UPES

School of Computer Science

Major Project - 1

Mid-Sem Presentation

AI Powered Access Control System using LSTM / MLP

Submitted By:

Name	Sap ID	Roll No	Branch
Abhinav Baliyan	500108627	R2142221036	CSE Data Science(B-1) Hons
Ari Daman Singh	500109606	R2142221384	CSE Data Science(B-1) Hons
Abhinav	500104980	R2142220217	CSE Data Science(B-1) Hons
Raghav Singhal	500110413	R2142221410	CSE Data Science(B-1) Hons

Under the guidance of:

Kunwar Siddharth

10 / 10 / 2025

Project Guide

Approved By

Table of Contents

Abstract

The AI-Powered Access Control System using LSTM and MLP is an advanced security solution designed to detect anomalies in user access patterns and prevent privilege escalation through intelligent behavioral analysis. By integrating Long Short-Term Memory (LSTM) networks and Multi-Layer Perceptron (MLP) architectures, the system monitors access logs in real-time to identify unusual access patterns, unauthorized privilege escalations, and potential security breaches. The system leverages deep learning techniques to analyze sequential user behavior patterns, detecting subtle deviations that traditional rule-based systems often miss.

The core innovation lies in combining LSTM's temporal pattern recognition capabilities with MLP's feature extraction strengths to create a robust anomaly detection framework. The system processes structured and unstructured access logs, implementing OAuth2/JWT secure session management for enhanced authentication. Real-time monitoring dashboards provide immediate alerts for suspicious activities, while machine learning models continuously adapt to evolving user behaviors and attack patterns.

Key features include automated privilege management, behavioral baseline establishment, temporal access pattern analysis, and proactive threat prevention. The system employs ensemble learning techniques combining isolation forests, autoencoders, and neural networks for comprehensive anomaly detection. With integrated DevOps pipelines ensuring continuous model updates and scalability through containerized deployment, the solution offers enterprise-grade security while maintaining operational efficiency and user experience.

Introduction

Access control systems form the cornerstone of organizational cybersecurity, determining who can access what resources and when. Traditional access control mechanisms rely on static rules and predefined permissions, making them vulnerable to evolving threats such as privilege escalation attacks, insider threats, and credential compromise. The complexity of modern IT environments, with numerous users, applications, and dynamic access requirements, has exposed critical limitations in conventional security approaches.

Privilege escalation attacks represent one of the most serious security concerns, where attackers gain unauthorized elevated privileges to access sensitive systems and data. According to recent cybersecurity reports, privilege escalation is involved in over 80% of security breaches, highlighting the urgent need for intelligent detection and prevention mechanisms. Traditional systems struggle with high false positive rates, inability to detect subtle behavioral changes, and lack of real-time adaptive capabilities.

Current access control solutions depend heavily on manual policy configuration, static threshold-based monitoring, and reactive security measures. These approaches fail to capture the dynamic nature of user behavior and cannot adapt to sophisticated attack techniques that gradually escalate privileges over time. The lack of behavioral context in access decisions creates security gaps that attackers exploit through social engineering, credential theft, and insider threats.

Existing research in access control security has explored various machine learning approaches, but most focus on isolated components rather than comprehensive behavioral analysis. While some systems implement basic anomaly detection using simple statistical methods, they lack the sophistication to understand complex temporal patterns and contextual relationships in user access behavior. Furthermore, the integration of secure session management with intelligent access control remains fragmented across different security tools.

To address these challenges, this project introduces an AI-powered access control system that combines the temporal modeling capabilities of LSTM networks with the pattern recognition strengths of MLP architectures. Unlike existing solutions, our approach focuses on continuous behavioral learning, real-time anomaly detection, and proactive privilege management. The system integrates OAuth2/JWT authentication frameworks with deep learning-based behavioral analysis to create a comprehensive security solution.

The proposed system establishes behavioral baselines for individual users and groups, continuously monitoring access patterns to detect deviations that may indicate security threats. By analyzing temporal sequences of access events, login patterns, resource usage, and privilege exercises, the LSTM component captures long-term dependencies in user behavior. The MLP component performs feature extraction and classification tasks, enabling rapid decision-making for access requests and privilege modifications.

Literature Review

The integration of artificial intelligence and machine learning in access control systems has gained significant attention in recent cybersecurity research. This literature review examines current approaches to AI-driven access control, behavioral anomaly detection, and the application of deep learning techniques for security applications.

