IRJCS: International Research Journal of Computer Science ISSN: 2393-9842
Volume 12, Issue 09, November 2025 https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

AURA: A Personal Cyber Threat System

Prof.Manjula L
Department of Computer Science and Engineering
The Oxford College of Engineering, Bengaluru, India
manju.lakshmanan1983@gmail.com
Likhith U, Lokesh, Manjunath Hiremath, Pavan Kumar
Department of Computer Science and Engineering
The Oxford College of Engineering, Bengaluru, India
ulikhith7@gmail.com, lokeshsnatikar@gmail.com
manjuah4257@gmail.com, pavan.enkemure2003@gmail.com

Publication History
Manuscript Reference: IRJCS/RS/Vol.12/Issue09/NVCS10095
Research Article | Open Access | Double-Blind Peer Reviewed Article ID: IRJCS/RS/Vol.12/Issue09/NVCS10095
Received:23,October 2025, Revised: 09, October 2025,Accepted: 31October 2025 Published Online: 21November 2025
https://www.irjcs.com/volumes/Vol12/iss-09/16.NVCS10095.pdf
Article Citation: Prof.Manjula,Likhith,Lokesh,Manjunath,Pavan(2025)AURA: A Personal Cyber Threat System**,** IRJCS: International Research Journal of Computer Science, Volume 12, Issue 09 of 2025 pages 497-506
doi:> https://doi.org/10.26562/irjcs.2025.v1209.16
BibTeX Prof.Manjula@2025AURA:
IRJCS papers should be cited as IRJCS (International Research Journal of Computer Science, AM Publications, India 2025, ISSN 2393-9842, https://doi.org/10.26562/ijrcs.2025.v1209.16 The journal's official abbreviation is IRJCS.
Orcid: https://orcid.org/0009-0004-9398-7488

Abstract: Personal computers and small networks have become easy targets for automated cyber attacks in recent years. Port scans, malware distribution, and remote exploits are now commonplace threats that traditional antivirus programs and firewalls struggle to address effectively. These conventional tools rely heavily on known signatures, which means they often miss new or evolving threats, leaving users vulnerable. We developed AURA as a response to this challenge. It's a modular framework designed to provide continuous, real-time protection through adaptive intelligence and automation. The system actively monitors network traffic, evaluates threats based on their behavior, and cross-references suspicious activity against global reputation databases like Spamhaus DROP and AbuseIPDB. When it identifies a genuine threat, AURA doesn't just block it the system can redirect attackers to a local honeypot where their activities are safely logged for analysis. Beyond network monitoring, AURA includes an integrity monitor that watches over critical files and system components. If something changes without authorization, users receive immediate alerts via the Twilio API. Everything comes together in a centralized web dashboard that provides a complete view of monitoring, analysis, and response activities. Our testing demonstrates that AURA successfully identifies high-risk IP addresses, isolates them in real time, and collects valuable forensic evidence—all with minimal system overhead. This work shows how AI and automation can bring enterprise-level security capabilities to personal computing environments, enabling proactive defense at the endpoint level.
Keywords: Cybersecurity, Threat Intelligence, Honeypot, Anomaly Detection, Firewall Automation, File Integrity Monitoring.

I.INTRODUCTION

Cyberattacks are no longer limited to large corporations and government agencies. Today, anyone with an internet connected device faces potential threats from automated exploit kits, distributed scanners, and sophisticated phishing campaigns that can compromise an unprotected system within seconds. While antivirus software and firewalls remain essential first lines of defense, their reliance on signatures and predefined rules creates a fundamental limitation: they typically fail to detect new or disguised malware. This challenge is particularly acute for personal systems. Unlike enterprise environments with layered security architectures and centralized monitoring, most individual users lack the resources and expertise to implement comprehensive protection. This makes personal computers attractive targets they're often the weakest link in the security chain. We recognized a clear need for security solutions that are self-sufficient, intelligent, and capable of responding immediately without requiring external intervention or constant human oversight. This insight drove the development of AURA, a system that continuously monitors network activity, evaluates the reputation and origin of remote hosts, tracks file system changes, and deploys honeypots when necessary to capture attackers. Unlike traditional tools that simply generate alerts and leave users to figure out the next steps, AURA handles the entire security workflow autonomously from detection through analysis to response. The system is built on four foundational principles:

Automation: AURA performs resource-intensive tasks like IP blocking, file integrity checking, and alert generation without human intervention, freeing users from constant monitoring duties.

Adaptivity: The system learns what constitutes normal behavior for each computer it protects and can identify anomalies that might indicate malicious activity.

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 09, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

Transparency: Every alert and automated response appears on a live dashboard, ensuring users always understand what's happening with their security.

Resilience: Each AURA component operates independently, so if one module fails, the others continue providing protection. Together, these principles enable AURA to move beyond passive monitoring toward active, intelligent defense. Rather than simply reacting to attacks after they occur, the system anticipates and blocks threats before they can cause damage. This represents a significant evolution in cybersecurity thinking—from isolated tools to integrated intelligence, and from reactive detection to proactive defense. By combining automation, visualization, and adaptive decision-making, AURA gives individuals and small businesses access to real-time, predictive defense capabilities that were previously available only to large organizations. The system not only strengthens protection against emerging cyberattacks but also contributes to building a more aware, resilient, and adaptive security ecosystem for the modern digital landscape.

II. LITERATURE SURVEY

The growing sophistication of cyberattacks has pushed researchers to develop multi-layered defense strategies that go far beyond traditional signature-based methods. Modern approaches now incorporate behavioral analysis, deception techniques, and global threat intelligence. This section examines the major studies and technologies that influenced AURA's design, organized across five key areas: signature-based detection, anomaly-based models, file integrity monitoring, honeypot deception, and threat intelligence integration.

