© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718981

Honeyshield - Real-Time Honeypot Monitoring & Alert System

ANUJ ANTIL[1], SHIVAM KUMAR[2], SHAKTI ARORA[3]
[1]*Student Department of Computer Science and Engineering (Cybersecurity) Kurukshetra University, Kurukshetra, India Technology Education and Research Integrated Institution Kurukshetra University Kurukshetra, India*
[2]*Student Department of Computer Science and Engineering (Cybersecurity), Kurukshetra University, Kurukshetra, India Technology Education and Research Integrated Institution*
[3]*Professor Department of Computer Science and Engineering (Cyber Security) Kurukshetra University, Kurukshetra, India Panipat Institute of Engineering and Technology Samalkha, Panipat, India*

Abstract- ***HoneyShield serves as a lightweight system for intrusion detection and monitoring. It relies on honeypots to spot malicious activities. The setup simulates a vulnerable environment and captures what attackers do in real time. Those deceptive endpoints look real enough to draw in attackers. They end up doing things like reconnaissance or trying to log in or run commands. Every move gets logged as a structured event. Then a dynamic model scores the risk and sorts it into low, medium, or high categories. Administrators get a boost in awareness from the real-time alerts. Those pop up as visual warnings whenever high-risk stuff shows up. The frontend feels intuitive for keeping an eye on sessions. You can check timelines of events or download the full chains of interactions. It uses a modular build with React up front and Python in the back. Simplicity and clear analysis come first in the design. Tests in experiments prove it catches odd patterns reliably. Things like repeated failed logins or weird file grabs or commands without permission stand out. Overall, HoneyShield shows how these honeypots traps help with early detection in today's security setups.***

Categories and Subject Descriptors
C.2.0 — Computer-Communication Networks: Security and protection
C.2.3 — Network Operations: Network monitoring, Network management
D.4.6 — Operating Systems: Security and Protection – Invasive software, Access controls
K.6.5 — Management of Computing and Information Systems: Security and protection
K.4.4 — Computers and Society: Electronic surveillance, Security
I.5.4 — Pattern Recognition: Applications – Anomaly detection, Behavior analysis

General Terms ***Security, Design, Reliability, Experimentation, Measurement, Performance, Verification***

Keywords- ***Honeypot, Intrusion Detection System (IDS), Cybersecurity, Deception Technology, Threat Monitoring, Risk Scoring, Attack Behavior Analysis, Network Security, Event Logging, Real-Time Alerting, HoneyShield.***

I. INTRODUCTION

The swift development of digital infrastructure has greatly amplified not just the prevalence but the complexity of cyber-attacks on individuals, organizations, and critical services. Today's attackers utilize sophisticated processes like brute-force authentication, credential harvesting, conspicuous escalation of privilege, and automated scanning to exploit flaws in system defenses.

The traditional suite of security software, including firewalls, antivirus programs, and signature-based intrusion-detection systems, fail to recognize new or developing threats primarily because these programs operate on predefined rules that are decided and known attack signatures. This shortcoming makes it clear that we need to move toward proactive, behavior-centric security approaches that can capture new or unknown patterns of attacks.

Honeypots have evolved into a successful deception-based approach to studying and examining the

behavior of attackers, as well as to providing better intrusion detection. A honeypot is a decoy system, created to look vulnerable and lure attackers to use or interact with it, while keeping the production network separate from exposure.

In this way, security analysts can observe attacker interactions with the decoy honeypot, as well as develop adversary methodologies, tools, and intent. However, current honeypot products generate enormous amounts of unstructured logs, have no real time detection capabilities, and provide inadequate visualization mechanisms; this makes using them both laborious and impractical to implement effectively.

HoneyShield, the system proposed in this study, is designed to overcome these limitations by integrating the use of honeypots and a risk-based intrusion detection and real-time monitoring system. HoneyShield captures attacker behaviors within a controlled environment and stores it as structured event chains for forensic analysis. The HoneyShield system also uses a dynamic risk-scoring mechanism on user actions, such as repeated failed login attempts,

unauthorized file access attempts, and suspicious command attempts, to assess the possible severity of an intruder's attempted intrusion. Once a high-risk action has been detected, the HoneyShield system will produce an immediate alert on the security analysts screen, allowing for fast knowledge and action.

