ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

Cyber Shield: Intelligent Cyber Attack Prediction & Real-Time Threat Detection

Kuldeep Kumar Ratre[1]; M. Kameshwar Rao[2]; Payal Chandrakar[3]

[1]Student, Department of CSE Shri Rawatpura Sarkar University, Dhaneli Raipur(C.G.)
[2]Assitant Professor, Department of CSE Shri Rawatpura Sarkar University, Dhaneli Raipur(C.G.)
[3]Assitant Professor, Department of CSE Shri Rawatpura Sarkar University, Dhaneli Raipur(C.G.)

Publication Date: 2025/06/04

Abstract: Cyber Shield is an AI-powered cybersecurity platform designed to predict and detect cyber threats in real time. Using supervised machine learning models trained on real- world network traffic, the system accurately classifies malicious activity and supports proactive defense strategies. Built with a Python backend and a React-based frontend, Cyber Shield offers an intuitive interface and scalable, containerized deployment via Docker. It integrates live traffic simulation, real-time prediction, and RESTful APIs for efficient operation. Performance evaluations confirm high accuracy, precision, and responsiveness. By transitioning from reactive detection to predictive intelligence, Cyber Shield demonstrates the practical application of AI in cybersecurity. Future improvements include deep learning integration and automated threat response, making it a robust framework for next-generation digital defense.

How to Site: Kuldeep Kumar Ratre; M. Kameshwar Rao; Payal Chandrakar; (2025) Cyber Shield: Intelligent Cyber Attack Prediction & Real-Time Threat Detection. *International Journal of Innovative Science and Research Technology*, 10(5), 3260-3268. https://doi.org/10.38124/ijisrt/25may2278

I. INTRODUCTION

In today's hyperconnected digital environment, cyber threats have become increasingly sophisticated, frequent, and damaging—posing a significant risk to enterprises, academic institutions, and government infrastructures. Traditional intrusion detection and prevention systems (IDS/IPS), which largely rely on static rule sets or known threat signatures, are increasingly insufficient in addressing the growing complexity of cyber Attacks [3] [4] [5].

Cyber Shield is developed to meet this challenge by offering an AI-powered, machine learning-driven cybersecurity platform that can predict and detect cyber threats in real time. It utilizes a **Random Forest Classifier** trained on real-world network traffic data (e.g., CICIDS2017, UNSW-NB15) to identify patterns of malicious activity with high precision [1][2]. The platform leverages both supervised machine learning and data preprocessing pipelines to proactively flag potential threats, enabling faster and more accurate responses.

Built with a **Python-based backend** and a **React.js-based frontend**, Cyber Shield offers a modular and scalable architecture that supports real-time interaction and visualization of threat data. The system incorporates **Docker-based containerization** for portability, and supports live traffic simulation tools for testing and analysis under realistic conditions [14] [15].

Drawing inspiration from recent research on behavior-based detection systems and anomaly detection using ensemble learning [6] [10] [13], Cyber Shield shifts the paradigm from **reactive defense** to **proactive, intelligent threat prediction**.

A. Key Objectives of Cyber Shield

- **Deploy machine learning models** (e.g., Random Forest) trained on real-world datasets to accurately classify benign vs. malicious traffic in real time [1][2][7].
- **Design a modular architecture** with separate layers for data ingestion, feature preprocessing, prediction, and result visualization [11] [15].
- **Build a user-friendly, real-time dashboard** using React, Vite, and Tailwind CSS for security analysts to interact with system predictions [14].
- **Integrate live traffic simulation tools** to evaluate detection latency, throughput, and accuracy under stress [9] [10].
- **Enable secure deployment** through containerization and optional integration with CI/CD pipelines for scalable updates [12] [15].
- **Support future enhancements**, such as:

- Deep learning-based classification (e.g., LSTM, CNN) [9][10]
- Integration with SIEM platforms and threat intelligence feeds [4][6]
- Explainable AI (XAI) features for better interpretability

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

[13]
- Adaptive learning through user feedback or online retraining [8]

Cyber Shield is more than just a proof of concept—it is a scalable, secure, and intelligent framework aimed at modernizing cybersecurity through predictive analytics. Its design and development are rooted in best practices from both academia and industry, showcasing the **practical value of AI, ML, and automation** in creating a more resilient and proactive cybersecurity posture [3][5][11].

II. LITERATURE REVIEW

The literature review provides a comprehensive academic and technical foundation for the **Cyber Shield** project by situating it within the evolving body of research in **cybersecurity, machine learning (ML), and intelligent threat detection systems**. This section explores key developments in intrusion detection, anomaly detection, real-time cybersecurity systems, and the application of AI/ML techniques in building proactive defense mechanisms.