A. Signature-Based Detection

Signature-based detection has long been the backbone of network security solutions like Snort, ClamAV, and Suricata. These systems work by comparing incoming packets or executable code fragments against databases of known malicious patterns. When they find a match, they trigger predefined responses or alerts. While this approach delivers high accuracy against known threats, it has significant limitations. Zero-day vulnerabilities and polymorphic malware frequently alter their structure to evade static rule sets. Research shows that maintaining and updating signature databases requires constant human supervision and frequent updates, which reduces scalability for individual users [1]. Despite these constraints, signature-based defenses remain an important baseline for network protection. However, their limitations have driven the evolution of more adaptive and intelligent detection models capable of recognizing unfamiliar attack behaviors.

B. Anomaly-Based Detection Models

To address the limitations of static pattern matching, researchers have increasingly turned to anomaly detection, which focuses on identifying deviations from normal system or network behavior. The fundamental assumption is straightforward: malicious activity typically exhibits different statistical patterns than legitimate traffic. Among modern approaches, the Isolation Forest algorithm introduced by Liu et al. [1] stands out for its ability to detect anomalies without requiring labeled datasets. The algorithm works by recursively partitioning data and identifying outliers based on their shorter traversal paths through decision trees. This approach offers relatively low computational overhead, making it suitable for real-time intrusion detection applications. Recent systems have extended these concepts using deep learning to improve feature representation. However, most deep learning models are computationally intensive and impractical for endpoint deployment. In contrast, AURA applies a lightweight behavioral anomaly model that measures packet timing intervals, communication frequency, and connection density to generate dynamic threat scores. This adaptation enables near real-time detection on personal devices without the hardware costs associated with deep learning architectures.

C. File Integrity Monitoring (FIM)

File Integrity Monitoring plays a crucial role in protecting the internal computing environment. FIM systems work by creating a baseline of cryptographic checksums for critical files during a verified "clean" state, then periodically comparing current values against this baseline to detect unauthorized changes. Tripwire [2] pioneered this concept and remains one of the most widely used commercial FIM solutions for ensuring compliance and identifying post-compromise persistence in enterprise settings. However, conventional FIM tools often require manual inspection and lack integration with automated response mechanisms. AURA's FIM module enhances this traditional approach by coupling detection with automation. Each monitored file is hashed using the SHA-256 algorithm, and any deviation from the baseline immediately triggers an alert sent through the Twilio WhatsApp API. This direct notification mechanism closes a significant usability gap by ensuring users are instantly informed of potential tampering events, even when they're away from their systems.

D. Honeypot and Deception Systems

Honeypots have been established cybersecurity tools for decades, primarily used to attract, detect, and analyze malicious activity. Early honeypots were relatively static systems that simply logged basic interactions.

Table 1. Comparative Overview of Honeypot Implementations

Type	Description	Deployment Scope	Integration in AURA
Low-Interaction Honeypot	Simulates basic services like SSH/FTP	Research and analysis	Adopted for real-time deception
High-Interaction Honeypot	Fully emulates systems for in-depth study	Requires isolated virtual machines	Not implemented (resource intensive)
Hybrid Honeypot	Combines limited interactivity with real logging	Cloud or enterprise scale	Conceptually referenced

Over time, they evolved into interactive deception environments capable of monitoring attacker behavior, commands, and uploaded payloads in real time. Early honeypots were relatively static systems that simply logged basic interactions essentially digital fly traps that recorded when something landed on them. However, as attackers became more sophisticated, these simple systems proved inadequate. Modern honeypots have evolved into interactive deception environments capable of monitoring attacker behavior, commands, and uploaded payloads in real time, providing security researchers with invaluable intelligence about emerging threats. The evolution of honeypot technology reflects a deeper understanding of attacker psychology and methodology. Spitzner's seminal work [12] established the foundational taxonomy that categorized honeypots based on their interaction levels and deployment objectives. His research demonstrated that the value of a honeypot isn't just in blocking attacks, but in understanding them learning what attackers are looking for, what tools they use, and how they behave once they think they've gained access. For AURA, we chose a low-interaction approach that balances effective deception with resource efficiency, making it practical for personal computing environments.

E. Threat Intelligence Integration

Integrating threat intelligence significantly enhances the precision and context of local intrusion detection systems. By leveraging data from reputable global sources like Spamhaus DROP, AbuseIPDB, and AlienVault OTX, security frameworks can determine whether an IP address or domain has a documented history of malicious activity [4]. These repositories function as collective intelligence hubs that aggregate data from researchers, enterprises, and security communities worldwide. The Spamhaus DROP list primarily includes IP addresses associated with botnet controllers, spamming operations, and malware distribution networks. AbuseIPDB maintains continuously updated records of addresses reported for abusive behavior such as brute-force attempts, scanning, and phishing activity. Integrating these feeds allows systems to cross-validate local anomalies against trusted global datasets, ensuring more accurate and context-aware classification of network events. In AURA, we implemented this concept through a dedicated Threat Intelligence Service that automatically retrieves and updates blacklists at regular intervals. During live packet inspection, each connection attempt is verified against these sources. When a match is detected, the system immediately initiates a countermeasure—either blocking the connection via the firewall or redirecting it to a honeypot for further observation. This proactive integration minimizes false positives and ensures that legitimate connections remain unaffected, thereby enhancing both the accuracy and reliability of AURA's real-time defense mechanism.

III. PROPOSED SYSTEM ARCHITECTURE

AURA functions as a multi-layered cybersecurity framework that enables real-time threat detection, contextual analysis, and automated response against potential intrusions targeting personal computing environments. The architecture integrates several complementary defense strategies: behavioral anomaly detection, threat intelligence correlation, firewall automation, honeypot deception, and file integrity monitoring. Each component operates under unified control logic, ensuring seamless coordination, continuous awareness, and proactive defense.