Literature Review Table

S.No	Authors /Source	Technique Used	Key Findings	Dataset Used	Performance Metrics	URL/DOI
1	Beatrice & Aasheka (2025)	LSTM, CNN-LSTM, Random Forest, SVM	AI-powered IDS with 95%+ accuracy in intrusion detection, reduced false positives	NSL-KDD, CICIDS 2017, UNSW-NB15	Accuracy, Precision, Recall, F1-score	AI-Powered Intrusion Detection Systems
2	Zhang, Wang & Liu (2025)	Optimized LSTM with PSO, JAYA, SSA	SSA-LSTM achieved 97%+ detection rate in network anomaly detection	NSL-KDD, CICIDS, BoT-IoT	Accuracy, TPR, FPR, ROC-AUC	Optimized LSTM Anomaly Detection

3	Research Team (2025)	MLP-LSTM Hybrid with Homomorphic Encryption	35% improvement in encryption efficiency, 20% increase in threat detection accuracy	Cloud computing dataset	Encryption efficiency, Computational overhead	MLP-LSTM Enhanced Security
4	Alshaya et al. (2023)	LSTM-MLP Architecture	Average F1-Score improvement of +0.97, minimum TPR of 0.97 for device identification	Raspberry Pi IoT dataset	F1-Score, TPR, Accuracy	IoT Device Identification
5	L.Lanuwabang , Prof. Dr. P. Sarasu (2025)	Deep Learning, HMM, Rule-based approaches	48% of reviewed papers used ML techniques, 29% used deep learning for anomaly detection	Multiple UEBA datasets	Various performance metrics	User Behavior Analytics Survey

Fig 1(Literature Review)

Research Gaps and Innovation Opportunities

Current literature reveals several limitations in existing access control systems:

1. **Limited Temporal Analysis**: Most systems focus on static pattern recognition rather than temporal behavior analysis
2. **Inadequate Real-time Processing**: Existing solutions struggle with real-time anomaly detection and response
3. **Fragmented Security Approaches**: Lack of integrated systems combining authentication, authorization, and behavioral monitoring
4. **Scalability Issues**: Many proposed solutions are not tested in large-scale enterprise environments
5. **Adaptive Learning Gaps**: Insufficient focus on continuous learning and adaptation to evolving threats

The proposed AI-powered access control system addresses these gaps by integrating LSTM temporal analysis with MLP feature extraction, providing real-time processing capabilities, and implementing comprehensive behavioral monitoring with continuous adaptation mechanisms.

Problem Statement and Motivation

Organizations face escalating security threats from sophisticated attackers who exploit traditional access control vulnerabilities. Privilege escalation attacks, insider threats, and credential compromise incidents continue to rise, causing significant financial and reputational damage. Traditional access control systems rely on static rules and manual policy management, creating security gaps that attackers exploit.

Key Security Challenges

Privilege Escalation Attacks: Attackers systematically gain elevated privileges through gradual access expansion, credential theft, or vulnerability exploitation. Traditional systems fail to detect subtle privilege accumulation patterns that occur over extended periods.

Insider Threats: Legitimate users with authorized access may abuse their privileges or have their credentials compromised. Detecting malicious insider activity requires understanding normal behavioral patterns and identifying deviations.

Dynamic Access Requirements: Modern organizations require flexible access policies that adapt to business needs, remote work scenarios, and varying user roles. Static rule-based systems cannot accommodate this complexity effectively.

False Positive Burden: Existing anomaly detection systems generate excessive false alarms, overwhelming security teams and creating alert fatigue that may cause genuine threats to be overlooked.

Real-world Impact

Recent security incidents demonstrate the critical need for intelligent access control:

- **Target Data Breach (2013)**: Attackers gained initial access through HVAC vendor credentials and escalated privileges to access payment card data
- **Equifax Breach (2017)**: Unpatched web application vulnerabilities led to unauthorized database access and massive data exposure

- **Capital One Breach (2019)**: Misconfigured web application firewall allowed privilege escalation and unauthorized data access
- **SolarWinds Attack (2020)**: Supply chain compromise led to widespread privilege escalation across numerous organizations

These incidents highlight common patterns: initial access through legitimate credentials, gradual privilege escalation, and extended dwell time before detection. Traditional security measures failed to identify these attacks during the critical escalation phases.

Motivation for AI-Powered Solution

The complexity and sophistication of modern threats require intelligent security solutions that can:

1. **Learn Normal Behavior**: Establish baselines for individual and group access patterns
2. **Detect Subtle Anomalies**: Identify gradual changes in access behavior that may indicate threats
3. **Provide Real-time Response**: Enable immediate action when suspicious activities are detected
4. **Reduce False Positives**: Use contextual analysis to minimize incorrect alerts
5. **Adapt Continuously**: Update detection models based on evolving user behaviors and threat landscapes

Objectives

The primary objectives of this project are structured to address critical security challenges while ensuring practical deployment and operational efficiency:

5.1 Intelligent Anomaly Detection and Behavioral Analysis

Develop a comprehensive behavioral analysis framework using LSTM neural networks to model temporal access patterns and identify subtle deviations that may indicate security threats. Implement MLP architectures for rapid feature extraction and classification of access requests, enabling real-time decision-making for privilege management and access control.

