEE Access*, vol. 8, pp. 127–135, 2020.

[4] S. Kashyap, R. Sharma, and V. Kumar, "Evaluating Dockerized Honeypots for Scalable Intelligence Collection," *Journal of Network Security*, vol. 14, no. 2,
pp. 45–54, 2022.

[5] M. Rahman, S. Akhtar, and K. Hossain, "Integration of Honeypots with SIEM Platforms for Real-Time Threat Correlation," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 334–347, 2024.

[6] J. Sequeira and M. George, "A Survey on Honeypots for Network Intrusion Detection," *Procedia Computer Science*, vol. 132, pp. 1218–1227, 2018.

[7] A. Kulkarni, S. Patel, and D. Singh, "Building a Scalable Multi-Service Honeypot using Open Source Tools," *International Journal of Advanced Research in Computer Science*, vol. 12, no. 4, pp. 59–67, 2021.

[8] R. Mukkamala and A. Sung, "Feature Selection for Intrusion Detection with Neural Networks and Support Vector Machines," *Transportation Research Record*, vol.