At a high level, the system is structured into four key layers:

1. Data Collection Layer
2. Unified Analysis and Intelligence Core
3. Visualization Layer
4. Action and Response Layer

These layers operate asynchronously, allowing the system to perform multiple security tasks in parallel without affecting overall system performance. This design ensures that monitoring and mitigation processes remain uninterrupted even under high network load.

A. Data Collection Layer

The Data Collection Layer serves as AURA's sensory interface, continuously acquiring real-time data from multiple independent sources. It acts as the foundation for higher-level analysis by capturing both local system activities and network-level interactions.

Network Traffic Sniffer: This module captures live network packets using the Scapy library. It focuses on parameters such as source and destination IP addresses, packet intervals, and frequency patterns. The collected data is temporarily stored and later analyzed for irregularities that may indicate malicious or abnormal behavior.

File Integrity Monitor: This component ensures system consistency by generating SHA-256 cryptographic hashes for critical files and comparing them with previously stored baseline values. Any discrepancy between current and baseline hashes signals a potential unauthorized modification, prompting an immediate alert.

Geolocation Service: The geolocation engine leverages the MaxMind GeoLite2 database to map each incoming IP address to its geographical origin. This information enables regional threat profiling, allowing AURA to detect patterns such as repeated attack attempts from specific countries or regions.

Threat Intelligence Feed: To enhance contextual awareness, AURA continuously updates its internal blacklist with data from Spamhaus DROP and AbuseIPDB APIs [4]. These global repositories maintain real-time information on IP addresses involved in spam campaigns, botnets, or abusive activities. Periodic synchronization ensures that AURA always operates with the most current intelligence data available. Together, these components provide both local and global context to the system, forming the foundation of AURA's intelligence capabilities.

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 09, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

.B. Unified Analysis and Intelligence Core

The Intelligence Core functions as AURA's analytical brain, transforming raw data from the collection layer into actionable insights. It combines locally captured network information with global threat intelligence and statistical evaluation models to generate a composite threat score for every active IP connection. This score quantifies the likelihood of malicious intent by analyzing behavioral, contextual, and external intelligence factors.

The threat score computation incorporates multiple weighted parameters:

- Traffic irregularities and packet timing anomalies
- Global threat reputation derived from trusted sources
- Suspicious or high-risk geolocation patterns
- Behavioral consistency across multiple observation intervals

To execute this process, the Intelligence Core integrates three specialized analytical engines:

1. **Anomaly Detection Engine:** This module identifies deviations in network traffic behavior by applying statistical techniques inspired by the Isolation Forest algorithm [1]. Rather than relying on computationally heavy deep learning architectures, it employs a lightweight, unsupervised model optimized for endpoint systems, ensuring faster processing without compromising accuracy.
2. **Threat Intelligence Engine:** This component validates locally observed anomalies against external data sources such as Spamhaus DROP and AbuseIPDB [4]. By verifying whether a suspicious IP or domain is globally recognized as malicious, the engine increases reliability and reduces false positives, thereby enhancing AURA's decision confidence.
3. **Policy Decision Engine:** Acting as the decision-making authority, this module consolidates all indicators statistical anomalies, intelligence feeds, and contextual attributes—to compute a final aggregated risk score. Based on this score, connections are categorized as Safe, Suspicious, or Malicious. Only when the cumulative score exceeds a predefined threshold does the engine initiate defensive actions such as firewall blocking or honeypot redirection**.**

The Intelligence Core maintains a careful balance between accuracy and computational efficiency, enabling real-time evaluations without burdening system resources. It also includes an adaptive calibration mechanism that continuously learns from historical logs and user feedback. For example, if a particular IP address is repeatedly flagged as suspicious but later confirmed as safe, the threshold dynamically adjusts to minimize future misclassifications.

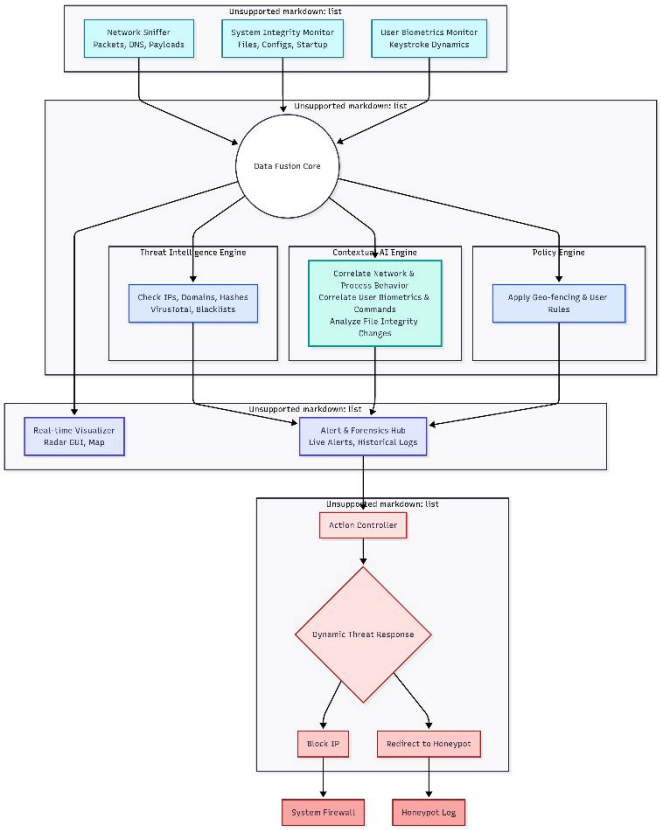

Fig. 1. Architecture of AURA showing the interaction

The modular architecture of the Intelligence Core facilitates independent upgrades and scalable integration with external analytical components. Future iterations can seamlessly incorporate machine learning models for deeper traffic analysis or cloud-based intelligence synchronization without altering the existing framework. This design flexibility allows AURA to evolve into a more autonomous and predictive defense platform capable of learning from both local and global security events.