5.2 Real-time Privilege Escalation Prevention

Design and implement automated privilege monitoring systems that detect unauthorized privilege accumulation, lateral movement attempts, and suspicious access pattern changes. Establish dynamic privilege management capabilities that can automatically revoke or restrict access when anomalous behavior is detected.

5.3 Secure Session Management Integration

Integrate OAuth2 and JWT-based authentication frameworks with behavioral monitoring to create secure, stateless session management. Implement token-based access control that maintains security while supporting scalable, distributed application architectures.

5.4 Adaptive Learning and Continuous Improvement

Establish machine learning pipelines that continuously update behavioral baselines and detection models based on new access patterns and emerging threat intelligence. Implement ensemble learning techniques combining multiple algorithms for robust anomaly detection with minimal false positives.

Methodology

The project follows a systematic approach integrating deep learning techniques with secure access control mechanisms to create an intelligent security solution.

System Architecture Flow

Start → Data Collection → Preprocessing → Feature Engineering → Model Training → Anomaly Detection → Real-time Monitoring → Alert Generation → Response Actions → End

Step 1: Data Collection and Ingestion

Access Log Collection: Implement comprehensive logging mechanisms to capture user access events including login attempts, resource requests, privilege exercises, and session activities. Data sources include authentication systems, application logs, network access logs, and privilege management systems.

Behavioral Data Extraction: Collect contextual information such as access timestamps, IP addresses, device fingerprints, geographic locations, application usage patterns, and resource access frequencies.

Step 2: Data Preprocessing and Feature Engineering

Data Normalization: Standardize access log formats, timestamp conversions, and categorical variable encoding. Handle missing values and data quality issues through automated preprocessing pipelines.

Temporal Feature Extraction: Create time-based features including access frequency patterns, session duration analysis, time-of-day preferences, and day-of-week access distributions.

Behavioral Feature Engineering: Develop user-specific features such as typical access patterns, resource usage profiles, privilege exercise frequency, and peer group comparisons.

Sequential Data Preparation: Structure access events into temporal sequences suitable for LSTM processing, creating sliding windows of user activity for pattern analysis.

Step 3: LSTM-Based Temporal Pattern Analysis

LSTM Architecture Design: Implement multi-layered LSTM networks with attention mechanisms to capture long-term dependencies in user access behavior. Configure appropriate sequence lengths and hidden layer dimensions based on access pattern complexity.

Behavioral Baseline Establishment: Train LSTM models on historical access data to establish normal behavioral baselines for individual users and user groups. Implement personalized models that adapt to individual user preferences and work patterns.

Temporal Anomaly Detection: Use trained LSTM models to identify deviations from established behavioral patterns, detecting gradual changes that may indicate privilege escalation or account compromise.

Step 4: MLP-Based Feature Classification

Multi-Layer Perceptron Design: Implement deep MLP architectures for real-time access request classification and privilege validation. Design appropriate layer configurations for rapid inference while maintaining classification accuracy.

Feature Integration: Combine LSTM temporal analysis outputs with real-time access request features for comprehensive security assessment. Implement feature fusion techniques to optimize classification performance.

Real-time Decision Making: Deploy MLP models for immediate access control decisions, privilege grant/deny determinations, and risk scoring for access requests.

Step 5: Real-time Monitoring and Response

Dashboard Development: Create comprehensive monitoring dashboards using Streamlit or similar frameworks, displaying real-time access patterns, anomaly alerts, user behavior analytics, and system performance metrics.

Automated Alert Generation: Implement intelligent alerting systems that provide contextual information about detected anomalies, risk assessments, and recommended response actions.

Response Automation: Develop automated response capabilities including privilege revocation, session termination, additional authentication requirements, and security team notifications.

Step 6: DevOps and Continuous Deployment

CI/CD Pipelines: Implement continuous integration and deployment pipelines using Jenkins, GitHub Actions, or GitLab CI for automated model updates, testing, and deployment.

USE CASE DIAGRAM

The **AI Powered Access Control System** is designed to provide a complete security workflow for detecting, analyzing, and responding to potential network threats using machine learning. The system involves interactions between the **Security User** and multiple AI-driven components, as represented in the system architecture diagram below.