As a part of the usability features, HoneyShield also integrates both an advanced React-based dashboard to visualize active user sessions, event timeline, and risk levels, allowing the user to easily and quickly understand a potential intrusion. The HoneyShield system does not require a user to manually parse through log files to get useful information, but can get actionable information instantaneously through the REST API and structure back-end in Python, increasing usability in academic research setting and as a system for production use.

In general, HoneyShield shows how you can combine deception-driven monitoring, risk-based analysis, and interactive visualization to produce an effective intrusion detection and attacker behavior observation system, as well as continue to advance the development of early warning cyber security systems.

II. TYPES OF HONEYPOTS

Honeypots can be classified in several ways, depending on the interaction level, deployment purpose, and application context. To begin, honeypots may be classified, based on the type of interaction, into three groups: low-interaction honeypots provide limited simulated services; medium-interaction honeypots provide limited services that allow for some type of attacker behavior capture; and, high-interaction honeypots provide a realistic experience for the attacker, allows real systems to be monitored, and in return, provides maximum data capture. However, they also carry a greater risk of losing the system being attacked, some maintenance burden, and can be more operationally expensive.

The purpose of the honeypot deployment introduces the differentiation of a production honeypot versus a research honeypot. A production honeypot is deployed within an organization's networked system to better detect intrusion and deflect the attacker; whereas, research honeypots are deployed for use in researching the vulnerabilities associated with various / emerging attack patterns, malware behavior, and/or adversarial techniques.

The other alternative apart from a production versus research honeypot, is simply a deployment strategy. The pure honeypot can also be adopted as a method of deployment. A pure honeypot is a fully observable system with extensive monitoring; additionally, a distributed honeypot can provide a benign means of attack deterrence, as it spreads out the simulators across multiple network points, rather than a centralized single location; not to be confused with a honey farm, a honey farm is a central observation facility, or analysis, of distributed sensors.

A honeypot can also be classified by application domains, such as web honeypots, malware honeypots, currency honeypots, database honeypots, e-mail honeypots, and ICS/SCDA honeypots; they all

absorb types of attacks. Thus, honeypots by definition can serve as effective tools for both intrusion detection, threat analysis and cyber security research.

III. RESEARCH HONEYPOTS

Research honeypots are sophisticated systems used to analyze an attacker's behavior, new threats, and new methods of intrusion. They are different from production honeypots in that production honeypots are concerned about operational security and advance warnings about threats, whereas research honeypots generate huge amounts of data to advance cybersecurity research, ease malware analysis, and build threat intelligence.

Research honeypots typically involve high-interaction or medium-interaction environments where attackers can fully interact with true services, operating systems, and applications. Researchers glean information from attacks that include new zero-day exploits and automated attack tool use, C2 exploit behavior, privilege escalation techniques, and lateral movement stems and techniques.

Research honeypots are often found in cyber labs, university research sites, and security organizations, as their use often puts great demands on monitoring, logging, and isolation, to emplace compromised honeypots that cannot be used to attack other external systems.

Research honeypots populations are actively looking to engage sophisticated attackers, to develop datasets that we can leverage for academic studies, their use in signature creation for IDS tools, probes to advance or develop anomaly-detect algorithms, and redefine defensive measures and processes.

Research honeypots serve as a tool to create knowledge rather than as a means to protect with intention, which makes them a unique element in advancing modern cybersecurity research.

IV. PRODUCTION HONEYPOT

Production honeypots are security tools used in real active networks to detect malicious activity and to decrease the chances of intrusion. Production honeypots are designed for practical detection of threats, early-warning alarms, and to divert attackers from sensitive resources. Production honeypots simulate weak services or systems with low to medium-levels of interaction that allow security teams to notice attacks without putting the organization at undue risk.