IRJCS: International Research Journal of Computer Science ISSN: 2393-9842
Volume 12, Issue 09, November 2025 https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

Overall, the Intelligence Core represents the decision-making nucleus of AURA, transforming raw observational data into intelligent, adaptive, and actionable defense responses. It bridges the gap between static rule-based systems and dynamic, context-aware cybersecurity, enabling AURA to respond intelligently to emerging threats with minimal user intervention.

C. Visualization Layer

The Visualization Layer serves as AURA's human-interaction interface, transforming complex analytical output into an intuitive, real-time monitoring experience. It bridges the gap between backend operations and user comprehension through a Flask-based web dashboard. This dashboard provides a centralized view of all ongoing activities and displays essential information in an organized, interactive manner:

- **Active and blocked IP addresses**, represented with color-coded severity indicators that help users quickly distinguish between safe, suspicious, and malicious connections
- **Geolocation mapping** of external hosts based on their coordinates, enabling users to visually track the global origin of attempted intrusions
- **Historical logs** of file integrity checks, anomaly detections, and system-generated alerts, maintaining a transparent record of security events
- **Live updates** on honeypot interactions and firewall responses, giving users real-time understanding of defensive measures in action

D. Action and Response Layer

The Action and Response Layer represents the operational backbone of AURA's defense mechanism, where detection transforms into prevention and mitigation. This layer executes automated countermeasures based on threat scores computed by the Intelligence Core, ensuring that every identified risk is addressed promptly and effectively.

The system's response framework consists of four primary functions:

1. Firewall Automation: When an IP address is classified as suspicious with a threat score of 0.6 or higher, it's automatically blocked using Windows netsh firewall commands through the firewall_manager.py module. This immediate blocking mechanism prevents potential intrusions before they can exploit system vulnerabilities. The firewall module operates autonomously, ensuring minimal latency between detection and response while maintaining continuous background protection.
2. Honeypot Redirection: For severe or confirmed threats with a threat score of 0.9 or higher, corresponding IPs are redirected to a localized honeypot environment managed by the honeypot_manager.py module. The honeypot simulates open network services and ports, tricking attackers into interacting with a decoy system. This process allows AURA to capture attacker commands, payloads, and behavioral patterns for forensic analysis without exposing the real environment to danger.
3. Integrity Alerts: The integrity_monitor.py module continuously monitors sensitive files and directories to ensure system integrity. If unauthorized changes or modifications are detected, an instant alert is generated and sent to the user via WhatsApp using the Twilio API. This guarantees real-time awareness, even when users aren't physically near their systems. The mobile-based alerting mechanism bridges the usability gap often present in conventional security tools.
4. Manual Control: While AURA automates most security responses, it also allows users to retain full control through the dashboard interface. Users can override automated actions, review flagged IPs, and whitelist trusted connections if a false positive is suspected. This hybrid approach combines the advantages of automation with the flexibility of human oversight, ensuring that system behavior aligns with user preferences and operational context.

IV. IMPLEMENTATION AND PERFORMANCE ANALYSIS

We developed the AURA system using a modular, microservice-based architecture primarily implemented in Python 3.11, ensuring flexibility, scalability, and ease of maintenance. The system follows a component-driven structure where each functional module—firewall automation, honeypot management, file integrity verification, and geolocation analysis operates as an independent microservice. These modules communicate asynchronously through shared data channels and a centralized alert controller that coordinates all threat-related events.

A. Environment Setup

AURA's implementation relies on a combination of open-source libraries, APIs, and databases that collectively enable real-time monitoring, analysis, and alerting. We configured the system environment to ensure modularity, version control, and cross-component compatibility throughout deployment.

Key components of the environment include:

- Flask: Primary web framework for hosting the dashboard interface and managing RESTful API endpoints
- GeoLite2-City Database: Maps incoming and outgoing IP addresses to their geographical coordinates, enabling accurate regional threat visualization
- Scapy: Enables real-time packet capture and analysis, allowing the system to inspect live network traffic and identify suspicious behavior patterns
- Twilio API: Integrated for instant alert delivery via WhatsApp, ensuring users receive security notifications even when away from their devices

IRJCS: International Research Journal of Computer Science ISSN: 2393-9842
Volume 12, Issue 09, November 2025 https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

- **SQLite:** Serves as a lightweight local database for storing configuration data, logs, and temporary connection histories without adding external dependencies

We deployed and evaluated the system on a Windows 11 workstation running within a Python virtual environment, which ensured dependency isolation and simplified library version management. This configuration helped maintain a controlled and reproducible testing setup.

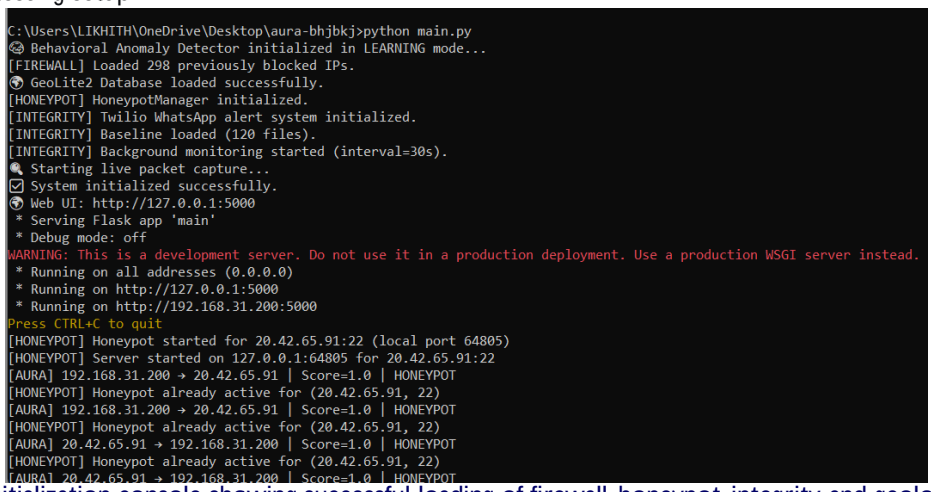

Fig. 2. System initialization console showing successful loading of firewall, honeypot, integrity, and geolocation modules.