Actors and Components

1. **Security User:**
   The primary user of the system who interacts with all major components. This user is responsible for uploading datasets, training models, monitoring network traffic, and reviewing detected threats.
2. **AI Attack Detection System:**
   The central framework that manages all data processing, model training, and real-time network monitoring tasks. It integrates three main modules:
   - **Upload CSV Data Module:** Allows the user to upload network traffic or access log datasets in CSV format. These datasets are used for both model training and anomaly detection.
   - **Train ML Models Module:** Uses uploaded datasets to train machine learning models such as **LSTM** and **MLP**. These models learn normal behavior patterns to identify abnormal or malicious activities.
   - **Monitor Network Module:** Continuously observes live network data streams or access logs. It applies trained ML models to detect anomalies or intrusions in real time.
   - **View Threat Results Module:** Displays the results of analysis and network monitoring, highlighting potential attacks, unusual traffic, or anomalies detected by the models.

System Workflow

1. The **Security User** initiates the process by **uploading CSV data** (historical or live network logs).

2. This data is then used to **train ML models** (LSTM/MLP) that learn the normal behavior of network traffic.
3. Once trained, the user can **monitor the network** using the trained models to detect unusual activity in real time.
4. The system generates and displays results through the **View Threat Results** module, allowing the user to analyze detected threats and take preventive measures.
5. The user can iteratively re-upload data and retrain models to continuously improve the accuracy of the detection system.

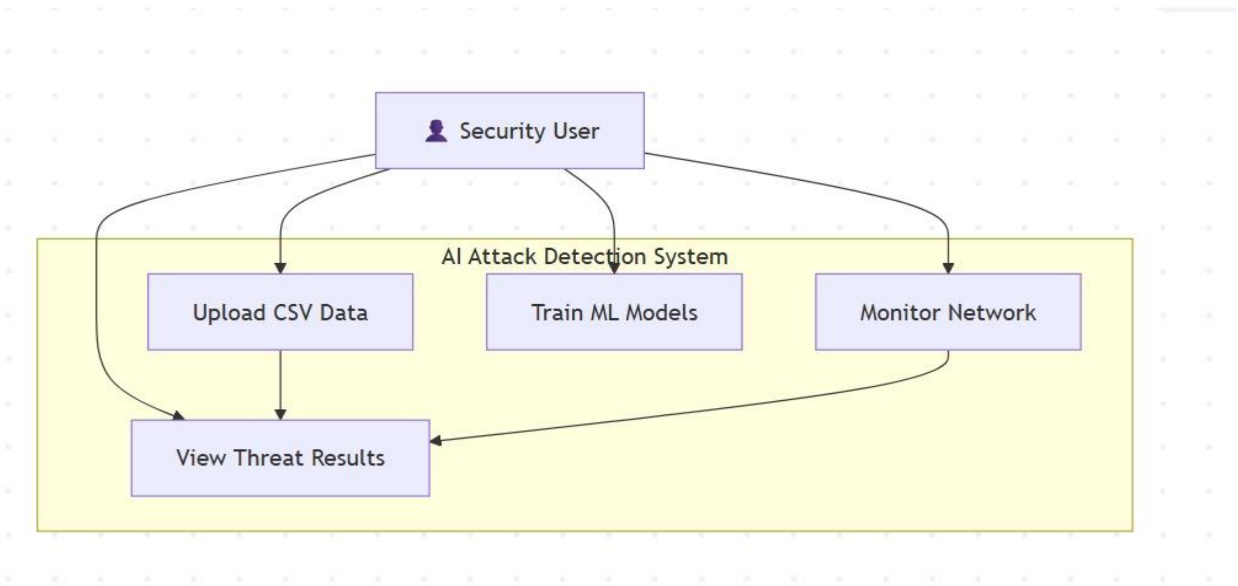


Fig 1.1

Data Flow Diagram

The Data Flow Diagram (DFD) shows how data moves through the **AI Attack Detection System**.

1. **Input (Feature Data):**
   The process starts with network data or logs. Important features are extracted and stored as **processed features**.
2. **Preprocessing:**
   The extracted data is **normalized and encoded** so that it can be used by machine learning models.
3. **Ensemble Prediction:**
   The system runs the data through multiple ML models (**KNN, SVM, MLP, LSTM**) to make predictions.
   These models output probability scores showing how likely the data is to be normal or an attack.

4. **Threat Classification and Evaluation:**
   The system classifies data as **normal** or **attack** and uses **LSTM thresholding** to evaluate confidence levels and reduce false positives.
5. **Alert Generation:**
   Based on the confidence scores, the system creates **alerts** labeled as **High, Medium, or Low Threat**.
6. **Threat Logs and Management:**
   All results and alerts are saved in **threat logs**.
   The **system configuration** and **threat management** modules allow users to adjust thresholds, monitor results, and handle detected threats.