With their lightweight and manageable capabilities, production honeypots can fit behind enterprise equipment in a DMZ, cloud environment, or internal network. Production honeypots signal high-value security alerts with minimal false positive noise because virtually any interaction with a honeypot is suspicious by default. Production honeypots help security teams notice port-scanning behaviors, dictionary attacks, malware, and unauthorized access attempts in real-time.

Although production honeypots provide less detail about behavioral data than research honeypots, their understanding is actionable intelligence that enables securing IDS detection, incident response, and improves overall security posture of the organization.

V. LITERATURE REVIEW

During a long duration of time, deception and honeypot technologies have given a unique visibility to the actions of the attacker as a complement to network defense measures and is the only technology that fulfills visibility and data needs that are not met with conventional perimeter controls and signature-based detectors.

Early conceptual work on honeypots concerned the definition of honeypots as network decoys which are based upon simulated vulnerabilities, to lure adversaries and facilitate observation of actors to be investigated in the future. Commercial and open-source initiatives like Honeyd were among the pioneers of offering lightweight implementation of network services and successfully implementing low-interaction honeypots at scale with a minimal overhead.

Along with the low-interaction honeypots, more realistic settings were increasingly created to

generate more attacker tactics, techniques and procedures (TTPs) by the introduction of systems that reflected utilization of medium- and high-interaction such as Kippa, Cowrie, Dionaea and service specific honeypots. Various properties of the usefulness of honeypots have been investigated using research methodologies.

To begin with, empirical evidence has been presented by research that honeypots are useful in enticing automated scanning, exploits attempt, credential theft, and malware propagation activity that generate datasets to facilitate forensic examination and signatures. Second, research has investigated trade-offs of coverage, manageability and fidelity of the interactions of interest through studies of honeypot deployment techniques (distributed deployment vs.

centralized/honey-farm). Distributed deployments enhance the variety of attacks that are witnessed and heighten storage and coordination requirements. With increased complexity of deployment, the analysis that is centralized often allows in-depth correlation of attacks to come up with patterns of attacks more relevant to them.

The lack of communication between the gathering of raw logs and the transformation of raw logs into actionable intelligence is a persistent problem in the field. A lot of honeypot authors generate huge volumes of raw log data, which is unstructured or semi-structured, and unless automated analytics are used or visualized in an intuitive way, these logs are likely to be of limited value.

Initial efforts were based on the log normalization, feature extraction and automated classification methods (rule-based and machine-learning based), to distinguish between benign background noise and actual malicious activity. Signature based correlation is very effective in identifying known threats, but very weak in the areas of zero-day attacks, polymorphic attacks, but anomaly based and behavior-based correlations give a direction toward the detection of unknown patterns, but the correlations also have a high level of false positive.

A discrete, yet similar, body of literature makes reference to real-time alerting and human in-the-loop maneuvers. A number of papers restate that alerts that display information in a time-sensitive manner, display information succinctly, (i.e. dashboards, aggregate risk scores and prioritized alerts, etc.) are major factors that make the honeypots useful to operational security teams.

The ability to enable the automated escalation, quarantine, or additional forensic capture were prescribed as integration with security information and event management (SIEM) tools or orchestration. In the end, however, experience on the ground and in practice also indicate that work breaking into the IT infrastructures will always have a latency around its detection, events with low priority will saturate the attention of the analysts, and little context to enable them to prioritize incidences quickly.

The computationally wise, session or user level risk scoring models have been developed in the IDS and the fraud detection literature. Heuristics (such as whether there were repeated (successful or unsuccessful) authentication attempts, access to sensitive files, or execution of suspicious commands to files of interest) that are lightweight are also attractive in resource-constrained systems (they are more comprehensible as well as computationally cheap).

Although machine learning and statistical, anomaly detectors may also offer finer grained discrimination, but there is a growing need of labeled training data and parameter threshold tuning by a user to prevent instability during live operational environments.

Finally, there are operation constraints mentioned in the literature such as operational constraints of high-interaction honeypots that need further isolation and monitoring to ensure they are not launched/command-and-control (C2) additional operation event; or capture an attackers payload provides privacy and legal consideration to the data capture, and there are usability related to the presentation of intelligence/dashboards that provide batched/summarized intelligence to an analyst, as opposed to raw logs.