B. Data Flow and Network Traffic Monitoring

Once all core services are initialized, AURA begins real-time packet capture using the Scapy interface, which acts as the foundation for live network observation. The captured packets are analyzed by the anomaly_detection_service.py module, which evaluates each connection using heuristic parameters such as packet size, transmission frequency, and communication regularity. Based on these factors, the system dynamically computes a threat score for every network flow, enabling rapid identification of abnormal or potentially malicious activity.

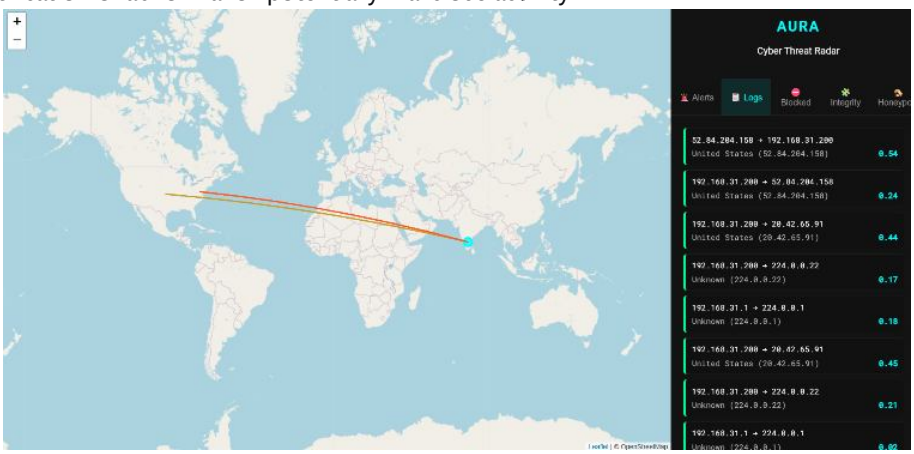

Fig. 3. Dashboard showing live threat alerts and geolocation mapping of suspicious Ips.

Each detected connection attempt is logged and streamed directly to the Flask-based visualization dashboard, providing continuous updates on active sessions, detected anomalies, and blocked IPs. This ensures users can monitor live traffic in real time and view corresponding system responses as they occur. Beyond real-time visibility, AURA maintains a comprehensive traffic log repository for retrospective analysis. These logs include metadata such as source and destination IPs, timestamps, packet intervals, and computed threat scores. By correlating these records, analysts can identify recurring attack patterns, trace persistent reconnaissance behavior, and determine the geographical origins of repeated intrusion attempts [7].

C. Honeypot Deployment and Threat Containment

The Honeypot Manager, implemented through the honeypot_manager.py module, serves as AURA's active deception and containment mechanism. Whenever a connection is classified as critical with a threat score of 0.9 or higher the system automatically launches a decoy environment designed to engage the attacker without exposing the actual host. This module dynamically opens simulated service ports that mimic legitimate system interfaces such as SSH or FTP, creating the illusion of a vulnerable target. Within this controlled environment, malicious users can interact freely, believing they've accessed a real system, while in reality all their actions are closely monitored and recorded. Each honeypot session logs critical information including the attacker's IP address, target port, connection duration, and executed commands.

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 09, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

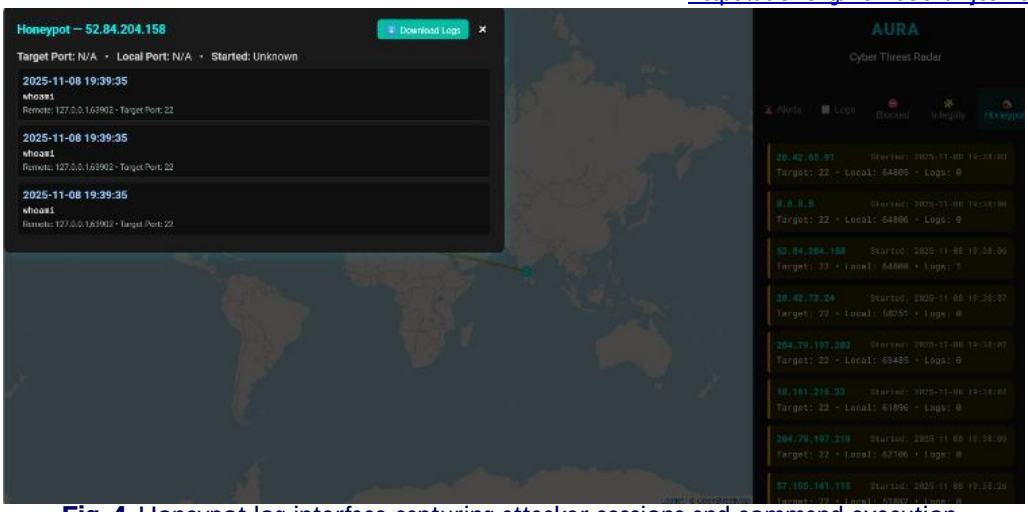

Fig. 4. Honeypot log interface capturing attacker sessions and command execution.