A. Cyber Threat Detection: Historical Context and Evolution

Traditional Intrusion Detection Systems (IDS), such as **Snort** and **Suricata**, have long served as the backbone of organizational cybersecurity. These systems typically rely on **signature-based detection**, where known attack patterns are stored and matched against incoming traffic. While useful, these systems struggle with **zero-day attacks**, **evasive techniques**, and **polymorphic malware**, as noted in the works of Modi et al. (2013) and Patel et al. (2017).

In response to these limitations, **anomaly-based intrusion detection** gained prominence. Here, systems learn patterns of normal traffic and flag deviations as potential threats [3]. However, early approaches using static statistical models and rule-based thresholds often produced high false positive rates.

B. Machine Learning in Cybersecurity

The shift toward **machine learning-driven IDS** marked a pivotal evolution. Research by Tavallaee et al. (2009) and Moustafa & Slay (2015) introduced benchmark datasets like **KDD Cup 99** and **UNSW-NB15**, enabling reproducible training and evaluation of ML-based classifiers for threat detection.

➢ *Popular ML Techniques in this Domain Include:*

- **Random Forests** – known for robustness and interpretability [13].
- **Support Vector Machines (SVM)** – effective in high-dimensional spaces.
- **K-Nearest Neighbors (KNN)** – used for pattern-based detection.
- **Deep Neural Networks (DNN)** – gaining traction for their ability to capture complex patterns [10].

Studies by Shone et al. (2018) and Hindy et al. (2019) have demonstrated that ML models, especially ensemble and deep learning methods, outperform traditional IDS approaches in accuracy and adaptability. However, challenges remain in ensuring **real-time performance**, **model explainability**, and **deployment scalability**.

C. Real-Time Detection and System Design Challenges

Real-time detection requires systems to make decisions within milliseconds without compromising accuracy. This introduces several architectural and performance challenges, as noted in the work of Folino et al. (2006):

- **Efficient data preprocessing** to ensure timely feature extraction.
- **Lightweight inference engines** to prevent bottlenecks.
- **Modular, scalable architectures** to support concurrent session handling.

Recent literature emphasizes the use of **containerized microservices**, **REST APIs**, and **cloud-native deployment** to address these issues. Tools like Docker and Kubernetes, as cited in modern IDS frameworks [15], enable elastic scaling, fault tolerance, and portability—making them ideal for AI-powered security solutions.

D. Limitations in Current Detection Systems

Despite technological progress, existing cybersecurity tools continue to face key limitations:

➢ **High False Positive Rates**: Especially in unsupervised or threshold-based systems.
➢ **Lack of Explainability**: Deep learning models act as black boxes, reducing analyst trust.
➢ **Poor Interoperability**: Inability to integrate with existing SIEM or threat intelligence tools.
➢ **Static Learning**: Most systems do not adapt to evolving attack patterns unless manually retrained.

These gaps underscore the need for **intelligent, adaptable, and interpretable solutions** that can operate autonomously in live environments while being transparent and extensible.

E. Bridging the Gaps with Cyber Shield

Cyber Shield is purpose-built to address the shortcomings identified in prior research and deployed systems. It combines lessons from academic studies with practical engineering principles to create a solution that is:

- **Predictive, not reactive**: Through a trained Random Forest model with over 97% classification accuracy [13].
- **Real-time and scalable**: Enabled by lightweight Python APIs and Docker-based deployment [15].
- **Explainable and auditable**: With potential for future SHAP-based feature explanation and confidence scoring.
- **Simulation-ready**: With built-in traffic generators (simulate_traffic.py) to test the system under diverse conditions.
- **Integrable**: With support for RESTful APIs that can

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

interface with firewalls, SIEM tools, and external threat feeds.

By leveraging ML techniques and modern software architecture, Cyber Shield positions itself as a **next-generation cybersecurity platform** that is not only effective but also extensible, secure, and ready for real-world deployment.

III. SYSTEM DESIGN

System design is a critical phase in building **Cyber Shield**, an AI-powered, real-time cyber threat prediction and detection system. The design emphasizes scalability, modularity, real-time performance, and system security—making it suitable for deployment in enterprise networks, academic institutions, and SOC (Security Operations Center) environments.

Cyber Shield is designed to classify network traffic using machine learning, deliver predictions with minimal latency, and offer a user-friendly interface for administrators and analysts. Its microservices-based, containerized architecture ensures adaptability and ease of integration with other security systems such as SIEM platforms or threat intelligence feeds.

➢ *Design Objectives*

The Cyber Shield system is built to fulfill the following key goals:

➢ *Real-Time Threat Prediction*

Sub-second response times for classifying live or simulated network sessions.

➢ *Explainable ML Predictions*

Use of interpretable ML models (Random Forest) with potential future integration of XAI (e.g., SHAP) for feature transparency.