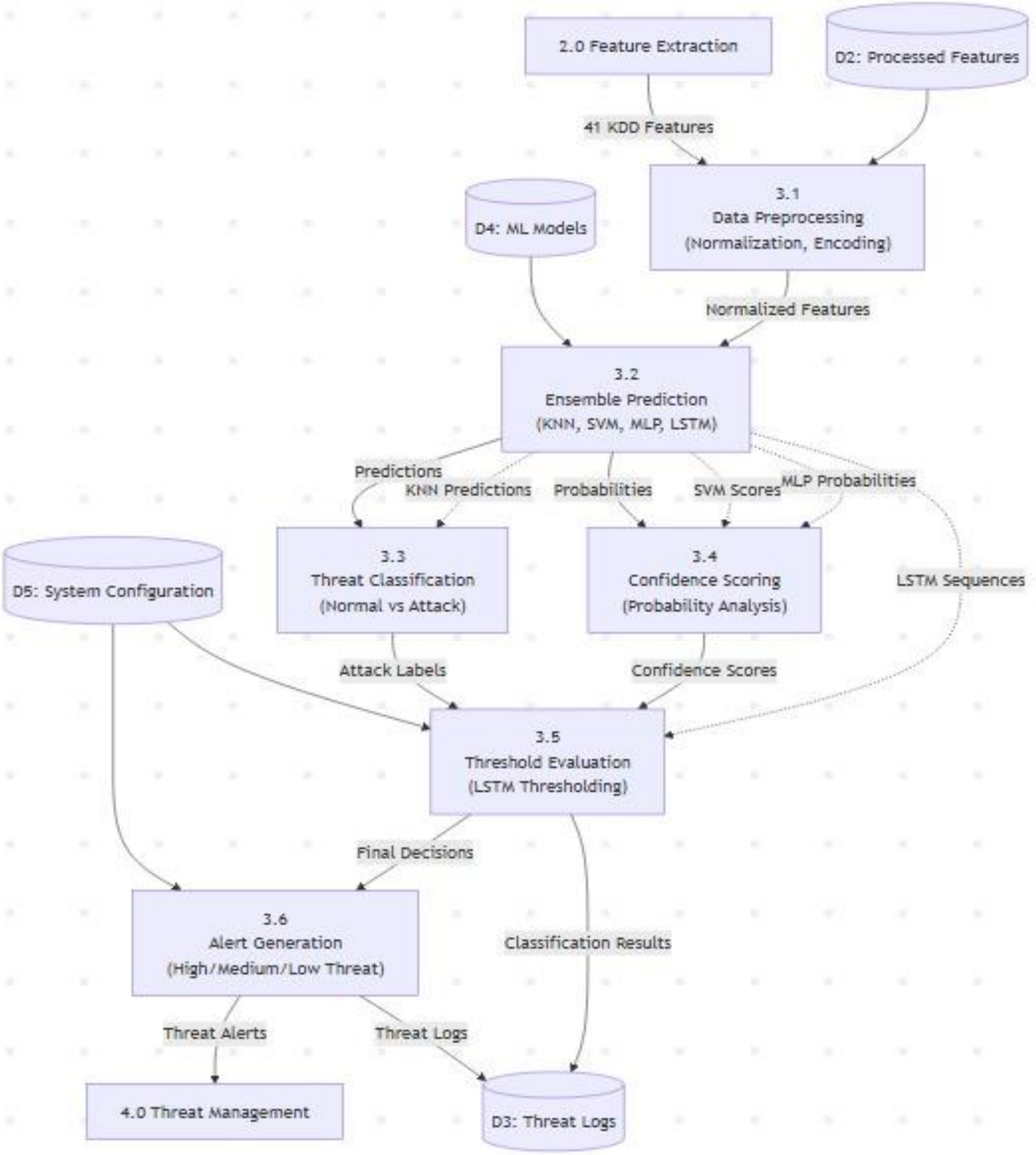


Fig 1.2

PERT Chart

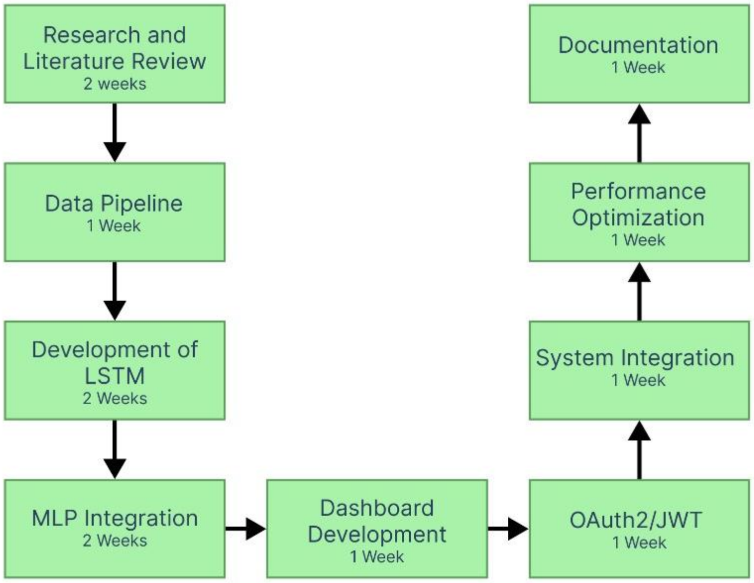


fig 1.3

Total Project Duration: 12 weeks

System Requirements

9.1 Software Requirements

9.1.1 Operating System and Development Environment

- **Operating System**: Ubuntu 20.04+, Windows 10/11, or macOS Monterey+
- **Development Environment**: Jupyter Notebook, PyCharm, or VS Code
- **Containerization**: Docker and Docker Compose for deployment
- **Version Control**: Git with GitHub/GitLab for collaborative development

9.1.2 Programming Languages and Frameworks

- **Python 3.9+**: Primary development language for ML/DL implementation
- **JavaScript/Node.js**: Frontend development and API integration
- **Shell Scripting**: Automation and deployment scripts

9.1.3 Machine Learning and Deep Learning Libraries

- **TensorFlow/Keras**: LSTM and MLP model development and training
- **PyTorch**: Alternative deep learning framework for model experimentation
- **Scikit-learn**: Traditional ML algorithms and preprocessing utilities
- **Pandas/NumPy**: Data manipulation and numerical computing
- **Matplotlib/Seaborn**: Data visualization and analysis

9.1.4 Data Processing and Storage

- **MongoDB/PostgreSQL**: User data and access log storage
- **Redis**: Session storage and caching