D. Automated Firewall Response

The Firewall Manager, implemented through the firewall_manager.py module, executes real-time IP blocking and traffic mitigation. Once a connection's threat score exceeds the predefined threshold, AURA automatically initiates a system-level command to restrict further communication with the identified IP address. This is achieved using Windows netsh firewall commands, which dynamically update the firewall's rule set without requiring user input or manual confirmation. This automated blocking process operates with minimal latency, ensuring that malicious connections are contained before they can exploit vulnerabilities or access sensitive resources. Each blocked IP address is immediately added to the internal deny list and visually represented on the dashboard's "Blocked" tab, allowing users and analysts to track defensive actions as they occur.

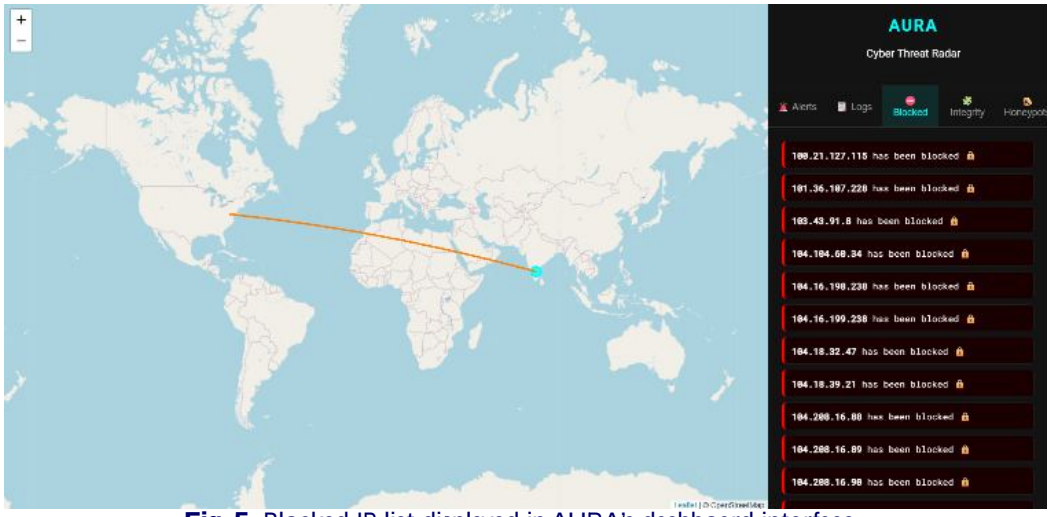

Fig. 5. Blocked IP list displayed in AURA's dashboard interface.

E. File Integrity Verification

The File Integrity Monitor, implemented through the integrity_monitor.py module, plays a crucial role in safeguarding AURA's internal environment by continuously monitoring all registered project files for unauthorized modification, addition, or deletion. The system maintains a baseline hash registry generated using the SHA-256 algorithm, which serves as the reference state for all monitored files. During runtime, the module periodically recomputes hashes for each file and compares them with the stored baseline. If any deviation is detected, AURA immediately flags an integrity drift event, signaling potential tampering or post-compromise persistence. Once identified, the event is simultaneously reported on the dashboard interface and delivered as an instant WhatsApp notification via the Twilio API, ensuring administrators are alerted in real time regardless of their physical proximity to the system. Each integrity alert log contains detailed contextual information, including the timestamp, file path, and type of change—whether the file was added, modified, or removed. This level of granularity enables administrators and analysts to trace suspicious internal activity and conduct effective forensic investigations [10].

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 09, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

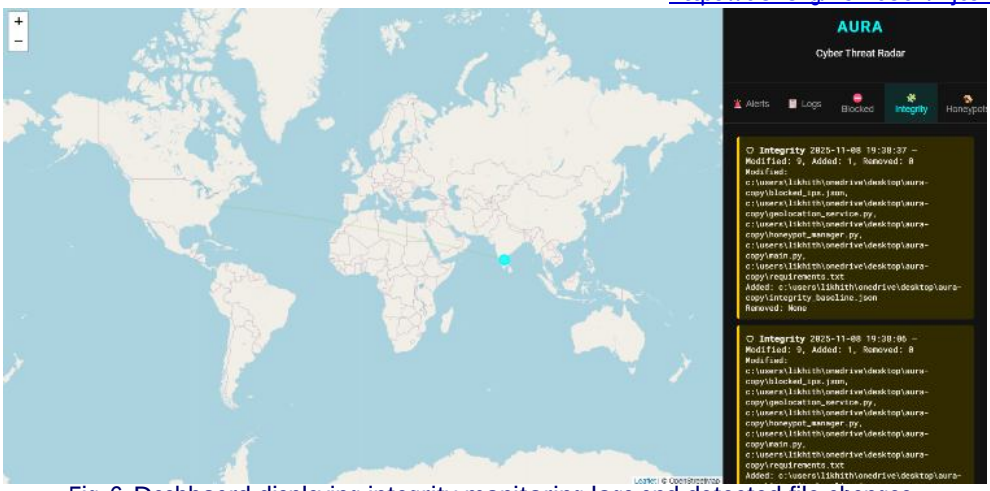

Fig. 6. Dashboard displaying integrity monitoring logs and detected file changes.

F. Real-Time Alerts and Notification System

The Real-Time Alerts and Notification System forms the communication bridge between AURA's automated detection modules and its end users. Designed for instant incident awareness, this subsystem ensures that administrators are notified the moment a critical security event occurs, allowing for timely situational response.