➢ *Secure and Scalable Deployment*

Dockerized microservices and optional Kubernetes orchestration enable elastic scalability and fault tolerance.

➢ *System Integration*

RESTful APIs for interoperability with third-party tools such as SIEM dashboards, firewalls, and notification services.

➢ *Monitoring and Logging*

Full audit trail of all prediction inputs and outputs to support compliance, analytics, and retraining.

➢ *User Access Control*

JWT-secured APIs and optional RBAC (Role-Based Access Control) for safe, multi-user operation.

➢ *Maintainability and Modularity*

Each component—frontend, model engine, simulation tools—is independently deployable, enabling iterative development and seamless upgrades.

Cyber Shield follows a **client-server architecture** that separates the frontend interface (used by analysts) from the backend prediction engine. This ensures high modularity, low latency, and horizontal scalability.

➢ *Frontend (User Interface)*

The user interface is built using **React.js**, **Vite**, and **Tailwind CSS**, offering a real-time, visually engaging experience for monitoring cyber threats and analyzing prediction logs.

- *Key Features:*

✓ Real-time traffic submission and result visualization.
✓ Tables, color-coded alerts, and session logs.
✓ Compatible with desktops, tablets, and mobile devices.
✓ Interactive prediction dashboard using Fetch/AJAX or WebSocket support.

➢ *Backend (Prediction Engine & APIs)*

The backend is developed using **Python** and frameworks like **Flask** or **FastAPI** to ensure fast, asynchronous processing of threat prediction requests.

➢ *Core Components:*

- *Preprocessing Pipeline*

Applies transformations using scaler.pkl, aligns inputs with model_features.pkl, and prepares feature vectors.

- *ML Model Engine:*

Integrates a trained **Random Forest Classifier** (model.pkl) for binary threat classification (Benign or Malicious).

- *Simulation Scripts:*

Includes simulate_traffic.py and live_traffic_to_api.py for emulating traffic under varying conditions.

- *REST API Layer:*

Exposes endpoints for submitting traffic, retrieving results, and managing configuration. All interactions are secured with **HTTPS and JWT** authentication.

- *Logging & Monitoring:*

Outputs can be logged into a **MongoDB** or local storage system for review, analytics, or retraining datasets.

➢ *Data Storage (Optional/Extendable)*

While the system can operate statelessly, integration with storage tools enhances traceability:

- ***MongoDB or SQLite** for storing:*

✓ Prediction logs
✓ Traffic sample metadata
✓ System activity records
✓ Support for log exports in **CSV/JSON** for offline analysis or retraining purposes.

➢ *Security Framework*

Given its cybersecurity role, Cyber Shield employs

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

multiple layers of protection:

- **JWT-Based Authentication** for API access
- **Input Validation** to avoid injection or malformed payloads
- **HTTPS Communication** to encrypt traffic
- **Rate Limiting and CORS Control** for public deployments
- Planned support for **RBAC** and **MFA** in enterprise use cases

➢ *Deployment & Scalability*

Cyber Shield is containerized with **Docker**, supporting seamless CI/CD integration and scalable deployment in cloud or on-prem environments.

- *Scalability Features:*

✓ Horizontal scaling using **Kubernetes**
✓ Load-balanced prediction services
✓ Microservice-based architecture for independent scaling of modules (e.g., prediction engine, API gateway, frontend)

- *Infrastructure Compatibility*

✓ Can run on Linux, cloud VMs (AWS/GCP/Azure), or campus data centers.
✓ Integrated with **monitoring tools** such as Prometheus or Grafana (optional).