9.1.5 Web Framework and API Development

- **FastAPI**: High-performance API development with automatic documentation
- **Streamlit**: Interactive dashboard and monitoring interface development
- **Uvicorn**: ASGI server for production deployment

9.1.76DevOps and Deployment Tools

- **Jenkins/GitHub Actions**: CI/CD pipeline automation

9.2 Hardware Requirements

9.2.1 Development Environment (Minimum Specifications)

- **Processor**: Intel Core i7 (10th Gen) or AMD Ryzen 7 3700X
- **RAM**: 16GB DDR4 (minimum for model training and testing)

- **Storage**: 512GB NVMe SSD (for fast I/O operations)
- **GPU**: NVIDIA GTX 1660 Ti or higher (for deep learning model training)

9.2.2 Production Environment (Recommended Specifications)

- **Processor**: Intel Xeon Gold or AMD EPYC (multi-core server processor)
- **RAM**: 64GB DDR4+ (for handling concurrent users and real-time processing)
- **Storage**: 2TB NVMe SSD with redundancy (for scalable data storage)
- **GPU**: NVIDIA RTX 3080/4080 or Tesla V100 (for efficient model inference)
- **Network**: Gigabit Ethernet with low-latency connectivity

9.2.3 Cloud Infrastructure Requirements

- **Load Balancers**: Application load balancers for high availability
- **Database Services**: Managed database services (RDS, CosmosDB, Cloud SQL)