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718981

VI. METHODOLOGY

The process of creating HoneyShield consists of four main structures: (1) honeypot environment configuration, (2) acquisition and event chaining of data, (3) risk-based analysis of behavior, and (4) data visualization, alerts, and real-time. Each of the four structures is specifically designed to deliver accuracy in capturing adversarial behavior, provide useful insight into operation intelligence, and maintain a lightweight footprint for research and practical use.

Honeypot Environment Configuration

HoneyShield utilizes a controlled, isolated, service-emulating environment that renders the environment attractive enough for unauthorized engagement. It implements a medium-interaction SSH honeypot which emulates aspects of a legitimate server interface while maintaining strong containment practices to avoid lateral movement.

The honeypot accepts simulated credentials, logs all attack activities and commands executed, and if unauthorized access is attempted, will record attempts to access restricted files or system configurations. The environment is structured to appear just real enough, including actual banners, filesystem structure, and responses to conditions of engagement, to entice the attack while ensuring that nothing is able to be compromised or breached within the network system.

Event Capture and Tamper-Evident Logging

All interactions occurring in the honeypot are recorded as individual events: timestamps, commands attempted, privilege escalation attempts, files accessed, and so on. HoneyShield has a chained-hash logging design in which each event stores a cryptographic hash of the previous event, allowing for tamper-evident integrity by ensuring that any unauthorized changes to the log will be detected.

Each event is sent to the backend API, which stores the logs in a structured database to allow for reconstruction of attack sessions and later forensic analysis.

Session Reconstruction and Data Normalization

Captured events are grouped into attacker sessions, representing continuous activity originated by a common source. The backend normalizes the sessions in order to remove redundant noise, standardize commands, and allow for extraction of key behavioral indicators such as command category, command frequency, and command severity. The backend also closes sessions with long idle gaps to ensure accurate segmentation of attacker sessions and provide “clean” data for analysis.

VII. RISK SCORING MODEL

HoneyShield utilizes a streamlined, explainable heuristics concept-based model for risk scoring. Each action collected is assigned a score based on the action's degree of threat; repeated successful authentication failures, category commands, file system probing, privilege escalation attempts, and malware-based patterns each have a score associated with them. The scores determine a risk value during the entire session. The risk value is used to place the session in Low, Medium, High thresholds.

These processes are simpler, and less computationally complex than more complicated machine-learning models. This approach is ideal for real, or near real-time deployment and justifies use during academic assessments since it is intuitive when assigning risk.

Real-Time Dashboard and Alerting System

The frontend with React and Vite provides an reactive graphical monitoring interface for active sessions and sessions within a time frame. The monitoring dashboard continuously polls the backend application for events and updates the score for each action in real time.

If an action within a session reaches the threshold for risk, a high-priority visual alert is generated for the analyst to be notified without delay. Metadata cards showcase metadata about the session, including, incidents of origin IP, the chain of events, command history, and a timestamp. This arrangement presents the analyst with an overall present view of the current situation and with risk data for immediate intervention.

© JUN 2026 | IRE Journals | Volume 9 Issue 12 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I12-1718981

System Deployment and Testing
The system is deployed using a modular architecture: the honeypot and backend run on python/FastAPI, while the frontend can run independently and must communicate via RESTful APIs. Testing consists of creating controlled SSH-based attacks, executing reconnaissance commands, attempting privilege escalation, and simulating malware activity. The controlled attack scenarios tests the integrity of logging, the constancy of risk scoring, the accuracy of alerts, and the responsiveness of the front-end interface.