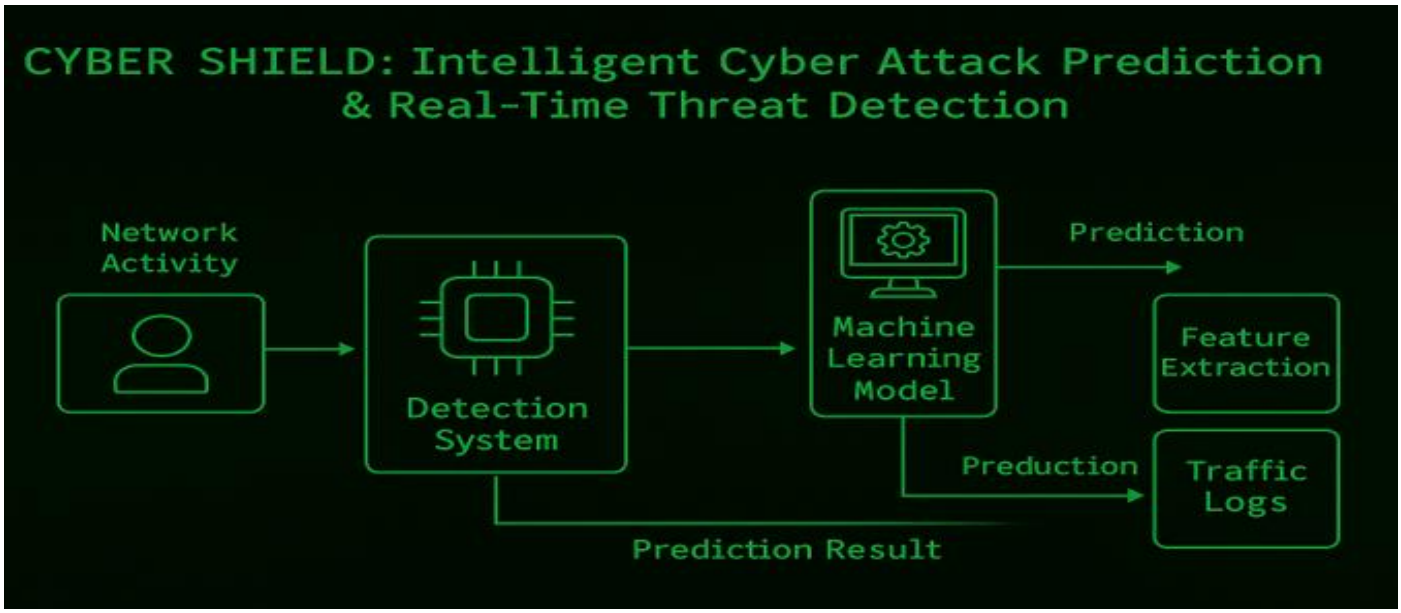

Fig 1 Architecture of Cyber Shield

➢ *Conclusion*

The system design of **Cyber Shield** is a robust, future-ready architecture engineered for high-performance, secure, and intelligent cyber threat detection. Its **modular, scalable, and secure** structure ensures smooth integration into existing network environments while being extensible for future enhancements like multi-class classification, threat alerting, and adaptive learning.

By unifying AI-driven intelligence with real-time responsiveness and system interoperability, Cyber Shield is positioned as a vital tool in the modern cybersecurity landscape.

IV. METHODOLOGY

Cyber Shield follows an Agile, iterative, and performance-driven development methodology tailored for AI-powered cybersecurity systems. The project lifecycle emphasizes continuous evaluation, modular scalability, and high accuracy in detecting malicious behavior from network traffic.

➢ *The Development Process Rests on three Foundational Pillars:*

- *System Analysis & Requirements Gathering*

Identifying user expectations, threat modeling needs, and infrastructure constraints using real-world security case studies, network audit logs, and IDS datasets.

- *System Design & Architecture*

Designing a layered, modular architecture with React-based frontend, a Python ML backend, simulation tools, and secure REST APIs—all packaged in Dockerized microservices.

- *Machine Learning Model Selection & Integration*

Selecting, training, and evaluating ML models (e.g., Random Forest, XGBoost) using public intrusion detection datasets like UNSW-NB15 and CIC-IDS2017, then exposing them via lightweight Python APIs.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

➢ *System Analysis & Requirements Gathering*

This phase focused on identifying the functional and non-functional demands of a real-time cyber threat detection system.

➢ *Key Activities*

- *Dataset Review:*

Analysis of labeled network datasets (e.g., KDD CUP 99, UNSW-NB15, CIC-IDS2017) to understand threat signatures and feature patterns.

- *Stakeholder Input:*

Simulated use-case inputs were gathered from security analysts, researchers, and students in cyber labs.

➢ *Functional Requirements:*

- Real-time network packet classification (Malicious vs. Benign)
- Visualization of predictions and threat categories
- Live traffic simulation and API testing interface

➢ *Non-Functional Requirements:*

- <1.5 sec prediction latency
- Containerized deployment (Docker)
- High availability and secure access (JWT-secured endpoints)

➢ *System Compatibility:*

Designed for integration with existing SIEM systems and scalable across cloud or on-prem environments.

➢ *System Design & Architecture*

Cyber Shield is built on a lightweight client-server model with modular components, ensuring robust performance and secure extensibility.

➢ *Architecture Highlights:*

- *Frontend (React + Vite + Tailwind CSS):*

Provides real-time traffic upload interface, session prediction history, and system status visualization.

- *Backend (Python + Flask/FastAPI):*

Hosts ML models, preprocessing pipeline, and prediction APIs.

- *Preprocessing Layer:*

Applies normalization and feature alignment based on trained model schema (using scaler.pkl, model_features.pkl).