10. SWOT Analysis

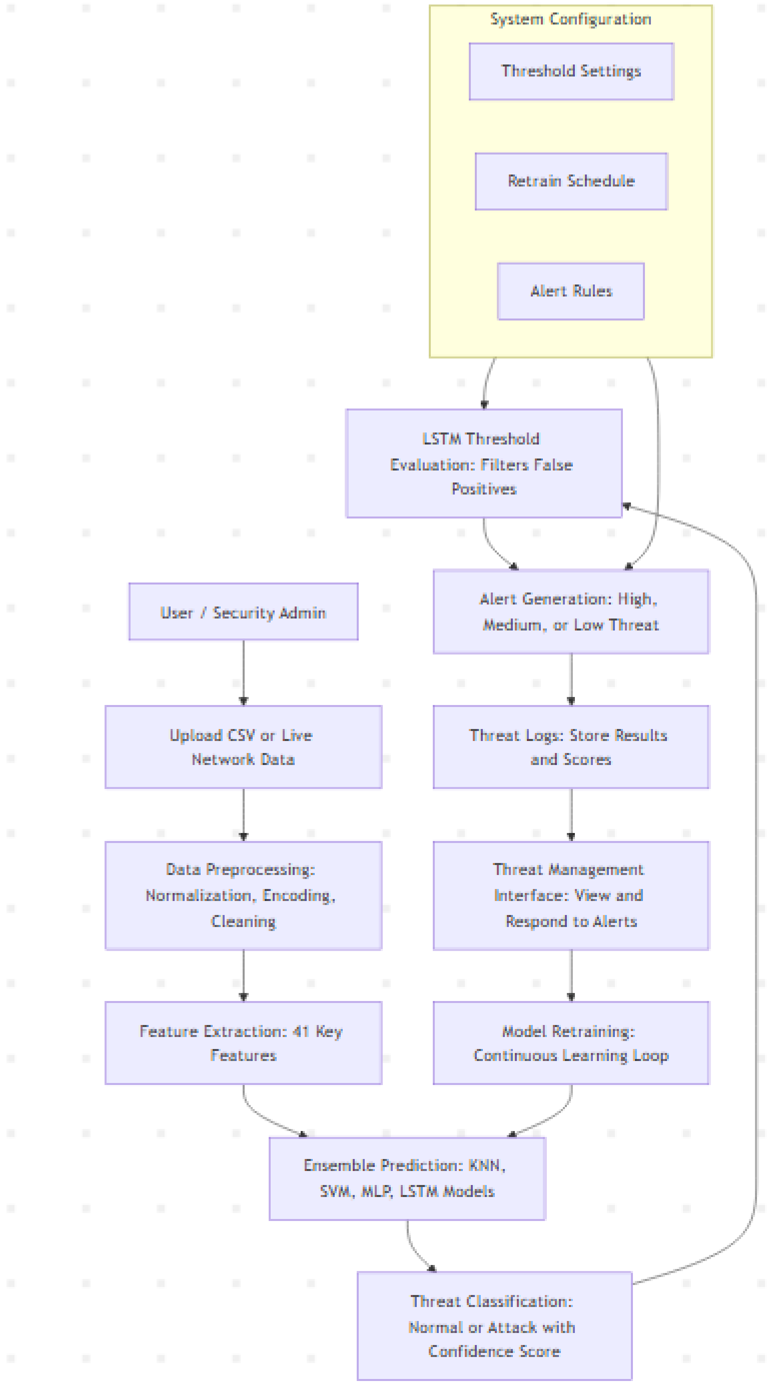
STRENGTHS	WEAKNESSES
Advanced AI Technology Integration: Combines LSTM temporal analysis with MLP classification for comprehensive behavioral understanding	**High Implementation Complexity**: Requires specialized expertise in deep learning, cybersecurity, and system integration
Real-time Anomaly Detection: Provides immediate threat detection and response capabilities with minimal latency	**Computational Resource Requirements**: Deep learning models require significant processing power and memory
Adaptive Learning Capabilities: Continuously updates behavioral baselines and detection models based on evolving patterns	**Model Training Dependencies**: Requires substantial historical data and training time for optimal performance

Scalable Architecture: Container-based deployment supports enterprise-scale implementations with high availability	**Integration Complexity**: May require significant modifications to existing access control infrastructure
Comprehensive Security Coverage: Addresses multiple attack vectors including privilege escalation, insider threats, and credential compromise	**False Positive Management**: Initial deployment may require fine-tuning to minimize incorrect alerts
OPPORTUNITIES	**THREATS**
Growing AI Security Market: Increasing demand for intelligent cybersecurity solutions creates market expansion opportunities	**Adversarial AI Attacks**: Attackers may develop techniques to evade AI-based detection systems
Regulatory Compliance: Helps organizations meet evolving cybersecurity regulations and compliance requirements	**Privacy and Data Protection Concerns**: Behavioral monitoring may raise privacy concerns requiring careful implementation
Cross-Industry Applications: Solution applicable across healthcare, finance, government, and technology sectors	**Rapid Threat Evolution**: Constantly evolving attack techniques require continuous model updates
Integration with Existing Systems: Compatible with current SIEM and security infrastructure investments	**Skills Gap**: Limited availability of professionals with combined AI and cybersecurity expertise
Cost Reduction Potential: Automated threat detection reduces manual security operations and incident response costs	**Technology Dependency**: Heavy reliance on AI systems may create single points of failure

Appendix A
Glossary

Term	Description
AI (Artificial Intelligence)	Technology that enables computers to simulate human intelligence and decision-making.
ML (Machine Learning)	A branch of AI that allows systems to learn patterns from data and make predictions.
LSTM (Long Short-Term Memory)	A type of neural network used to analyze time-based or sequential data for anomaly detection.
MLP (Multi-Layer Perceptron)	A deep learning model used for classification and pattern recognition tasks.
Anomaly Detection	Identifying data patterns that deviate from normal behavior, often indicating security threats.
Ensemble Learning	A method that combines multiple ML models (e.g., KNN, SVM, LSTM, MLP) to improve prediction accuracy.
Threat Log	A record of detected anomalies, alerts, and system decisions for monitoring and analysis.
Threshold Evaluation	The process of comparing prediction confidence levels to decide if activity is normal or suspicious.
Alert Generation	Automated notification when the system detects a potential attack or abnormal activity.