VIII. RESULTS

The HoneyShield system was developed by testing it under controlled attack conditions and regular usage tests.
to assess the accuracy of detection, log reliability and alert responsiveness. The tests indicate that the system is effective in identifying malicious behavior and, at the same time, providing sound.

real-time performance monitoring. System Under Attack Performance.
Some simulated interactions with attackers were performed, such as: brute-force login. attempt, directory-probe, command-enumerate and download/access.

confidential files.
It was observed that the honeypot received 100 percent of the attack commands injected which depicts complete.

transparency into counteractivity. The order of events was preserved by the chained-hash logging of events. authenticity of events that enabled the correct re-construction of the attacker sessions without any lapses in their action.

Accuracy of Risk Scoring
The HoneyShield heuristic risk-scoring model identified the risk categories of the sessions correctly. The system validated and was able to identify:

Easy activity (non-complicated scans).
Medium risk activity (accessing files, uploading suspicious files)
High risk (Actions involving more than one attempt at intrusion, or more than one attempt attempted and unsuccessful)
The scoring of the 30 controlled attacks was as follows:

96.6% correct identification of high-risk sessions.
Behavior of brute-force-attack based behavior was successfully detected 100% of the time. 0 misdeeds on evil command chains.

These findings suggest that despite a simple explainable rule, HoneyShield did not do badly. to monitor intrusion in real-time.
Alerts and real time monitoring.

The React dashboard was able to update the session data at the rate of 2s. In testing for high-risk activity:

- There was an activation of alerts in 1200ms to 200ms of malicious activity.
- The pop-up alert was only seen when there was an attacker-event and this was always done by the pop-up alert.

appropriately.

- Once the risk-check was updated, no alerts were provided to normal user interaction. This proves that HoneyShield generates quick detection and it does not flood an. analyst having fake pop-up notifications.

IX. DISCUSSION

The results of the HoneyShield system thus prove the effectiveness of honeypot-based intrusion detection to capture real-time malicious behavior and analyze it across the broader organization. Specifically, attacker engagement, risk scoring, and real-time alerting are features of the system that were recorded very effectively.

This points to the fact that techniques of deception are still very relevant in contemporary cybersecurity. The tamper-evident chain hash logging increased the fairness and forensic reliability of the collected data by making event sequences impossible to be modified by an attacker without detection. Although the heuristic risk model accurately detected brute-force attempts and high-risk actions during testing, its

rule-based nature may constrain its efficacy and detection scope against capable or stealthier attacker strategies.

The UI provided meaningful session monitoring and alerts that helped analysts enhance their situational awareness, while overall the UI's value is determined by how well it is tuned to limit false alerts.

Limitations observed in this study naturally include a simplified honeypot environment, no collective network-level monitoring, and of course, the lack of automation as a response mechanism. Altogether, data indicates that HoneyShield provides an effective, lightweight, and practical way of carrying out intrusion monitoring while supporting improvements in the future, such as using a machine-learning detection method, hitherto the honeypot-based concepts.

X. CONCLUSION

HoneyShield is a best example of how honeypot technology, combine with real-time intrusion monitoring and risk-based alerting, can be used to enhance the security posture. Capturing attacker interactions in a controlled environment allows the system to provide the most valuable insight in attack behavior without compromising actual infrastructure.

Meanwhile, the chain-hashed logging mechanism enhances forensic integrity and ensures that all recorded events remain tamper-evident and trustworthy. In summary, HoneyShield giving a practical and accessible solution for early threat detection by combining a lightweight backend, interactive frontend dashboard, and clear risk-scoring model.

The system performs reliably to identify brute-force and suspicious activity patterns; however, its heuristic approach can still be improved to detect more sophisticated or evasive attacks. Overall, HoneyShield was able to effectively achieve its goal as a honeypot-based intrusion detection and monitoring system, providing a lot of operational value for defenders and a solid foundation for further research and improvements.

REFERENCES

[1] Mairh, A., Barik, D., Jena, D., & Verma, K. (2011). Honeypot in Network Security: A Survey. International Conference on Communication, Computing & Security (ICCCS '11).

[2] GeeksforGeeks. (n.d.). Honeypots in Cyber Security – Introduction, Types & Working. Retrieved from https://www.geeksforgeeks.org

[3] Denning, D. E. (1987). An Intrusion-Detection Model. IEEE Transactions on Software Engineering.