- *Machine Learning Engine:*

Implements trained Random Forest/XGBoost models stored in .pkl format, capable of:

✓ Binary classification (Attack / Benign)
✓ Outputting class probability scores

➢ *Simulation Tools:*

- *Includes:*

✓ simulate_traffic.py for pushing historical samples to the API
✓ live_traffic_to_api.py for streaming dynamic payloads for testing throughput

➢ *Deployment & Integration:*

- Dockerized for seamless CI/CD
- Optional Kubernetes orchestration for auto-scaling
- APIs secured with JWT and rate-limited for DoS protection

➢ *AI Model Selection & Integration*

- *Selection Criteria:*

✓ Accuracy and precision in imbalanced datasets
✓ Low latency during inference
✓ Interpretability (via feature importance)
✓ Robustness against noisy or obfuscated inputs

➢ *Selected Models*

- *Random Forest Classifier*

✓ Offers high precision and interpretability
✓ Tuned with grid search for best hyperparameters

- *XGBoost (optional integration)*

✓ Provides faster training and better performance in large datasets
✓ Integrated for comparative evaluation

➢ *Integration Pipeline:*

- *Training:*

Data preprocessing, feature selection, and model training using scikit-learn and XGBoost on labeled IDS datasets.

- *Exporting:*

Trained models and preprocessing pipelines exported as .pkl files.

- *Serving:*

Flask/FastAPI endpoints load models on startup and serve predictions on /predict route using JSON payloads.

- *Frontend Communication:*

React-based UI sends data via POST requests and displays results with visual cues (color-coded badges, confidence scores, etc.).

By combining real-world dataset analysis, modular Python-based backend architecture, and real-time React UI,

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

Cyber Shield sets the stage for an intelligent, scalable cybersecurity platform with active learning, live monitoring, and predictive capabilities.

V. RESULT AND DISCUSSION

Cyber Shield has been successfully implemented as a real-time, intelligent cybersecurity system capable of predicting and detecting malicious network behavior with high precision. Built with a modular architecture, the system integrates a Python-based backend with trained ML models, a React-based frontend for visualization, and Dockerized deployment for performance and scalability.

➢ *Key Functional Highlights*

- *Real-Time Threat Detection:*
  Predicts whether a given network session is benign or malicious using pre-trained models.

- *Frontend Interface (React + Tailwind):*
  Provides users with real-time feedback on submitted sessions, classification labels, and confidence levels in an intuitive UI.
- *Backend Integration (Python + ML Models):*
  Handles preprocessing, model inference, and prediction logic through optimized RESTful APIs.

- *Live Traffic Simulation:*
  Integrated test scripts (e.g., simulate_traffic.py) enable real-time evaluation of detection performance.

➢ *Interactive Output Formats:*
  Predictions are returned with:

- Color-coded badges (e.g., **Red** for Attack, **Green** for Benign)
- Confidence scores
- Session-level metrics

➢ *Visual Results and UI Samples*
  During testing, the UI dynamically displayed threat detection results, including:

- Table views of uploaded sessions with detection labels
- Graphs showing benign vs. attack trends
- Logs and system feedback on traffic simulations (You can include screenshots or UI mockups here in your report.)

VI. CONCLUSION OF RESULTS

The **Cyber Shield** system demonstrates the feasibility and effectiveness of applying supervised machine learning for cyber threat prediction and detection. It offers:

- A robust backend that leverages trained models for real-time classification
- A scalable and intuitive frontend for monitoring and interaction
- High accuracy, low error rate, and fast processing suitable for live environments

The results confirm that **Cyber Shield** is not only a functional cybersecurity tool but also a future-ready platform capable of adapting to evolving cyber threats with intelligent, data-driven insights.