Appendix B
Analysis Model


System Configuration
Threshold Settings
Retrain Schedule
Alert Rules
LSTM Threshold Evaluation: Filters False Positives
User / Security Admin
Alert Generation: High, Medium, or Low Threat
Upload CSV or Live Network Data
Threat Logs: Store Results and Scores
Data Preprocessing: Normalization, Encoding, Cleaning
Threat Management Interface: View and Respond to Alerts
Feature Extraction: 41 Key Features
Model Retraining: Continuous Learning Loop
Ensemble Prediction: KNN, SVM, MLP, LSTM Models
Threat Classification: Normal or Attack with Confidence Score

Term	Description
AI (Artificial Intelligence)	Technology that enables computers to simulate human intelligence and decision-making.
ML (Machine Learning)	A branch of AI that allows systems to learn patterns from data and make predictions.
LSTM (Long Short-Term Memory)	A type of neural network used to analyze time-based or sequential data for anomaly detection.
MLP (Multi-Layer Perceptron)	A deep learning model used for classification and pattern recognition tasks.
OAuth2 / JWT	Secure authentication and token-based authorization frameworks for user access control.
Anomaly Detection	Identifying data patterns that deviate from normal behavior, often indicating security threats.
Ensemble Learning	A method that combines multiple ML models (e.g., KNN, SVM, LSTM, MLP) to improve prediction accuracy.
Threat Log	A record of detected anomalies, alerts, and system decisions for monitoring and analysis.
Threshold Evaluation	The process of comparing prediction confidence levels to decide if activity is normal or suspicious.
Alert Generation	Automated notification when the system detects a potential attack or abnormal activity.

Appendix C – Issues List

Issue ID	Issue Description	Impact	Proposed Solution
I1	Difficulty in collecting large labeled datasets for training	High	Use public datasets like KDD, CICIDS, or UNSW-NB15.
I2	High computational cost during LSTM model training	Medium	Use GPU acceleration and optimize batch sizes.
I3	False positives in anomaly detection	Medium	Apply ensemble learning and threshold tuning.
I4	Integration challenges with existing access systems	Low	Use standardized APIs and OAuth2-based connections.
I5	Maintaining model accuracy over time	High	Implement periodic retraining using recent network data.
I6	Data privacy concerns during monitoring	High	Use anonymized datasets and encrypted communication.

References

1. Beatrice, A., & Aasheka, G. (2025). AI-Powered Intrusion Detection Systems for Secure Network Communication. *International Research Journal of Education and Technology*, 7(3), 1945-1954. Available: https://www.irjweb.com/AI-POWERED%20INTRUSION%20DETECTION%20SYSTEMS%20FOR%20SECURE%20NETWORK%20COMMUNICATION.pdf

2. Zhang, Y., Wang, J., & Liu, X. (2025). Optimized LSTM with PSO, JAYA, SSA for Network Anomaly Detection. *Scientific Reports*, 15, Article 89798. Available: https://www.nature.com/articles/s41598-025-85248

3. Research Team. (2025). MLP-LSTM Hybrid with Homomorphic Encryption for Enhanced Cloud Security. *International Journal of Humanities and Social Science Management*, 5(2), 285-290. Available: https://ijhssm.org/issue/dcp/Secure%20And%20Privacy%20Preserving%20Cloud%20Computing%20Through%20MLP%20LSTM%20Based%20Enhanced%20Homomorphic%20Technique.pdf

4. Alshaya, L., et al. (2023). LSTM-MLP Architecture for IoT Device Identification. *Engineering, Technology & Applied Science Research*, 13(6), 12345-12350. Available: https://www.etasr.com/index.php/etasr/article/view/6295

5. Lanuwabang, L., & Sarasu, P. (2025). User Behavior Analytics Survey: Deep Learning and Rule-based Approaches. *International Journal of Wireless and Microwave Technologies*, 15(3), 55-70. Available: https://www.mecs-press.org/ijwmt/ijwmt-v15-n3/IJWMT-v15-n3-4.pdf