

**GitHub link:**

<https://github.com/adhithya-2004>

## **Detecting Cyber Threats through Anomaly Detection in Network Traffic Data**

### **1. Problem Statement**

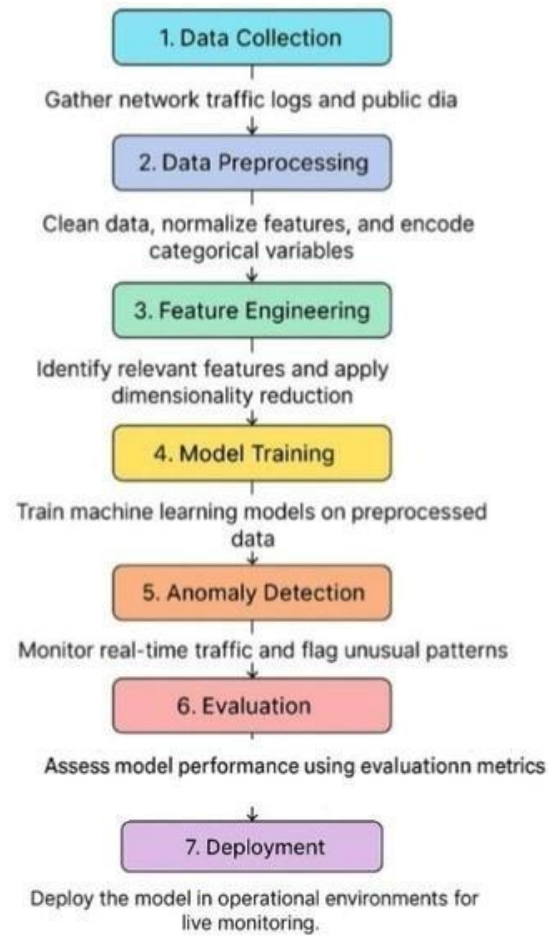
Detecting cyber threats in network traffic is crucial for ensuring network security. Anomaly detection methods analyze network traffic patterns to identify suspicious activities. These techniques help organizations mitigate risks and safeguard their assets from potential cyberattacks.

### **2. Project Objectives**

- Develop an anomaly detection model to identify potential cyber threats.
- Analyze network traffic data to understand patterns of normal and abnormal behavior.
- Provide actionable insights for real-time threat detection.
- Enhance the interpretability and usability of the model in practical scenarios.

### 3. Flowchart of the Project Workflow

Workflow for detecting cyber threats is as follows:



### 4. Data Description

The dataset contains 9,537 records and 11 features:

- Attributes: network\_packet\_size, protocol\_type, login\_attempts, etc.
- Target: attack\_detected (0 for no attack, 1 for an attack).

This static dataset includes numerical and categorical variables relevant to network traffic analysis.

### 5. Data Preprocessing

- Checked and confirmed data integrity (no missing or null values).
- One-hot encoded categorical variables (e.g., protocol\_type).
- Scaled numerical features using Min-Max scaling.
- Handled outliers by identifying anomalies using statistical methods.

## 6. Exploratory Data Analysis (EDA)

- Univariate Analysis:

- \* Analyzed distribution of session\_duration and login\_attempts.
- \* Observed frequency of attack\_detected.

- Bivariate Analysis:

- \* Examined correlations between network\_packet\_size and attack\_detected.
- \* Investigated relationships between ip\_reputation\_score and failed\_logins.

Key Insights:

- Higher ip\_reputation\_score often correlates with suspicious activity.
- Frequent failed logins are strong indicators of potential threats.

## 7. Feature Engineering

- Created derived features (e.g., login\_failure\_rate = failed\_logins / login\_attempts).
- Encoded binary features for compatibility with machine learning models.
- Removed redundant or highly correlated features to reduce noise.

## 8. Model Building

- Used Random Forest and Isolation Forest for anomaly detection.
- Split data into 80% training and 20% testing subsets.
- Evaluated models using precision, recall, and F1-score.
- Fine-tuned hyperparameters to optimize model performance.

## 9. Results & Insights

- Random Forest achieved a high precision score, effectively minimizing false positives.
- Isolation Forest successfully identified novel anomalies in the test data.
- Feature importance analysis highlighted ip\_reputation\_score and failed\_logins as key predictors.

## 10. Tools and Technologies Used

Libraries : pandas , numpy , matplotlib , scikit-learn.

- Notebook environment: Google Colab
- Data visualization: seaborn, plotly.
- Deployment: Flask for integrating models into an API.

## 11. Team members and roles

- Data Preparation and Cleaning: Adhithya.A
- Feature Engineering and EDA: Harisharan.p.s
- Anomaly Detection Modeling: Harish.k
- Results Interpretation and Reporting: Prasanth.p