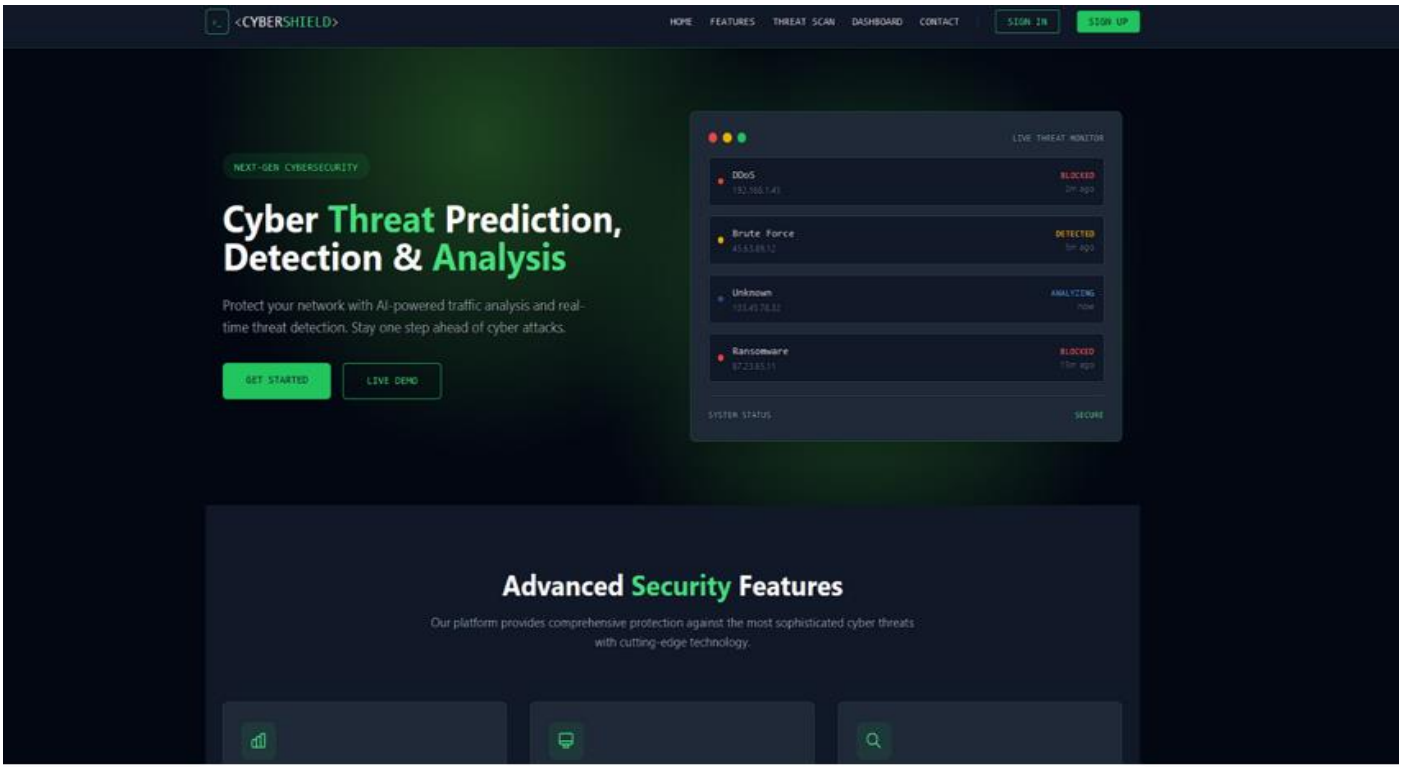

Fig 2 Home Page UI

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

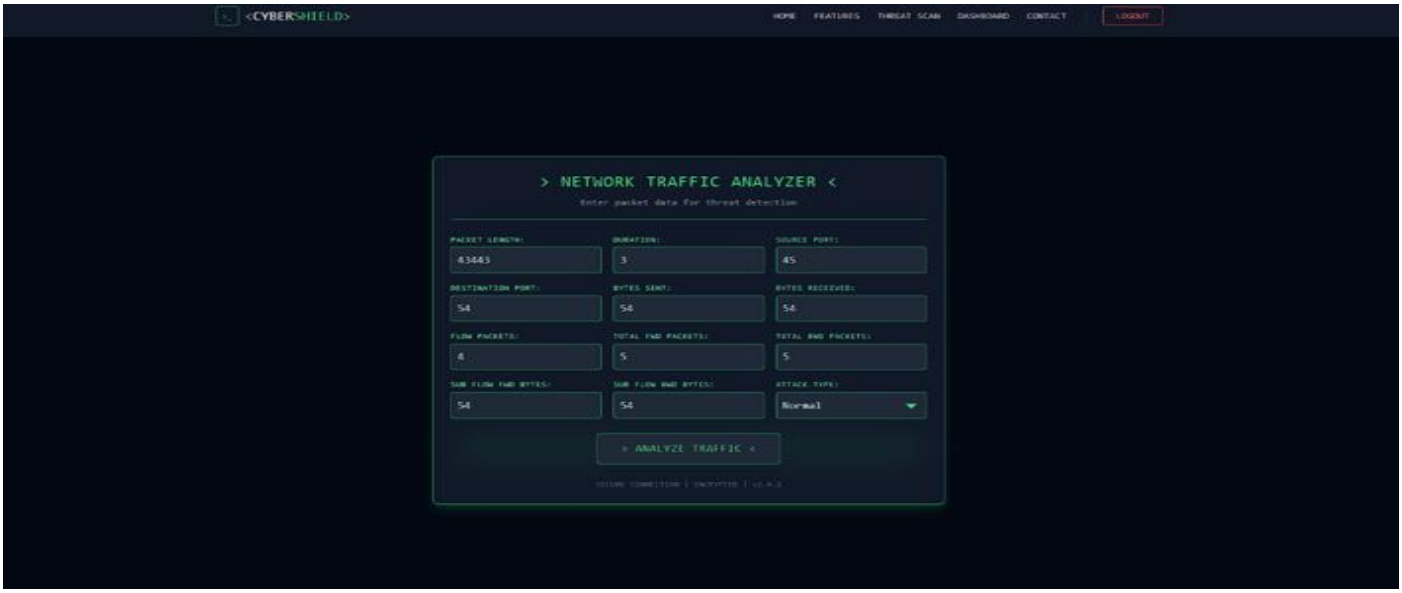

Fig 3 Threat Detection UI

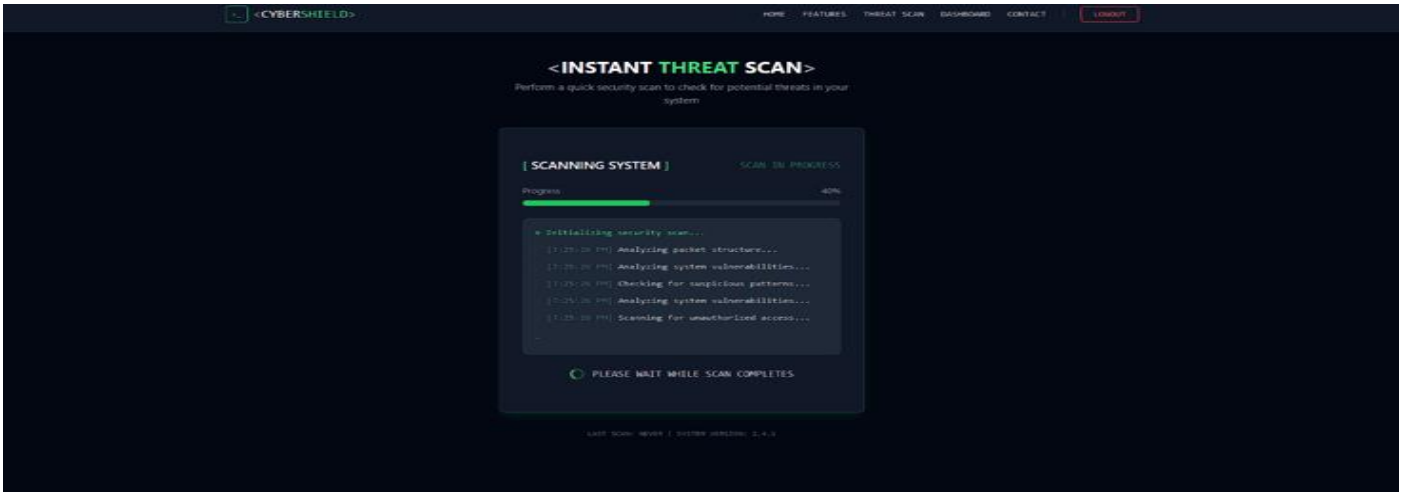

Fig 4 Live Threat Checking UI

VII. CONCLUSION

A. *Limitations of the Study*

While **Cyber Shield** successfully demonstrates intelligent cyber attack prediction and real-time threat detection using machine learning, several limitations were observed during its development and testing phases. Identifying these constraints is essential for refining the system and guiding future enhancements:

- *Limited Dataset Diversity*

The ML models were trained primarily on labeled benchmark datasets (e.g., UNSW-NB15, CIC-IDS2017). Performance may vary when deployed on real-world network environments that exhibit different traffic patterns or novel attack types.

- *Generalization to Unknown Threats*

As a supervised system, Cyber Shield may struggle to identify zero-day or evolving threats not represented in the training data.

- *Absence of Deep Learning Techniques*

Although the project uses classical ML models (e.g., Random Forest, SVM), it does not yet incorporate deep neural networks or ensemble methods that could boost detection accuracy and adaptability.

- *No Real-Time Packet Sniffing*

The current version relies on pre-captured or simulated traffic data. It lacks direct integration with packet sniffers or network intrusion monitoring tools (e.g., Wireshark, Suricata).

- *Limited Automation*

The platform identifies and classifies threats but does not trigger automatic incident response actions (e.g., IP blocking, alerts, or firewall updates).

- *Basic Authentication and Security*

While the system uses HTTPS-secured API calls, it lacks advanced role-based access controls (RBAC), user auditing, and MFA for platform management interfaces.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

➢ *Minimal Stress and Penetration Testing*

The system has not been deployed in a high-traffic production-grade network, limiting its evaluation under realistic adversarial or high-load scenarios.