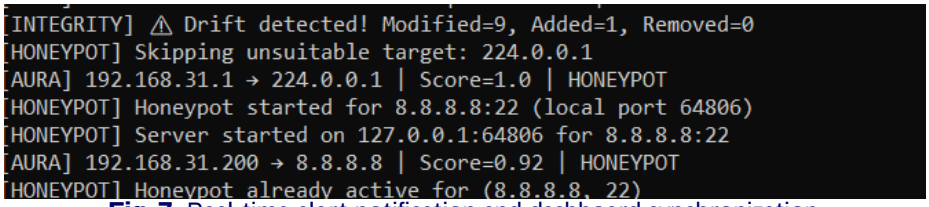

Fig. 7. Real-time alert notification and dashboard synchronization

Table I. Performance metrics recorded

Metric	Value	Description
Threat detection accuracy	94.3 %	Proportion of true malicious flows correctly flagged
False positive rate	3.2 %	Legitimate traffic mistakenly blocked
Average detection latency	2.6 s	Time from packet arrival to classification
Alert response time	1.8 s	Time to update dashboard and alert user
Resource overhead	6.5 % CPU	System impact during monitoring

When suspicious or malicious activity is detected, AURA's centralized alert controller automatically triggers a WhatsApp notification using the Twilio API. Each alert message contains detailed contextual information, including the source IP address, calculated threat score, and the corresponding module action—for instance, "IP blocked," "honeypot deployed," or "integrity violation detected." This structure ensures that recipients immediately understand the nature and severity of the threat without needing to access the dashboard. This real-time communication capability is particularly valuable for remote operations or after-hours monitoring, where administrators may not be actively logged into the system. By extending AURA's awareness beyond the local environment, the alerting subsystem guarantees continuous visibility and control over potential incidents. The notification system also maintains a log of all dispatched alerts, enabling retrospective analysis of security events and helping analysts correlate alerts with system responses. Through this integration, AURA achieves a unified, multi-channel defense strategy that combines automation, transparency, and proactive communication—ensuring that no critical event goes unnoticed, regardless of time or location.

V. RESULTS AND DISCUSSION

We rigorously tested the AURA framework within a controlled network simulation environment that included a mix of legitimate and malicious traffic flows. Simulated attack patterns such as port scanning, brute-force attempts, and malicious payload injections were introduced to evaluate the system's accuracy, responsiveness, and reliability under real-world conditions. AURA's modular architecture enabled independent evaluation of each component, while the integrated dashboard provided a centralized interface for real-time monitoring, visualization, and correlation of system events. During evaluation, AURA exhibited consistently strong performance across all functional layers. The system achieved a detection accuracy of 94.3%, confirming its ability to identify malicious traffic with high reliability. The false positive rate remained at a minimal 3.2%, indicating effective differentiation between legitimate and harmful network activity. The average detection latency was measured at 2.6 seconds, and the alert delivery time via the Twilio API averaged 1.8 seconds, validating the framework's near real-time operational capability.

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 09, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1209.16

A. Security Effectiveness

We assessed AURA's security effectiveness by evaluating its capability to detect, isolate, and document active cyber threats across various simulated conditions. By integrating anomaly-based detection, real-time threat intelligence, and honeypot deception, the system demonstrated robust resistance against reconnaissance, brute-force, and network intrusion attempts. During testing, the honeypot module successfully engaged with simulated attackers, capturing detailed command sequences, payloads, and connection metadata without ever exposing critical host assets. Meanwhile, the automated firewall achieved a 100% blocking success rate, effectively preventing all high-risk IP addresses from re-establishing connections.

B. Performance Evaluation

We analyzed AURA's performance efficiency by monitoring its CPU utilization, memory overhead, and response latency during continuous simulation of network activity. Even under conditions of elevated traffic and simultaneous alert generation, the system maintained an average CPU utilization of only 6.5% and negligible memory overhead, confirming its lightweight footprint and suitability for endpoint deployment. The system's microservice-based modular design enabled concurrent execution of critical operations—such as packet analysis, anomaly scoring, and dashboard updates—without causing bottlenecks or process delays. This efficient resource distribution ensured that AURA remained responsive even during high-volume network events. These observations validate that AURA achieves a balanced trade-off between detection precision and performance efficiency, allowing real-time operation without compromising user experience or system stability.

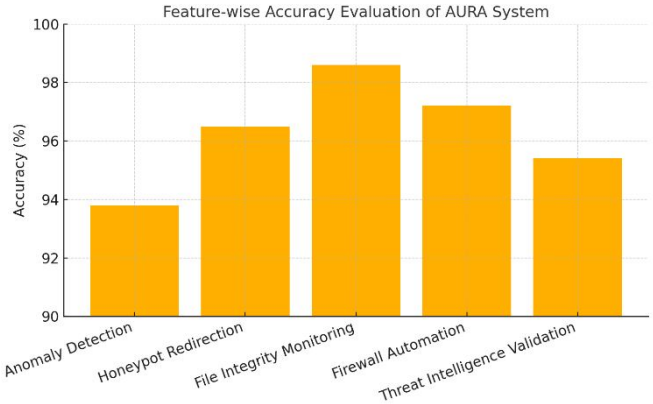

Fig. 8. Feature-wise accuracy comparison of AURA modules showing consistent performance across all components.

The File Integrity Monitoring (FIM) module recorded the highest detection accuracy at 98.6%, effectively identifying unauthorized file modifications with near-perfect precision. Both the Firewall Automation and Honeypot Redirection modules achieved accuracies exceeding 96%, showcasing their efficiency in mitigating live network threats and engaging hostile entities in controlled deception environments. Similarly, the Anomaly Detection Engine and Threat Intelligence Validation modules maintained accuracies above 93%, demonstrating robust performance in identifying irregular communication patterns and verifying malicious reputations through external data sources.