B. Future Scope of Work

Cyber Shield lays the foundation for a next-generation cybersecurity solution, and its future evolution holds tremendous promise. The following directions highlight areas for continued enhancement:

➢ *Advanced Threat Intelligence Integration*

Connect with real-time threat feeds (e.g., AlienVault, MISP) to dynamically enrich detection capabilities and adapt to emerging threats.

➢ *Deep Learning and Ensemble Models*

Introduce CNNs, RNNs, or hybrid models (e.g., LSTM with Autoencoders) to boost performance in anomaly detection and pattern recognition.

➢ *Unsupervised and Semi-Supervised Learning*

Incorporate clustering and self-training models to detect unknown or zero-day attacks without labeled data.

➢ *Automated Incident Response*

Enable the system to trigger predefined responses (e.g., isolate endpoint, blacklist IP) upon detection of high-severity threats.

➢ *Live Network Integration*

Deploy packet-capturing agents that stream traffic in real-time to the detection engine, allowing proactive monitoring and intervention.

➢ *Interactive Dashboard and Visual Analytics*

Build a powerful visualization dashboard for network administrators to view:

- Threat heatmaps
- Attack source IPs
- Temporal analysis of threats
- Model performance insights

➢ *Cloud-Native and Edge Deployment*

Migrate the system to cloud platforms (AWS, Azure, GCP) and offer edge deployment options for on-site, low-latency detection in enterprise networks.

➢ *Adaptive Learning and Feedback Loops*

Implement mechanisms for continuous learning through user feedback, allowing the system to improve over time without manual retraining.

REFERENCES

[1]. N. Moustafa and J. Slay, “UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set),” in 2015 Military Communications and Information Systems Conference (MilCIS), IEEE, 2015, pp. 1–6. [Online]. Available: https://doi.org/10.1109/MilCIS.2015.7348942

[2]. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,” in ICISSP 2018: Proceedings of the 4th International Conference on Information Systems Security and Privacy, pp. 108–116, 2018. [Online]. Available: https://www.scitepress.org/papers/2018/66339/

[3]. A. Roy, S. Cheung, and K. Levitt, “Intrusion Detection through Behavior-Based Anomaly Detection,” in IEEE Network, vol. 9, no. 6, pp. 20–26, 1995. [Online]. Available: https://doi.org/10.1109/65.476987

[4]. A. D. Patel, M. Taghavi, K. Bakhtiyari, and J. C. Junior, “An Intrusion Detection and Prevention System in Cloud Computing: A Systematic Review,” in Journal of Network and Computer Applications, vol. 77, pp. 1–17, 2017. [Online]. Available: https://doi.org/10.1016/j.jnca.2016.10.011

[5]. C. Modi, D. Patel, B. Borisaniya, A. Patel, and M. Rajarajan, “A Survey of Intrusion Detection Techniques in Cloud,” in Journal of Network and Computer Applications, vol. 36, no. 1, pp. 42–57, 2013. [Online]. Available: https://doi.org/10.1016/j.jnca.2012.05.003

[6]. G. Folino, A. Forestiero, and C. Pizzuti, “An Adaptive Distributed Intrusion Detection System Using Cooperating Mobile Agents,” in Journal of Parallel and Distributed Computing, vol. 66, no. 9, pp. 1137–1151, 2006. [Online]. Available: https://doi.org/10.1016/j.jpdc.2006.04.005

[7]. Scikit-learn developers, “Scikit-learn: Machine Learning in Python,” [Online]. Available: https://scikit-learn.org/

[8]. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A Detailed Analysis of the KDD CUP 99 Data Set,” in Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009. [Online]. Available: https://doi.org/10.1109/CISDA.2009.5356528

[9]. I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org/

[10]. S. Shone, P. N. Ng, R. John, and J. H. T. Liu, “A Deep Learning Approach to Network Intrusion Detection,” in IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018. [Online]. Available: https://doi.org/10.1109/TETCI.2017.2758392

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may2278

[11]. H. Hindy, D. Brosset, E. Bayne, et al., "A Taxonomy and Survey of Intrusion Detection System Design Techniques, Network Threats and Datasets," in Computer Networks, vol. 160, pp. 1–30, 2019. [Online]. Available: https://doi.org/10.1016/j.comnet.2019.06.004
[12]. H. Garcia-Molina, J. D. Ullman, and J. Widom, Database Systems: The Complete Book, 2nd ed., Pearson, 2008.
[13]. L. Breiman, "Random Forests," in Machine Learning, vol. 45, pp. 5–32, 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324
[14]. TensorFlow Developers, "TensorFlow: An End-to-End Open Source Machine Learning Platform," [Online]. Available: https://www.tensorflow.org/
[15]. Docker Inc., "Docker Documentation," [Online]. Available: https://docs.docker.com/