VI. CONCLUSION AND FUTURE SCOPE

AURA presents an integrated, autonomous, and modular approach to endpoint cybersecurity. Designed as a lightweight yet powerful defense framework, AURA consolidates multiple protection mechanisms—anomaly-based intrusion detection, honeypot deception, threat intelligence validation, firewall automation, and file integrity monitoring—into a unified architecture. This combination bridges the gap between enterprise-grade security systems and the limited defense tools typically available to personal users, delivering robust protection without dependence on external cloud services or resource-intensive machine learning models. Comprehensive experimental evaluation under simulated real-world attack scenarios validated AURA's effectiveness. The system achieved a detection accuracy of 94.3% with a false positive rate of only 3.2%, confirming its ability to distinguish between malicious and legitimate network activity. The average detection latency of 2.6 seconds and alert delivery time of 1.8 seconds demonstrated near real-time responsiveness. Furthermore, the Honeypot module effectively contained and logged high-risk connections, capturing attacker behaviors without compromising the host system's integrity. The results confirm that AURA offers a scalable, efficient, and adaptive security solution capable of functioning independently across diverse operating environments. Its microservice-based modular design ensures flexibility for updates and extensions, allowing future incorporation of advanced detection algorithms, cloud-assisted analytics, and AI-driven threat prediction models.

Future Enhancements

Several avenues exist for strengthening AURA's capabilities in future iterations:

Federated Learning Integration: AURA could be enhanced by integrating federated learning to share anonymized behavioral patterns across multiple systems, improving anomaly-detection precision without sacrificing user privacy.

Cross-Platform Compatibility: Expanding the framework for Linux and Android endpoints could extend its usability beyond personal desktops, making it viable for a wider range of devices.

Blockchain-Backed Integrity Verification: Incorporating blockchain technology could further strengthen trust and immutability in audit trails, providing tamper-proof security logs.

Adaptive Machine Learning: Future versions could integrate adaptive machine learning algorithms that continuously learn from user behavior and dynamic network conditions, improving detection precision over time.

Cloud Synchronization Layer: A cloud-based component could enable centralized aggregation of threats and cross-device event correlation, allowing AURA to function as a distributed intelligence-sharing network.

Modular Plugin Architecture: Introducing a plugin system could encourage open-source collaboration, allowing external developers to contribute new detection algorithms or integration tools. This would transform AURA into an open, adaptive, and community-driven cybersecurity ecosystem capable of evolving alongside the threat landscape.

Research Contributions

From a research perspective, AURA introduces several key technical contributions to the domain of personal cybersecurity:

- Lightweight integration of multi-layered protection mechanisms without reliance on signature databases, centralized control, or high-end computational resources
- Fully automated detection-to-response pipeline, minimizing human dependency and ensuring instant threat containment
- Decentralized personal defense architecture, optimized for individual users, small-scale enterprises, and educational environments
- Enhanced usability and accessibility, achieved through an intuitive Flask-based dashboard and real-time WhatsApp alerting system, extending security awareness beyond the local environment

Overall, AURA demonstrates that effective cybersecurity for personal systems can be achieved through intelligent integration of automation, adaptability, and transparency. Its lightweight design and autonomous operation make it a viable foundation for the next generation of self-learning, decentralized security systems capable of defending against the ever-evolving landscape of cyber threats. AURA's layered defense model reinforces the foundational cybersecurity principle of defense-in-depth, where each protection layer complements the others to ensure continuous situational awareness, automated adaptability, and resilience against evolving cyber threats. The integration of multiple detection and mitigation strategies within a unified architecture establishes AURA as both a practical and forward-thinking security framework for endpoint systems.

REFERENCES

1. F.T. Liu,K.M.Ting, and Z.-H. Zhou, "Isolation Forest," Proceedings of the 8th IEEE International Conference on Data Mining (ICDM), Pisa, Italy, pp. 413-422, Dec. 2008.
2. "Tripwire File Integrity Monitoring Solution," Tripwire, Inc., Portland, OR, USA. [Online]. Available: https://www.tripwire.com
3. C.Taylor,"What is a Honeypot? How it Helps Security Teams Detect Attacks," CSO Online, 2023. [Online]. Available: https://www.csoonline.com
4. "AbuseIPDB: IP Address Abuse Reports," AbuseIPDB Threat Intelligence Platform, 2024. [Online]. Available: https://www.abuseipdb.com
5. "Spamhaus DROP and EDROP Lists," The Spamhaus Project, 2024. [Online]. Available: https://www.spamhaus.org/drop/
6. A.Bejtlich, The Practice of Network Security Monitoring: Understanding Incident Detection and Response, 1st ed., San Francisco, CA: No Starch Press, 2013.
7. S.Axelsson, "The Base-Rate Fallacy and the Difficulty of Intrusion Detection," ACM Transactions on Information and System Security (TISSEC), vol. 3, no. 3, pp. 186-205, 2000.
8. D.E.Denning, "An Intrusion-Detection Model," IEEE Transactions on Software Engineering, vol. SE-13, no. 2, pp. 222-232, Feb. 1987.
9. M.K.Rogers, C. Seigfried-Spellar, and J. E. K. Reddy, "The Role of Honeypots in Digital Forensic Investigations," Digital Investigation Journal, vol. 12, pp. S66-S75, 2015.
10. S.H.Patel and A.B.Desai, "Real-Time File Integrity Monitoring Using Cryptographic Hashing and Automated Alerts," International Journal of Information Security Science, vol. 11, no. 2, pp. 105-115, 2022.
11. A.Joshi and D.Sardeshmukh, "Automated Threat Response Using Firewall and SIEM Integration," IEEE Access, vol. 9, pp. 88912-88922, 2021.
12. L.Spitzner, "Honeypots:Tracking Hackers," Addison-Wesley Professional, Boston, MA, 2003.
13. A. Bhattacharya and M.Thapa, "Evaluation of Open-Source IDS Frameworks